Irwin King
Jun Wang
Laiwan Chan
DeLiang Wang (Eds.)

LNCS 4234

Neural Information Processing

13th International Conference, ICONIP 2006
Hong Kong, China, October 2006
Proceedings, Part III

3 Part III

Springer

Irwin King Jun Wang Laiwan Chan
DeLiang Wang (Eds.)

Neural Information Processing

13th International Conference, ICONIP 2006
Hong Kong, China, October 3-6, 2006
Proceedings, Part III

Springer

Volume Editors

Irwin King
Laiwan Chan
Chinese University of Hong Kong
Department of Computer Science and Engineering
Shatin, New Territories, Hong Kong
E-mail:{king,lwchan}@cse.cuhk.edu.hk

Jun Wang
Chinese University of Hong Kong
Department of Automation and Computer-Aided Engineering
Shatin, New Territories, Hong Kong
E-mail: jwang@acae.cuhk.edu.hk

DeLiang Wang
Ohio State University
Department of Computer Science and Engineering
Columbus, Ohio, USA
E-mail: dwang@cse.ohio-state.edu

Library of Congress Control Number: 2006933758

CR Subject Classification (1998): F.1, I.2, I.5, I.4, G.3, J.3, C.2.1, C.1.3, C.3

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

ISSN 0302-9743
ISBN-10 3-540-46484-0 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-46484-6 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11893295 06/3142 5 4 3 2 1 0

Preface

This book and its companion volumes constitute the Proceedings of the 13th International Conference on Neural Information Processing (ICONIP 2006) held in Hong Kong during October 3–6, 2006. ICONIP is the annual flagship conference of the Asia Pacific Neural Network Assembly (APNNA) with the past events held in Seoul (1994), Beijing (1995), Hong Kong (1996), Dunedin (1997), Kitakyushu (1998), Perth (1999), Taejon (2000), Shanghai (2001), Singapore (2002), Istanbul (2003), Calcutta (2004), and Taipei (2005). Over the years, ICONIP has matured into a well-established series of international conference on neural information processing and related fields in the Asia and Pacific regions. Following the tradition, ICONIP 2006 provided an academic forum for the participants to disseminate their new research findings and discuss emerging areas of research. It also created a stimulating environment for the participants to interact and exchange information on future challenges and opportunities of neural network research.

ICONIP 2006 received 1,175 submissions from about 2,000 authors in 42 countries and regions (Argentina, Australia, Austria, Bangladesh, Belgium, Brazil, Canada, China, Hong Kong, Macao, Taiwan, Colombia, Costa Rica, Croatia, Egypt, Finland, France, Germany, Greece, India, Iran, Ireland, Israel, Italy, Japan, South Korea, Malaysia, Mexico, New Zealand, Poland, Portugal, Qatar, Romania, Russian Federation, Singapore, South Africa, Spain, Sweden, Thailand, Turkey, UK, and USA) across six continents (Asia, Europe, North America, South America, Africa, and Oceania). Based on rigorous reviews by the Program Committee members and reviewers, 386 high-quality papers were selected for publication in the proceedings with the acceptance rate being less than 33%. The papers are organized in 22 cohesive sections covering all major topics of neural network research and development. In addition to the contributed papers, the ICONIP 2006 technical program included two plenary speeches by Shun-ichi Amari and Russell Eberhart. In addition, the ICONIP 2006 program included invited talks by the leaders of technical co-sponsors such as Wlodzislaw Duch (President of the European Neural Network Society), Vincenzo Piuri (President of the IEEE Computational Intelligence Society), and Shiro Usui (President of the Japanese Neural Network Society), DeLiang Wang (President of the International Neural Network Society), and Shoujue Wang (President of the China Neural Networks Council). In addition, ICONIP 2006 launched the APNNA Presidential Lecture Series with invited talks by past APNNA Presidents and the K.C. Wong Distinguished Lecture Series with invited talks by eminent Chinese scholars. Furthermore, the program also included six excellent tutorials, open to all conference delegates to attend, by Amir Atiya, Russell Eberhart, Mahesan Niranjan, Alex Smola, Koji Tsuda, and Xuegong Zhang. Besides the regular sessions, ICONIP 2006 also featured ten special sessions focusing on some emerging topics.

ICONIP 2006 would not have achieved its success without the generous contributions of many volunteers and organizations. ICONIP 2006 organizers would like to express sincere thanks to APNNA for the sponsorship, to the China Neural Networks Council, European Neural Network Society, IEEE Computational Intelligence Society, IEEE Hong Kong Section, International Neural Network Society, and Japanese Neural Network Society for their technical co-sponsorship, to the Chinese University of Hong Kong for its financial and logistic supports, and to the K.C. Wong Education Foundation of Hong Kong for its financial support. The organizers would also like to thank the members of the Advisory Committee for their guidance, the members of the International Program Committee and additional reviewers for reviewing the papers, and members of the Publications Committee for checking the accepted papers in a short period of time. Particularly, the organizers would like to thank the proceedings publisher, Springer, for publishing the proceedings in the prestigious series of *Lecture Notes in Computer Science*. Special mention must be made of a group of dedicated students and associates, Haixuan Yang, Zhenjiang Lin, Zenglin Xu, Xiang Peng, Po Shan Cheng, and Terence Wong, who worked tirelessly and relentlessly behind the scene to make the mission possible. There are still many more colleagues, associates, friends, and supporters who helped us in immeasurable ways; we express our sincere thanks to them all. Last but not the least, the organizers would like to thank all the speakers and authors for their active participation at ICONIP 2006, which made it a great success.

October 2006

Irwin King
Jun Wang
Laiwan Chan
DeLiang Wang

Organization

General Chair and Co-chair

Jun Wang, Hong Kong
Laiwan Chan, Hong Kong

Organizing Chair

Man-Wai Mak, Hong Kong

Finance and Registration Chair

Kai-Pui Lam, Hong Kong

Workshops and Tutorials Chair

James Kwok, Hong Kong

Publications and Special Sessions Chair and Co-chair

Frank H. Leung, Hong Kong
Jianwei Zhang, Germany

Publicity Chair and Co-chairs

Jeffrey Xu Yu, Hong Kong
Derong Liu, USA
Chris C. Yang, Hong Kong
Wlodzislaw Duch, Poland

Local Arrangements Chair and Co-chair

Andrew Chi-Sing Leung, Hong Kong
Eric Yu, Hong Kong

Secretary

Haixuan Yang, Hong Kong

Program Chair and Co-chair

Irwin King, Hong Kong
DeLiang Wang, USA

Program Committee

Shigeo Abe, Japan
Peter Andras, UK
Sabri Arik, Turkey
Abdesselam Bouzerdoum, Australia
Ke Chen, UK
Liang Chen, Canada
Luonan Chen, Japan
Zheru Chi, Hong Kong
Sung-Bae Cho, Korea
Sungzoon Cho, Korea
Seungjin Choi, Korea
Andrzej Cichocki, Japan
Chuangyin Dang, Hong Kong
Wai-Keung Fung, Canada
Takeshi Furuhashi, Japan
Artur dAvila Garcez, UK
Daniel W.C. Ho, Hong Kong
Edward Ho, Hong Kong
Sanqing Hu, USA
Guang-Bin Huang, Singapore
Kaizhu Huang, China
Malik Magdon Ismail, USA
Takashi Kanamaru, Japan
James Kwok, Hong Kong
James Lam, Hong Kong
Kai-Pui Lam, Hong Kong
Doheon Lee, Korea
Minho Lee, Korea
Andrew Leung, Hong Kong
Frank Leung, Hong Kong
Yangmin Li, Macau
Xun Liang, China
Yanchun Liang, China
Xiaofeng Liao, China
Chih-Jen Lin, Taiwan
Xiuwen Liu, USA
Bao-Liang Lu, China
Wenlian Lu, China
Jinwen Ma, China
Man-Wai Mak, Hong Kong
Sushmita Mitra, India
Paul Pang, New Zealand
Jagath C. Rajapakse, Singapore
Bertram Shi, Hong Kong
Daming Shi, Singapore
Michael Small, Hong Kong
Michael Stiber, USA
Ponnuthurai N. Suganthan, Singapore
Fuchun Sun, China
Ron Sun, USA
Johan A.K. Suykens, Belgium
Norikazu Takahashi, Japan
Michel Verleysen, Belgium
Si Wu, UK
Chris Yang, Hong Kong
Hujun Yin, UK
Eric Yu, Hong Kong
Jeffrey Yu, Hong Kong
Gerson Zaverucha, Brazil
Byoung-Tak Zhang, Korea
Liqing Zhang, China

Reviewers

Shotaro Akaho
Toshio Akimitsu
Damminda Alahakoon
Aimee Betker
Charles Brown
Gavin Brown
Jianting Cao
Jinde Cao
Hyi-Taek Ceong
Pat Chan
Samuel Chan
Aiyou Chen
Hongjun Chen
Lihui Chen
Shu-Heng Chen
Xue-Wen Chen
Chong-Ho Choi
Jin-Young Choi
M.H. Chu
Sven Crone
Bruce Curry
Rohit Dhawan
Deniz Erdogmus
Ken Ferens
Robert Fildes
Tetsuo Furukawa
John Q. Gan

Kosuke Hamaguchi
Yangbo He
Steven Hoi
Pingkui Hou
Zeng-Guang Hou
Justin Huang
Ya-Chi Huang
Kunhuang Huarng
Arthur Hsu
Kazushi Ikeda
Masumi Ishikawa
Jaeseung Jeong
Liu Ju
Christian Jutten
Mahmoud Kaboudan
Sotaro Kawata
Dae-Won Kim
Dong-Hwa Kim
Cleve Ku
Shuichi Kurogi
Cherry Lam
Stanley Lam
Toby Lam
Hyoung-Joo Lee
Raymond Lee
Yuh-Jye Lee
Chi-Hong Leung
Bresley Lim
Heui-Seok Lim
Hsuan-Tien Lin
Wei Lin
Wilfred Lin
Rujie Liu
Xiuxin Liu
Xiwei Liu
Zhi-Yong Liu
Hongtao Lu
Xuerong Mao
Naoki Masuda
Yicong Meng
Zhiqing Meng
Yutaka Nakamura
Nicolas Navet
Raymond Ng
Rock Ng
Edith Ngai
Minh-Nhut Nguyen
Kyosuke Nishida
Yugang Niu
YewSoon Ong
Neyir Ozcan
Keeneth Pao
Ju H. Park
Mario Pavone
Renzo Perfetti
Dinh-Tuan Pham
Tu-Minh Phuong
Libin Rong
Akihiro Sato
Xizhong Shen
Jinhua Sheng
Qiang Sheng
Xizhi Shi
Noritaka Shigei
Hyunjung Shin
Vimal Singh
Vladimir Spinko
Robert Stahlbock
Hiromichi Suetant
Jun Sun
Yanfeng Sun
Takashi Takenouchi
Yin Tang
Thomas Trappenberg
Chueh-Yung Tsao
Satoki Uchiyama
Feng Wan
Dan Wang
Rubin Wang
Ruiqi Wang
Yong Wang
Hua Wen
Michael K.Y. Wong
Chunguo Wu
Guoding Wu
Qingxiang Wu
Wei Wu
Cheng Xiang
Botong Xu
Xu Xu
Lin Yan
Shaoze Yan
Simon X. Yang
Michael Yiu
Junichiro Yoshimoto
Enzhe Yu
Fenghua Yuan
Huaguang Zhang
Jianyu Zhang
Kun Zhang
Liqing Zhang
Peter G. Zhang
Ya Zhang
Ding-Xuan Zhou
Jian Zhou
Jin Zhou
Jianke Zhu

Table of Contents – Part III

DRFE: Dynamic Recursive Feature Elimination for Gene Identification Based on Random Forest

Ha-Nam Nguyen and Syng-Yup Ohn

Department of Computer Engineering
Hankuk Aviation University, Seoul, Korea
{nghanam, syohn}@hau.ac.kr

Abstract. Determining the relevant features is a combinatorial task in various fields of machine learning such as text mining, bioinformatics, pattern recognition, etc. Several scholars have developed various methods to extract the relevant features but no method is really superior. Breiman proposed Random Forest to classify a pattern based on CART tree algorithm and his method turns out good results compared to other classifiers. Taking advantages of Random Forest and using wrapper approach which was first introduced by Kohavi *et. al*, we propose an algorithm named Dynamic Recursive Feature Elimination (DRFE) to find the optimal subset of features for reducing noise of the data and increasing the performance of classifiers. In our method, we use Random Forest as induced classifier and develop our own defined feature elimination function by adding extra terms to the feature scoring. We conducted experiments with two public datasets: Colon cancer and Leukemia cancer. The experimental results of the real world data showed that the proposed method has higher prediction rate compared to the baseline algorithm. The obtained results are comparable and sometimes have better performance than the widely used classification methods in the same literature of feature selection.

1 Introduction

Machine learning techniques have been widely used in various fields such as text mining, network security and especially in bioinformatics. There are wide ranges of learning algorithms that have been studied and developed, i.e. Decision Trees, K Nearest-Neighbor, Support Vector Machine, etc. These existing learning algorithms do well in most cases. However, as the number of features in a dataset is large, the performance of these algorithms is degraded. In that case, the whole set of features of a dataset usually over-describes the data relationships. Thus, an important issue is how to select a relevant subset of features based on their criteria. A good feature selection method should heighten the success probability of the learning methods [1, 2]. In other words, this mechanism helps to eliminate noises or non-representative features which can impede the recognition process.

Recently, Random Forest (RF) was proposed based on an ensemble of CART tree classifications [3]. This method turns out better results compared to other classifiers including Adaboost, Support Vector Machine and Neural Network. Researchers applied RF as a feature selection method [4, 5]. Some tried RF directly [4] and others adapted it for relevance feedback [5]. The approach presented in [5] attempts to address this problem with correlation techniques. In this paper, we introduce a new method of feature selection based on Recursive Feature Elimination. The proposed method reduces the set of features via feature ranking criterion. This criterion re-evaluates the importance of features according to the Gini index [6, 7] and the correlation of training and validation accuracy which are obtained from RF algorithm. By that way, we take both feature contribution and correlation of training error into account. We applied the proposed algorithm to classify several datasets such as Colon cancer and Leukemia cancer. The DRFE showed better classification accuracy than RF and sometimes it showed better results compared to other studies.

The rest of this paper is organized as follows. In section 2 we describe feature selection approaches. In Section 3 we briefly review RF and its characteristics that will be used in proposed method. The framework of proposed method is presented in Section 4. Details of the new feature elimination method will be introduced in Section 5. Section 6 explains the experimental design of proposed method and the analysis of obtained results. Some concluding remarks are given in Section 7.

2 Feature Selection Problem

In this section, we briefly summarize the space dimension reduction and feature selection methodologies. Feature selection approach has been shown as a very effective way in removing redundant and irrelevant features, so that it increases the efficiency of the learning task and improves learning performance such as learning time, convergence rate, accuracy, etc. A lot of studies have focused on feature selection literature [1, 2, 8-11]. As mentioned in [1, 2], there are two ways to determine the starting point in a searching space. The first strategy might start with nothing and successively adds relevance features called *forward selection*. Another one, named *backward elimination*, starts with all features and successively removes irrelevance ones.

There are two different approaches used for feature selection, i.e. Filter approach and Wrapper approach [1, 2]. The Filter approach considers the feature selection process as precursor stage of learning algorithms. The most disadvantage of this approach is that there is no relationship between the feature selection process and the performance of learning algorithms. The second approach focuses on a specific machine learning algorithm. It evaluates the selected feature subset based on the goodness of learning algorithms such as the accuracy, recall and precision values. The disadvantage of this approach is high computation cost. Some researchers tried to propose methods that can speed up the evaluating process to decrease this cost. Some studies used both filter and wrapper approaches in their algorithms called hybrid approaches [9, 10, 12-14]. In these methods, the feature criteria or randomly selection methods are used to choose the candidate feature subsets. The cross validation mechanism is employed to decide the final best subset among the whole candidate subsets [6].

3 Random Forest

Random Forest is a special kind of ensemble learning techniques [3]. It builds an ensemble of CART tree classifications using bagging mechanism [6]. By using bagging, each node of trees only selects a small subset of features for the split, which enables the algorithm to create classifiers for high dimensional data very quickly. One have to specify the number of randomly selected features (*mtry*) at each split. The default value is *sqrt*(*p*) for classification where *p* is number of features. The Gini index [6, 7] is used as the splitting criterion. The largest possible tree is grown and not pruned. One should choose the big enough number of trees (*ntree*) to ensure that every input feature is predicted at least several times. The root node of each tree in the forest keeps a bootstrap sample from the original data as the training set. The out-of-bag (OOB) estimates are based on roughly one third of the original data set. By contrasting these OOB predictions with the training set outcomes, one can arrive at an estimation of the predicting error rate, which is referred to as the OOB estimate of error rate.

To represent what is the out-of-bag (OOB) estimate method, we assume a method for building a classifier from training set. We can construct classifiers $H(x, T_k)$ based on bootstrap training set T_k from given training set T. The out-of-bag classifier of each sample (x, y) in training set is defined as the aggregate of the vote only over those classifiers for which T_k does not contain that sample. Thus the out-of-bag estimation of the generalization error is the error rate of the out-of-bag classifier on the training set.

The Gini index is defined as squared probabilities of membership for each target category in the node.

$$gini(N) = \frac{1}{2}\left(1 - \sum_j p(\omega_j)^2\right) \qquad (1)$$

where $p(\omega_j)$ is the relative frequency of class ω_j at node *N*. It means if all the samples are on the same category, the impurity is zero; otherwise it is positive value. Some algorithm such as CART [6], SLIQ [18], and RF [3, 7] were used Gini index as splitting criterion. It tries to minimize the *impurity* of the nodes resulting from split. In Random forest, the Gini decrease for each individual variable over all trees in the forest gives a fast variable important that is often very consistent with the permutation importance measure [3, 7].

4 Proposed Approach

The proposed method used *Random Forest module* to estimate the performance consisting of the cross validation accuracy and the importance of each feature in training data set. Even though RF robust against over-fitting problem itself [3, 6], our approach can not inherit this characteristic. To deal with the over-fitting problem, we use n-fold cross validation technique to minimize generalization error [6].

The *Feature Evaluation* module computes the feature importance ranking values according to the obtained results from *Random Forest module* (see Equation 2). The irrelevant feature(s) are eliminated and only important features are survived by mean of feature ranking value. The survival features are again used as input data of *Random Forest module*. This process is repeatedly executed until it satisfies the desired criteria.

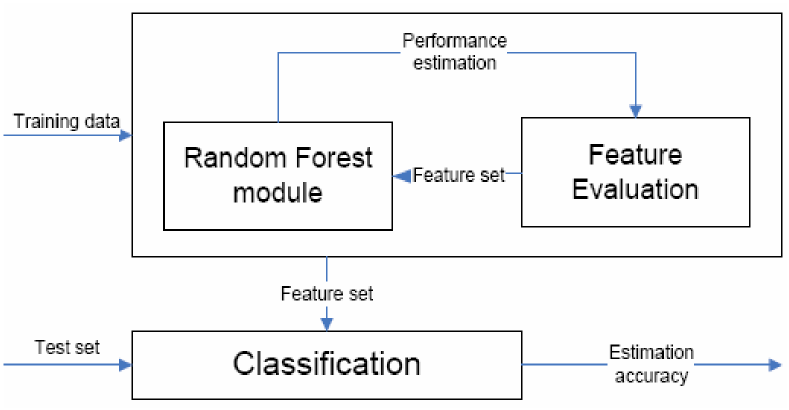


Fig. 1. The main procedures of our approach

The set of features, which is a result of learning phase, is used as a filter of test dataset in classification phase. The detail of proposed algorithm will be presented in next section. The overall procedure of our approach is shown in Fig. 1.

5 Dynamic Recursive Feature Elimination Algorithm

When computing the ranking criteria in wrapper approaches, they usually concentrate much on the accuracies of the features, but not much on the correlation of the features. A feature with good ranking criteria may not create a good result. Also the combination of several features with good ranking criteria, may not give out a good result. To remedy the problem, we propose a procedure named Dynamic Recursive Feature Elimination (DRFE).

1. Train data by Random Forest with the cross validation
2. Calculate the ranking criterion for all features F_i^{rank} where i=1..n (n is the number of features).
3. Remove feature by using *DynamicFeatureElimination* function (for computational reasons, it may be more efficient if we remove several features at a time)
4. Back to step 1 until reach the desired criteria.

In step 1, we use Random Forest with n-folders cross vadilation to train the classifier. In the j^{th} cross validation, we will obtain a turtle (F_j, A_j^{learn}, $A_j^{validation}$) that are the feature importance, the learning accuracy and the validation accuracy, respectively. We will use those values to compute the ranking criterion in step 2.

The cores of our algorithm are presented in step 2. In this step, we use the results from step 1 to build ranking criterions which will be used in step 3. The ranking criterion of feature i^{th} is computed as follow

$$F_i^{rank} = \sum_{j=1}^{n} F_{i,j} \times \frac{(A_j^{learn} + A_j^{validation})}{|A_j^{learn} - A_j^{validation}| + \varepsilon} \tag{2}$$

where j=1,.., n is the number of cross validation folders, $F_{i,j}$, A_j^{learn} and $A_j^{validation}$ are the feature importance in terms of the node impurity which can be computed by Gini inpurity, the learning accuracy and the validation accuracy of feature j-th obtained from *RandomForest* module, respectively. ε is the real number with very small value. The first factor ($F_{i,j}$) is presented the Gini decrease for each feature over all trees in the forest when we train data by RF. Obviously, the higher decrease of $F_{i,j}$ is obtained, the better rank of feature we have [3, 6] . We use the second factor to deal with the overfitting issue [6] as well as the desire of high accuracy. The numerator of the factor presents for our desire to have high accuracy. The more this value we get, the better the rank of the feature is. We want to have a high accuracy in learning and also want not too fit the training data which called overfitting problem. To solve this issue, we applied the n-folder cross validation technique [6]. We can see that the less difference between the learning accuracy and the validation accuracy is, the result is the more stability of accuracy. In the other words, the target of denominator is to reduce overfitting problem. In the case that the learning accuracy is equal to the validation accuracy, the difference is equal to 0, we use ε with very small value to avoid the fraction coming to ∞. We want to choose the feature with both high stability and high accuracy. To deal with this problem, the procedure choose a feature subset only if the validation of this selected feature subset is higher than the validation of the previous selected feature set. This heuristic method ensures that the feature set we chose always have better accuracy. As a result of step 2, we have an ordered-list of ranking criterion of features.

In step 3, we propose our feature elimination strategy based on backward approach. The proposed feature elimination strategy depends on both ranking criterion and validation accuracy. The ranking criterion makes the order of features be eliminated and the validation accuracy is used to decide whether the chosen subset of features is permanently eliminated. In normal case, our method eliminates features having the smallest value of ranking criterion. The new subset is validated by *RandomForest* module. The obtained validation accuracy plays a role of decision making. It is used to evaluate whether the selected subset is accepted as new candidate of features. If the obtained validation accuracy is lower than the previous selected subset accuracy, it tries to eliminate other features based on their rank values.

This iteration is stopped whenever the validation accuracy of the new subset is higher than the previous selected subset accuracy. If there is no feature to create new subset and no better validation accuracy, the current subset of features is considered as the final result of our learning algorithm. Otherwise the procedure goes back to step 1.

6 Experiments

We tested the proposed algorithm with several datasets including two public datasets (Leukemia and Colon cancer) to validate our approach. In this section, we represent the description of used datasets, our experimental configurations, and some evaluations about the experimental results.

6.1 Datasets

The colon cancer dataset contains gene expression information extracted from DNA microarrays [1]. The dataset consists of 62 samples in which 22 are normal samples and 40 are cancer tissue samples, each has 2000 features. We randomly choose 31 samples for training set and the remaining 31 samples were used as testing set. (Availble at: http://sdmc.lit.org.sg/GEDatasets/Data/ColonTumor.zip).

The leukemia dataset consists of 72 samples divided into two classes ALL and AML [15]. There are 47 ALL and 25 AML samples and each contains 7129 features. This dataset was divided into a training set with 38 samples (27 ALL and 11 AML) and a testing set with 34 samples (20 ALL and 14 AML) (Availble at: http://sdmc.lit.org.sg/GEDatasets/ Data/ALL-AML_Leukemia.zip).

6.2 Experimental Environments

Our proposed algorithm was coded using R language (http://www.r-project.org; R Development Core Team, 2004), and *RandomForest* packages (from A. Liaw and M. Wiener) for random forest module. All experiments are conducted on a Pentium IV 1.8 GHz personal computer. The learning and validation accuracies were determined by means of 4-fold cross validation. The data was randomly split into a training set and a testing set. In this paper, we used RF with the original dataset as the base-line method. The proposed algorithm and the base-line algorithm were executed with the same training and testing datasets to compare the efficiency of the two methods.

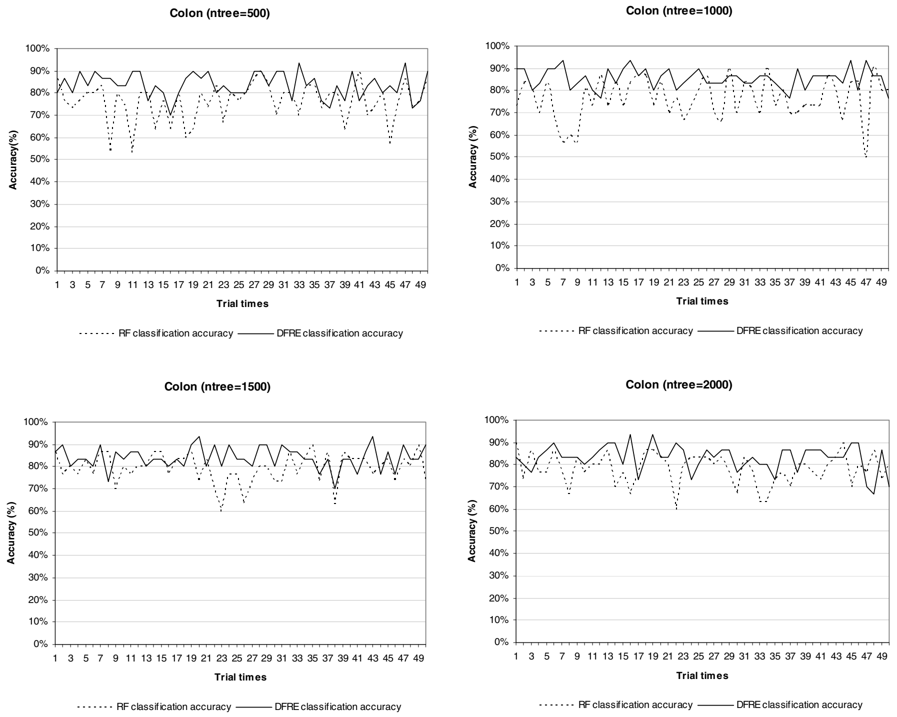


Fig. 2. The comparison of classification accuracy between DRFE (dash line) and RF (dash-dot line) via 50 trials with parameter *ntree* = {500, 1000, 1500, 2000} in case of Colon dataset

6.3 Experimental Results and Analysis

6.3.1 Colon Cancer

The data was randomly divided into a training set of 50 samples and a testing set of 12 for 50 times, and our final results were averaged over these 50 independent trials (Fig. 2). In our experiments, we use the default value for the *ntry* parameter (see Sec. 3) and the *ntree* parameter was tried with different values of 500, 1000, 1500, and 2000.

The summary of classification results are depicted in Table 1. The classification accuracy of the proposed algorithm is significantly better than the baseline one. Table 2 presents the average number of selected features obtained from all experiments. As mentioned above, several features are eliminated each iteration because of speed-up (Sec. 5). The proposed method achieves accuracy of 85.5% when performing on about 141 genes predictors retained after using the DRFE procedure. This number of genes only makes up about 7.1% (141/2000) of the overall genes. The method not only increases the classification accuracy but also reduces the standard deviation values (Table 1).

Table 1. The average classification rate of Colon cancer over 50 trials (average % of classification accuracy ±standard deviation)

Tree number	500	1000	1500	2000
RF only	75.6±8.9	76.0±9	79.3±6.8	78.0±7.1
DRFE	83.5±5.6	**85.5±4.5**	84.0±5.1	83.0±6.0

Some studies have done in terms of feature selection approaches. The comparison of those studies' results and our approach's result are depicted in Table 2. Our method sometimes showed better results compared to the old ones. In addition, the standard deviation values of the proposed method are much lower than both RF (see Table 1) and other methods (Table 2). It shows that the proposed method turned out more stable results than previous ones.

Table 2. The best prediction rate of some studies in case of Colon dataset

Type of classifier	Prediction rate (%)
GA\SVM [9]	84.7±9.1
Bootstrapped GA\SVM [10]	80.0
Combined kernel for SVM [16]	75.33±7.0
DRFE	**85.5±4.5**

6.3.2 Leukemia Cancer

As mentioned in Sec. 6.1, the Leukemia dataset is already divided into training and testing set. To setup the 50 independent trials, we randomly selected 4000 features among 7129 given set of features. In this experiment, the *ntree* parameter was set to 1000, 2000, 3000, and 4000. By applying DRFE, the classification accuracies are significantly improved in all 50 trials (Fig. 3).

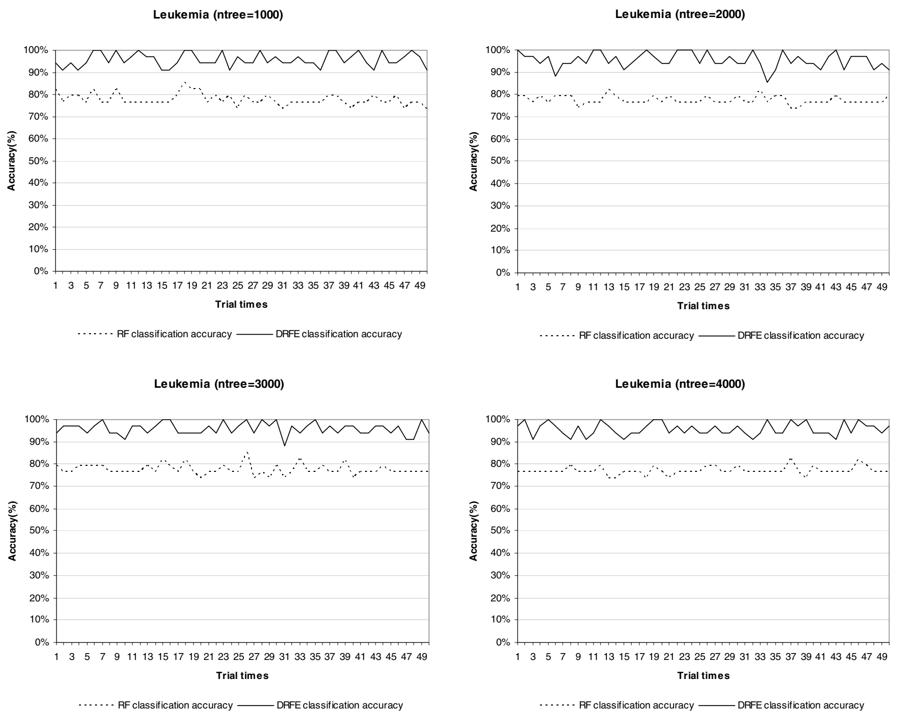


Fig. 3. The comparison of classification accuracy between DRFE (dash line) and RF (dash-dot line) via 50 trials with parameter *ntree* = {1000, 2000, 3000, 4000} in case of Leukemia dataset

The summary of classification results are depicted in Table 3. In those experiments, the tree number parameters do not significantly affect the classified results. We selected 50 as the number of feature elimination which is called *Step* parameter (Step=50). Our proposed method achieved the accuracy of 95.94% when performing on about 55 genes predictors retained by using DRFE procedure. This number of obtained genes only makes up about 0.77% (55/7129) of the whole set of genes.

Table 3. Classification results of leukemia cancer (average % of classification accuracy ±standard deviation)

Tree number	1000	2000	3000	4000
RF only	77.59±2.6	77.41±1.9	77.47±2.5	76.88±1.9
DRFE	95.71±3.1	95.53±3.3	**95.94±2.7**	95.76±2.8

And again, we compare the prediction results of our method and some other studies' results performed on Leukemia dataset (Table 4). The table shows the classification accuracy of our method is much higher than these studies' one.

Table 4. The best prediction rate of some studies in case of Leukemia data set

Type of classifier	Prediction rate (%)
Weighted voting[8]	94.1
Bootstrapped GA\SVM [10]	97.0
Combined kernel for SVM [16]	85.3±3.0
Multi-domain gating network [17]	75.0
DRFE	**95.94±2.7**

7 Conclusions

In this paper, we introduced the novel method in terms of feature selection. The RF algorithm itself is particularly suited for analyzing high-dimensional dataset. It can easily deal with a large number of features as well as a small number of training samples. Our method not only employed RF by mean of conventional REF but also made it fluently adapt to feature elimination task by using the DRFE procedure. Based on the defined ranking criterion and the dynamic feature elimination strategy, the proposed method obtains higher classification accuracy and more stable results than the original RF. The experiments achieved a high recognition accuracy of 85.5%±4.5 when performing on Colon cancer dataset with only a subset of 141 genes and the accuracy of 95.94%±2.7 in case of Leukemia cancer using a subset of 67 genes. The experimental results also showed a significant improvement in the classification accuracy compare to the original RF algorithm especially in case of Leukemia cancer dataset.

Acknowledgement

This research was supported by RIC (Regional Innovation Center) in Hankuk Aviation University. RIC is a Kyounggi-Province Regional Research Center designated by Korea Science and Engineering Foundation and Ministry of Science & Technology.

References

1. Kohavi, R. and John, G.H.: Wrappers for Feature Subset Selection, Artificial Intelligence (1997) pages: 273-324
2. Blum, A. L. and Langley, P.: Selection of Relevant Features and Examples in Machine Learning, Artificial Intelligence, (1997) pages: 245-271
3. Breiman, L.: Random forest, Machine Learning, vol. *45* (2001) pages: 5–32.
4. Torkkola, K., Venkatesan, S., Huan Liu: Sensor selection for maneuver classification, Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (2004) page(s):636 - 641
5. Yimin Wu, Aidong Zhang: Feature selection for classifying high-dimensional numerical data, Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2 (2004) Pages: 251-258

6. Duda, R. O., Hart, P. E., Stork, D. G.: Pattern Classification (2nd Edition), John Wiley & Sons Inc. (2001)
7. Breiman, L., Friedman, J. H., Olshen, R. A., Stone, C. J.: *Classification and Regression Trees*, Chapman and Hall, New York (1984)
8. Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, J. P., Mesirov, J., Coller, H., Loh, M. L., Downing, J.R., Caligiuri, M. A., Bloomfield, C. D., and Lander, E.: Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring," Science, vol. 286 (1999) pages: 531-537.
9. Fröhlich, H., Chapelle, O., and Schölkopf, B.: Feature Selection for Support Vector Machines by Means of Genetic Algorithms, 15th IEEE International Conference on Tools with Artificial Intelligence (2003) pages: 142
10. Chen, Xue-wen: Gene Selection for Cancer Classification Using Bootstrapped Genetic Algorithms and Support Vector Machines, IEEE Computer Society Bioinformatics Conference (2003) pages: 504
11. Zhang, H., Yu, Chang-Yung and Singer, B.: Cell and tumor classification using gene expression data: Construction of forests, Proceeding of the National Academy of Sciences of the United States of America, vol. 100 (2003) pages: 4168-4172
12. Das, S.: Filters, wrappers and a boosting-based hybrid for feature selection, Proceedings of the 18th ICML (2001)
13. Ng, A. Y.: On feature selection: learning with exponentially many irrelevant features as training examples", Proceedings of the Fifteenth International Conference on Machine Learning (1998)
14. Xing, E., Jordan, M., and Carp, R.: Feature selection for highdimensional genomic microarray data", Proc. of the 18th ICML (2001)
15. Alon, U., Barkai, N., Notterman, D., Gish, K., Ybarra, S., Mack, D., and Levine, A.: Broad Patterns of Gene Expression Revealed by Clustering Analysis of Tumor and Normal Colon Tissues Probed by Oligonucleotide Arrays, Proceedings of National Academy of Sciences of the United States of American, vol 96 (1999) pages: 6745-6750.
16. Nguyen, H.-N, Ohn, S.-Y, Park, J., and Park, K.-S.: Combined Kernel Function Approach in SVM for Diagnosis of Cancer, Proceedings of the First International Conference on Natural Computation (2005)
17. Su, T., Basu, M., Toure, A.: Multi-Domain Gating Network for Classification of Cancer Cells using Gene Expression Data, Proceedings of the International Joint Conference on Neural Networks (2002) pages: 286-289
18. Mehta M., Agrawal R., Rissanen J.: SLIQ: A Fast Scalable Classifier for Data Mining, Proceeding of the International Conference on Extending Database Technology (1996) pages: 18-32

Gene Feature Extraction Using T-Test Statistics and Kernel Partial Least Squares

Shutao Li[1], Chen Liao[1], and James T. Kwok[2]

[1] College of Electrical and Information Engineering
Hunan University
Changsha 410082, China
[2] Department of Computer Science
Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong
shutao_li@yahoo.com.cn, lc337199@sina.com, jamesk@cs.ust.hk

Abstract. In this paper, we propose a gene extraction method by using two standard feature extraction methods, namely the T-test method and kernel partial least squares (KPLS), in tandem. First, a preprocessing step based on the T-test method is used to filter irrelevant and noisy genes. KPLS is then used to extract features with high information content. Finally, the extracted features are fed into a classifier. Experiments are performed on three benchmark datasets: breast cancer, ALL/AML leukemia and colon cancer. While using either the T-test method or KPLS does not yield satisfactory results, experimental results demonstrate that using these two together can significantly boost classification accuracy, and this simple combination can obtain state-of-the-art performance on all three datasets.

1 Introduction

Gene expression studies by DNA microarrays provide unprecedented chances because researchers can measure the expression level of tens of thousands of genes simultaneously. Using this microarray technology, a comprehensive understanding of exactly which genes are being expressed in a specific tissue under various conditions can now be obtained [3].

However, since the gene dataset usually includes only a few samples but with thousands or even tens of thousands of genes, such a limited availability of high-dimensional samples is particularly problematic for training most classifiers. As such, oftentimes, dimensionality reduction has to be employed. Ideally, a good dimensionality reduction method should eliminate genes that are irrelevant, redundant, or noisy for classification, while at the same time retain all the highly discriminative genes [11].

In general, there are three approaches to gene (feature) extraction, namely, the filter, wrapper and embedded approaches. In the filter approach, genes are selected according to the intrinsic characteristics. It works as a preprocessing step without the incorporation of any learning algorithm. Examples include the nearest shrunken centroid method, TNoM-score based method and the T-statistics method [8]. In the wrapper approach, a learning algorithm is used to score the feature subsets based on the resultant predictive power, and an optimal feature subset is searched for a specific classifier [4]. Examples include recursive feature elimination, and genetic algorithm-based algorithms.

In this paper, we propose a new gene extraction method based on the filter approach. First, genes are preprocessed by the T-test method to filter irrelevant and noisy genes. Then, kernel partial least squares (KPLS) is used to extract features with high information content and discriminative power. The rest of this paper is organized as follows. In Section 2, we first review the T-test method and KPLS. The new gene extraction method is presented in Section 3. Section 4 then presents the experimental results, which is followed by some concluding remarks.

2 Review

In the following, we suppose that a microarray dataset containing n samples is given, with each sample x represented by the expression levels of m genes.

2.1 T-Test Method

The method is based on the t-statistics [7]. Denote the two classes as positive ($+$) class and negative ($-$) class. For each feature x_j, we compute the mean μ_j^+ (respectively μ_j^-) and standard deviation δ_j^+ (respectively δ_j^-) for the $+$ class (respectively, $-$ class) samples. Then a score $T(x_j)$ can be obtained as:

$$T(x_j) = \frac{|\mu_j^+ - \mu_j^-|}{\sqrt{\frac{(\delta_j^+)^2}{n_+} + \frac{(\delta_j^-)^2}{n_-}}},$$

where n_+ and n_- are the numbers of samples in the positive and negative classes respectively.

2.2 Kernel Partial Least Squares (KPLS)

Given a set of input samples $\{x_i\}_{i=1}^n$ (where each $x_i \in \mathbb{R}^m$) and the corresponding set of outputs $\{y_i\}_{i=1}^n$ (where $y_i \in \mathbb{R}$). Here, only one-dimensional output is needed because only two-class classification is considered. With the use of a kernel, a nonlinear transformation of the input samples $\{x_i\}_{i=1}^n$ from the original input space into a feature space $\mathcal{F}$ is obtained, i.e. mapping $\phi : x_i \in \mathbb{R}^m \rightarrow \phi(x_i) \in \mathcal{F}$. The aim of KPLS is then to construct a linear PLS model in this kernel-induced feature space $\mathcal{F}$. Effectively, a nonlinear kernel PLS in the original input space is obtained and the mutual orthogonality of the score vectors can be retained.

Let Φ be the $n \times m'$ matrix of input samples in the feature space $\mathcal{F}$, and its ith row be the vector $\phi(x_i)^T$. Let m' be the dimensionality of $\phi(x_i)$, which can be infinite. Denote ϕ' the $n \times m'$ deflated dataset and Y' the $n \times 1$ deflated class label. Then the rule of deflation is

$$\begin{aligned} \phi' &= \phi - t(t^T\phi), \\ Y' &= Y - t(t^T Y). \end{aligned} \tag{1}$$

Here, t is the score vector (component) which is obtained in the following way. Let w and c be the weight vectors. The process starts with random initialization of the Y-score u and then iterates the following steps until convergence:

```
1: $w = X^T u/(u^T u)$;
2: $\|w\| \to 1$;
3: $t = Xw$;
4: $c = Y^T t/t^T t$;
5: $u = Yc/(c^T c)$;
6: Repeat steps 1.-5.
```

The process is iterated for Fac times. Subsequently, the deflated dataset can be obtained from the original dataset and the PLS component, while the deflated class label be obtained from the original class labels and the PLS component.

Denote the sequence of t's and u's obtained $n \times 1$ vectors $t_1, t_2, \ldots, t_{Fac}$ and $u_1, u_2, \ldots, u_{Fac}$, respectively. Moreover, let $T = [t_1, t_2, \ldots, t_{Fac}]$ and $U = [u_1, u_2, \ldots, u_{Fac}]$. The "kernel trick" can then be utilized instead of explicitly mapping the input data, and results in: $K = \Phi\Phi^T$, where K stands for the $n \times n$ kernel matrix: $K(i,j) = k(x_i, x_j)$, where k is the kernel function. K can now be directly used in the deflation instead of ϕ, as

$$K' = (I_n - tt^T)K(I_n - tt^T). \tag{2}$$

Here, K' is the deflated kernel matrix and I_n is n-dimensional identity matrix. Now Eq.(2) takes the place of Eq.(1). So deflated kernel matrix is obtained by the original kernel matrix and the PLS component. In kernel PLS, the assumption that the variables of X have zero mean in linear PLS should also hold. The procedure must be applied to centralize the mapped data in the feature space $\mathcal{F}$ as:

$$K = (I_n - \frac{1}{n}1_n 1_n^T)K(I_n - \frac{1}{n}1_n 1_n^T).$$

Here, 1_n is the $n \times 1$ vector with all elements equal to one. Given a set of test samples $\{z_i\}_{i=1}^n$ (where $z_i \in \mathbb{R}^m$), its projection into the feature space is

$$D_p = K_t U(T^T K U)^{-1},$$

where $D_p = [d_1, d_2, \ldots, d_{n_i}]^T$ is a $n_t \times p$ matrix, the columns of D_p are the p KPLS components and the rows of D_p are the n_t test samples in the reduced-dimensional space. K_t is the $n_t \times n$ kernel matrix defined on the test set such that $K_t(i,j) = k(z_i, x_j)$. $T^T K U$ is an upper triangular matrix and thus invertible. The centralized test set kernel Gram matrix K_t can be calculated by [10,9]

$$K_t = (K_t - \frac{1}{n}1_n 1_n^T)K(I_n - \frac{1}{n}1_n 1_n^T).$$

3 Gene Extraction Using T-Test and KPLS

While one can simply use the T-test method or KPLS described in Section 2 for gene extraction, neither of them yields satisfactory performance in practice[1]. In this paper, we propose using the T-test and KPLS in tandem in performing gene extraction. Its key steps are:

1: (Preprocessing using T-test): Since the samples are divided into two classes, one can compute the score for each gene by using the T-statistics. Those genes with scores greater than a predefined threshold T are considered as discriminatory and are selected. On the other hand, those genes whose scores are smaller than T are considered as irrelevant/noisy and are thus eliminated.
2: (Feature extraction using KPLS): The features extracted in the first step are further filtered by using KPLS.
3: (Training and Classification): Using the features extracted, a new training set is formed which is then used to train a classifier. The trained classifier can then be used for predictions on the test set.

A schematic diagram of the whole process is shown in Figure 1.

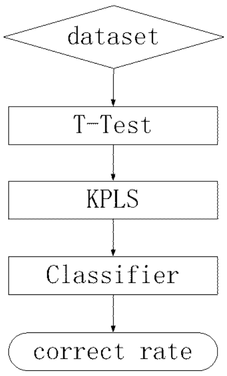


Fig. 1. The combined process of gene extraction and classification

4 Experiments

4.1 Setup

In this section, we evaluate the performance of the proposed gene selection method on three benchmark datasets:

1. Breast cancer dataset: It contains 7,129 genes and 38 samples. 18 of these samples are ER+ (estrogen receptor) while the remaining 20 are ER− [12].

[1] This will be experimentally demonstrated on several benchmark datasets in Section 4.

Table 1. Parameter used in the classifiers

	breast cancer	leukemia	colon cancer
K in K-NN	3	13	14
number of hidden units in NN	3	3	3
soft-margin parameter (C) in SVM	1	10	100

Table 2. Testing accuracies (%) when either the T-test or KPLS is used

		breast cancer	leukemia	colon cancer
T-test only	$T = 1000$	86.8	97.2	82.3
	$T = 2200$	76.3	93.1	79.0
	$T = 2500$	65.8	95.8	82.3
KPLS only	$\gamma = 2$	89.5	93.1	88.7
	$\gamma = 5$	89.5	93.1	85.5

Table 3. Testing accuracy (%) on the breast cancer dataset

T	γ	Fac	K-NN	NN	SVM
500	100	10	94.7	97.4	97.4
		15	94.7	97.4	97.4
		20	76.3	97.4	97.4
	200	10	94.7	100.0	100.0
		15	94.7	100.0	100.0
		20	84.2	100.0	100.0
	300	10	94.7	100.0	100.0
		15	92.1	100.0	100.0
		20	86.8	100.0	100.0
1000	100	10	92.1	100.0	100.0
		15	89.5	100.0	100.0
		20	94.7	100.0	100.0
	200	10	94.7	100.0	100.0
		15	97.4	100.0	100.0
		20	97.4	100.0	100.0
	300	10	92.1	100.0	100.0
		15	92.1	100.0	100.0
		20	97.4	100.0	100.0
1500	100	10	81.6	86.8	86.8
		15	84.2	86.8	84.2
		20	81.6	89.4	86.8
	200	10	81.6	84.2	84.2
		15	81.6	86.8	81.6
		20	84.2	86.8	84.2
	300	10	81.6	81.6	84.2
		15	81.6	81.6	84.2
		20	81.6	84.2	84.2

Table 4. Testing accuracy (%) on the leukemia dataset

T	γ	Fac	K-NN	NN	SVM
2000	100	10	97.2	98.6	98.6
		15	95.8	98.6	98.6
		20	97.2	98.6	98.6
	200	10	97.2	98.6	98.6
		15	98.6	98.6	98.6
		20	95.8	98.6	98.6
	300	10	97.2	98.6	98.6
		15	98.6	98.6	98.6
		20	98.6	98.6	98.6
2500	100	10	94.5	98.6	98.6
		15	95.8	98.6	98.6
		20	91.7	98.6	98.6
	200	10	94.4	100.0	100.0
		15	97.2	100.0	100.0
		20	83.3	100.0	100.0
	300	10	94.4	100.0	100.0
		15	94.4	100.0	100.0
		20	83.3	100.0	100.0
3000	100	10	95.8	100.0	100.0
		15	90.3	100.0	100.0
		20	86.1	100.0	100.0
	200	10	95.8	100.0	100.0
		15	90.3	100.0	100.0
		20	79.2	100.0	100.0
	300	10	95.8	98.6	98.6
		15	90.3	98.6	98.6
		20	76.4	98.6	98.6

2. Leukemia dataset: It contains 7,129 genes and 72 samples. 47 of these samples are of Acute Myeloid Leukemia (AML) and the remaining 25 are of Acute Lymphoblastic Leukemia (ALL) [5].
3. Colon cancer dataset: It contains 2,000 genes and 62 samples. 22 of these samples are of normal colon tissues and the remaining 40 are of tumor tissues [1].

Using the genes selected, the following classifiers are constructed and compared in the experiments:

1. K-nearest neighbor classifier (k-NN).
2. Feedforward neural network (NN) with a single layer of hidden units. Here, we use the logistic function for the hidden units and the linear function for the output units. Back-propagation with adaptive learning rate and momentum is used for training.
3. Support vector machine (SVM). In the experiments, the linear kernel is always used.

Table 5. Testing accuracy (%) on the colon cancer dataset

T	γ	Fac	K-NN	NN	SVM
1700	100	2	87.1	88.7	90.3
		5	88.7	88.7	88.7
		10	87.1	88.7	88.7
	200	2	87.1	88.7	90.3
		5	88.7	88.7	88.7
		10	87.1	88.7	88.7
	300	2	87.1	90.3	90.3
		5	88.7	82.3	90.3
		10	83.9	90.3	90.3
2200	100	2	87.1	91.9	90.3
		5	91.9	87.1	90.3
		10	90.3	91.9	91.9
	200	2	87.1	88.7	90.3
		5	91.9	87.1	90.3
		10	90.3	90.3	90.3
	300	2	87.1	90.3	90.3
		5	91.9	87.1	90.3
		10	87.1	90.3	90.3
2500	100	2	88.7	88.7	88.7
		5	87.1	79.0	87.1
		10	82.3	82.3	85.5
	200	2	88.7	87.1	88.7
		5	87.1	85.5	87.1
		10	80.7	83.9	80.7
	300	2	88.7	90.3	88.7
		5	87.1	83.9	87.1
		10	82.3	83.9	85.5

Each of these classifiers involves some parameters. The parameter settings used on the different datasets are shown in Table 1. Because of the small training set size, leave-one-out (LOO) cross validation is used to obtain the testing accuracy. Both gene selection and classification are put together in each LOO iteration, i.e., they are trained on the training subset and then the performance of the classifier with the selected features is assessed with the left out examples.

4.2 Results

There are three adjustable parameters in the proposed method:

1. The threshold T associated with the T-test method;
2. The width parameter γ in the Gaussian kernel

$$k(x, y) = \exp(-\|x - y\|^2/\gamma),$$

   used in KPLS;
3. The number (Fac) of score vectors used in KPLS.

Table 6. Testing accuracies (%) obtained by the various methods as reported in the literature

classifier	breast cancer	leukemia	colon cancer
Adaboost (decision stumps) [2]	-	95.8	72.6
SVM (quadratic kernel) [2]	-	95.8	74.2
SVM (linear kernel) [2]	97.4	94.4	77.4
RVM (linear kernel) [6]	94.7	94.4	80.6
RVM (no kernel) [6]	89.5	97.2	88.7
logistic regression (no kernel) [6]	-	97.2	71.0
sparse probit regression (quadratic kernel) [6]	-	95.8	84.6
sparse probit regression (linear kernel) [6]	97.4	97.2	91.9
sparse probit regression (no kernel) [6]	84.2	97.2	85.5
JCFO (quadratic kernel) [6]	-	98.6	88.7
JCFO (linear kernel) [6]	97.4	**100.0**	**96.8**
proposed method	**100.0**	**100.0**	91.9

As a baseline, we first study the individual performance of using either the T-test method and KPLS for gene extraction. Here, only the SVM is used as the classifier. As can be seen from Table 2, the accuracy is not high. Moreover, the performance is not stable when different parameter settings are used.

We now study the performance of the proposed method that uses both the T-test method and KPLS in tandem. Testing accuracies, at different parameter settings, on the three datasets are shown in Tables 3, 4 and 5, respectively. As can be seen, the proposed method can reach the best classification performance of 100% on both the breast cancer and leukemia datasets. On the colon cancer dataset, it can also reach 91.9%.

Besides, on comparing the three classifiers used, we can conclude that the neural network can attain the same performance as the SVM. However, its training time is observed to be much longer than that of the SVM. On the other hand, the K-NN classifier does not perform as well in our experiments.

We now compare the performance of the proposed method with those of the other methods as reported in the literature. Note that all these methods are evaluated using leave-one-out cross-validation and so their classification accuracies can be directly compared. As can be seen in Table 6, the proposed method, which attains the best classification accuracy (of 100%) on both the breast cancer and leukemia datasets, outperforms most of the methods. Note that the *Joint Classifier and Feature Optimization* (JCFO) method [6] (using the linear kernel) can also attain 100% on the Leukemia dataset. However, JCFO relies on the Expectation-Maximization (EM) algorithm [6] and is much slower than the proposed method.

5 Conclusions

In this paper, we propose a new gene extraction scheme based on the T-test method and KPLS. Experiments are performed on the breast cancer, leukemia and colon cancer datasets. While the use of either the T-test method or KPLS for gene extraction does not yield satisfactory results, the proposed method, which uses both the T-test method and KPLS in tandem, shows superior classification performance on all three datasets. The proposed gene extraction method thus proves to be a reliable gene extraction method.

Acknowledgment

This paper is supported by the National Nature Science Foundation of China (No. 6040204) and Program for New Century Excellent Talents in University.

References

1. U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, and A.J. Levine. Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Science*, 96:6745–6750, 1999.
2. A. Ben-Dor, L. Bruhn, N. Friedman, I. Nachman, M. Schummer, and Z. Yakhini. Tissue classification with gene expression profiles. In *Proceedings of the Fourth Annual International Conference on Computational Molecular Biology*, pages 54–64, 2000.
3. H. Chai and C. Domeniconi. An evaluation of gene selection methods for multi-class microarray data classification. In *Proceedings of the Second European Workshop on Data Mining and Text Mining for Bioinformatics*, pages 3–10, Pisa, Italy, September 2004.
4. K. Duan and J.C. Rajapakse. A variant of SVM-RFE for gene selection in cancer classification with expression data. In *Proceedings of the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, pages 49–55, 2004.
5. T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, and E.S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.
6. B. Krishnapuram, L. Carin, and A. Hartemink. Gene expression analysis: Joint feature selection and classifier design. In B. Schölkopf, K. Tsuda, and J.-P. Vert, editors, *Kernel Methods in Computational Biology*, pages 299–318. MIT, 2004.
7. H. Liu, J. Li, and L. Wong. A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns. *Genome Informatics*, 13:51–60, 2002.
8. B. Ni and J. Liu. A hybrid filter/wrapper gene selection method for microarray classification. In *Proceedings of International Conference on Machine Learning and Cybernetics*, pages 2537–2542, 2004.
9. R. Rosipal. Kernel partial least squares for nonlinear regression and discrimination. *Neural Network World*, 13(3):291–300, 2003.

10. R. Rosipal, L.J. Trejo, and B. Matthews. Kernel PLS-SVC for linear and nonlinear classification. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 640–647, Washington, D.C., USA, August 2003.
11. Y. Tang, Y.-Q. Zhang, and Z. Huang. FCM-SVM-RFE gene feature selection algorithm for leukemia classification from microarray gene expression data. In *Proceedings of IEEE International Conference on Fuzzy Systems*, pages 97–101, 2005.
12. M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J.A. Olson Jr., J.R. Marks, and J.R. Nevins. Predicting the clinical status of human breast cancer by using gene expression profiles. *Proceedings of the National Academy of Science*, 98(20):11462–11467, 2001.

An Empirical Analysis of Under-Sampling Techniques to Balance a Protein Structural Class Dataset

Marcilio C.P. de Souto[1], Valnaide G. Bittencourt[2], and Jose A.F. Costa[3]

[1] Department of Informatics and Applied Mathematics
[2] Department of Computing and Automation
[3] Department of Electric Engineering
Federal University of Rio Grande do Norte
Campus Universitario
59072-970 Natal-RN, Brazil
Phone: +55-84-3215-3815; Fax: +55-84-3215-3813
marcilio@dimap.ufrn.br, valnaide@dca.ufrn.br,
alfredo@dee.ufrn.br

Abstract. There have been a great deal of research on learning from imbalanced datasets. Among the widely used methods proposed to solve such a problem, the most common are based either on under or over sampling of the original dataset. In this work, we evaluate several methods of under-sampling, such as Tomek Links, with the goal of improving the performance of the classifiers generated by different ML algorithms (decision trees, support vector machines, among others) applied to problem of determining the structural similarity of proteins.

1 Introduction

There are several aspects that can have impact on the performance of classification systems. One of the them regards the non-uniform distribution of instances among the classes. When such a difference in the instance distributions per class is large, we have the problem of imbalanced dataset [1]. In this case, the classifiers generated, via the Machine Learning (ML) algorithms, often have difficulty in classifying the concept associated with minority class (the class with the smallest number of instances).

There have been a great deal of research on this area of learning from imbalanced datasets [2,3,4,5]. Among the widely used methods proposed to solve such a problem, the most common are based either on under or over sampling of the original imbalanced dataset. In this work, we evaluate several methods of under-sampling with the goal of improving the performance of the classifiers generated by different ML algorithms applied to problem of determining the structural similarity of proteins [6,7,8].

More specifically, we address the problem of recognizing structural class in protein folds by using ML algorithms such as: Decision Trees, Naive Bayes Classifiers, k-Nearest Neighbor, Support Vector Machines and Artificial Neural Networks. Proteins are said to have a common fold, which is a common three-dimension (3-D) pattern, if they have the same major secondary structure[1] in the same arrangement and with the

[1] Secondary structure refers to local structural elements such as hairpins, helixes, beta-pleated sheets, among others.

same topology [9]. In our analysis, the focus is on structural predictions in the context of a particular classification of the 3-D folds, by using the taxonomy of the Structure Classification of Protein (SCOP) database [10].

The SCOP database is a hierarchical classification of known protein structures, organized according to their evolutionary and structural relationship. Such a database is divided into four hierarchical levels: class, fold, superfamily, and family. In our work, we will focus on class level, at which the proteins are labeled according to the following structural classes: *all-*α, *all-*β, α/β, α+β, and *small*.

The current datasets, as explained in Section 2.2, formed based on the previous classification are clearly imbalanced. For example, in the dataset presented in [8], the majority class (α/β) contains 203 instances, whereas two classes is composed of, respectively, only 46 and 45 instances (α+β and *small*). In fact, experimental works have shown that a significant number of proteins in class α+β is incorrectly classified as either from class *all-*β or from class α/β [11].

2 Material and Methods

In order to minimize the problems caused by the non-uniform distribution of instances in our data set, we will apply to it - priori to the learning phase - different under-sampling techniques: Random, Tomek Links, One-sided selection (OSS), Neighborhood Cleaning Rule (NCL) and Condensed Nearest Neighbor Rule (CNN). Also, we proposed a modification to the original CNN.

We constrain our analysis to under-sampling techniques for some specific characteristics of our dataset. For example, as it will be explained in Section 2.2, one of the properties of the dataset we analyze is that the similarity degree in terms of sequence of the proteins is set to be less than 40%. Thus, such a requirement makes it inviable the use of over-sampling techniques (e.g., creating new instances by interpolation), for this property could be violated.

2.1 Under-Sampling Techniques

In this work, we evaluate six different methods of under-sampling to balance the class distribution on the training dataset. The methods used are described next.

– **Random.** This is a non-heuristic method that aims to balance class distribution through the random elimination of instances belonging to the majority class.
– **Tomek Links [12].** Tomek links can be defined as follows. Given two instances $\mathbf{x}_i$ and $\mathbf{x}_j$ belonging to different classes, and $d(\mathbf{x}_i,\mathbf{x}_j)$ is the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$. A $(\mathbf{x}_i,\mathbf{x}_j)$ pair is a Tomek link if there is no instance $\mathbf{x}_k$, such that $d(\mathbf{x}_k,\mathbf{x}_i) < d(\mathbf{x}_i,\mathbf{x}_j)$ or $d(\mathbf{x}_k,\mathbf{x}_j) < d(\mathbf{x}_i,\mathbf{x}_j)$. If two instances $(\mathbf{x}_i,\mathbf{x}_j)$ form a Tomek link, then either one is noise or both ones are borderline. Tomek links can be used as an under-sampling method or as a data cleaning method. As an under-sampling method, only instances belonging to the majority class are eliminated, and as a data cleaning method, instances of both classes are removed.
– **Condensed Nearest Neighbor Rule (CNN) [13].** CNN is used to find a consistent subset of instances, not necessarily the smallest one. A subset $T' \subseteq T$ is consistent

with *T* if using a 1-Nearest Neighbor (1-NN), *T'* correctly classifies the instances in T. The rationale is to eliminate the instances from the majority class that are distant from the decision border, for they could be considered less relevant for learning.

An algorithm to create a subset *T'* from *T* as an under-sampling method is the following [5]. First, randomly draw one instance from the majority class and all instances from the minority class. Add these instance to *T'*. Afterwards, build a 1-NN with the instances in *T'*, and classify the instances in *T*. Every misclassified instance from *T* is moved to *T'*.

- **Complement Condensed Nearest Neighbor Rule (CNNc).** We proposed a modification to the original CNN. In our method, CNNc, in contrast to the CNN, every correctly classified instance from *T* is moved to *T'*. The aim to remove instance close to the decision border, which could confuse the learning algorithm.
- **One-sided selection (OSS) [14].** OSS aims at creating a training dataset composed only by "safe instances. In order to do so, this technique removes instances that are noisy, redundant, or near that decision border. As the other under-sampling techniques previously described, OSS removes only instances from the majority class.
- **Neighborhood Cleaning Rule (NCL) [15].** NCL is a modification of the Wilson's Edited Nearest Neighbor Rule (ENN). ENN [16] removes instances whose class label differs from the class given by a 3-NN.

  For a two-class problem the algorithm can be described in the following way. For each instance $\mathbf{x}_i$ in the training set, its three nearest neighbors are found. If $\mathbf{x}_i$ belongs to the majority class and the classification given by the 3-NN contradicts the original class of $\mathbf{x}_i$, then $\mathbf{x}_i$ is removed. If $\mathbf{x}_i$ belongs to the minority class and its 3-NN misclassifies $\mathbf{x}_i$, then the nearest neighbors that belong to the majority class are removed.

2.2 Datasets

We use the dataset[2] in [8]. This dataset is a modification of the one created and used in [7]. Ding and Dubchak's dataset[3] is formed by a training set (Ntrain) and a test set (Ntest). The training set was extracted from the (PDB selects) sets [17] and comprises 313 proteins from 27 most populated SCOP folds (more than seven instances for each fold). For this set, all the pairwise sequence identities are less than 35%. The test set was extracted from the PDB 40D [10]. Such a set contains 386 representatives (excluding the sequences already in the training set) of the same 27 SCOP folds with the pairwise sequence identities less than 35%.

The features used in the learning system were extracted from protein sequences according to the method described in [18]. In that work, they considered several features for predicting protein folds using global description of the chain of amino acids representing proteins. Such descriptors were computed from physical, chemical, and structural properties of the constituent amino acids: hydrophobicity, polarity, polarizability, predicted secondary structure, normalized van der Waals volume and amino acid composition of the protein sequence.

[2] http://www.brc.dcs.gla.ac.uk/~ actan/eKISS/data.htm
[3] http://www.nersc.gov/~cding/protein/

These descriptors essentially represent the frequencies with which such properties change along the sequence and their distribution in the chain. Each of this properties is described by a vector with 21 continuous attributes (with exception of the amino acid composition, for there are only 20 amino acids). In our work, in order to form an input pattern, we use all descriptors with all their features at once and one attribute representing the length of the protein, that is, our input vector has 126 attributes.

It is important to point out that [8] cleaned their data set in relation to previous dataset in [7] by removing errors from both training and testing examples. Also the authors applied the protein fold classification according to SCOP 1.61 [10] and Astral 1.61 [19] with sequence identity less than 40% (November 2002). After performing this stage, the resulting data set contained 582 instances distributed in 25 SCOP folds or, in a higher hierarchical level, four SCOP structural classes (*all-*α, *all-*β, α/β, $\alpha+\beta$) plus a class for those proteins not in any of the former classes (*small*) - Table 1. In our work, we will use this data set with the four SCOP classification (plus the additional class *small*) as the classes to be learned by our ML techniques.

Table 1. Number of instances per class

Structural Class	**#Instances**
all-α	111
all-β	177
α/β	203
$\alpha+\beta$	46
small	45
Total	582

2.3 Evaluation

The comparison of two supervised learning methods is, often, accomplished by analyzing the statistical significance of the difference between the mean of the classification error rate, on independent test sets, of the methods evaluated. In order to evaluate the mean of the error rate, several (distinct) datasets are needed. However, the number of datasets available is often limited. One way to overcome this problem is to divide the data sets into training and test sets by the use of a k-fold cross validation procedure [20,21,22].

This procedure can be used to compare supervised methods, even if only one data set is available. The procedure works as follows. The data set is divided into k disjoint equal size sets. Then, training is performed in k steps, each time using a different fold as the test set and the union of the remaining folds as the training set. Applying the distinct algorithms to the same folds with k at least equal to 10, the statistical significance of the differences between the methods can be measured, based on the mean of the error rate from the test sets [20,21,22]. Besides the mean of the global classification error rate, we evaluate the mean of the classification error rate for the majority and minority classes.

3 Experiments

In our experiments, in order to generate the new training sets, each one of the under-sampling techniques was applied to the different partitions obtained with the 10-fold stratified cross validation (always keeping one of the 10 partitions unmodified - the one to be used as test set). Based on this, for each ML algorithm, we had eight experiments: one with the original dataset [8] (without the use of the under-sampling technique) and seven experiments (one for each under-sampling technique).

In this work, in order to deal with problems with more than two class, we will follow the strategy in [5]: we will refer as majority class the composition of instances of classes *all-*α, *all-*β and α/β, whereas the minority class will be composed by instances of the classes α+β, and *small*. As a computing tool to implementation of the techniques of data pre-processing, we used Matlab 7.0.4 Released 14 and for running the ML algorithms, the software Weka 3.4 [21].

The values for the parameters of the ML algorithms were chosen as follows. For example, for an algorithm with only one parameter, an initial value for such a parameter was chosen followed by the run of the algorithm. Then, experiments with a larger and smaller value were also performed. If with the initially chosen value the classifier obtained had the best results (in terms of validation error), then no more experiments were performed. Otherwise, the same process was repeated for the parameter value with the best result so far.

Using the previous procedure, we arrived to the following values for the parameters (the parameters not mentioned were set to their default values) of the ML algorithms for each of the dataset considered (WEKA implementation):

– Naive Bayes Classifier (**NB**): *KernelEstimator = true*.
– k-Nearest Neighbor (***k*-NN**): *distance Weighting = 1/distance*.
– Decision Tree (**DT**): all parameters were set to their default.
– Artificial Neural Network (multi-layer perceptron with backpropagation) (**NN**): maximum number of iteration = 1000, momentum = 0.9, size of the validation set = 10%, for all the seven datasets; number of hidden neurons for the original, CNN and OSS datasets was set to 30; for the Tomek Links, Random and NCL datasets to 20; and for the case of CNNc dataset to 5. Learning rate for the Random, CNN, NCL and OSS datasets was set to 0.001; for the original and CNNc datasets to 0.1; and for Tomek Links dataset to 0.01.
– Support Vector Machines (**SVM**): c was set to 1 to all the seven datasets. The parameter *expoent* was set to 1 for the original, Random, CNN, CNNc and OSS datasets, and to 2 for the NCL and Tomek Links datasets.

4 Results

Table 2 illustrates the results[4] obtained with original dataset and the application, priori to learning, of the six different under-sampling techniques: Random, *Tomek Links*,

[4] Mean and standard deviation of the classification error rate for each of the five ML algorithms used.

CNN, CNNc, NCL e OSS. For the original dataset, according to the values in this table and hypothesis tests, we can conclude that SVM presented a significantly smaller error rate (17.01%) when compared to each of the other methods. With respect to the other learning algorithms, the null hypothesis was rejected in favor of NN when compared, respectively, to NB, k-NN, and DT. The hypothesis null was also rejected in favor of NB when compared to k-NN and DT. In contrast, there was no evidence to state the difference of performance between k-NN and DT.

Table 2. Mean (standard deviation) of the global classification error rate

	DT	***k*-NN**	**NB**	**SVM**	**NN**
Original Dataset	24.58 ± 5.94	24.74 ± 5.20	21.65 ± 4.24	17.01 ± 3.00	18.32 ± 3.15
Random	35.91 ± 7.37	32.65 ± 6.15	34.39 ± 5.91	26.27 ± 4.29	26.28 ± 4.41
Tomek Links	24.41 ± 4.31	26.17 ± 5.40	22.60 ± 3.95	19.07 ± 4.22	18.73 ± 5.17
CNN	37.47 ± 7.13	34.71 ± 5.49	25.59 ± 7.43	25.43 ± 5.06	23.72 ± 5.21
CNNc	32.61 ± 9.08	36.18 ± 8.88	32.74 ± 6.27	31.57 ± 9.09	30.53 ± 9.56
NCL	26.99 ± 6.28	30.45 ± 4.32	25.05 ± 7.84	21.18 ± 3.33	21.36 ± 5.60
OSS	29.16 ± 5.19	33.67 ± 6.03	25.90 ± 6.57	23.21 ± 5.51	23.05 ± 4.34

Tables 3 and 4 illustrate, respectively, the mean (standard deviation) of the classification error rate for the minority and majority classes. Here, we consider as majority, due to the quantity of instances, the class α/β. Whereas, as minority, we consider the class α+β, despite of it having one more instances that the class *small* - also, as observed in first section of this paper, experimental works have shown that a significant number of proteins in class α+β is incorrectly classified as either from class *all-*β or from class α/β [11].

Table 3. Mean (standard deviation) of the classification error rate for $\alpha + \beta$

	DT	***k*-NN**	**NB**	**SVM**	**NN**
Original Dataset	74.50 ± 25.65	82.50 ± 13.39	76.50 ± 18.86	72.00 ± 24.15	69.00 ± 15.78
Random	50.00 ± 23.33	70.00 ± 24.73	55.00 ± 28.23	48.00 ± 29.77	45.00 ± 26.88
Tomek Links	62.50 ± 13.52	75.00 ± 12.39	67.50 ± 19.76	50.50 ± 16.08	54.50 ± 15.87
CNN	61.50 ± 15.86	76.00 ± 11.25	57.50 ± 16.63	69.50 ± 10.88	62.50 ± 9.52
CNNc	41.50 ± 18.46	63.00 ± 22.63	37.50 ± 15.83	37.00 ± 17.48	20.00 ± 15.29
NCL	44.40 ± 13.33	72.00 ± 15.07	58.50 ± 12.04	43.00 ± 13.63	40.50 ± 12.17
OSS	56.00 ± 13.38	70.50 ± 20.79	55.50 ± 17.47	52.50 ± 19.47	40.50 ± 17.27

According to the previous tables, as well as hypotheses tests, one can observe, for example, that the results obtained with the Random technique were comparable to those achieved with the OSS technique. Despite of not using any heuristic, the Random technique has the merit to remove with the same probability any kind of instances (noise, redundant data and so on).

The application of the Tomek Links under-sampling techniques generated often the dataset with which the ML algorithms obtained the smallest global classification error

Table 4. Mean (standard deviation) of the classification error rate for α/β

	DT	***k*-NN**	**NB**	**SVM**	**NN**
Original Dataset	20.76 ± 9.35	5.93 ± 6.11	8.39 ± 7.33	11.85 ± 5.88	9.40 ± 5.45
Random	35.43 ± 8.24	29.36 ± 5.30	21.24 ± 13.08	36.50 ± 10.66	19.31 ± 10.22
Tomek Links	16.79 ± 8.08	6.43 ± 4.72	11.86 ± 7.44	8.88 ± 6.11	11.88 ± 6.88
CNN	36.50 ± 18.78	24.55 ± 14.42	13.38 ± 10.51	15.26 ± 6.65	20.21 ± 6.43
CNNc	29.17 ± 12.88	32.50 ± 18.69	17.40 ± 16.83	27.02 ± 15.85	25.26 ± 17.46
NCL	20.29 ± 10.57	3.93 ± 3.86	12.21 ± 7.45	9.86 ± 6.00	6.93 ± 4.70
OSS	33.28 ± 9.83	25.50 ± 10.52	14.60 ± 9.45	14.86 ± 7.53	18.31 ± 6.84

rate. Besides, the dataset formed with this technique produced the smallest classification error rate for the majority class α/β, when compared to those obtained with the other under-sampling techniques. However, in terms of the classification error rate for the minority class $\alpha + \beta$, the Tomek Links techniques showed a very poor result.

The NCL technique showed the more consistent results for the criterion analyzed (improvement in the accuracy of the classifiers built). One of the reasons for this could be the fact that such a technique can be more robust to remove noisy instances. In fact, differently from the other under-sampling techniques used, the classifiers built with this NLC dataset, when compared to the original dataset, showed a reduction of both the error in the minority class (about 30%) and in the majority class (around 3%) - the other under-sampling techniques in general caused an increase in incorrect classification rate related majority class when compared to the original data base (about 4% greater).

In contrast to the NCL technique, the dataset generated with the CNNc, when presented to the ML algorithms, achieved the largest global and majority class classification error rates. Nevertheless, the classifiers built using the dataset created with this technique presented the smallest classification error rate for the minority class.

Among all the under-sampling techniques applied to our dataset, the CNN technique presented the worst results. One of the reasons for this is the fact that the CNN technique eliminates instances from the majority class that are distant from the decision border, for they could be considered less relevant for learning. In the case of the dataset analyzed in this paper, even the instances distant from the decision border seem to give important information for the induction of the classifier.

5 Final Remarks

In this work, we evaluated several under-sampling techniques for balancing datasets. In general, the ML algorithms that received as inputs the pre-processed datasets (application of one of the under-sampling techniques) produced classifiers that were more accurate with respect to minority class ($\alpha + \beta$). From a mean of the classification error rate of 75% for the minority class for all classifiers (Table 3 - original dataset), the use of under-sampling techniques reduced this rate in 54.2%. However, the under-sampling techniques led to an increase of the mean of the classification error rate to 19.40% - on average - for the majority class for all classifiers. Likewise, these techniques led to an increase of global classification error rate from an average of 21.24% to 27.73%.

Based on the previous results, one can observe that, in a general way, there is a trade-off between the errors for majority and minority classes. Nevertheless, the decrease in the classification error rate for the class $\alpha + \beta$ (21% on average) is significantly larger than the increase of classification error rate for the class α/β (around 8%). The NCL technique was the one to create the datasets with which the ML methods got the best results. It was able to reduce the classification errors for both majority and minority classes. However it was not able to reduce the global error due to the increase of wrong classification of other classes, mainly of the class *all-*β.

Probably, the decision of not removing any instance of the minority class instances (even noisy ones) could have influenced in the results. However, the small quantities of instances of that class led us to such a decision (as in [23]). Furthermore, the fact that our dataset presents many classes (five) could make harder the learning process. This happens because the system, after de application of pre-processing techniques, could turn different the concept of minority and majority classes - on literature the under-sampling techniques application for problems with two unbalance classes, being rare its application to many classes problems.

References

1. Japkowicz, N.: Learning from imbalanced data sets: A comparison of various strategies. In: Proc. of the AAAI Worrkshop on Learning from Imbalanced Data Sets. (2000) 10–15
2. Fawcett, T., Provost, J.: Adaptive fraud detection. Data Mining and Knowledge Discovery **1** (1997) 291–316
3. Kubat, M., Holte, R., Matwin, S.: Machine learning for the detection of oil spills in satellite radar images. Machine Learning **30** (1998) 195–215
4. Japkowicz, N., Stephen, S.: The class imbalance problem: A systematic study. Intelligent data Analyis **6** (2002) 429–449
5. Batista, G.E.A.P.A., Prati, R.C., Monard, M.C.: A study of the behavior of several methods for balancing machine learning training data. SIGKDD Explorations **6** (2004) 20–29
6. Baldi, P., Brunak, S.: Bioinformatics: the Machine Learning approach. Second edn. MIT Press (2001)
7. Ding, C., Dubchak, I.: Multi-class protein fold recognition using support vector machines and neural networks. Bioinformatics **17** (2001) 349–358
8. Tan, A., Gilbert, D., Deville, Y.: Multi-class protein fold classification using a new ensemble machine learning approach. Genome Informatics **14** (2003) 206–217
9. Craven, M.W., Mural, R.J., Hauser, L.J., Uberbacher, E.C.: Predicting protein folding classes without overly relying on homology. In: Proc. of ISBM. (1995) 98–106
10. Lo Conte, L., Ailey, B., Hubbard, T., Brenner, S., Murzin, A., Chotia, C.: SCOP: a structural classification of proteins database. Nucleic Acids Research **28** (2000) 257–259
11. Chinnasamy, A., Sung, W., Mittal, A.: Protein structure and fold prediction using tree-augmented nave bayesian classifier. In: Proc. of the Pacific Symposium on Biocomputing. Volume 9. (2004) 387–398
12. Tomek, I.: Two modifications of CNN. IEEE Transactions on Systems, Man, and Communications **6** (1976) 769–772
13. Hart, P.: The condensed nearest neighbor rule. IEEE Transactions on Information Theory **14** (1968) 515–516
14. Batista, G.E.A.P.A., Carvalho, A.C.P.L.F., Monard, M.C.: Applying one-sided selection to unbalanced datasets. In: Proc. of the Mexican International Conference on Artificial Intelligence. (2000) 315–325

15. Laurikkala, J.: Improving identification of dificult small classes by balancing class distribution. A-2001-2, University of Tampere (2001)
16. Wilson, D.: Asymptotic properties of nearest neighbor rules using edited data. IEEE Transactions on Systems, Man, and Communications **2** (1972) 408–421
17. Hobohm, U., Sander, C.: Enlarged representative set of proteins. Protein Science **3** (1994) 522–524
18. Dubchak, I., Muchnik, I., Kim, S.: Protein folding class predictor for SCOP: Approach based on global descriptors. In: Proc. of the Intelligent Systems for Molecular Biology. (1997) 104–107
19. Chandonia, J.M., Walker, N., Lo Conte, L., Koehl, P., Levitt, M., Brenner, S.: Astral compendium enhancements. Nucleic Acids Research **30** (2002) 260–263
20. Mitchell, T.: Machine Learning. McGraw Hill (1997)
21. Witten, I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann Publishers (2005)
22. Dietterich, T.: Approximate statistical test for comparing supervised classification learning algorithms. Neural Computation **10** (1998) 1895–1923
23. Batista, G.E.A.P.A.: Pr-processamento de dados em Aprendizado de Mquina Supervisionado. PhD thesis, Universidade de So Paulo, Instituto de Cincias Matemticas e de Computao (2003)

Prediction of Protein Interaction with Neural Network-Based Feature Association Rule Mining

Jae-Hong Eom and Byoung-Tak Zhang

Biointelligence Lab., School of Computer Science and Engineering,
Seoul National University,
Seoul 151-744, South Korea
{jheom, btzhang}@bi.snu.ac.kr

Abstract. Prediction of protein interactions is one of the central problems in post–genomic biology. In this paper, we present an association rule-based protein interaction prediction method. We adopted neural network to cluster protein interaction data, and used information theory based feature selection method to reduce protein feature dimension. After model training, feature association rules are generated to interaction prediction by decoding a set of learned weights of trained neural network and by mining association rules. For model training, an initial network model was constructed with public *Yeast* protein interaction data considering their functional categories, set of features, and interaction partners. The prediction performance was compared with traditional simple association rule mining method. The experimental results show that proposed method has about 96.1% interaction prediction accuracy compared to simple association mining approach which achieved about 91.4% accuracy.

1 Introduction

It is known that protein–protein interactions (PPIs) are fundamental reactions in the organisms and play important roles by determining biological processes. Therefore, comprehensive description and analysis of PPIs would significantly contribute to the understanding of biological phenomena and problems. After the completion of the genome sequence of yeast (*Saccharomyces cerevisiae*), researchers have undertaken the task of functional analysis of the yeast genome comprising more than 6,300 proteins [1], and abundant interaction data have been produced by many research groups. Thus, fresh methods to discover novel knowledge from the interaction data through the analysis of these data are needed.

A variety of attempts have been tried to predict protein functions and interactions with various data such as gene expression, PPI data, and literature analysis. Analysis of gene expression data through clustering also adopted to predict functions of unannotated proteins based on the idea that genes with similar functions are likely to be co-expressed [2, 3]. Park *et al.* [4] analyzed interactions between protein domains in terms of the interactions between structural families of evolutionarily related domains. Iossifov *et al.* [5] and Ng *et al.* [6] inferred new interaction from existing interaction data. Even though there are many other approaches for analyzing and predicting protein interactions, however, many approaches to protein interaction analysis suffered from high dimensional property of data which have thousand of features [7].

In this paper, we propose an adaptive neural network based feature association mining method for PPI prediction. We used additional association rules for interaction prediction those are generated by decoding set of learned weights of neural network. We presumed that association rules decoded from neural network would make the prediction procedure more robust for unexpected error factors by accounting relatively robust characteristic of neural networks (e.g., error factors would be false positive or negative interactions those are provided to the prediction model).

Basically, we use adaptive resonance theory (ART) [8] as an adaptive neural network clustering model to build prediction model. We used ART-1 [9], modified version of ART [10], to cluster binary vectors. The advantage of using ART-1 algorithm for grouping of feature abundant interaction data is that it adapts the changes in new protein interactions without losing key information learned from other interactions trained previously. We assumed 'protein–protein interaction' of yeast as 'feature–to–feature' association of each interacting proteins. To analyze PPIs with respect to their interaction class with their feature association, we use as many features as possible from several major public databases such as (Munich Information Center for Protein Sequences) MIPS and SGD (Saccharomyces Genome Database) [11, 12] to build rich feature vector for each protein interaction. We used the same approach of Rangarajan *et al.* [13] for clustering model design and we also use the same feature selection filter of Yu *et al.* [14] to reduce computational complexity and improve the overall learning performance by eliminating non-informative features.

This paper is organized as follows. In Section 2, we introduce feature selection filter and describe overall architecture of ART-1 based protein interaction clustering model. In Section 3, we present detailed neural network training method with PPI data and the decoding method of association rules extracted from trained network. In Section 4, we present the representation scheme of protein interaction for neural network input, association mining, and experimental results. Finally, concluding remarks and future works are given in Section 5.

2 Feature Dimension Reduction and Protein Cluster Learning

Feature Dimension Reduction by Feature Selection

A set of massive features for each protein and interacting pairs are built by utilizing several public protein databases [11, 12, 15, 16, 17]. Generally, feature selection is necessary when dealing with such high dimensional (feature dimension) data. In our study, set of features having no information of its association with other proteins are removed by applying feature selection. To filter out non-informative features we applied entropy and information gain-based measure, *symmetrical uncertainty* (*SU*-value), as a measure of feature correlation [18]. The procedures of the correlation-based feature dimension reduction filter of Eom *et al.* [7] used for our application.

Enriching Protein Features by Neural Network-Based Cluster Learning

We use ART-1 neural network to group the class of PPIs by their 13 functional classes and the class of interacting counterparts. In our ART-1 based clustering, a protein interaction is represented as a prototype vector that is a generalized representation of a set of features of each interacting proteins. The degree of similarity between the members of each cluster can be controlled by changing the value of the vigilance parameter ρ. We analyzed the cluster formed by using the ART-1 technique by varying the vigilance parameter between the values 0.2 and 0.8. Figure 1 represents the architecture of ART-1 based clustering model and the PPI_i stand for each protein interaction and it includes set of features of two interacting proteins. The overall procedure for clustering protein interactions with the ART-1 based clustering model is described in the Appendix. The basic layout of this procedure is identical with the work of Rangarajan *et al.* [13]. The set of weights of trained neural network were decoded as a form of association rule with the 'weight-to-rule' decoding procedures described in Figure 3 to enrich the protein features.

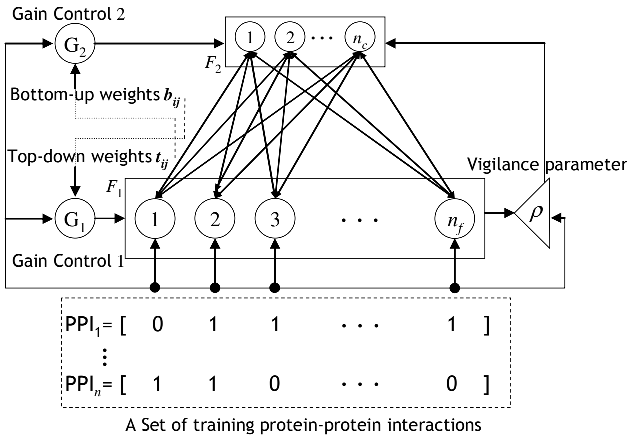


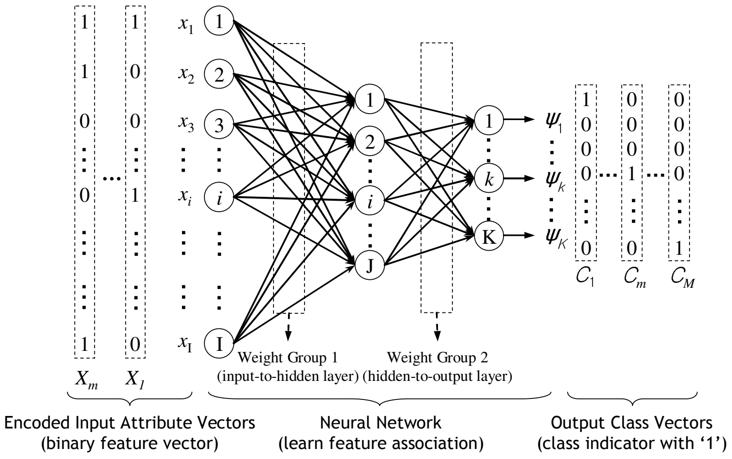
Fig. 1. The schematic architecture of neural network (ART-1) based clustering model (More detailed model constructions are described in [19])

3 Rule Extraction from Trained Neural Network

Learning Feature Associations with Neural Network

A supervised artificial neural network (ANN) uses a set of training examples or records. These records include N attributes. Each attribute, A_n (n = 1, 2, … , N), can be encoded into a fixed length binary substring $\{x_1 \ldots x_i \ldots x_{m(n)}\}$, where $m(n)$ is the number of possible values for an attribute A_n. The element $x_i = 1$ if its corresponding attribute value exists, while all the other elements = 0. Then, the proposed number of input nodes, $\boldsymbol{I}$, in the input layer of ANN can be given by $I = \sum_{n=1}^{N} m(n)$.

The input attributes vectors, $\boldsymbol{X}_m$, to the input layer can be rewritten as $\boldsymbol{X}_m = \{x_1 \ldots x_i \ldots x_I\}_m$, $m = (1,2,\ldots, M)$ where M is the total number of input training patterns. The output class vector, $\boldsymbol{C}_k(k = 1, 2, \ldots, K)$, can be encoded as a bit vector of a fixed length K as follows $C_k\{\psi_1 \ldots \psi_k \ldots \psi_K\}$ where K is the number of different possible classes. If the output vector belongs to class$_k$ then the element ψ_k is equal to 1 while all the other elements in the vector are zeros. Therefore, the proposed number of output nodes in the output layer of ANN is *K*. Accordingly the input and the output nodes of the ANN are determined and the structure of the ANN is shown in Figure 2. The ANN is trained on the encoded vectors of the input attributes and the corresponding vectors of the output classes. The training of ANN is processed until the convergence rate between the actual and the desired output will be achieved. The convergence rate can be improved by changing the number of iterations, the number of hidden nodes (***J***), the learning rate, and the momentum rate.



X_m
X_I
x_1
x_2
x_3
x_i
x_I
Weight Group 1 (input-to-hidden layer)
Weight Group 2 (hidden-to-output layer)
ψ_1
ψ_k
ψ_K
C_1
C_m
C_M
Encoded Input Attribute Vectors (binary feature vector)
Neural Network (learn feature association)
Output Class Vectors (class indicator with '1')

Fig. 2. The structure of the artificial neural network for feature association learning. The two weight groups (WG1 and WG2) are decoded into association rule after network training.

By ANN training, two groups of weights are obtained. The first group, $(\mathrm{WG1})_{i,j}$, is the weights between the input node *i* and the hidden node *j*. The second group, $(\mathrm{WG2})_{j,k}$, is the weights of the hidden node *j* and output node *k*. A sigmoid is used for the activation function of the hidden and output nodes. Then, the total input to the *j*–th hidden node (IHN_j) and the output of the *j*–th hidden node (OHN_j) are given by

$$IHN_j = \sum_{i=1}^{I} x_i (WG1)_{i,j} \ , \ OHN_j = \frac{1}{1+e^{-\left[\sum_{i=1}^{I} x_i (WG1)_{i,j}\right]}}. \tag{1}$$

Nextly, the total input to the *k*–th output node, ION_k, is given by

$$ION_k = \sum_{j=1}^{J} (WG2)_{j,k} \frac{1}{1+e^{-\left[\sum_{i=1}^{I} x_i (WG1)_{i,j}\right]}}. \tag{2}$$

Then, the final value of the *k*–th output node, ψ_k, is given by

$$\psi_k = \left\{ \frac{1}{1+e^{-\left[\sum_{j=1}^{J} WG2_{j,k} \left(\frac{1}{1+e^{-\left[\sum_{i=1}^{I} x_i (WG1)_{i,j}\right]}} \right)\right]}} \right\}. \tag{3}$$

The function, $\psi_k = f(x_i, (\mathrm{WG1})_{i,j}, (\mathrm{WG2})_{j,k})$ is an exponential function in x_i since $(\mathrm{WG1})_{i,j}$, $(\mathrm{WG2})_{j,k}$ are constants and its maximum output value is equal to one. Then, we can say that "An input vector, $\boldsymbol{X_m}$, belongs to a *class*$_k$ *iff* $\boldsymbol{\psi_k} \in \boldsymbol{C_m} = 1$ and all other elements in $\boldsymbol{C_m} = 0$."

```
With the given parameters,
  - $A$: set of attributes.   - $\alpha$: set of attributes (conditional), $\beta$: set of result attributes (result).
  - $n$: the number of total attribute, $\gamma$: the length of feature $n$.
  - $G$: set of the best $b$ chromosome, $g$: a chromosome in $G$.
  - $b$: the number of total chromosome ($|G| = b$).
  - $\mu$: the number of total rule found by association rule mining.

Repeat Step 1 to Step 5, for all $g$ in $G$.
  1. Create temporary empty rule $t$: $\{\alpha\} \rightarrow \{\beta\}$, and Set $\alpha = \beta = \varphi$.
  2. Divide best chromosome into $2n$ segments.
     (Each segment in 1 to $n$ is corresponds to each attribute of $A_n$ for condition of rule).
     (Each segment in $n$+1 to $2n$ is corresponds to each attribute of $A_n$ for result of rule).
  3. For all $i$, $i = 1$ to $n$.
     3.1 For all $j$, $j = 1$ to $\gamma$.
         3.1.1 If the corresponding bit of conditional chromosome is equal to '1',
               $\alpha \leftarrow \alpha \cup A_j$.
     3.2 Connect all feature in $\alpha$ with operator 'AND'.
  4. For all $i$, $i = n+1$ to $2n$.
     4.1 For all $j$, $j = 1$ to $\gamma$.
         4.1.1 If the corresponding bit of result chromosome is equal to '1',
               $\beta \leftarrow \beta \cup A_j$.
     4.2 Connect all feature in $\beta$ with operator 'AND'.
  5. For all $k$, $k = 1$ to $\mu$.
     5.1 If any $R(k) \equiv t$ then $R \leftarrow R - R(k)$ else $R \leftarrow R \cup t$.
Return final rule set $R$
($R$ = rules mined by association mining + rules decoded by top $b$ chromosome decoding).
```

Fig. 3. The rule decoding procedures from the selected best chromosome

Deriving Association Rules from Trained Network with GA-Based Decoding

To extract relations (rules) among the input attributes, $\boldsymbol{X_m}$ relating to a specific *class*$_k$ one must find the input vector, which maximizes $\boldsymbol{\psi_k}$. This is an optimization problem and can be stated as $\psi_k(x_i)$ by considering binary data feature vector $\boldsymbol{x}$. In $\psi_k(x_i)$, x_i are binary values (0 or 1). Since the objective function $\psi_k(x_i)$ is nonlinear and the constraints are binary, it is a nonlinear integer optimization problem. Genetic algorithm (GA) can be used to solve this optimization problem by maximizing the objective function $\psi_k(x_i)$. In this paper, we used conventional generational-GA procedures with this objective function $\psi_k(x_i)$ to find the best chromosome which provided as an input of neural network and produce best network output (i.e. highest prediction accuracy).

After we obtain best chromosomes which produces best network output, we decoded these chromosome into the form of association rules (here, we call this association rule as '*neural feature association rule*' because they are extracted from trained neural network). To extract a rule for $class_k$ from the best chromosomes selected by GA procedures, we decoded them with several procedures presented in Figure 3.

4 Experimental Results

Protein Interopaction as Binary Feature Vector

An interaction is represented as a pair of two proteins that directly binds to each other. This protein interaction is represented by binary feature vector of interacting proteins and their associations. Figure 4 describes this interaction representation processes. Interactions prepared through these processes are provided to the neural network-based clustering and to the prediction model to group each protein interaction class and learn the association of features which generalize the interactions. Then, the constructed cluster prototype is used to predict the classes of protein interactions presented in the test step. The 13 functional categories of interacting protein from MIPS [11] which is known for the most reliable curated protein interaction database in current literature are used to evaluate the category classes clustering accuracy.

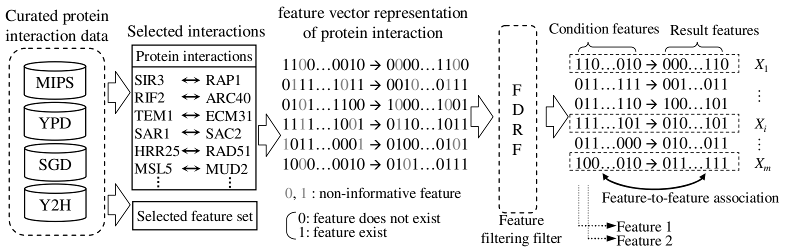


Fig. 4. The feature vector representation of protein interactions. Each interaction is represented as a binary feature vector (whether the feature exists or not). The feature dimension reduction filter (FDRF) marks those features as 'don't care' which have *SU* value less than given *SU* threshold δ to remove non-informative features so as to improve the performance of clustering model. The marked features are regarded when train clustering model. The resulting vectors of interactions are provided to the neural network learning model as network input, describe in Figure 2, for model training, testing, and generation of neural feature association rules.

Data Sets

Each *yeast* protein has various functions or characteristics which are called 'feature.' In this paper, set of features of each protein are collected from public genome databases [11, 12, 15, 16, 17]. We use similar features of protein interaction of Oyama *et al.* [20] which include EC numbers (from SWISS-PROT), SWISSPROT/PIR keywords, PROSITE motifs, bias of the amino acids, segment cluster, and amino acid patterns, etc. A major protein pairs of the interactions are also obtained from the same data source of Oyama *et al.* [20]. These dataset include various experimental data such as

YPD and Y2H by Ito *et al.* [16] and Uetz *et al.* [17]. Additionally, we used SGD to construct more abundant feature set [12]. Table 1 shows the statistics of each interaction data source and the number of features before and after the application of FDRF.

Table 1. The statistics for the dataset

Data Source	# of interactions	# of initial features	# of filtered features
MIPS [11]	10,641	6,232 (total)	1,293 (total)
YPD [15]	2,952		
SGD [12]	1,482		
Y2H (Ito *et al.*) [16]	957		
Y2H (Uetz *et al.*) [17]	5,086		

Experiment Procedures

First, we predicted the classes of new PPIs with neural network for their 13 functional categories obtained from MIPS [11]. The accuracy of class prediction is measured whether the predicted class of interaction is correctly corresponds to the class of MIPS. After this step, we constructed feature association rule from this trained neural network with similar procedure with Figure 3.

Next, we trained another neural network with PPI data represented as binary feature vector according to the method in Figure 4. After the model training, we extracted again feature association rules from the model with the procedure in Figure 3. Then we predicted test PPIs with these two set of association rules and measured the prediction accuracy of each approaches with 10-fold cross-validation.

Results

Table 2 show the interaction prediction performance of various combination of associantion mining, information theory based feature filtering, and exploitation of rules derived from trained neural network.

Table 2. The comparison of prediction accuracies of the proposed methods. The effect of the FDRF-based feature selection and neural network-based are shown in terms of prediction accuracy. For filtered interaction vectors by FDRF, the feature association-based prediction model with neural association rule (☆) shows the best performance (*Asc*: association rule based prediction. *FDRF + Asc.*: prediction based on association rule mined from filtered feature vectors. *Asc. + N-Asc.*: rule based prediction with association rule and the rule derived from trained neural network. *FDRF + Asc. + N-Asc.*: combination of all methods).

Prediction method	Number of interactions			Accuracy (|P|/|T|)
	Training set Size	Test set (T)	Predicted correctly (P)	
Asc. (△)	4,628	463	423	91.4 %
FDRF + Asc. (▽)	4,628	463	439	94.8 %
Asc. + N-Asc. (◇)	4,628	463	432	93.3 %
FDRF + Asc. + N-Asc. (☆)	4,628	463	445	96.1 %

In Table 2, simple association mining approach (△) achieved the lowest performance. The number of total feature used in this approach was 6,232. This is quite high feature dimension. So, we can guess that it may includes lots of non-informative and redundant features and these features may affect the prediction accuracy in negative way by interfering correct rule mining. This assumption confirmed by investigating the result of second approach, FDRF + Asc. (▽), association mining with non-informative and redundant feature filtering. This feature filtering approach improved overall prediction performance about 3.4% than the first approach. But the third approach, Asc. + N-Asc. (◇), prediction with the rules from association rule mining and the rule derived from trained neural network only improved overall prediction performance about 1.9% than the first approach.

This result can be explained again with the feature dimension problem. In this third approach, there also exist redundant and non-informative garbage features which decrease the prediction performance. But in this approach, eventhough there still lots of garbage features, the over all performance improved about 1.9%. This is the effect of the rule exploitation derived from trained neural network. This inference can be confirmed again by investigating the result of fourth approach, FDRF + Asc. + N-Asc (☆), prediction with the rule from association mining and the rule derived from trained neural network along with feature filtering. Non-informative and redundant features are filtered out in this approach. Consequently, this approach improved over all prediction accuracy up to about 4.7%. These results are outperform other several approaches including *k*-NN (86.4%), support vector machine (93.3), structure and sequence conservation-based prediction (88.5%), and generative stochastic model with MCMC estimation (94.8%) in prediction accuracy [21].

Thus, we can say that both the information theory-based feature filtering and the exploitation of the rule derived from trained neural network and conventional association rule mining methods are helpful for improving overall performance of feature-to-feature association-based PPI prediction. By considering these experimental results, the proposed approaches will be useful as a data preprocessing and prediction methods especially when we handle the data which have many features.

5 Conclusions

We presented neural network based protein interaction learning and association rule mining method from feature set and trained neural network model for PPI prediction task. Also we applied information theory-based feature selection procedure to improve the performance of trained feature association learning model. The proposed method (combination of all methods) achieved accuracy improvement about 4.7%. From the experimental results, it is suggested that the neural network-based feature association learning model could be used for more detailed investigation of the PPIs by learning the hidden patterns of the data having many features and implicit associations among them. From the results, we can conclude the proposed method is suitable for efficient analysis of PPIs through learning their hidden 'feature associations.'

However, to overcome the false positive rates of current public interaction database is one of the important issues for more reliable prediction. The computational complexities caused by using neural network and GA is another issues to resolve for efficient predictions. Also, more biological features such as pseudo amino acid composition or protein localization facts will be also helpful for improving overall prediction accuracy and should be considered in the future works.

References

1. Goffeau, A., Barrell, B.G., *et al.*: Life with 6000 genes. *Science* **274** (1996) 546–67.
2. Eisen, M.B., Spellman, P.T., *et al.*: Cluster analysis and display of genomewide expression patterns. *Proc. Natl. Acad. Sci. USA* **95** (1998) 14863–68.
3. Pavlidis, P. andWeston, J.: Gene functional classification from heterogeneous data. In *Pro. of the 5th Int'l Conf. on Comp. Molecular Biology* (2001) 249–55.
4. Park, J., *et al.*: Mapping protein family interactions: intramolecular and intermolecular protein family interaction repertoires in the PDB and yeast. *J. Mol. Biol.* **307** (2001) 929–39.
5. Iossifov, I., Krauthammer, M., *et al.* Probabilistic inference of molecular networks from noisy data sources. *Bioinformatics* **20**(8) (2004) 1205–13.
6. Ng, S.K., Zhang, Z., *et al.* Integrative approach for computationally inferring protein domain interactions. *Bioinformatics* **19**(8) (2003) 923–29.
7. Eom, J.-H., *et al.*: Prediction of implicit protein–protein interaction by optimal associative feature mining. *Lecture Notes in Computer Science* **3177** (2004) 85–91.
8. Carpenter, G.A. and Grossberg, S.: A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision*, *Graphics and Image Processing* **37** (1987) 54–115.
9. Barbara, M.: ART1 and pattern clustering. In proceedings of the 1988 connectionist models summer 1988. Morgan Kaufmann (1988) 174–85.
10. Heins, L.G. and Tauritz, D.R.: Adaptive resonance theory (ART): an introduction. *Internal Report* 95-35, Dept. of Comp. Sci., Leiden University, Netherlands (1995) 174–85.
11. Mewes, H.W., Amid, C., *et al.*: MIPS: analysis and annotation of proteins from whole genomes. *Nucleic Acids Res.* **32**(1) (2004) D41–44.
12. Christie, K.R., Weng, S., *et al.*: Saccharomyces Genome Database (SGD) provides tools to identify and analyze sequences from Saccharomyces cerevisiae and related sequences from other organisms. *Nucleic Acids Res.* **32**(1) (2004) D311–14.
13. Rangarajan, S.K., Phoha, V.V., *et al.*: Adaptive neural network clustering of web users. *IEEE Computer* **37**(4) (2004) 34–40.
14. Yu, L. and Liu, H.: Feature selection for high dimensional data: a fast correlation-based filter solution. In *Proceeding of ICML-03* (2003) 856–863.
15. Csank, C., Costanzo, M.C., *et al.*: Three yeast proteome databases: YPD, PombePD, and CalPD (MycoPathPD). *Methods Enzymol.* 350 (2002) 347–73.
16. Ito, T., Chiba, T., *et al.*: A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc. Natl Acad. Sci. USA* **98** (2001) 4569–4574.
17. Uetz, P., Giot, L., *et al.*: A comprehensive analysis of protein–protein interactions in Saccharomyces cerevisiae. *Nature* **403** (2000) 623–627.
18. Press, W.H., Flannery, B.P. *et al.*: Numerical recipes in C: The Art of Scientific Computing (2nd Ed.). *Cambridge University Press*, Cambridge (1992) 633–634.
19. Eom, J.-H., *et al.*: Adaptive Neural Network based Clustering of Yeast Protein-Protein Interactions. *Lecture Notes in Computer Science* **3356** (2004) 49–57.
20. Oyama, T., Kitano, K., *et al.*: Extraction of knowledge on protein–protein interaction by association rule discovery. *Bioinformatics* **18** (2002) 705–14.
21. Eom, J.-H. and Zhang, B.-T.: Prediction of implicit protein–protein interaction using optimal associative feature rule, *Journal of Korea Information Science Society: Software and Application* **33**(4) (2006) 365–77.

Appendix

The procedures of ART-1 based protein interaction clustering.

```
Given array of input protein interaction vectors **PPI** and vigilance parameter $\rho$,

1. **Initialize**
   1.1 **Set** the value of gain control $G_1$ and $G_2$,
       $G_1, G_1 = \begin{cases} 1 \; if \;\; input \;\; PPI_I \neq 0 \; and \;\; output \; from \; F_2 \;\; Layer = 0 \\ 0 \; for \;\; all \;\; other \;\; cases \end{cases}$
   1.2 **Set** all nodes in $F_1$ layer and $F_2$ layer to 0.
   1.3 **Set** all weight of top-down weight matrix, $t_{ji} = 1$.
   1.4 **Set** all weight of bottom-up weight matrix,
       $b_{ij} = \left(1/(n_f + 1)\right)$ ($n_f$ = the size of the input feature vector).
   1.5 **Set** the vigilance parameter $\rho$ (0.2 to 0.7).

2. **Repeat** Step 2.1 to 2.7, for all protein-protein interaction vector $PPI_I$.
   2.1 **Read** randomly chosen interaction vector
       $PPI_I = (P_1, P_2, \cdots, P_{I=n_f})$, where $P_1 = 0 \; or \; 1$.
   2.2 **Compute Input** $y_j$ for each node in $F_2$, $y_j = \sum_{i=1}^{n_f} P_i \times b_{ij}$.
   2.3 **Determine** ***k***, $y_k = \sum_{j=1}^{\# \; of \; nodes \; \text{in} \; F_2} \max(y_j)$.
   2.4 **Compute Activation**, $X_k^* = (X_1^*, X_2^*, \cdots, X_{i=5,240}^*)$ for the node $k$ in $F_1$
       Where, $X_i^* = t_{ki} \times P_i \;\; (i = 1, \cdots, n_f)$.
   2.5 **Calculate similarity** $\delta$, between $X_i^*$ and $PPI_i$: $\delta = \frac{\|X_k^*\|}{\|PPI_I\|} = \left( \sum_{i=1}^{n_f} X_i^* \Big/ \sum_{i=1}^{n_f} P_i \right)$.
   2.6 **Update weight** of top-down weight matrix with $PPI_I$ and node *k*.
       If $\delta > \rho$, update top-down weight of node *k*, $t_{ki}(new) = t_{ki} \times P_i$ where $i = 1, \cdots, n_f$.
   2.7 **Create a new node** in $F_2$ layer
       2.7.1 **Create** a new node *l*.
       2.7.2 **Initialize** top-down weight $t_{li}$ to the current input feature pattern.
       2.7.3 **Initialize** button-up weight for the new node *l*.
             $b_{il}(new) = \left( X_i^* \Big/ (0.5 + \sum_{i=1}^{n_f} X_i^*) \right)$, where $i = 1 \cdots n_f$.
```

Prediction of Protein Secondary Structure Using Nonlinear Method

Silvia Botelho, Gisele Simas, and Patricia Silveira

FURG
Av. Itália Km 8, Rio Grande, RS, Brazil
{silviacb, gisele, patricia}@ee.furg.br

Abstract. This paper presents the use of neural networks for the prediction of protein Secondary Structure. We propose a pre-processing stage based on the method of Cascaded Nonlinear Components Analysis (C-NLPCA), in order to get a dimensional reduction of the data which may consider its nonlinearity. Then, the reduced data are placed in predictor networks and its results are combined. For the verification of possible improvements brought by the use of C-NLPCA, a set of tests was done and the results will be demonstrated in this paper. The C-NLPCA revealed to be efficient, propelling a new field of research.

1 Introduction

This paper investigates the use of artificial neural networks (NNs) as pre-processing stage in the prediction/classification of protein Secondary Structure, from Blast profiles [1] gotten through their sequence of amino acids. This classification serves to reduce searching space in other methods of prediction of protein Tertiary Structure that, in turn, is strictly related with their functionality [10].

Considering the large dimension of the data to be treated, a stage of pre-processing can increase the performance of the predictor. [5] use the traditional techniques of Principal Components Analysis (PCA) for dimensional reduction, showing that pre-processing methods aiming at the dimensional reduction of the data can contribute enough for the improvement of the classifications supplied for the predictors systems.

This paper suggests the use of a pre-processing stage of data, considering, of original form, the use of neural networks through of the Cascaded Nonlinear Components Analysis (C-NLPCA) method in the pre-processing of data [2]. The use of this method is concerned with the utilization of the nonlinear potential of neural networks, in order to acquire useful information for the classification phase, together with the reduction. Moreover, the use of the C-NLPCA prevents the occurrence of an undesired simplification of the variability of data, which could be caused by the use of a linear method such as the PCA.

Aiming at validation of the proposal, once data are reduced, these serve as input for three neural networks with different topologies, trained by the Resilient Propagation (RPROP) method [13] in epoch, each NN independently finds the classification of Secondary Structure for each amino acid. Then, these classifications are combined for the attainment of better results.

This paper has 7 sections. Firstly we present an overview of Problem of Prediction of Protein Secondary Structure. Blast Profiles are detailed in section 3. The fourth section is dedicated to introduce our pre-processing method for data reduction, following by the classifier explanation, in section 5. Our implementation, tests and results are analyzed in the sixth section, finally we present the conclusion and future works.

2 The Prediction of Secondary Structure

The proteins are responsible for the structural and architectural function of the cell and they are composed by the numerous amino acids join themselves by peptide bonds (chains) [10]. The sequence of amino acids and its peptide bonds is called Primary Structure. While that the Secondary Structure corresponds to the foldings occurred for the establishment of hydrogen bonds between distant amino acids. The knowledge of the Secondary Structure of a protein serves to reduce the searching space of its Tertiary Structure, space configuration of the protein, therefore, facilitating in the briefing of its functionality. The detailed knowledge of proteins, in turn, would allow to to discover alternative treatments for diverse diseases. Thus, the problem of prediction of protein Secondary Structure consists of, from its Primary Structure (sequence of amino acids), classifying each of its constituent amino acids in one of the recurrent substructures in the three-dimensional conformation of the protein which can be grouped in: α-helixes, β-sheets and coils.

Related Works. The algorithms that simulate the folding of the sequences, developed until the moment, can not accurately simulate the laws that control this process. Therefore, Artificial Neural Networks (NNs) show as a promising method and come frequent being used, demonstrating good results [4,6,14].

[12] presented a study to choose the best configuration of NN for prediction of Secondary Structure. The proposed solution suggests that the sequences of proteins must be covered by overlapped windows, being the prediction related with the central element of the window. From the success reached by [12], other solutions had been searched, applying different algorithms of training, topologies of NN, and treatment of the input data [14]. A significant advance was achieved with the use of more biological information as input for the NNs, more specifically with the use of sequence profiles of proteins as input for the prediction. In contrast of sequences that supply local information only, the profiles looking for distant relations between diverse sequences of a bank. An example of this profit can be evaluated in predictor PHD [15], which used frequency profiles as input.

Amongst most recent publications, the significant improvement obtained through the alteration of the type of input data are detached, introducing more divergent information, particularly the PSI_BLAST profiles. This type of input was proposed initially by Jones in predictor PSIPRED [6]. The PSI_BLAST [1] can find distant homologous in sequences of proteins stored in a data base.

Another existing predictor is CONSENSUS, which was developed by [3]. This predictor compares and combines four others: DSC [8], PHD [15], NSSP [17], and PREDATOR [3], without making reductions of data. While that, [5], taking into account the high dimension of the data to be treat, used a technique of statistical reduction of input data, applying the Principal Components Analysis (PCA) in the PSI_BLAST profiles. The results were compared with the studies of [3], and the constructed new predictor by [5] presented better results.

Considering all predictors that use PSI_BLAST profiles as input data, the GMC [4] is the predictor that has presented the best performance. However this one does not use dimensional reduction methods.

3 Using Blast Profiles in the Prediction

The proposal of this study is to evaluate the improvements that the use of a nonlinear method for dimensional reduction of data can bring in the prediction of Secondary Structure. Therefore, a similar procedure to those of [5], which used a linear method, was carried through. In summary, a sequence of proteins is presented to the PSI_BLAST and the matrix of scores produced by this is re-passed to a pre-processing stage. In such a stage, the input data are reduced and, to follow, passed to the three classifiers, which receive the same input values. The results obtained by the classifiers are, then, presented for the stage of combination of rules, which gives the final classification.

Attainment of Blast Profiles. The Primary Structure of a protein can be represented by a sequence of characters (strings) of variable size, where each character represents one of the 20 amino acids. The order where these amino acids are displayed in the sequence represents the occurred polypeptide bonds.

The 396 sequences of proteins present in the CB396 [11] are submitted to PSI_Blast research. The software Position Specific Iterated - Basic Local Alignment Search Tool (PSI_Blast) [1] searches for similar proteins in the data base selected, i. e., formed by similar sequences of amino acid, taking into account the order of their position in the sequence. The result of this search is organized through an increasing order of similarity; this method is called alignment.

The system also returns as output PSI_Blast profiles. Each amino acid that composes the protein occupies a position in the sequence. Thus, all the amino acids in this position, in all the sequences that compose the alignment, are compared, and specific values are attributed according to the occurrences found, these values forms the PSI_Blast profile. The searched protein receives a score that is stored in the PSSM Position Specific Score Matrix. Such a matrix has the dimension $p \times n$, where p represents 20 amino acids and n the number amino acid that composes protein. These PSSM matrices are re-passed to the reduction stage, which will be described in the next section.

4 Dimensional Reduction

Supposing that neural networks can be used as a good approach to classify protein Secondary Structures [4,6,14], the selection of relevant information on the input sequences of the classifier neural networks and its process of codification are crucial steps for the effective prediction of the NN. Neural Networks with few inputs have few weights to be adjusted, lead to the best generalizations, to a faster training, and prevent saturation.

Regarding this, the performance of the predictor NN can be increase, reducing the number of its inputs. A reduction stage can improve the performance, eliminating redundant information present in the data base. We proposed the use of Cascaded Nonlinear Components Analysis (C-NLPCA), an approach based on neural networks for dimensional reduction of large data set. In our approach C-NLPCA receives the PSSM score matrix eliminating redundant information presented in this matrix. Thus, the PSSM score matrix of each protein is covered by overlapped windows of size w, displaced until reaching the original size of the protein. The dimension of each window is $p = 20 * w$ (representing the frequencies of 20 existing amino acids x w amino acids belonging to the window). The reduction stage reduces from p to k the number of data (dimension of the window). Later, each set of k components is forwarded to the classification stage, which has the task to predict the corresponding Secondary Structure to the central amino acid of the window.

4.1 Principal Components Analysis (PCA)

The PCA is a technique that can be used to supply to a statistic analysis of the data set, being used as pre-processing stage to the prediction [5]. This analysis is concerned with the extraction of the factors that better represent the structure of interdependence between variables of large dimensions. Therefore, all the variables are analyzed simultaneously, each one in relation to all the others, aiming at determining factors (principal components) that maximize the explanation of variability existing in the data. However, the PCA is indicated for the analysis of variables that have linear relations. Each principal component brings statistic information different from the others. However, the first principal components are such more relevant that we can even disdain the others [7].

The Expectation Maximization (EM) algorithm can be used to extract the linear principal components, see [16] for more details. This algorithm has $O(knp)$ complexity, where k represents the number of auto-vectors to be learned. The algorithm EM has the following steps: $X = (C^T C)^{-1} C^T Y$; $C^{new} = Y X^T (X X^T)^{-1}$

Y represents the input matrix, with dimension $20 \times n$; X represents the matrix of unknown states, $k \times n$, $k =$ number of principal components; C is the matrix of the k principal components. The steps of the algorithm EM are repeated until occurring convergence of the matrix of components C, so that n must always be bigger than k.

4.2 Cascaded Nonlinear Components Analysis (C-NLPCA)

Nonlinear behavior of input data, in the problem of Secondary Structure prediction, can be noticed by the good performance of neural networks in this application [4,6,14]. We intend also to extend this nonlinear treatment for the pre-processing stage. In this paper, it is suggested the use of C-NLPCA, see details in [2], to provide a nonlinear mapping of the data, since linear methods, such as the PCA, may introduce undesirable simplifications in the analysis of variables with nonlinear relations. In previous work, we have proposed the C-NLPCA method. It is based on the cascading in layers of NLPCAs simple NNs [9], aiming the nonlinear treatment of the data of high dimension.

The NLPCA Analysis. Neural networks for the analysis of nonlinear principal components, called NLPCAs, are Multi-Layers-Perceptron NNs composed by five layers: input (p neurons), hidden of codification (m neurons), bottleneck (r neurons), hidden of decoding (m neurons) and output (p neurons). Neurons of the codification and decoding layers use nonlinear functions, while those of input, bottleneck and output use linear functions of activation. The inputs in a p-dimensional space, when forward to the bottleneck layer, are mapped by the r-dimensional space before reproducing the outputs. After the stage of training, the activation values of the bottleneck layer neuron supply the nonlinear principal components.

The NLPCA neural network is auto-associative and it is trained to get a mapping between input and output, which minimize the function:

$min \sum_{i=1}^{n} |\overrightarrow{X_i} - \overrightarrow{X_i'}|$, where $\overrightarrow{X_i'}$ is the output of the NN for each $\overrightarrow{X_i}$ input.

The output produced by the expansion will contain the cumulative error of n samples, so the $\overrightarrow{X_i} - \overrightarrow{X_i'}$ residue value can be used for the attainment of the second principal component and, thus, successively.

From NLPCAs sets to C-NLPCA System. Due to the intrinsic problems of saturation of the neural networks, the applicability of the NLPCA is restricted to the cases where $p \ll n$[1]. In order to avoid such limitation, we have proposed an architecture where NLPCAs are grouped in layers, see [2] for more details. In the reduction stage, data of p initial dimensions are grouped in a series of small NLPCAs neural networks of $p\prime$ dimensions. Each NLPCA reduces its respective inputs of $p\prime$ for 1 dimension. The reduced data (principal locals) again are grouped and reduced successively in subsequent layers. Such layers are called reduction layers. Next, the bottleneck neuron of the last C-NLPCA neural network will supply the first global principal component (C-NLPC) of the set of input of p dimension. Such NLPCA neural network is called by bottleneck layer. This treatment process of information in diverse NNs and its respective combinations are called cascading.

The Expansion Stage. After of the successive reduction layers and of the bottleneck layer, a set of MLP neural networks composes the expansion stage. The output values gotten by bottleneck NLPCA are used as input for MLP NNs with 1 input neuron and $p\prime$ output neurons. These NNs, designated as expansion NNs, go being disposed successively, layer to layer, until the reproduction of output sets whose final dimension is equal to the presented to the system: p. The expansion layers are symmetrical to the reduction layers, being that during the expansion, there is not a training of neural networks; only the value of the

[1] A more formal relation can be established in function of parameters number (weights and bias) of NLPCA and of the number of the samples n presented to the NN. This relation establishes that $2pm + 2mr + 2m + r + p \leq n$.

principal component is propagated using a NN composed by the three last layers of each symmetrical NLPCA of the reduction.

Since every principal local are combined successively, in the stage of reduction, the C-NLPCA considers all the relations of neighborhood between the variables. As well as the relations between all the windows that compose PSSM matrix of a given protein are also analyzed in their use as samples in the training.

5 Classification

Aiming at the validation of use of the C-NLPCA in the pre-processing stage of predictors, we have also implemented a classification stage [4]. The proposed classifier associates three MLPs NNs with distinct topologies, objectifying the choice of a better local minimum. The distinction of the NNs is associated with the amount of neurons of the hidden layer. Each output value is associated according to established norms in the stage of combination of rules. The training of NNs is made by epoch, based on the Resilient Propagation - RPROP [13].

The combination of the three neural networks of different topologies intends to improve the prediction of the Secondary Structures [3]. The rules used for the classification of results are such: Voting, Average, Product, Maximum and Minimum [5]. After the combination of NNs, the results are evaluated using a reduced variation of the Jack-Knife process applied to subgroups of proteins, see [5] for more details. The exactness of each of each subgroups is measured by the value of Q3, obtained through the equation $Q3 = \frac{correct\_number\_amino_acids}{total\_number\_amino_acids} \times 100$.

6 Tests and Results

A tool in C++ was developed aiming at to implement the pre-processing stage in C-NLPCA and the classifier neural networks. Profiles of window of size $w = 13$ were analyzed, resulting in data of dimensions of 260 elements ($13 * 20$ amino acid) to be processed and reduced. The reduced data were placed in predictor NNs with 30, 35 and 40 neurons in the hidden layer.

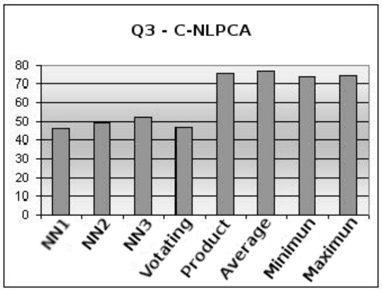
For the effect of comparison with other studies and in function of the characteristics of the presented data, the $k = 80$ first principal components were extracted. Thus, a dimensional reduction from 260 to $r = 1$ was gotten 80 times, being that, in each repetition, the residue is normalized and injected as input for the calculation of the next component. Several tests are done, regarding the performance of the proposal. The obtained results confirm the effectiveness of the method. The table 1 shows a comparison between our proposal (with C-NLPCA reduction) and others.

The results obtained by the adopted predictor in this work using the reduction by the C-NLPCA presented better results than the GMC (without dimensional reduction) and PCA methods. This is due to the fact that the dimensional reduction took into account the nonlinearity of data, besides providing a more effective training of the classificatory neural network. From original 260 components ($p = 260$), the nonlinear reduction process have resulted in 80 Cascaded Nonlinear Principal Components (C-NLPCs). Figure 1 shows the value of each

Table 1. The best result of our C-NLPCA Approach

Method	Q3
PHD [15]	71.9%
DSC [8]	68.4%
PREDATOR [3]	68.6%
NNSSP [17]	71.4%
CONSENSUS [3]	72.9%
GMC [4]	75.9%
PCA [5]	73,8%
C-NLPCA	76.1%

Table 2. The values of the 80 PCs



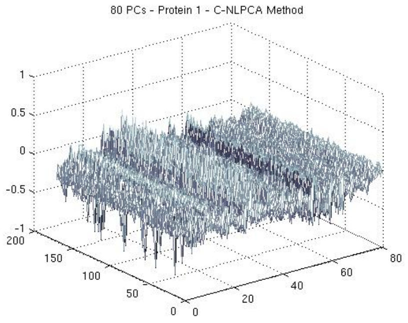


Fig. 1. The values of the 80 C-NLPCs

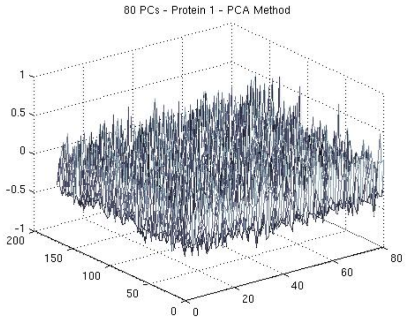


Fig. 2. The values of the 80 PCs

one C-NLPC. We can compare them with Linear Principal Components (PCs), see figure 2. A more smoothed variability is found in C-NLPCs than PCs.

Regarding the classifier stage, the combination rules had contributed in the increase of precision of the results. Figure/table 2 shows that the application of the "average rule" have gotten the best results. The neural network with the most neurons in the hidden layer (NN3 in figure/table 2) presents a better performance in relation to others.

7 Conclusions

This paper presented a study on computational methods for the classification of protein Secondary Structures in: α-helixes, β-sheets and coils. It was investigate the use of the C-NLPCA method as dimensional reduction stage of the input data of the classifiers.

The protein Primary Structures contained in CB396 bank were submitted to the software PSI_Blast that located distant homologous proteins. From the alignment of these proteins, the PSI_Blast generated a PSSM scores matrix. Such a matrix was passed to a reduction stage and the reduced data were passed through classifier NNs with different topologies, whose outputs were combined.

The obtained results with C-NLPCA confirm the effectiveness in the dimensional reduction of data for the prediction of Secondary Structures. Fact that stimulates researches on the application of this method. Studies can be done concerning the topology of the neural networks used in C–NLPCA. As well as, the way of attaining C-NLPCs of higher order.

References

1. S. Altchul, T. Madden, A. Shaffer, J. Zhang, Z. Zhang, W. Miller, and D. Lipman. Gapped blast and psi-blast: A new generation of protein database search programs. *Nucleic Acids Research*, 1997.
2. S. S. C. Botelho, R. A. Bem, I. L. Almeida, and M. M. Mata. C-nlpca: Extracting non-linear principal components of image datasets. *Simpsio Brasileiro de Sensoriamento Remoto*, 2005.
3. A. J Cuff and J. G. Barton. Evaluation and improvement of multiple sequence methods for protein secondary structure prediction. *Proteins Structure, Function and Genetics*, 34:508–519, 1999.
4. K. S. Guimaraes, J. C. B. Melo, and G. D. C. Cavalcanti. Combining few neural networks for effective secondary structure prediction. *Proceedings of the Third IEEE Symposion on Bioinformatics and Bioengineering*, pages 415–420, 2003a.
5. K. S. Guimaraes, J. C. B. Melo, and G. D. C. Cavalcanti. Pca feature extraction for protein structure prediction. *International Joint Conference on Neural Networks*, pages 2952–2957, 2003b.
6. D. T. Jones. Protein secondary structure prediction based on position-specific scoring matrices. *on J. Mol. Biol.*, 292:195–202, 1999.
7. R. Khattree and D. N. Naik. *Multivariate data reduction and discrimination with SAS software.* Cary, NC: SAS Institute Inc., 2000.
8. R. King and M. Sternberg. Identification and application of the concepts important for accurate and reliable protein secondary structure prediction. *Proteins Science*, 5:2298–2310, 1996.
9. M. A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AlChe Jounal*, 37:233–43, 1991.
10. A.L. Lehninger. *Principles of biochemist.* Editora Sarvier, So Paulo, 1984.
11. University of Dundee and The Barton Group. Cb396. *http://www.compbio.dundee.ac.uk/*, 20 jul 2005.
12. N. Qian and T. J. Setnowski. *Predicting the Secondary Structure of Globular Proteins Using Neural Network Models.* Baltimore, 1988.
13. M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. *Proceedings of the International Conference of Neural Networks*, pages 586–591, 1993.
14. B. Rost. Review: Protein secondary structure prediction continues to rise. *Journal of Structural Biology*, 134:204–218, 2001.
15. B. Rost and C. Sander. Combining evolutionary information and neural network to predict secondary structure. *Proteins*, 19:55–72, 1994.
16. S. Roweis. *Algorithms for PCA and SPCA, in Advances in Neural Information Processing Systems.* M. I. Jordan and M. J. Kearns and S. A. Solla, 1998.
17. A. Salamov and M. Solovyev. Prediction of protein secondary structure by combining nearest-neighbor algorithm and multiple sequence alignments. *Journal of Molecular Biology*, 247:11–15, 1995.

Clustering Analysis for Bacillus Genus Using Fourier Transform and Self-Organizing Map

Cheng-Chang Jeng[1], I-Ching Yang[1], Kun-Lin Hsieh[1], and Chun-Nan Lin[2]

[1] Systematic and Theoretical Science Research Group, National Taitung University, 684 Chunghua Rd., Sec. 1, Taitung, Taiwan
{cjeng, icyang, klhsieh}@nttu.edu.tw
[2] Department of Management Information System, National Chung Cheng University, 168, University Rd., Min-Hsiung Chia-Yi, Taiwan
a103p2@yahoo.com.tw

Abstract. Because the lengths of nucleotide sequences for microorganisms are various, it is difficult to directly compare the complete nucleotide sequences among microorganisms. In this study, we adopted a method that can convert DNA sequences of microorganisms into numerical form then applied Fourier transform to the numerical DNA sequences in order to investigate the distributions of nucleotides. Also, a visualization scheme for transformed DNA sequences was proposed to help visually categorize microorganisms. Furthermore, the well-known neural network technique Self-Organizing Map (SOM) was applied to the transformed DNA sequences to draw conclusions of taxonomic relationships among the bacteria of Bacillus genus. The results show that the relationships among the bacteria are corresponding to recent biological findings.

Keywords: DNA sequence, Bacillus, Fourier transform, Self-organizing map.

1 Introduction

The classification of bacteria may lead to the understanding of evolution of microorganisms. Before the advances in molecular biology, biologists had difficulties in differentiating microorganisms or inferring their evolutionary relationships with morphological features or microbial fossils [1]. Until 1965, Zuckerandl and Pauling [2] suggested that molecules can be documents of evolutionary history for cellular lifeforms. Later in 1977, Woese and Fox [3] selected macromolecule,16S ribosomal RNA (rRNA), to establish and infer phylogenetic relationships among microorganisms. Based on differentiating the similarities and differences in 16S rRNA, Woese and his colleagues showed that root of lifeforms can be categorized into a three-domain system: Eucarya, Archaea, and Bacteria [4].

However, there are still some criticisms that argue inconsistent decisions made by Woese and his colleagues on categorizing organisms [5]. Also, some research based on the analysis of other genes and proteins found different topology of phylogenetic tree [6]. These research suggested that adopting a single gene, rRNA, to infer phylogenetic relationships is unable to fulfill the requirements of phylogenetic classification. Moreover, Delong and Pace [1] expected that comparative information on the genome structure, content, and organization may help obtain more understanding the evolutionary processes of microorganisms.

To understand genome structure, as we known, the biological organisms are frequently assumed to have the different coding structure of nucleotide sequences. In other words, it is possible to analyze the coding structure of nucleotide sequences to mine the classification information of microorganisms. However, the lengths of nucleotide sequences for microorganisms are various. It is difficult to directly compare the complete nucleotide sequences among microorganisms. If we would like to compute a DNA sequence at once, transformation of nucleotides will be a necessary action to transfer the DNA sequence into a numerical form with fixed length. In order to systematically transform DNA sequences, some researchers have investigated the method of power spectrums of nucleotides in DNA sequences, and reveals useful information and structures of genomes [7-9]. For this reason, it may carry out microorganism classification based on the power spectrums of complete DNA sequences. Since biological organisms may differ on the spectral densities of nucleotides that reflect the topological state of genomic DNA, a procedure for classifying microorganisms will be viewed as a meaningful issue to be studied. Some researchers [10,11] have provided a simple classification of archaea and bacteria based on the discussions of peak intensities at 10-11 bp periodicities in the nucleotide spectrums. The results of those studies roughly show classification of bacteria can be discriminated by comparing the differences among 10-11 bp periodicities in the power spectrums of nucleotide sequences. However, those processes of bacterial classification are rather heuristic than systematic. To discriminate or categorize microorganisms, SOM may be helpful here, because it can partition data into arbitrary groups without any a-priori knowledge or statistical assumption and is able to evaluate the distances among data and groups to visually investigate the similarity. And therefore, in this study, an integrated procedure, which provides an innovative method for microorganism classification, based on the combination of Fourier transform for DNA sequences and Self-Organizing Map to verify the classification of microorganisms and visually map the relationships among microorganisms.

2 Fourier Transform for DNA Sequences

A DNA sequence can be regarded as a combination of 4 nucleotides: A (adenine), C (cytosine), G (guanine), and T (thymine), and hence a DNA sequence with N nucleotides can be initially represented as a $1 \times N$ vector

$$D = [w_k] \ ; \mathrm{k} = 0 \ldots N-1 \tag{1}$$

where w_k denotes the symbol of the k*th* nucleotide. Furthermore, D can be translated to numerical form E_d by

$$E_d = [s_{dk}] \ ; s_{dk} = 1 \text{ if } w_k = d, \text{ or } s_{dk} = 0; \ \mathrm{k} = 0 \ldots N-1 \text{ and } d \in \{\mathrm{A}, \mathrm{C}, \mathrm{G}, \mathrm{T}\}. \tag{2}$$

Based on Eq. (2), it will be easy to obtain E_{A}, E_{C}, E_{G} and E_{T} for nucleotides A, C, G and T respectively [12].

The power spectrums for the numerical representation of nucleotides E_d is defined as $P_d(f_j) = |V_d(f_j)|^2 = V_d(f_j)V_d(f_j)^*$, where

$$V_d(f_j) = \frac{1}{N}\sum_{k=0}^{N-1} s_{dk}\exp(-2\pi ikf_j) \tag{3}$$

is the Fourier transform of E_d, and the frequency f_j is calculated by

$$f_j = \frac{j}{N}, j = 0 \ldots N-1, \tag{4}$$

and the corresponding periodicity g_j for f_j is

$$g_j = \frac{1}{f_j}. \tag{5}$$

Instead of computing Fourier transform for a complete DNA sequence at once, a complete DNA sequence can be sliced into M small trunks with equal size of N, that can be determined according to the user's option with the consideration with power of 2, for the computation of Fourier transform. And thereafter averaged power spectrums can be computed from the M number of $[P_d(f_j)]$, where $j = 0 \ldots N-1$. Based on the characteristics of Fourier transform, $[P_d(f_j)]$ will be an N-dimension symmetric vector. In practice, we also take $N/2$ scalars, $P_d(f_1) \ldots P_d\left(f_{\frac{N}{2}}\right)$, from $[P_d(f_j)]$ for the computation of averaged power spectrums. Note that the first scalar $P_d(f_0)$ is discarded because $f_0 = 0$ and g_0 will be undefined.

According to Eq. (1), for example, the vector D for an organism can be represented as D = [A, G, C, T, T, T, T, C, A, …, A, G, T, G, A, T, T, T, T, C]. Then the vector D for the organism can be further transformed to a binary form E_A by applying Eq. (2) and represented as E_A = [1, 0, 0, 0, 0, 0, 0, 0, 1, …, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0].

To verify the rationality and feasibility for the proposed procedure, we took data (it can be obtained from www.ncbi.nlm.nih.gov) to perform the analysis. As listed in Table 1, the sample materials consist of the complete DNA sequences from the 12 bacteria belonging to Bacillus genus, which consists of bacteria that usually cause acute fatal disease.

In this study, we took N = 2048 and consider a sequence if it has a length of 2048 bp or larger. Then, we use N=2048 to slice a complete DNA sequence into M small pieces. Then, the M slices of the DNA sequence were averaged. Let p_{dn} denotes the average power spectrum of nucleotide d for the n*th* organism in Table 1. The matrix of average power spectrum for the 12 organisms can be obtained by the Eq. (6)

$$Q_d = \lfloor p_{dn}^T \rfloor \tag{6}$$

Table 1. DNA sequences of 12 bacteria belonging to Bacillus genus

No.	Bacteria	Label	Accession No.	Bases (bps)
1	Bacillus anthracis A2012	B1	AAC01000001	5093554
2	Bacillus cereus ATCC 14579	B2	AE016877	5411809
3	Bacillus anthracis str. Ames	B3	AE016879	5227293
4	Bacillus cereus ATCC 10987	B4	AE017194	5224283
5	Bacillus anthracis str. Sterne	B5	AE017225	5228663
6	Bacillus licheniformis ATCC 14580	B6	AE017333	4222645
7	Bacillus anthracis str. 'Ames Ancestor'	B7	AE017334	5227419
8	Bacillus thuringiensis serovar konkukian str. 97-27	B8	AE017355	5237682
9	Bacillus subtilis subsp. Subtilis str. 168	B9	AL009126	4214630
10	Bacillus clausii KSM-K16	B10	AP00627	4303871
11	Bacillus halodurans C-125	B11	BA000004	4202352
12	Bacillus cereus E33L	B12	CP000001	5300915

Because the power spectrums converted from Fourier transform is symmetric, which means that we do not have to use the all 2048 power spectrums, the matrix Q_d shall be a 1024×12 matrix with the consideration of half number of power spectrums. After Q_d is obtained, the visual representation of Q_d is constructed.

3 Self-Organizing Map and Data Visualization

As shown above, using Fourier transform DNA sequences may be converted to multi-dimensional and numerical data. For a better understanding of differences among the 12 organisms, SOM is performed on each p_{dn}, which denotes the averaged power spectrums of nucleotide d for the nth organism in Table 1. SOM may be helpful here, because it can partition data into arbitrary groups without any a-priori knowledge or statistical assumption and is able to evaluate the distances among data and groups to visually investigate the similarity. On the other hand, since the topic about SOM is widely known in the field of ANN research and the general procedure for performing SOM can be found in Kohonen's publications as well as almost every book on ANN [13], we are not going to discuss details about the implementation of SOM in this study.

Despite the ability of categorizing data into clusters, the nature of SOM is to project multi-dimensional data onto a two dimensional map. For displaying the two dimensional map of SOM, a U-matrix [14] is an enhanced visualization tool that not only projects high dimensional data onto a two dimensional map but also represents distance structure and topological relationships among data. In a U-matrix the distance or similarity measurement between two adjacent neurons is obtained and the distance between two adjacent neurons is presented in the U-matrix. Usually, the distance or similarity measurement is defined as a Euclid distance. For a SOM with the topology of hexagonal $m\times n$ output neurons, the dimension of U-matrix is (2m-1) $\times$ (2n-1). Figure 2 shows the structure and representation of a U-matrix. In Fig. 1, d(A, B), d(A, C), and d(B, C) are the distance or similarity measures between neurons. Practically, using colors or grayscales to differentiate the lengths of distance, we can visualize the topological structure and relationships among the neurons, and therefore help find the meaningful clusters.

In order to construct the graph of U-matrix, software package developed by Vesanto and his colleagues [15] for the calculations and visualization of SOM is utilized in this study. This software package is a toolbox that implements SOM and its visualization in Matlab scientific computing environment. In the next section, the proposed methods and tools utilized in this study are combined to differentiate the microorganisms of Bacillus genus.

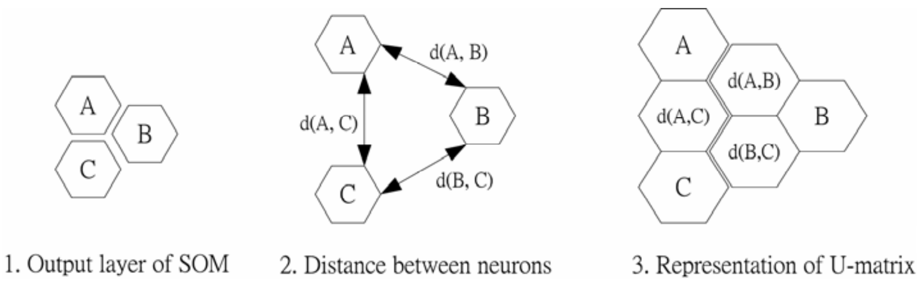


Fig. 1. The construction of a U-matrix for representing distance structure and topological relationships in a SOM

4 Results

In a DNA sequence, the base pairs A-T (adenine-thymine) and C-G (cytosine-guanine) are the widely known fact that the pair of nitrogenous bases connects the complemen-tary strands of DNA. In other words, the distribution and construction of nucleotides A should be identical to nucleotides T. Also, the distribution and construction of nucleotides C should be identical to nucleotides G. In other words, we don't need to observe the distributions of all 4 nucleotides. In this study, we chose A and C to demonstrate visuals. With the proposed procedures to transform the 12 DNA sequences into numerical forms, SOMs with topology of 3×4 hexagonal lattices are applied to the numerical forms of DNA sequences in order to investigate the similarity among the 12 microorganisms. Fig. 2 and 4 represent Q_A and Q_C, which illustrate visualizations for the power spectrums of nucleotides A (adenine) and C (cytosine) on the 12 microorganisms of Bacillus genus, respectively. Note that the grayscale bars located at the right side of Fig. 2 and 4 show the range of the measurements. These diagrams can visually help recognize the structural differences among the 12 microorganisms.

Fig. 3 and 5 show the distance structure and clustering relationships of nucleotides A and C on the 12 microorganisms. Note that white circles on the U-matrix graphs represent neurons that they are possible clustering cells (centers) for the 2 different types of nucleotides. Numbers marked in the cells of 2 U-matrix graphs demonstrate the distance measures among neurons, and the grayscale bars located at the right side of U-matrix graphs show the visual measurements for the graphs. As shown in the U-matrix graphs, the darker hexagons represent shorter distance among neurons. Based on the U-matrix visualization, brighter or lighter the hexagons imply higher the 'walls' that usually divide data into clusters. And therefore, we draw some dot lines to help distinguish areas separated by 'walls'.

5 Discussions

In Fig. 3, the suggested clusters provided by the U-matrix of nucleotide A can be concluded that B1, B2, B3, B4, B5, B7, B8, and B12 form a cluster, B6 and B9 form the second cluster, and B10 and B11 form the third cluster. Beyond that, in Fig. 5, which is the U-matrix of nucleotide C, provides slightly different clustering relationships. From right to left, we can observe that the hexagons are partitioned into 4 major regions. B1, B2, B3, B4, B5, B7, B8, and B12 are in a region that the measures of distance are relatively small. Then, as mentioned above, hexagons colored with light gray imply that there are 'walls' that divide data into clusters. In Fig. 5, B11 is separated from the cluster formed by B6, B9, and B10.

Apparently, no matter which nucleotide is applied to the method proposed in this paper, the U-matrices of nucleotides A and C, all suggest that bacteria B1, B2, B3, B4, B5, B7, B8, and B12 are very similar to each other. This result is corresponding to recent biological findings that these 8 microorganisms are members of the Bacillus cereus group of bacteria and have been shown that Bacillus anthracis and Bacillus thuringiensis should be considered as a direct lineage of Bacillus cereus by some biological and chemical analysis [16]. In particular, bacteria B6 and B9, which are Bacillus licheniformis and Bacillus subtilis, respectively, are close to each other based on the U-matrices of nucleotides A and C. Rey and his colleague [17] suggested that the two bacteria have extensive organizational similarity and should be considered that the two bacteria are in the same evolutional lineage.

Although in Fig. 3 the bacteria Bacillus clausii (B10) and Bacillus halodurans (B11) are categorized in the same cluster, in Figure 5 the two bacteria are relatively different. Based on the analysis in nucleotides C, Bacillus clausii and Bacillus halodurans seem to be two different species. Yet we find no biological research papers that consider putting Bacillus clausii and Bacillus halodurans together to investigate their evolutional relationships. Even then, it is noticeable that Bacillus clausii and Bacillus halodurans are relatively close to each other based on the analysis of nucleotide A. And, based on the analysis of nucleotidee C, Bacillus clausii (B10) seems to be close to Bacillus licheniformis (B6) and Bacillus subtilis (B9). Anyway, we found that clustering results for Bacillus clausii and Bacillus halodurans make unclear or incomprehensible. We suggest that the relationships between the two bacteria need to be studied further in some biological methods.

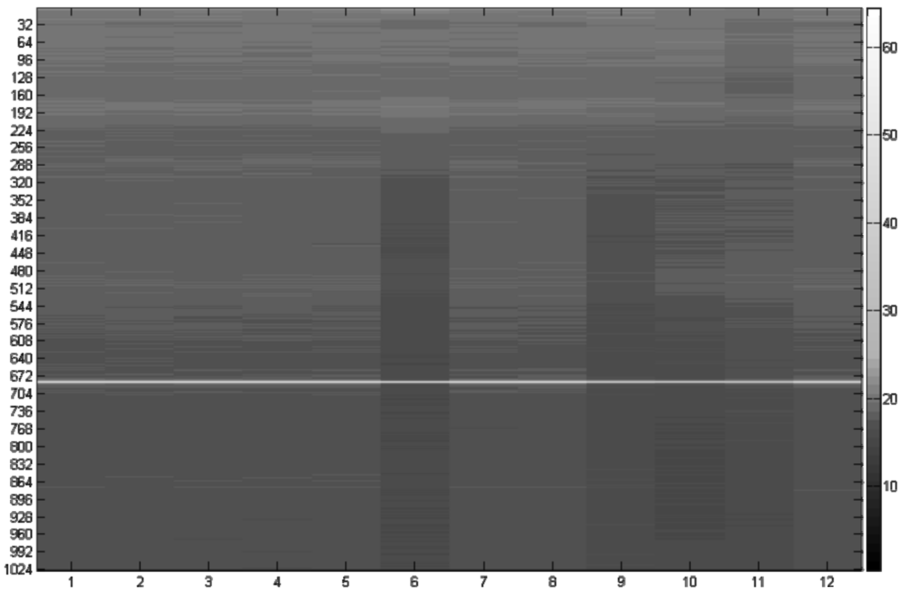


Fig. 2. Power spectrums for nucleotide A (adenine) on the 12 microorganisms of Bacillus genus

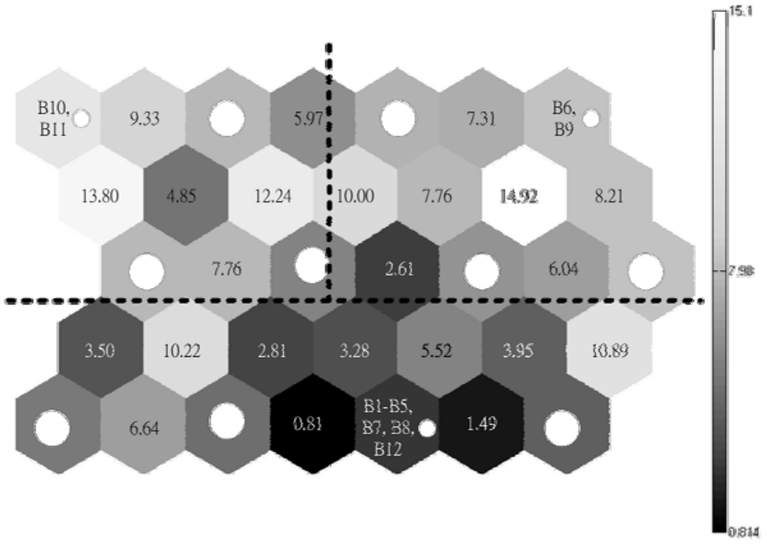


Fig. 3. U-matrix of nucleotide A for representing distance structure and clustering relationships of the 12 microorganisms

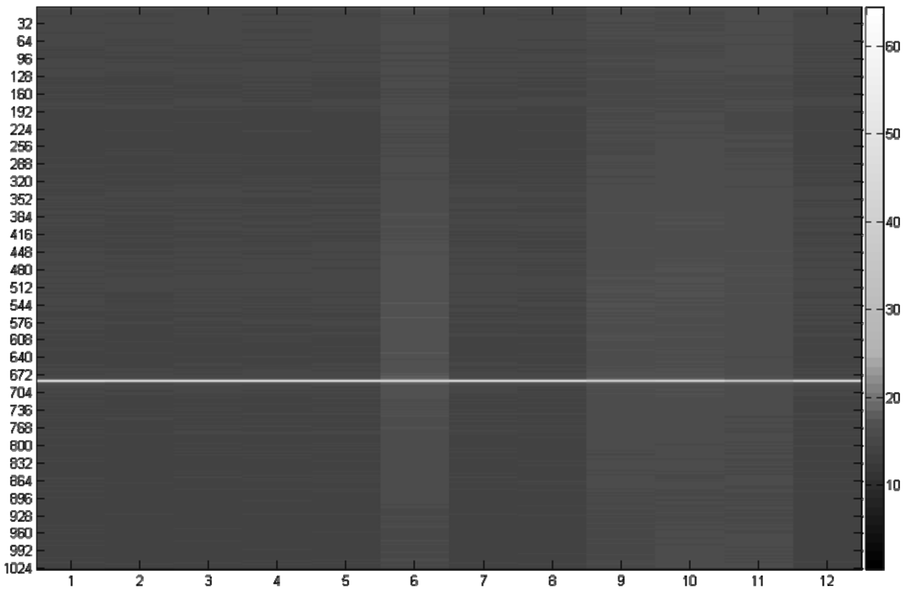


Fig. 4. Power spectrums for the nucleotide C (cytosine) on the 12 microorganisms of Bacillus genus

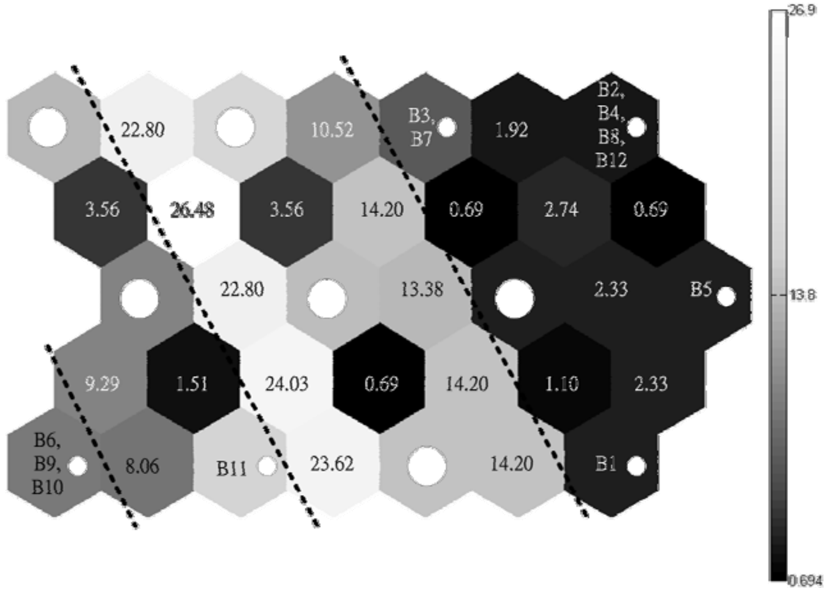


Fig. 5. U-matrix of nucleotide C for representing distance structure and clustering relationships of the 12 microorganisms

6 Conclusions

In this article, we proposed a method that combines Fourier transform and SOM to achieve the clustering analysis for microorganisms. Besides, a meaningful contribution, which is the clustering analysis of biological organisms, can be made by using Fourier transform and SOM to a complete DNA sequence without biochemical or biological procedures. In this study we found some significance. It will be given as follows: (1) The technique of power spectrum analysis effectively shrinks the dimensions of complete bacteria DNA sequences for the proposed procedure. (2) Instead of observing the spectral densities with traditional wave-like diagrams, Figures as Fig. 2 and 4 can provide an aspect that we can observe the distributions among the power spectrums of complete DNA sequences via a bird's eye view, which may be more intuitive and effective. (3) The taxonomic relationships among different microorganisms of Bacillus genus we demonstrated in this paper are comparable to biological research findings.

Acknowledgments. This research was supported by the National Science Council of Taiwan under contract number NSC 94-2416-H-143-001and NSC 94-2112-M-143-001.

References

1. Delong, E.F., Pace, N.R.: Environmental Diversity of Bacteria and Archaea. Systematic Biology 50 (2001) 470-478
2. Zuckerandl, E., Pauling, L.: Molecules as Documents of Evolutionary History. Journal of Theoretical Biology 8 (1965) 357-366
3. Woese, C.R., Fox, G.E.: Phylogenetic Structure of the Prokaryotic Domain: The Primary Kingdoms. Proceedings of the National Academy of Sciences of the United States of America 74 (1977) 5088-5090
4. Woese, C.R., Kandler, O., Wheelis, M.L.: Towards a Natural System of Organisms: Proposal for the Domains Archaea, Bacteria, and Eucarya. Proceedings of the National Academy of Sciences of the United States of America 87 (1990) 4576-4579
5. Mayr, E.: Two Empires or Three. Proceedings of the National Academy of Sciences of the United States of America 95 (1998) 9720-9723
6. Doolittle, W.F.: Phylogenetic Classification and the Universal Tree. Science 284 (1999) 2124-2128
7. Chen, Y.H., Nyeo, S.L., Yu, J.P.: Power-Laws in the Complete Sequences of Human Genome. Journal of Biological Systems 13 (2005) 105-115
8. De Sousa Vieira, M.: Statistics of DNA Sequences: A Low-Frequency Analysis. Physical Review E 60 (1999) 5932-5937
9. Isohata, Y., Hayashi, M.: Analyses of DNA Base Sequences for Eukaryotes in Terms of Power Spectrum Method. Japanese Journal of Applied Physics 44 (2005) 1143-1146
10. Fukushima, A., Ikemura, T., Kinouchi, M., Oshima, T., Kudo, Y., Mori, H., Kanaya, S.: Periodicity in Prokaryotic and Eukaryotic Genomes Identified by Power Spectrum Analysis. Gene 300 (2002) 203-211
11. Nyeo, S.L., Yang, I.C., Wu, C.H.: Spectral Classification of Archaeal and Bacterial Genomes. Journal of Biological Systems 10 (2002) 233-241

12. Voss, R.F.: Evolution of Long-Range Fractal Correlations And 1/f Noise in DNA Base Sequences. Physical Review Letter 68 (1992) 3805-3808
13. Kohonen, T.: Self-Organizing Maps. 3rd edn. Springer-Verlag, Berlin Heidelberg New York (2001)
14. Vesanto, J.: SOM-Based Visualization Methods. Intelligent Data Analysis 3 (1999) 111-126
15. Vesanto, J., Himberg, J., Alhoniemi, E., Parhankangas, J.: Self-Organizing Map in Matlab: The SOM Toolbox. Proceedings of the Matlab DSP Conference. Espoo, Finland (1999) 35-40
16. Helgason, E., Økstad, O.A., Caugant, D.A., Johansen, H.A., Fouet, A., Mock, M., Hegna, I. Kolstø, A.-B.: Bacillus Anthracis, Bacillus Cereus, and Bacillus Thuringiensis--One Species on the Basis of Genetic Evidence. Applied and Environmental Microbiology 66 (2000) 2627-2630
17. Rey, M.W. et al.: Complete Genome Sequence of the Industrial Bacterium Bacillus Licheniformis and Comparisons with Closely Related Bacillus Species. Genome Biology 5 (2004) Article R77

Recurrence Quantification Analysis of EEG Predicts Responses to Incision During Anesthesia

Liyu Huang[1,2], Weirong Wang[1,3], and Sekou Singare[2]

[1] Department of Biomedical Engineering, Xidian University, Xi'an, 710071, China
[2] Institute of Biomedical Engineering, Xi'an Jiaotong University, Xi'an, 710049, China
huangly@mail.xjtu.edu.cn
[3] Department of Medical Instrumentation, Shanhaidan Hospital, Xi'an, 710004. China
owl@mail.xidian.edu.cn

Abstract. The need for assessing the depth of anesthesia during surgical operations has existed since the introduction of anaesthesia, but sufficiently reliable method is still not found. This paper presents a new approach to detect depth of anaesthesia by using recurrence quantification analysis of electroencephalogram (EEG) and artificial neural network(ANN). The EEG recordings were collected from consenting patient prior to incision during isoflurane anaesthesia of different levels. The four measures of recurrence plot were extracted from each of eight-channel EEG time series. Prediction was made by means of ANN. The system was able to correctly classify purposeful responses in average accuracy of 87.76% of the cases.

1 Introduction

Anesthesia has been defined as the state in which, as a result of a drug-induced unconsciousness, the patient neither perceives nor recalls noxious stimulation. Therefore, an adequately anesthetized patient is unconscious, analgesic, and amnesic. In operation, depth of anesthesia(DOA) is always assessed by anaesthesiologists subjectively by means of observing the blood pressures(BP), heart rate(HR), somatic movement, facial grimacing, lacrimation and diaphoresis. The presence of some certain signs, such as movement, a rise in blood pressure, tearing, sweating, techycardia, pupillary dilation, indicates the inadequate level of anaesthesia. However, these clinical signs may be absent due to the use of some drugs, such as muscle relaxants, opioids, cholinergic and β-adrenergic antagonists, vasodilators, and antihypertensive agents. For lack of a reliable monitor of anesthetic depth, inadequate levels of anaesthesia occasionally occur, the overall incidence of recall most frequently stated is approximately 1%, cases of intraoperative awareness have been reported in the literature[1][2]. A new non-invasive monitoring technique for detection of DOA would be extremely valuable.

Since anesthesia can directly affect the brain's function, the central nervous system has been the center of attention during anaesthetic course, so analysis of the brain's neuroelectrical signals, such as electroencephalogram (EEG), seems to be a good choice to look for a correlation between the states of anesthesia and some characteristic parameters of the signals. Over the years, numerous efforts have been made by EEG analysis, such as time/frequency domain analysis[3-5], bispectral analysis[6,7], time-frequency distribution (wavelet transform)[8], however, the use of EEG as a measure of adequacy of anesthesia has achieved limited success.

In this paper, a new method is introduced to monitor DOA by using recurrence quantification analysis(RQA) of EEG and artificial neural network(ANN). The rest of this paper is organized as follows: the clinical EEG data acquisition under different levels of anesthetic concentrations is described first. And then, the reseach method is introduced. After a short review of the basic concept of recurrence plot (RP), some measures of the complexity is defined based on RP. After that we apply a four-layer ANN as a non-linear identifier for estimating DOA, according to the extracted measure values from RP of the EEGs. Clinical experiments were used to validate the DOA estimator design in real-time.

2 Materials and Data Acquisition

The data were obtained from experiments on 98 consenting patients(ASA grade I~II). The patients, 56 men and 42 women, ranging in age from 26 to 63 years(37.8±9.3), were scheduled for elective noncranial surgery. Patients must be excluded from the study if they had a history of hypertension, seizure disorder, or other chronic neurologic or psychiatric illness, or if they were taking antihypertensive medications.

Anesthesia was induced with propofol, followed by a muscle relaxant to facilitate intubation, and maintained with isoflurane in oxygen. The traditional physiological signs, electrocardiogram (ECG), the arterial blood pressure (BP), heart rate (HR), arterial oxygen saturation, End-tidal CO_2($EtCO_2$, by capnograph) were monitored and all gases(carbon dioxide, isoflurane, oxygen, nitrous oxide) were continuously monitored by mass spectrometer in all the experiments. The arterial pressure and heart rate were measured every minute non-invasively using a Dinamap monitor(Critikon). ECG was monitored using a Mennen Horizon monitor(USA).

A definition of the level of anesthesia from non-EEG criteria is very necessary for our study. During and after skin incision, which is a standard stimulation to test anaesthitic effect, each patient was observed carefully for 2 *min* to detect purposeful movement, changes in hemodynamic parameters and respiratory pattern. A gross purposeful movement, usually of the head and limbs, or grimace was considered a positive response, the EEG recording preceding the incision was then labelled as 0.0 for the responder. The response was also estimated as a positive one, in any two of the following three cases, when there were no movement and grimace, but

(1) a significant HR response (rise was greater than 15%);
(2) a significant BP response(rise was greater than 15%);
(3) a spontaneous change in respiratory pattern(change in $EtCO_2$ was more than 12mm Hg).

where the response was negative one, the EEG recording preceding the incision was labelled as 1.0 for non-responder.

The initial surgical incision was made after return of neuromuscular function as determined by a peripheral nerve stimulator and at least 30-*min* after induction of anesthesia, and after at least 10 *min* at a different end-tidal isoflurane concentrations of 1.0MAC(37 patients), 1.2MAC(33 patients) or 1.4MAC(28 patients).

The EEG was monitored using a HXD-I(E) (China) monitor for 2 *min* before incision. Eight channels of EEG were recorded using Ag/AgCl electrodes (Fp1, Fp2, C3, C4, T3, T4, O1 and O2, they were placed in accordance with the international standard 10~20 system, with the reference points at left earlobe and right earlobe). The EEG data were filtered with a high-pass filter at 0.4*Hz* and a low-pass filter at 70*Hz*, a 50*Hz* notch filter was also employed, and sampled at 250*Hz*, digitized to 12bits. The impedance on each of the EEG channel was required to be less than $2K\Omega$ at the start of each experiment. From 98 patient experiments, 98 distinct EEG recordings were collected prior to incision during different isoflurane anaesthesia levels, including 31 responder and 67 non-responder recordings. Each EEG recording was 2 *min* long(30,000 points).

3 Method

The proposed approach includes three different procedures. First, RPs are computed from each of eight-channel EEG time series, and then the complexity measures are extracted from every RP map. At last, prediction was made by ANN.

3.1 Recurrence Plot

A recurrence plot (RP) is a two-dimensional squared matrix with black and white dots and two timeaxes, where each black dot at the coordinates (t_1, t_2) represents a recurrence of the system's state $\vec{x}_i(t_1)$ at time t_2:

$$R(t_1,t_2)=\Theta(\varepsilon-\|\vec{x}(t_1)-\vec{x}(t_2)\|),\quad \vec{x}(t)\in\Re^m \tag{1}$$

where m is the dimension of the system (degrees of freedom), ε is a small threshold distance, $\|\cdot\|$ is a norm (e.g., the Euclidean norm) and $\Theta(x)$ is the Heaviside function[9].

3.2 Recurrence Quantification Analysis

Recurrence plots contain subtle patterns that are not easily ascertained by qualitative visual inspection. Zbilut and Webber have presented the recurrence quantification analysis (RQA) to quantify an RP [10], they define quantitative parameters using the recurrence point density and the diagonal structures in the recurrence plot, and Marwan et al. extend the definition on the vertical structures[11]. In our study, four of them are chosen to analyze. These measures are described briefly as follows:

1) *Recurrence rate(RR)*, quantifies a percentage of the plot occupied by recurrent points,

$$RR=\frac{1}{N^2}\sum_{i,j=1}^{N}R_{i,j} \tag{2}$$

2) *Determinism(DET)*, quantified a percentage between the recurrent points that form upward diagonal line segments and the entire set of recurrence points.

The diagonal line *l* consists of two or more points that are diagonally adjacent with no intervening white space, *P(l)* denote the number of lines with length *l,*

$$DET = \frac{\sum_{l=l_{\min}}^{N} lP(l)}{\sum_{i,j}^{N} R_{i,j}} \tag{3}$$

3) *Laminarity(LAM)*, quantified a percentage between the recurrence points forming the vertical line structures and the entire set of recurrence points, *P(v)* denote the number of lines with length *v,*

$$LAM = \frac{\sum_{v=v_{\min}}^{N} vP(v)}{\sum_{i,j}^{N} R_{i,j}} \tag{4}$$

4) *V-Entropy (V-ENTR)*, quantified a Shannon entropy of vertical line segment distributions,

$$V - ENTR = -\sum_{v=v_{\min}}^{N} p(v)\log_2 p(v) \tag{5}$$

where *p(l)* represents probabilities distribution of vertical line structures. the base of *log* here is taken to two, so *L-ENTR* and *V-ENTR* are in unit of bit.

A computation of RPs and these measures in small EEG windows moving along the recording time, using these values as the inputs to ANN, makes the real-time evaluation of DOA possible.

3.3 Artificial Neural Networks

In our study, the input to the ANN will be the 32 measure values(4 from one-channel EEG), extracted from the eight-channel corresponding EEG recordings with label 1.0 (for nonresponder) or 0.0 (for responder). Compared with three-layer(one hidden layer) ANNs, the four-layer ANN(two hidden layers) has a certain advantage in estimating DOA(see Table 2). The number of the second hidden units and the optimum number of clusters are determined according to analysis of input and output feature space[12, 13] and pseudo F-statistic(PFS) clustering technique[14]. The optimum structure of ANN is determined as 32-9-2-1. We build up the network in such a way that each layer is fully connected to the next layer.

Training and test of ANN used ‘leave-one-out’ strategy. That is, to train the ANN on samples from *n–1* rats and test on samples from the remaining one. This process is repeated *n* times and each time a different rat is left behind [15].

4 Results

The RP of the EEG recorded was computed and mapped for each patient with a time series of 512 points. Fig 1(a)~(c) show the RPs of three reponders with 1.0, 1.2, 1.4 MAC, respectively. (d)~(f) show the RPs of three non-reponders with 1.0, 1.2, 1.4 MAC, respectively.

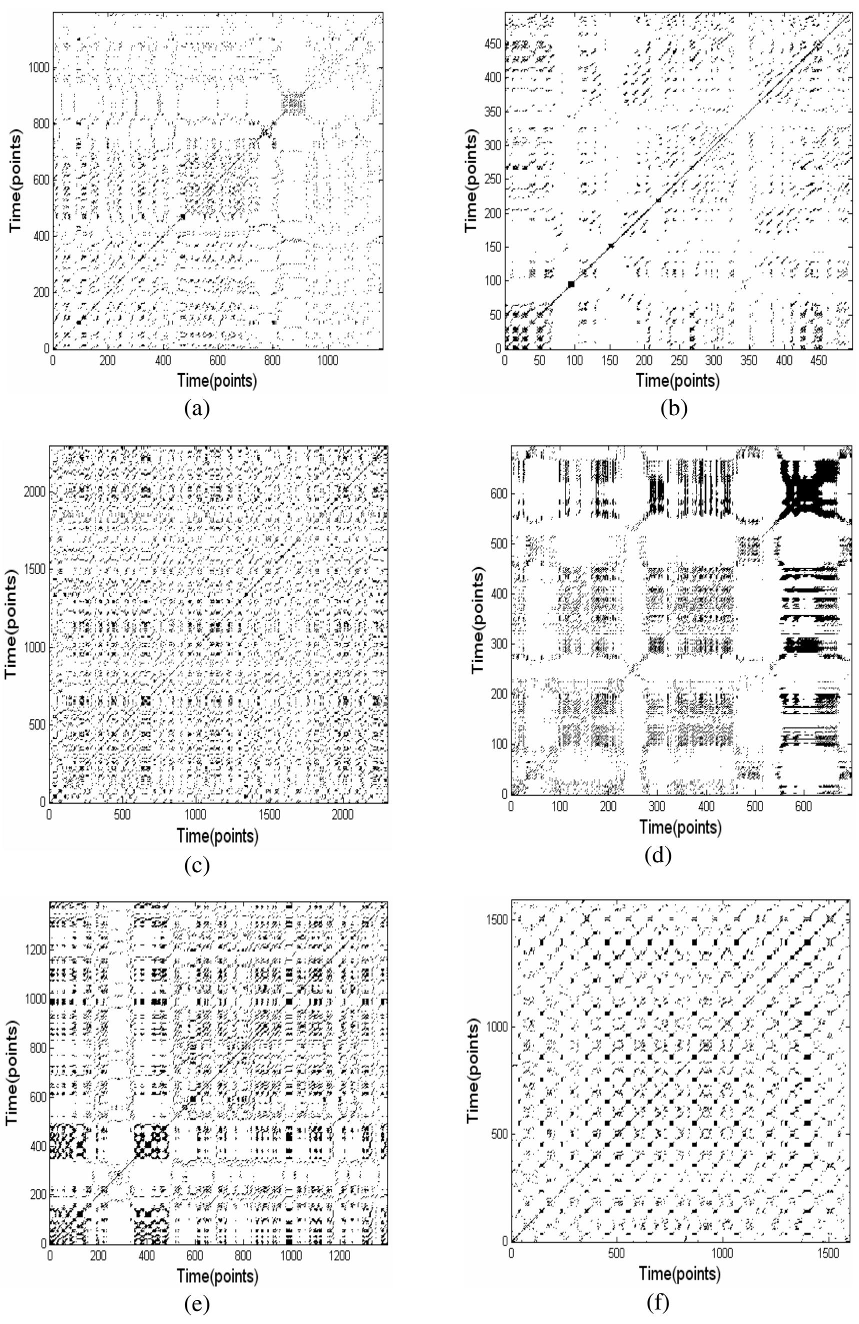


Fig. 1. RP of typical patients, including three responders with 1.0 MAC (a), 1.2 MAC (b), 1.4 MAC (c), and three non-responders with 1.0 MAC (d), 1.2 MAC (e), 1.4 MAC (f)

The results of testing our proposed system are shown in Table 1. In total, five response states are misclassified as non-response, and seven non-response states are misclassified as response. For response prediction, the average accuracy is 87.76%. We tested our system using different ANNs, for four-layer (32-9-2-1) and three-layer (32-9-1) ANNs, the accuracy is 87.76% and 83.67%, respectively(see Table 2).

It would be informative to briefly compare our system with the commonly used spectral analysis methods, such as the bispectral analysis (BIS)[6, 7], the spectral edge frequency (SEF) and the median frequency (MF) [16, 17]. For different prediction schemes, the comparison of performances are shown in Table 2.

Table 1. Testing results for proposed system, which employs four-layer ANN

Level of anesthesia	Size of training set	Number of patients	Misclassified rate for response state	Misclassified rate for non-response state	Accuracy
1.0MAC	97	37	4/21	2/16	83.78%
1.2MAC	97	33	1/7	3/26	87.88%
1.4MAC	97	28	0/3	2/25	92.86%

Table 2. Comparison of performances of different prediction schemes

Predicting schemes	Other conditions	Types of ANN employed	Average accuracy
RQA using ANN	98 patients and 98 EEG recordings at different anesthetic levels	Four-layer ANN (32-9-2-1)	87.76%
		Three-layer ANN (32-9-1)	83.67%
Spectral analysis[a]	BIS		83%
Spectral analysis[a]	BIS		84%
	SEF		79%
	MF		68%

[a] These results are cited from the references, paper [7] and [18], respectively, compared with our conclusion.

The accuracy for BIS, SEF and MF methods is 84%, 79% and 68%, respectively[7]. Another accuracy 83% was given out by another researcher, Sebel[18], also using bispectrum method. Some of the results are cited from the references (paper [7] and [18]) for comparison.

5 Discussion

RP is a valuable graphical tool for assessing the geometry of the dynamics exploiting non-linear dependencies specially in non-stationary time-series, therefore, it is particularly useful in the analysis of physiological data. These plots disclose distance relationships between points on a dynamical system providing a faithful representation of the time dependencies (correlations) contained in the data[9].

The passed studies show that adequacy of anesthesia is a complicated concept and it is difficult to accurately evaluate DOA by a single parameter[19]. The recurrence plot represents the recurrence of the phase space trajectory to a certain state, which is a fundamental property of deterministic dynamical systems[20, 21]. The chaotic or quasi-random behavior of EEG signals gives us the possibility to quantitatively study functional changes of the brain by means of RQA. In this paper, we propose a new approach to predict response to incision during anaesthesia using RQA. Compared with other schemes, our designed system has a better performance and the RQA is a potential way to assess the anaesthetic adequacy. The reason for this maybe is that our scheme combines the different nonlinear methods and enables the system to address the non-analytical, non-stationary, non-linear dynamical properties of the EEG. Our studies further indicate that, in most cases, the EEG contains sufficient information to estimate the DOA, the key is whether or not the method used is suited to the nature of the EEG signal properly.

The anesthetic dose versus the depth of anesthesia curve is highly nonlinear [22] which motivated the use of NN's as the basic classifiers in our scheme. Owing to the restricted experimental conditions, we have only 98 recordings from 98 patients for training and testing ANN. A good way to tackle such a dilemma would be to use the 'leave-one-out' method [15], it is a standard method to evaluating classification systems, especially in the case of small samples for training and test. As the training and test samples belong to different patents, there is not any bias in the results.

In summary, this study shows that recurrence quantification analysis of EEG is a good candidate for characterizing patients' brain activity under different depths of anesthesia. The measures will help us to better understand the non-linear dynamic character of brain's consciousness during anesthesia and enhance our ability to assess the DOA. Although the results of this initial study are significant, nevertheless, the work reported here is still preliminary. Our clinical experimental scheme may be further improved and our study has been limited to consider only isoflurane anaesthesia. Additional clinical studies need to be carried out on a wide patient population to further evaluate the proposed design, especially with different inhalational anaesthetic agents and with opioid anaesthesia, where awareness during surgery is a major concern.

Acknowledgment

This work was supported by China Postdoctoral Science Foundation and National Natural Science Foundation of China under grant NO. 60371023.

References

1. Moerman N., Bonke, B., Oosting, J.: Awareness and recall during general anesthesia: facts and feelings. Anesthesiology, 79 (1993) 454–464
2. Ghoneim, M.M., Block, R.L.: Learning and consciousness during general anaesthesia. Anesthesiology, 76 (1992) 279–305
3. Rubin, M.A., Freeman, H.: Brain potential changes in man during cyclopropane anesthesia. J. Neurophysiol. 3 (1940) 33–42
4. Thomsen, C.E., Christensen, K.N., Rosenflack, A.: Computerized monitoring of depth of anaesthesia with isoflurane, Br. J. Anaesthesia. 63(1989) 36–43
5. Sharma, A., Roy, R.J.: Design of a recognition system to predict movement during anaesthesia. IEEE Trans. on Biomed. Eng. 44(1997) 505–511
6. Kearse, L. A., Manberg, P., DeBros, F., (eds.): Bispectral analysis of the electroencephalo-gram during induction of anesthesia may predict hemodynamic responses to laryngoscopy and intubation. Electroencephalography and clinical Neurophysiology, 90(1994) 194–200
7. Vernon, J. M., Lang, E., Sebel, P. S., (eds.): Prediction of movement using bispectral electroencephalographic analysis during propofol/alfentanil or isoflurane/alfentanil anesthesia. Anesth. Analg. 80 (1995) 780–785
8. Nayak, A., Roy, R. J., Sharma, A.: Time-frequency spectral representation of the EEG as an aid in the detection of depth of anaesthesia. Ann. Biomed. Eng. 22(1994) 501–513
9. Eckmann, J. P., Kamphorst, S. O., Ruelle, D.: Recurrence plots of dynamical systems. Europhys. Lett. 4 (1987) 973–977
10. Zbilut, J. P., Webber, Jr. C.L.: Embeddings and delays as derived from quantification of recurrence plots. Phys. Lett. A 171 (1992) 199–203
11. Marwan, N., Wessel, N., Meyerfeldt, U., (eds.): Recurrence-plot-based measures of complexity and their application to heart-rate-variability data. Phys. Reviw. E 66(2002) 026702
12. Lippmann, R.P.: An Introduction to Computing with Neural nets. IEEE ASSP Magazine, April (1987) 4–22
13. Mirchandami, G., Cao, W.: On hidden nodes for neural nets. IEEE Trans. on Circuits and System, 36(1989) 661–664
14. Vogel, M. A., Wong, A. K. C., PFS clustering method. IEEE Trans. on Pattern Anal. Mach. Intell. 1(1979) 237–245
15. Fukunaga, K.: Introduction to Statistical Pattern Recognition. 2nd edition, Academic, San Diego, CA (1990)
16. Drummond, J. C., Brann, C. A., Perkins, D. E., (eds.): A comparison of median frequency, spectral edge frequency, a frequency band power ratio, total power, and dominance shift in determination of depth of anaesthesia. Acta Anaesthesiologica Scand.35(1991) 693–699
17. Rampil, I. J., Matteo, R. S.: Changes in EEG spectral edge frequency correlate with the hemodynamic response to laryngoscopy and intubation. Anesthesiology, 67(1987) 139–142
18. Sebel, P.S., Bowles, S.M., Saini, V., (eds.): EEG bispectrum predicts movement during thiopental/isoflurane anaesthesia. J. Clin. Monit. 11(1995) 83–91
19. Linkens, D. A.: Adaptive and intelligent control in anesthesia. IEEE Contr. Syst. Technol. Dec. (1992) 6–11
20. Argyris, J.H., Faust, G., Haase, M.: An Exploration of Chaos, North-Holland, Amsterdam (1994)
21. Ott, E.: Chaos in Dynamical Systems, Cambridge University Press, Cambridge (1993)
22. Quasha, A. L., Eger, E. I., Tinker, H. H.: Determination and applications of MAC, Anesthesiol. 53(1980) 315–334

Wavelet Spectral Entropy for Indication of Epileptic Seizure in Extracranial EEG

Xiaoli Li

Cercia, School of Computer Science, The University of Birmingham,
Edgbaston, Birmingham, B15 2TT, UK
xiaoli.avh@gmail.com

Abstract. Automated detection of epileptic seizures is very important for an EEG monitoring system. In this paper, a continuous wavelet transform is proposed to calculate the spectrum of scalp EEG data, the entropy and a scale-averaged wavelet power are extracted to indicate the epileptic seizures by using a moving window technique. The tests of five patients with different seizure types show wavelet spectral entropy and scale-averaged wavelet power are more efficiency than renormalized entropy and Kullback_Leiler (K-L) relative entropy to indicate the epileptic seizures. We suggest that the measures of wavelet spectral entropy and scale-averaged wavelet power should be contained to indicate the epileptic seizures in a new EEG monitoring system.

Keywords: Epileptic seizure; Wavelet analysis; Spectral entropy; EEG.

1 Introduction

It is known that continuous EEG recordings are widely applied to analyze / diagnose epilepsy patients [1], in particular automated detection of the epileptic seizures is very helpful and efficient for epilepsy diagnosis, unlike traditional methods that a trained technician reviews the entire EEG manually. Currently, various methods have been proposed, including linear methods [2], nonlinear methods [3-5], computational intelligence [6] and information theory [7]. The spectral analysis based method still is the most reliable for the detection of epileptic seizures. Since poor frequency resolution of the discrete Fourier transform (DFT), wavelet-based techniques are receiving growing interest given their ability to provide comparable spectral statistics that are local in time. At present, wavelet techniques become one of the most attractive methods to detect epileptic seizures [8-11]. Unfortunately, most of wavelet methods just focus on the energy information in the time frequency domain.

In this paper, spectral entropy of neural activity at the time –frequency domain is proposed to indicate the epileptic seizures. Traditionally, the entropy can characterize the degree of randomness of time sequence and quantify the difference between two probability distributions. Generally speaking, entropy is a measure of the uncertainty of a random process, or a complexity measure of a dynamical system [12]. In [7,13], renormalized entropy and Kulback-Leibler (K-L) entropy for EEG signals are proposed; it is found that there is a decrease of the renormalized and K-L entropy during the epileptic seizures. The following two disadvantages limit the detection of seizure. The spectral estimation with Fourier transform is possible to result in some of spouse information for a complex nonstationary EEG data. The second disadvantage of the renormalized entropy and K-L relative entropy is needed to determine a reference segment for each case. This paper presents wavelet spectral entropy and scale-averaged wavelet power to indicate the epileptic seizure. The five case studies showed that this method is better to indicate the epileptic seizures than renormalized and K-L entropy in EEG.

2 Methods

Wavelet analysis is a powerful tool to analyze EEG data. Projecting a time series into a time–scale space, the dominant modes of variability and its variation over time can be explored [14]. Previous works show Morlet wavelet transform can efficiently represent neural activity [15]. A Morlet wavelet function $\psi_0(t)$ is written as

$$\psi_0(t) = \pi^{-1/4} e^{i\omega_0 t} e^{-t^2/2} . \tag{1}$$

where ω_0 is the wavelet central angle frequency, often $\omega_0 \geq 6$, which is an optimal value to adjust the time – frequency resolution [16]. Given $\psi_0(t)$, a family of wavelet can be generated by a dilation, $\psi_s(t) = \frac{1}{\sqrt{2}} \psi_0(t/s), s \in (0,+\infty)$, s is called scale. Wavelet $\psi_s(t)$ can be taken as a parametric filter that is specified by the scale parameter s; and the duration of its impulse response increases with the increase of s.

Continuous wavelet transform (CWT) are performed through the convolution of a parent wavelet function $\psi_s(t)$ with the analyzed signal $x(t)$; it is:

$$W(s,\tau) = \frac{1}{\sqrt{s}} \int x(t) \psi_s^* (\frac{t-\tau}{s}) dt . \tag{2}$$

where s and τ denote the scale and translation; * denotes complex conjugation. By adjusting the scale s, a series of different frequency components in the signal can be extracted. The factor $\sqrt{s}$ is for energy normalization across the different scales. Through wavelet transforms, the information of the time series $x(t)$ is projected on the two dimension space (scale s and translation τ).

Given an EEG time series, X={x_n}, $n = 0 \ldots N$-1, with equal time spacing dt, the continuous wavelet transform of the discrete sequence is defined as the convolution of x_n with a scaled and translated version of $\psi_0(t)$; it is given by

$$W_s(n) = \sum_{n'=0}^{N-1} x_{n'} \psi_s * (\frac{(n'-n)dt}{s}) . \tag{3}$$

where * denotes the complex conjugate. Changing the wavelet scale *s* and translating along the localized time index n, a map can be constructed to show the amplitude of any feature versus the scale at a short time. Large values of the map (wavelet coefficients) reflect the combined effect of a large fluctuation of the time series and of a good matching of shape between the series and the wavelet function.

If a vertical slice through a wavelet plot is a measure of the local spectrum, the time-averaged wavelet spectrum over a certain period is

$$\overline{W_s}^2(n) = \frac{1}{n_a} \sum_{n=n_1}^{n_2} |W_s(n)|^2 . \tag{4}$$

where the index *n* is arbitrarily assigned to the midpoint of n_1 and n_2; and $n_a=n_2-n_1+1$ is the number of points averages over. By repeating (4) at each time step, we create a wavelet plot smoothed by a certain window. We define the global time-average of the instantaneous wavelet spectrum as follows:

$$I(s) = \overline{W}^2(s) = \frac{1}{N} \sum_{n=0}^{N-1} |W_s(n)|^2 . \tag{5}$$

Where *I*(*s*) is called wavelet spectrum, which is the expected value of the global time-average of the instantaneous power of $W_s(n)$. The entropy concept is applied to measure the fluctuation of wavelet spectrum, which is called wavelet spectrum entropy.

To examine fluctuations in power of a time series over a range of scales (a band), a scale-averaged wavelet power is defined as the weighted sum of the wavelet power spectrum over scales s_{j1} to s_{j2}:

$$\overline{W_n}^2 = \frac{d_j d_t}{C_d} \sum_{j=j1}^{j2} \frac{|W_n(s_j)|^2}{s_j} . \tag{6}$$

where, d_j is scale step; *dt* is the time space of the time series; the C_d is scale independence and is a constant for each wavelet function. The scale-averaged wavelet power is a time series of the average variance in a certain frequency band, which can be used to examine modulation of a time series.

In the following section, the wavelet spectrum entropy and scale-averaged wavelet power are used to analyze the epileptic EEG signals; these two feature values are taken as the indications of the epileptic seizures simultaneously. These two features contain the change degree and distribution of signal power at the time – scale domain.

3 Results and Discussions

The EEG data analyzed in this paper are derived from Dr. Alpo Vaerri (Tampere University of Technology). Sampling frequency of the EEG data is 200 Hz. The data is recorded by four bipolor montage, which are located in F8-C4, F7-C3, T6-02, and F5-01. EEG data of five patients with different epileptic seizures were collected by sampling frequency of 200 Hz. EEG of patient 1 contains petit mal epileptic discharges, patient 2's EEG contains irregular type epileptic discharges, patient 3's EEG contains three type epileptic discharges, patient 4's EEG contains epileptic seizures with EMG activity on the EEG, and patient 5's EEG contains psychometric epileptic seizures.

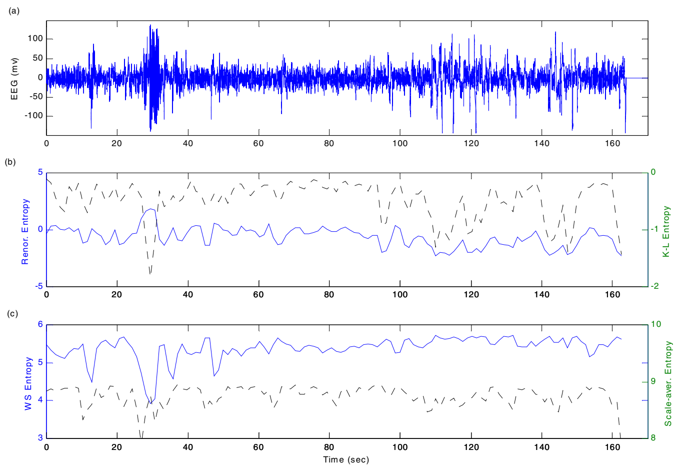


Fig. 1. (a) EEG recorded from the F8-C4 electrode of patient 1 (petit mal epileptic discharges); (b) Renormalized entropy (solid line) and K-L relative entropy (dot line) calculated with FT; (c) The normalized wavelet spectrum entropy (solid line) and scale - averaged wavelet power entropy (dot line) by using Morlet wavelet transform

This paper uses wavelet spectrum entropy and scale-averaged wavelet power to indicate the epileptic seizures. An EEG data is divided into consecutive segments of length N=512 with a 50% overlap. Fig. 1 shows an EEG signal of patient 1, a petit mal epileptic seizure occurs at the 30 second. In Fig. 1 (b), K-L relative entropy can indicate the petit mal epileptic seizures occurred at the 30 second, however, the spikes that occurs after 90 second are detected as well. The renormalized entropy cannot give significant indications for these seizures. Fig. 1 (c) shows the wavelet spectrum entropy and scale - averaged wavelet power based on Morlet wavelet that can indicate the seizure at the second of 30. The epileptic seizure occurs at the second 12 is also indicated, unlike the K-L relative entropy and renormalized wavelet spectrum. The scale-averaged wavelet power can basically indicate other petit mal epileptic seizures as well, although it is not clearer than K-L relative entropy. This is due to the fact that the renormalized entropy and K-L relative entropy have a reference segment for the case. If we can select a reference segment well, a good result could be obtained, otherwise the result is so much bad. This selection is not a very easy task in the practical implement because of the variation of cases.

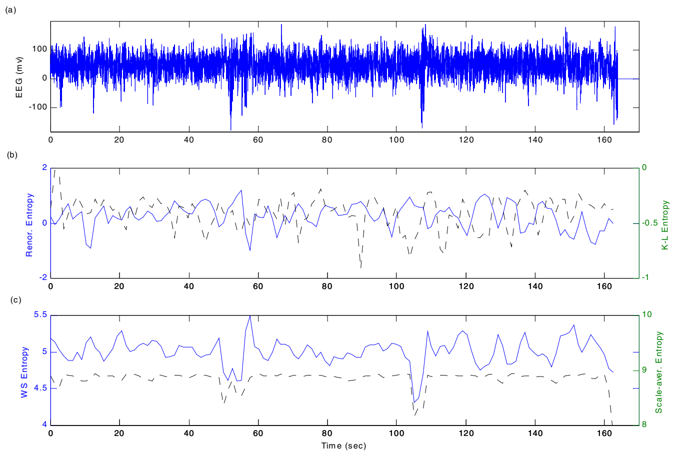


Fig. 2. (a) EEG recorded from the F8-C4 electrode of patient 2 (irregular type epileptic discharges); (b) Renormalized entropy (solid line) and K-L relative entropy (dot line) calculated with FT; (c) The normalized wavelet spectrum entropy (solid line) and scale-averaged wavelet power (dot line) by using Morlet wavelet transform

Fig. 2 shows the EEG signal of patient 2 with irregular type epileptic seizure. The EEG signal contains two irregular epileptic seizures during the 50-60 and 105-110 second. Fig. 2 (c) shows the normalized wavelet spectrum entropy and scale-averaged wavelet power can indicate these two seizures clearly. However, the renormalized entropy and K-L relative entropy fail to indicate these two seizures in Fig. 2 (b). This is resulted by the selection of the reference segment before running the detection method.

EEG of a patient with mixed epileptic seizures is shown in Fig. 3 (a). Most of the seizures of the patient could be indicated with the normalized wavelet spectrum entropy, seeing Fig. 3 (c), but the seizure occurs during 130-140 second. The scale-averaged wavelet power can indicate these seizures well, like the K-L relative entropy. Renormalized entropy only indicates the a few seizures, however.

Fig. 4 shows an epileptic EEG data of patient 4 with EMG activity. The patient occurs EMG activity during the 50-80 and 150-160 second, respectively. Fig. 4 (b) and (c) show the renormalized entropy, K-L relative entropy and normalized wavelet spectrum entropy did not overcome the effect of the EMG activity. However, that scale-averaged wavelet power is able to overcome this effect of the EMG activity, seeing the dot line in Fig. 4 (c), meanwhile the epileptic seizures are indicated completely. Fig. 5 (a) shows the EEG of a patient with psychomotor epileptic seizures. Seeing Fig. 5 (b) and (c), the normalized wavelet spectrum entropy is the best to indicate the seizures among other entropy. It is found that two hidden seizures that occur at the second 20 and 58 could be indicated by the normalized wavelet spectrum entropy (seeing the solid line of Fig. 5 (c)).

In this paper, we employ the wavelet spectrum entropy and scale-averaged wavelet power based on the Morlet wavelet transforms to indicate the seizures of five patients. It is found that theses two feature values could indicate the epileptic seizures effectively. Some of hidden seizures could be indicated, but also the effect of EMG

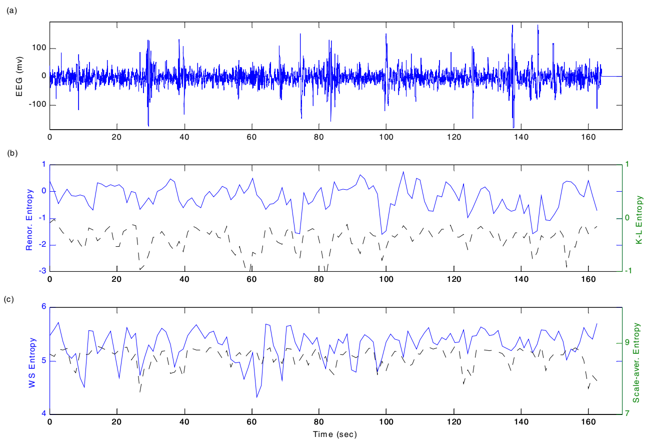


Fig. 3. (a) EEG recorded from the F8-C4 electrode of patient 3 (mixed epileptic seizures); (b) Renormalized entropy (solid line) and K-L relative entropy (dot line) calculated with FT; (c) The normalized wavelet spectrum entropy (solid line) and scale-averaged wavelet power (dot line) by using Morlet wavelet transform

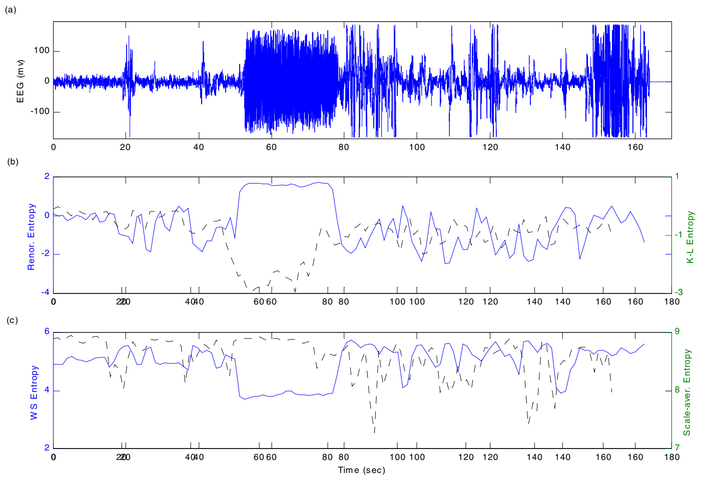


Fig. 4. (a) EEG recorded from the F8-C4 electrode of patient 4 (epileptic seizures with EMG activity); (b) Renormalized entropy (solid line) and K-L relative entropy (dot line) calculated with FT; (c) The normalized wavelet spectrum entropy (solid line) and scale- averaged wavelet power (dot line) by using Morlet wavelet transform

activity to detection of epileptic seizure could be avoided for scale-averaged wavelet power. It is noticed that the renormalized entropy and K-L relative entropy two methods need a segment of EEG as a reference. In the future work, we will combine the normalized wavelet spectrum entropy and scale-averaged wavelet power to come up with a new detection method of epileptic seizures in extracranial EEG.

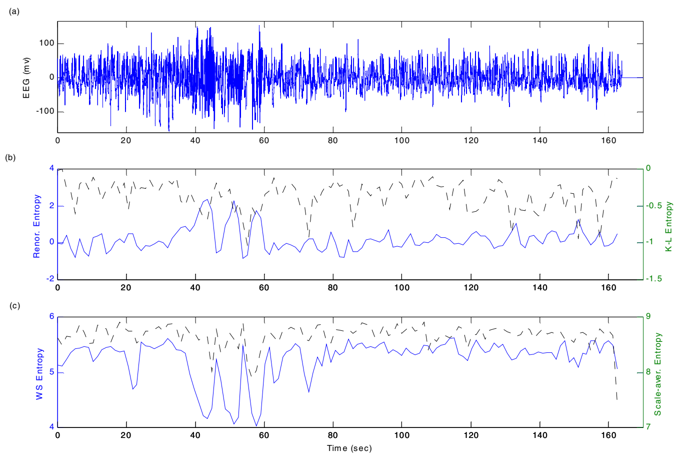


Fig. 5. (a) EEG recorded from the F8-C4 electrode of patient 5 (psychomotoric epileptic seizures); (b) Renormalized entropy (solid line) and K-L relative entropy (dot line) calculated with FT; (c) The normalized wavelet spectrum entropy (solid line) and scale-averaged wavelet power (dot line) by using Morlet wavelet transform

Acknowledgments. The work is supported by Cercia, The Centre of Excellence for Research in Computational Intelligence and Applications, in the School of Computer Science at the University of Birmingham, UK and National Natural Science Foundation of China (60575012).

References

1. Gotman, J.: Automatic seizures detection: improvements and evaluation. Electroenceph. Clin. Neurophysiol, **76** (1989) 317-324
2. Hilfiker, P., Egli, M.: Detection and evolution of rhythmic components in ictal EEG using short segment spectra and discriminate analysis. Electroenceph. Clin. Neurophsical, **82** (1992) 255-265
3. Babloyantz, A., Destexhe, A.: Low-dimensional chaos in an instance of epilepsy. Proc. Nat. Acad Sci, USA, **83** (1986) 3513-3517
4. Frank, G.W., Lookman, T., Nerenberg, MA.H, Essex, C., Lenieux, J., Blume, W.: Chaotic time series analysis of epileptic seizures. Physica D, **46** (1990) 427-438
5. Yaylali, H., Kocak, H., Jayakar, P.: Detection of seizures from small samples using nonlinear dynamic system theory. IEEE. Trans. Biomed. Eng., **43** (1996) 743-751
6. McGrogan, N., Tarassenko, L.: Neural network detection of epileptic seizures in the EEG. Research Report of Department of Eng. Sci., Oxford University, (1999)
7. Kopitzki, K., Warnke, P.C., Timmer, J.: Quantitive analysis by renormalized entropy of invasive EEG recordings in focal epilepsy. Phys. Rew. E. **58** (1998) 4895-4864
8. Blance, S., Quian, R.Q., Rosso, O.A., Kochen, S.: Time–frequency analysis of electro-encephalogram series. Phys. Rev. E, **51** (1995) 2624-2631
9. Blanco, S., D'Attellis, C.E., Isaacson, S.I., Rosso, O.A., Sirne, R.O.: Time-frequency analysis of electroencephalogram series. II. Gabor and wavelet transform. Phys. Rev. E, **54** (1996) 6661-6672

10. Kalayci, T., Özdamar, Ö.: Wavelet preprocessing for automated neural network detection of EEG spikes. IEEE Engineering in Medicine and Biology, **14** (1995) 160-166
11. Sirne, R.O., Isaacson, S.I., D'Attellis, C.E.: Data-reduction process for long-term EEGs. IEEE Engineering in Medicine and Biology, **18** (1999) 56-61
12. Saparin, P., Witt, A., Kurths, J., Anishenko, V.: The renormalized entropy - an appropriate complexity measure. Chaos, Solitons and Fractals **4** (1994) 1907-1916
13. Quian Quiroga, R., Arnhold, J., Lehnertz K., Grassberger, P.: Kulback-Leibler and renormalized entropies: Applications to electroencephalograms of epilepsy patients. Phy. Rev. E, 62 (2000) 8380-8386
14. Daubechies, I., The wavelet transform, time-frequency localization and signal analysis. IEEE Trans. Inf. Theory, **36** (1990) 961-1005
15. Li, X., Kapiris, P.G., Polygiannakis, J., Eftaxias, K.A. and Yao, X., Fractal spectral analysis of pre-epileptic seizures phase: in terms of criticality, Journal of Neural Engineering, **2**(2005) 11-16
16. Farge, M., Wavelet transforms and their application to turbulence. Annu., Rev. Fluid. Mech., **24** (1992) 395-457

The Study of Classification of Motor Imaginaries Based on Kurtosis of EEG*

Xiaopei Wu[1,2] and Zhongfu Ye[2]

[1] The Key Laboratory of Intelligent Computing & Signal Processing of MOE, Anhui University, Hefei 230039, Anhui province, China
wxp2001@ahu.edu.cn

[2] Department of Electronic Engineering and Information Science, USTC, Hefei 230037, Anhui province, China
zhongfuye@ustc.edu.cn

Abstract. In this paper, the kurtosis-based method for the classification of mental activities is proposed. The EEG signals were recorded during imagination of left or right hand movement. The kurtosis of EEG and its dynamic properties with respect to time are analyzed. The experiment results show that the kurtosis can reflect the EEG pattern changes of different motor imageries. According to the analysis and experiment results, a kurtosis based classifier for the classification of left and right movement imagination is designed. This classifier can achieves near 90% correct rate. As the kurtosis is computationally less demanding and can also be estimated in on-line way, so the new method proposed in this paper has the practicability in the application of brain-computer interface.

Keywords: EEG; kurtosis; motor imaginary; classification.

1 Introduction

The study of brain computer interface (BCI) is a hotspot in the field of biomedical engineering. Until now, about 40 research groups around the world are carrying out BCI research. The contents of BCI researches are to build the communication directly between human brain and computer (or other instruments) so that to realize the control of brain over apparatuses. EEG or ERP signals is commonly used as the carrier of control commands[1].

The research in this paper is concerned with the EEG pattern extraction during left and right hand movement imagination. Until now, these are some reports about this research. In the published papers, the researchers have selected different EEG patterns and adopted different methods for the classification of the left and right hand movement, such as : adaptive AR parameters, AR spectrum, mu and beta rhythm[2]-[7], etc. In this paper, the high order cumulant of EEG signal is investigated when the subject are imaging the left or right hand movement. The research in this paper shows

* This paper is supported by national science foundation of China (60271024), RENCAI foundation of Anhui province (2004Z028) and innovative research team of Anhui university.

that the distribution of normalized 4th order cumulant (kurtosis) of EEG on the scalp is changing obviously with different movement imagination (imaging left or right hand movement). For the purpose of dynamic estimation of kurtosis, we adopt a window sliding along the EEG data sequence to calculate the kurtosis of multi-channel EEG signals. Comparing the two kurtosis time courses of EEG in C3 and C4 while the subject is imaging the hand movement, we find that the kurtosis changes obviously with different imagination task. Based on these phenomena, we propose a classification algorithm based on kurtosis to classify the left and right hand movement imagination. For 160 trials of motor imagination, the rate of accurate classification is near 90%. Furthermore, the kurtosis estimation is in online way, so the computation is less demanding. We conclude that the kurtosis based classification method has the advantages in real time computation and classification.

2 The Online Estimation of Kurtosis

For a random signal $x(n)$ with zero mean, its kurtosis is defined as:

$$kurt = \frac{E[x^4(n)]}{(E[x^2(n)])^2} - 3 \tag{1}$$

we commonly use finite length samples of $x(n)$ to estimate the 2th and 4th order moments in Eq.1. Assume N is the length of samples, then:

$$m_2 = E[x^2(n)] \approx \frac{1}{N}\sum_{n=1}^{N} x^2(n) \tag{2a}$$

$$m_4 = E[x^4(n)] \approx \frac{1}{N}\sum_{n=1}^{N} x^4(n) \tag{2b}$$

But in many cases, the sample $x(1)$, $x(2),\ldots,x(N)$ is not fixed but new observations keep on coming. So we need an online estimation method of moments and kurtosis. Considering the requirement of the data analysis in this paper, two kinds of on line estimation methods are investigated.

2.1 The Online Estimation of Kurtosis Based on Sliding Analytic Window

Assume the length of the analytic window is N, n is the discrete time index. So the data samples in the window at time n is $x(n\text{-}N\text{+}1),\ldots,x(n\text{-}1),x(n)$. For the new incoming sample of $x(n+1)$, it is not difficult to get the online estimation of 2th and 4th moments at time n+1:

$$m_2(n+1) = m_2(n) - \frac{1}{N}\{x^2(n-N) - x^2(n+1)\} \tag{3a}$$

$$m_4(n+1) = m_4(n) - \frac{1}{N}\{x^4(n-N) - x^4(n+1)\} \tag{3b}$$

Where: $m_2(n)$ and $m_4(n)$ denote the 2th and 4th moments at time n.

$$m_2(n) = \frac{1}{N}\sum_{i=0}^{N-1} x^2(n-i)$$

$$m_4(n) = \frac{1}{N}\sum_{i=0}^{N-1} x^4(n-i)$$

Combining Eq.1 and Eq.3, we can get the online estimation of kurtosis:

$$kurt(n+1) = \frac{m_4(n) - \frac{1}{N}\left\{x^4(n-N) - x^4(n+1)\right\}}{\left[m_2(n) - \frac{1}{N}\left\{x^2(n-N) - x^2(n+1)\right\}\right]^2} - 3 \tag{4}$$

2.2 The Online Estimation of Kurtosis with Increasing Window Length

Another way for online estimation of kurtosis and moments is to fix the left side of the analytic window and the kurtosis is updated with the new incoming sample, (that means the length of the analytic window are increasing with the new incoming samples), the formulas for moment estimation are as follows:

$$m_2(n+1) = \frac{n}{n+1} m_2(n) + \frac{x^2(n+1)}{n+1} \tag{5a}$$

$$m_4(n+1) = \frac{n}{n+1} m_4(n) + \frac{x^4(n+1)}{n+1} \tag{5b}$$

where

$$m_2(n) = \frac{1}{n}\sum_{i=1}^{n} x^2(n)$$

$$m_4(n) = \frac{1}{n}\sum_{i=1}^{n} x^4(n)$$

Combining eq.1 and eq.5, we get another formula for online kurtosis estimation:

$$kurt(n+1) = \frac{\frac{n}{n+1} m_2(n) + \frac{x^2(n+1)}{n+1}}{\left[\frac{n}{n+1} m_4(n) + \frac{x^4(n)}{n+1}\right]^2} - 3 \tag{6}$$

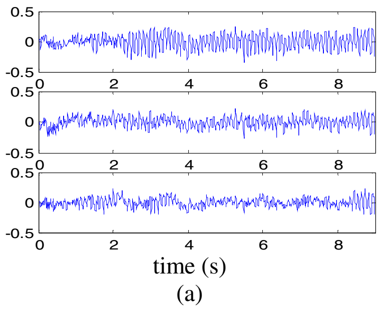
It is easy to find that either of Eq.(4) and (5) needs much less computation than directly estimating it in common way. In real world application, we can combine the two formulas to estimate the kurtosis and moments. For example, at the beginning of online data processing, the data length is less than the length of analytic window, so formula (6) can be used to estimate the kurtosis and moments , when data length is increasing to be or greater than the length of the analytic window, formula (5) can be used.

3 The Kurtosis Analysis and Classification of EEG During Imagination of Left and Right Hand Movement

3.1 EEG Data Description

The EEG data is from the Graze university of technology. The experimental procedure is as follows. The subjects were sitting in a relaxing chair looking at the centre of a monitor placed in front of them. The task was to control a feedback bar by means of imagery left or right hand movements. The order of left and right cues was random. The total time of data acquisition of each trial is 9s. During the first 2s, the subjects were quiet and didn't do any imagination. At t=2s an acoustic stimulus indicates the beginning of the trial, the trigger channel went from low to high, and a cross "+" as displayed for 1s; then at t=3s, an arrow (left or right) was displayed as cue. At the same time the subject was asked to move a bar into the direction of the cue. The data acquisition stopped at t=9s. Three bipolar EEG channels were measured over C3, Cz and C4. The EEG was sampled with 128Hz, it was filtered between 0.5 and 30Hz.

Fig.1 (a) (b) are two typical EEG signals while the subject imagines left and right hand movement respectively. It has been shown that the imagination of either a left or right hand movement results in an amplitude attenuation (desynchronization) of μ rhythm at the contra-lateral sensorimotor representation area and in some cases, in an amplitude increase(synchronization) at the ipsilateral hemisphere. We call these phenomena the event-related (de) synchronization (ERD, ERS), which are the base of an EEG-based brain computer interface(BCI). The brain states associated with motor imagery are transformed into control signals to implement the communication between brain and computer or other apparatuses.



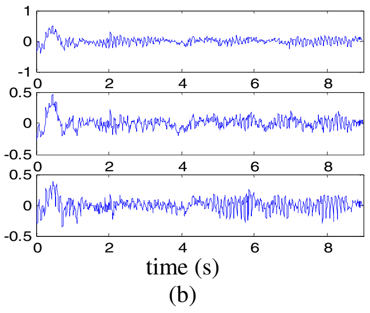
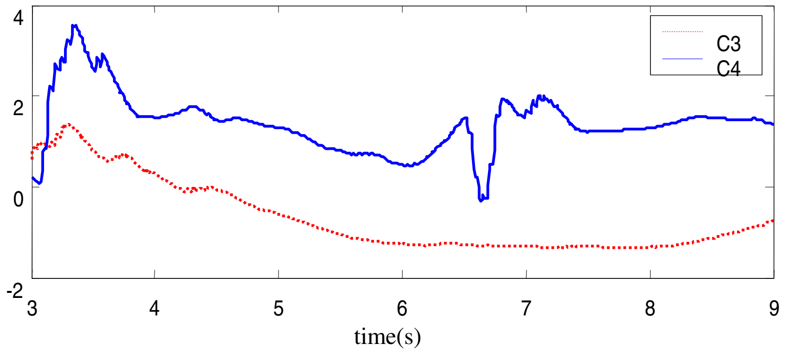


Fig. 1. Two trials of EEG of movement imagination, EEG channels are C3(top), CZ(middle), C4(bottom) respectively. (a)imagining left hand movement ; (b)imagining right hand movement

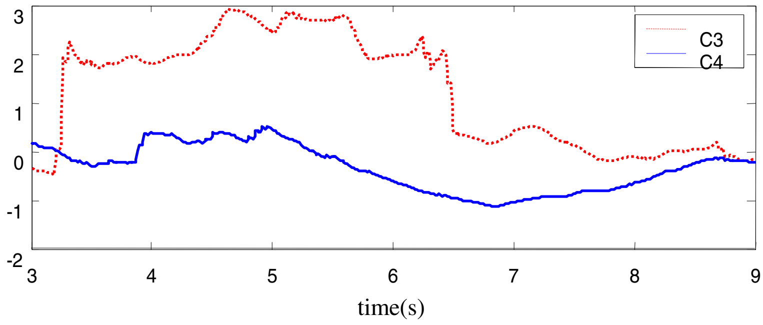
3.2 The Kurtosis of EEG During Imagination of Left and Right Hand Movement

The online estimation formula (4) is used to analyze the kurtosis of channel C3 and C4 in Fig.1. The length of window *N*=384. Fig.2 shows the time course of kurtosis (solid line for channel C3 and dotted line for Channel C4).

Comparing the time course of kurtosis of C3 and C4 in Fig.2(a),(b), while imagining the left hand movement, kurtosis of channel C3 is smaller than it of channel C4 in Fig.2(a) and vice versa while imagining the right hand movement (Fig.2(b)). The experiment results show that online kurtosis can reflect the changes of motor imagination tasks.



(a)



(b)

Fig. 2. Time course of kurtosis of EEG signals(C3, C4), the length of sliding window is 384: (a) imagining left hand movement; (b)imagining right hand movement

160 trials with class labels ('+' , 'o' for right and left) are employed to study the universality of using kurtosis for discrimination of left and hand movement. The length of window is 512 samples (4 seconds) and the time scope for analysis is between 3s and 8s. Only channels C3, C4 are analyzed. The experimental procedure is as follows: For each trial, after filtering the EEG signal with the bandpass filter (8-12Hz). The window of *N*=512 is sliding along EEG samples and the $kurt_{C3}(t)$, $kurt_{C4}(t)$ of the 512 samples contained in the window is calculated at each time *t*, then

we combine $kurt_{C3}(t)$, $kurt_{C4}(t)$ to be a 2-dimensional kurtosis vector $\mathbf{kurt}(t)=[kurt_{C3}(t),\ kurt_{C4}(t)]^T$. For 160 trials, we can get 160 kurtosis vectors totally at each time sample t. Fig 3 (a),(b),(c) show the scatter maps of the kurtosis vectors at t=4s, 6s, 8s respectively. Under the diagonal line in Fig.3 is the region of $kurt_{C3}(t)>kurt_{C4}(t)$, and vise versa. According to the three scatter maps, we can see that, with the increment of time, the scatter map is becoming more and more consistent with the true labels of left and right hand movements.

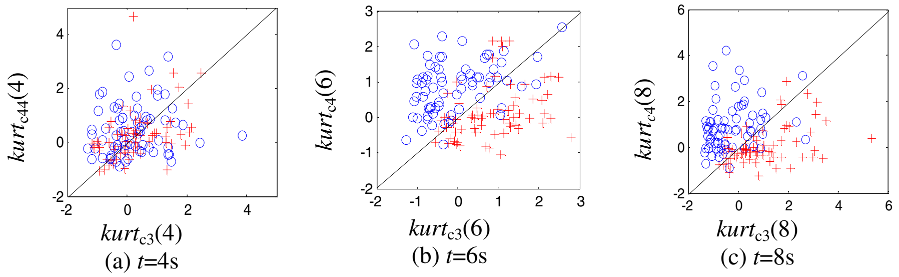


Fig. 3. The scatter maps of 4th order cumulants of (C_3 ,C_4) EEG signals at t=4s,6s,8s

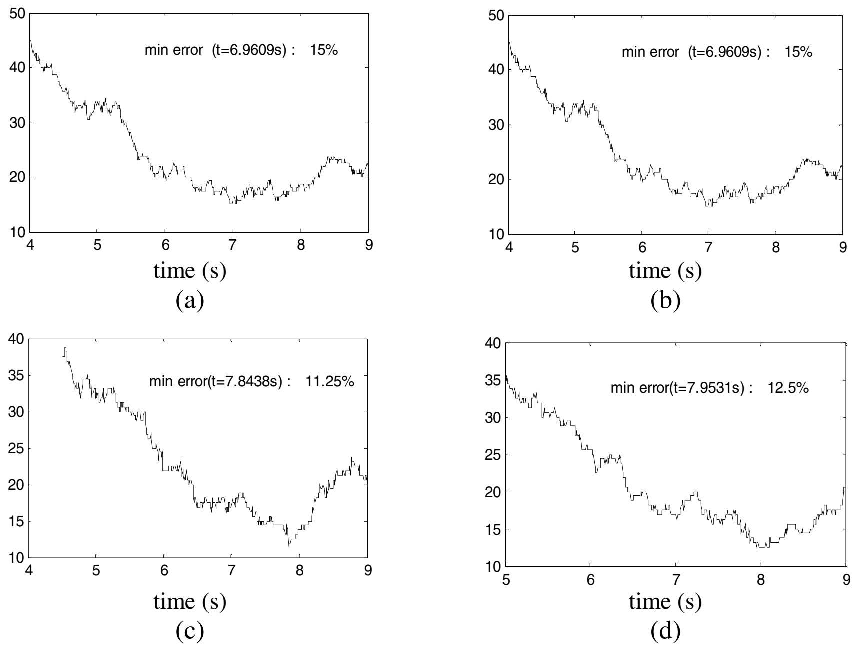


Fig. 4. Time courses of classification error rate for different length of analytic windows

For the purpose of choosing the best window's length N and the best decision time t in classification, a simple kurtosis-based classifier is proposed as follows:

$$kurt_{C3}(t) - kurt_{C4}(t)\begin{cases} > 0 & \text{right hand movement} \\ < 0 & \text{left hand movement} \end{cases} \tag{7}$$

In following experiments, the kurtosis of C_3 and C_4 from 160 trials is computed based on a window moving along the EEG samples from 0 to 9s. Then based on Eq.(7) , the classification error rate at each time point t is calculated. The length of window N is from 384 to 640 (3 to 5s), the interval of increment of N is 32 (0.25s).

Four representative time courses of the error rate with N=448, 512, 576, 640 (3.5, 4, 4.5, 5s) respectively are illustrated in Fig4. Comparing 4 time courses of error rate in Fig 4, we can see that the best decision time is between 6-8s. error rate reaches its minimal value at this time interval. Among all of the results with different N. The best result is achieved while N=576 at t=7.84s, and the minimal error rate is 11.25% (Fig.4c).

4 Discussion

It is worth while to mention that the power (second order moment)-based method is commonly to be used for the μ rhythm detection. But compared with the kurtosis based method, the former is not able to perform as good as the latter, especially in noisy environment. In our research, by using power-based method, we can only get 82.5% correct classification rate. Furthermore, the online estimation of window-based kurtosis proposed in this paper is easy to implement. So, we think that kurtosis-based classifier can be used for online detection of movement imaginations.

In our study, we also pick up the 18 trials which are not be classified correctly by kurtosis-based method and try to use other methods to classify them, such as power spectrum comparison, AR model based method, etc. But the results are not improved. By observing the EEG waveform of 18 trials, most of them don't contain ERD (RES) or contain very weak ERD (ERS) phenomena. We think that may be caused by the fatigue or distraction of subjects during the process of EEG data acquisition.

References

1. Eleanor A. Currana,b and Maria J. Stokesa : Learning to control brain activity: A review of the production and control of EEG components for driving brain–computer interface (BCI)systems. Brain and cognition, 2003,51(3): 326–336.
2. G. Pfurtscheller and A. Aranibar : Event-related cortical desynchronization detected by power measurements of scalp EEG. Electroencephalography and Clinical Neurophysiology, 1977, 42(8): 817–826.
3. G. Pfurtscheller and C. Neuper : Motor Imagery and Direct Brain-Computer Communication. Proceedings of IEEE, 2001,89(7):1123–1134,.
4. Keirn Z A and Aunon J I : A New Mode of Communication Between Man And His Surroundings. IEEE Transactions on Biomedical Engineering. 1990,37 (12) :1209–1214.
5. Schloegl A, Lugger K and Pfurtscheller G : Using Adaptive Autoregressive Parameters For a Brain-Computer-Interface Experiment. Proceedings 19th International Conference IEEE/EMBS Oct. 30 - Nov. 2, 1997 Chicago, IL. USA: 1533–1535

6. G. Pfurtscheller, C. Neuper and G.R. Muller,etc : Graz-BCI: State of the Art and Clinical Applications. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 2003,11(2):177–180.
7. J.R. Wolpaw, D.J. McFarland, and T.M. Vaughan : Brain-computer interface research at the Wadsworth Center. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 2000.8(2):222–226,

Automatic Detection of Critical Epochs in coma-EEG Using Independent Component Analysis and Higher Order Statistics

G. Inuso, F. La Foresta, N. Mammone, and F.C. Morabito

DIMET - *Mediterranea* University of Reggio Calabria
via Graziella Feo di Vito, I-89060 Reggio Calabria, Italy
fabio.laforesta@unirc.it
http://neurolab.ing.unirc.it

Abstract. Previous works showed that the joint use of Principal Component Analysis (PCA) and Independent Component Analysis (ICA) allows to extract a few meaningful dominant components from the EEG of patients in coma. A procedure for automatic critical epoch detection might support the doctor in the long time monitoring of the patients, this is why we are headed to find a procedure able to automatically quantify how much an epoch is critical or not. In this paper we propose a procedure based on the extraction of some features from the dominant components: the entropy and the kurtosis. This feature analysis allowed us to detect some epochs that are likely to be critical and that are worth inspecting by the expert in order to assess the possible restarting of the brain activity.

1 Introduction

Previous works showed that the joint use of Principal Component Analysis (PCA) and Independent Component Analysis (ICA) allows to extract meaningful dominant components (DCs) from the Electroencephalogram (EEG) of patients in coma [1]. The EEG is a technique that measures the electric field produced by the bioelectric impressed current density, associated with neuronal activity, through a set of scalp electrodes properly placed over the head [2]. The visual inspection of the EEG is worldwide accepted as a key step in the cerebral death assessment of patients in coma, because the EEG is strictly correlated with the electric activity of the brain [3,4]. The bioelectric pattern of the patient in coma is monitored for a long time in order to detect any electrical activity of the brain. In case no cerebral electrical spontaneous activity is observed, the patient will be thought to be dead.

Assessing the cerebral death may be troublesome because of the artifacts: artifacts are signals not related to the cognitive activity, they are generated by external factors (electrical line noise, interference) or by non-cognitive inner factors (muscle activity, eye blinking, eye movements). They are superimposed to the signals we want to analyse and it stand to reason that they are very unwelcome, especially if the signal we want to analyse is weak, such as the signals from patients in coma. Moreover, the intensive care environment, that is the typical environment which the coma signals are recorded in, is full of many electromedical working equipments [5,6], thus the external artifacts are very likely to occur.

Previous work showed that we can extract, from coma EEG, a few dominant components including almost the whole information content of the EEG itself [1]. This extraction is based on the joint use of PCA and ICA. The recording is divided into data segments (epochs) than each epoch is processed. PCA concentrates most of the information content in a few principal components and provides information about the dimension of the embedding space, then ICA is carried out in order to extract the independent components. This algorithm can concentrate the information content in a few dominant components, thus each epoch can be represented by them [7,8,9].

Since the time duration of the coma EEG may be very long, we headed to investigate a procedure for the automatic detection of the critical EEG epochs, in order to support the doctor in the long time monitoring. An automatic critical epoch detection procedure is supposed to label the epochs that the expert should pay particular attention to, so that he can figure out whether the detected activity is related to artifacts or to brain activity.

Thus we are headed to find a procedure able to automatically quantify how much an epoch is critical or not. In this paper we propose a method based on the extraction of some features from the dominant components: the entropy and the kurtosis. This feature analysis allowed us to detect some epochs that are likely to be critical and that are worth inspecting by the expert.

The paper is organized as follows: the second section explains how the dominant components are extracted from the EEG, the third section explains why we propose the entropy and the kurtosis as markers for the critical epoch detection, the fourth section reports the results and the last section reports the conclusions.

2 PCA and ICA for Dominant Components Extraction

Before processing the dataset by PCA and ICA, we divided the dataset 1sec non overlapping segments (epochs) and we extracted the dominant components from each epoch. PCA is a classic technique that, given a set of multidimensional observed variables, yields a smaller set of variables, with less redundancy, that are supposed to provide a reliable compressed representation of the data. Essentially, a set of correlated variables is transformed into a set of uncorrelated variables that are ordered by reducing variance. The uncorrelated variables are linear combinations of the original variables, and the last variables of this set can be removed with minimum loss of the information carried by the original data.

PCA is commonly used to reduce the dimensionality of a data set while retaining as much information as possible. PCA can be thought as a rotation of the existing axes to new positions in the space defined by the original variables. In this new rotation, there will be no correlation between the new variables.

Therefore, PCA provides a representation of the distribution of the original variables in the new multidimensional space. PCA can be performed finding the eigenvalues of the covariance matrix of the data [10] and calculating the information content η_i related to the i-th principal component (PC) by

$$\eta_i = \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j} \tag{1}$$

where λ_i is the eigenvalue related to the i-th PC and n is the amount of the PCs extracted from the n EEG rows.

ICA allows to find underlying factors from multivariate data. ICA extracts components that are both statistically independent and nongaussian [10]. Given N samples of the observed data vector $\mathbf{x}$, whose elements are the mixtures $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_m$, modeled by

$$\mathbf{x} = \mathbf{A}\mathbf{s} = \sum_{j=1}^{n} \boldsymbol{a}_j \boldsymbol{s}_j \tag{2}$$

where $\mathbf{A}$ is the unknown m-by-n mixing matrix with column vectors $\boldsymbol{a}_j, j = 1, 2, ..., n$, and $\mathbf{s}$ is an unknown n-dimensional source vector containing the source signals $\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_n$, which are assumed to be statistically independent. In general, the dimensionality m of the vector $\mathbf{x}$ and $\boldsymbol{a}_j$ can be different from n. We usually assume that the number of mixtures is at least equal to the number of sources ($m \geq n$), the mixing matrix $\mathbf{A}$ has full rank, and that at most one source $\boldsymbol{s}_j$ can be Gaussian [10]. ICA solves the Blind Source Separation (BSS) problem supposing that the sources are statistically independent from each other. In particular ICA estimates a n-by-m unmixing matrix $\mathbf{W}$ so that

$$\mathbf{u} = \mathbf{W}\mathbf{x} = \sum_{j=1}^{m} \boldsymbol{w}_j \boldsymbol{x}_j \tag{3}$$

where $\mathbf{u}$ is an estimation of the source vector $\mathbf{s}$. This model can be used in different situations, for example in multidimensional signal processing, where each sensor detects an unknown superimposition of unknown source signals at the time points $t = 1, 2, ..., N$. Many algorithms for ICA have been proposed in the last years. We exploited the Bell-Sejnowski INFOMAX algorithm [11,12]. In particular, we used the switching extended rule,

$$\varDelta\mathbf{W} \propto [\mathbf{I} - \mathbf{K}\tanh(\mathbf{u})\mathbf{u}^T - \mathbf{u}\mathbf{u}^T]\mathbf{W} \tag{4}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix and $\mathbf{K}$ is a $n \times n$ diagonal matrix, whose elements are k_i; for supergaussian sources $k_i = 1$, while $k_i = -1$ for subgaussian sources. The k_i are related to the sign of the $\boldsymbol{u}_i$ kurtosis:

$$k_i = sign(kurt(\boldsymbol{u}_i)) \tag{5}$$

The performance of ICA can be improved by means of a PCA preprocessing that makes the procedure less time consuming [10]. The PCA can be also used in order to reduce the data dimensionality and/or to perform whitening process.

3 Feature Extraction

Our goal is to automatically detect the critical epochs, in other words the epochs that it is worth being inspected carefully. When a patient is in coma, it is hopeful to detect any even tiny brain activity. The amplitude of the EEG of these patients is very tiny, the typical brain waves are not visible, thus in case of restarting of the brain activity, the epoch is supposed to have particular statistical characteristics compared to the rest of the epochs. A critical epoch will be "more random" because an unexpected event occurs, it will be "odd" with respect to the epochs in which no activity is observed. Any activity is hopeful to be detected, even some artifacts, for example the ocular or muscular artifacts. These artifacts are characterized by a peaky distribution and could be detected by a measure of the "peakyness". The parameters that are able to estimate the randomness and the peakyness are the entropy and the kurtosis, respectively. They have been proposed for the detection of the artifacts in normal EEG [13], we propose to use them for the detection of brain activity in coma EEG. In particular we propose to use the Renyi's definition of entropy. Since the dominant components include most of the information content of the epochs, in order to extract the features of each epoch we processed the corresponding dominant component. Once the features were extracted, they were normalized and compared: if at least one of the two features of a component (i.e. epoch) exceeded a fixed threshold, the component was marked as critical. We proposed a threshold for the two features, further investigation will be devoted to the optimization of this setting.

3.1 Kurtosis

Given a scalar random variable x, kurtosis has the following expression:

$$k = m_4 - 3m_2^2 \tag{6}$$

$$m_n = E\{(x - m_1)^n\} \tag{7}$$

where m_n is the n-order central moment of the variable and m_1 is the mean. Kurtosis is positive for "peaked" activity distributions, typical of eye blink, cardiac artifacts and muscular artifacts; kurtosis is negative for "flat" activity distributions, typical of noise [13]. We estimated the kurtosis of each dominant component and we normalized it to zero-mean and unit-variance with respect to all the dominant components. The epochs associated with a component whose entropy exceeded the 50% of the range of the normalized kurtosis, were marked as critical.

3.2 Renyi's Entropy

As a measure of randomness of the epoch, we used the differential entropy, in particular we used the Renyi's definition of entropy. For a random variable y, whose pdf is $f_Y(y)$, the Renyi's entropy is defined as:

$$H_{R_\alpha}(y) = \frac{1}{1-\alpha} \log \int_{-\infty}^{+\infty} f_Y^\alpha(y) dy \tag{8}$$

$$H_{R_\alpha}(y) = \frac{1}{1-\alpha} \log\{\frac{1}{N^\alpha} \sum_j [\sum_i k_\sigma(y - y_i)]^{\alpha-1}\} \tag{9}$$

where α is the order of the entropy and the expression (9) comes from the application of the kernel estimators [14]. If the random variable is concentrated in small temporal intervals, its differential entropy is small, indeed the variables whose probability densities take large values give a strong contribution to the integral in the (8), thus their entropy is small. This feature of the entropy help us to mark the signals which are concentrated in small temporal intervals with high probabilities and, therefore, which are very likely to account for restarting brain activity or artifacts. We estimated the Renyi's entropy of each dominant component, then we normalized these values to zero-mean and unit-variance with respect to all the components (i.e. the epochs). The epochs associated with a component whose entropy exceeded the 60% of the range of the normalized entropies, were marked as very random epochs. The order of the entropy was set at 2, in order to equally emphasize the sub-gaussian and the super-gaussian components [14].

4 Results

4.1 EEG Data Description

The EEG was recorded in an intensive care environment. We data were acquired through 19 electrodes placed on the scalp according to the international standard 10-20 by Jasper (the electrode montage and a 51 sec data segment is shown in Figure 1), we chose a monopolar montage, with the reference electrode placed in a neutral point of the scalp. The sampling rate was set at 256 Hz.

4.2 Dominant Components Extraction

The EEG recording was first divided into 1-sec epochs. For each epoch we performed the PCA and then we estimated the information content of each component. Looking at the PCs we can figure out which is the minimum dimension of the embedding space of our EEG data. Previous work showed that we can completely embed the coma EEG data in a 2÷3 dimensional space [1]. Once we checked that we could embed our data in low-dimensional space, we passed the whole dataset of principal components through ICA. The early two or three ICs of each epoch could account for the whole information content of the epoch itself, but we decided to test the procedure selecting only the first component (the dominant component), because it accounted for most of the information. Thus the ICs were ordered according to their variance and the first one was selected. Figure 1 shows the dominant components extracted from the epochs.

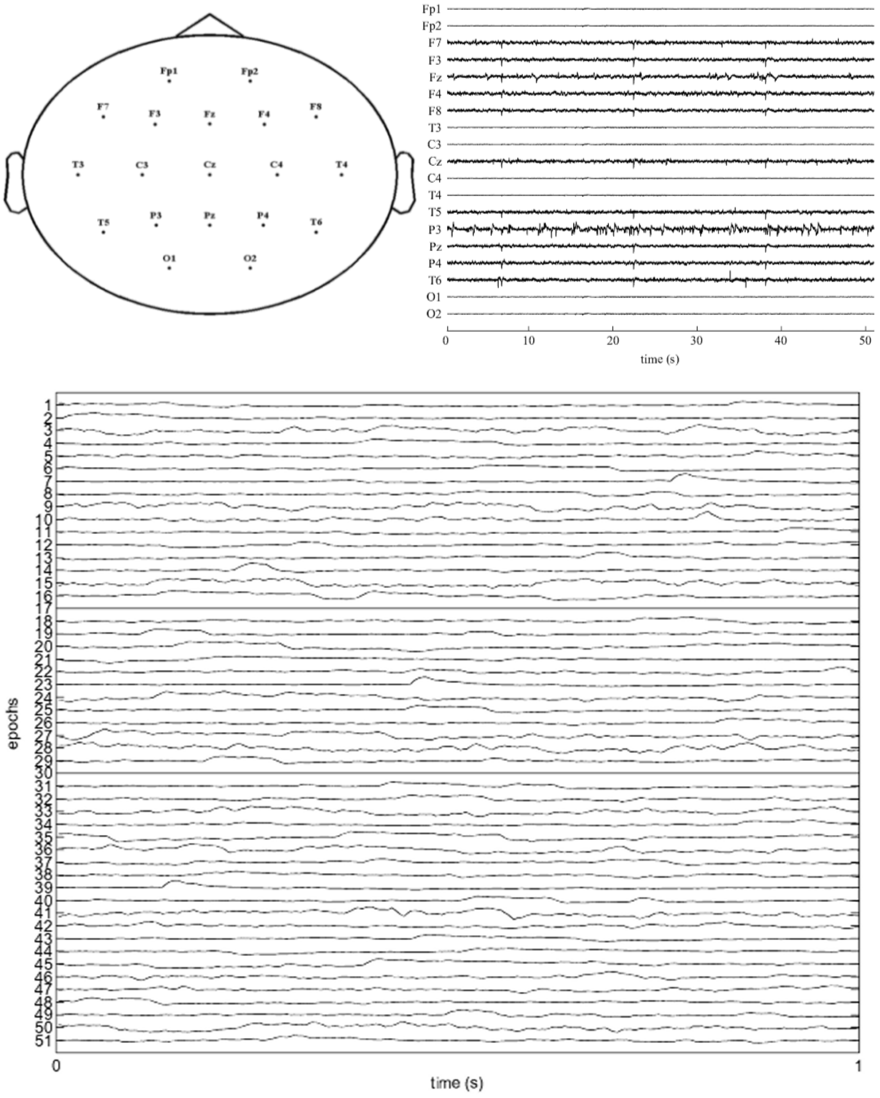


Fig. 1. (on the top) The electrode montage and a 51sec EEG segment from a patient in coma. (on the bottom) The dominant components extracted from the 51 epochs.

4.3 Feature Extraction

The set of dominant components was processed according to feature extraction procedure described in Section 3. The Renyi's entropy and the kurtosis were computed for each dominant component (i.e. epoch), they were normalized to zero-mean, unit-variance and amplitude in the range -1 to 1. If at a least one of the two features of a certain component exceed the fixed threshold, the

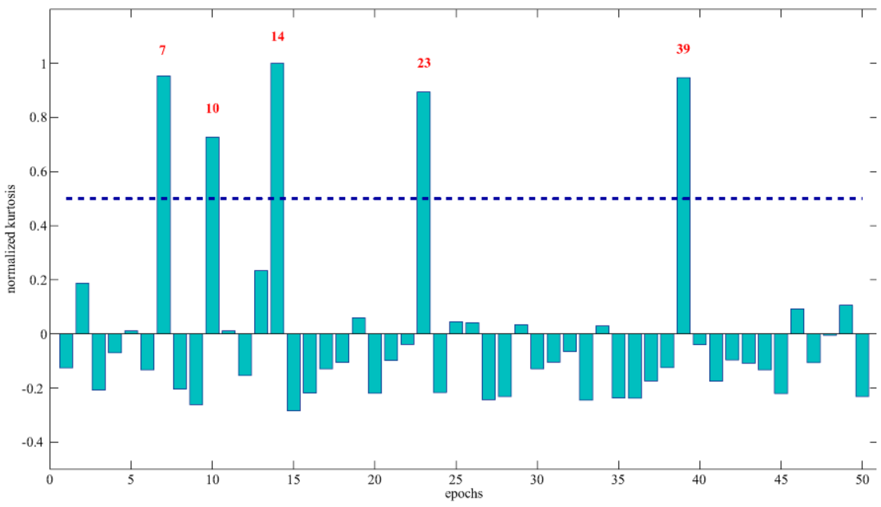


Fig. 2. The normalized kurtosis of the epochs. The kurtosis exceeds the threshold 0.5 for the epochs EP7, EP10, EP14, EP23, EP39.

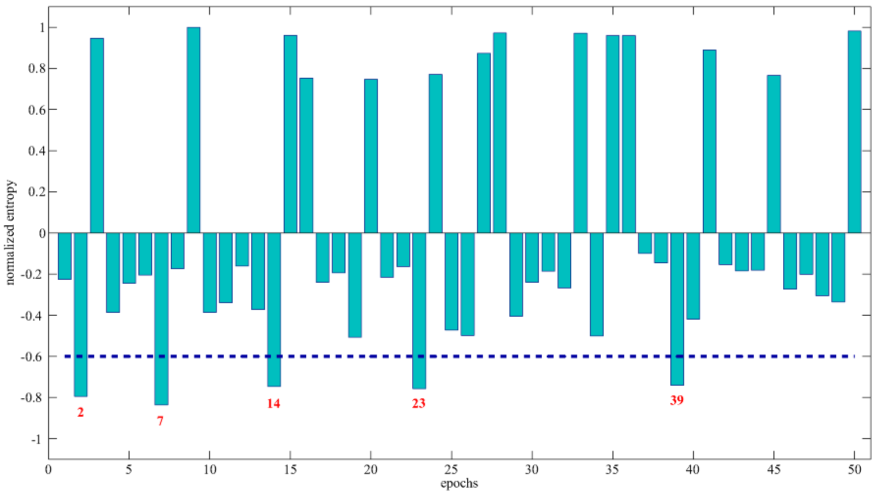


Fig. 3. The normalized Renyi's entropy of the epochs. The entropy exceeds the threshold -0.6 for the epochs EP2, EP7, EP14, EP23, EP39.

component was marked and the corresponding epoch was thought to be critical. The kurtosis was estimated according to the equations in Section 3.1 and the results are plotted in Figure 2: the critical epochs (EPs) are EP7, EP10, EP14, EP23, EP39. The Renyi's entropy was computed according to the equations in 3.2 and the results are plotted in Figure 3: the critical epochs (EPs) are EP2, EP7, EP10, EP14, EP23, EP39. We can point out that the epochs EP7, EP14,

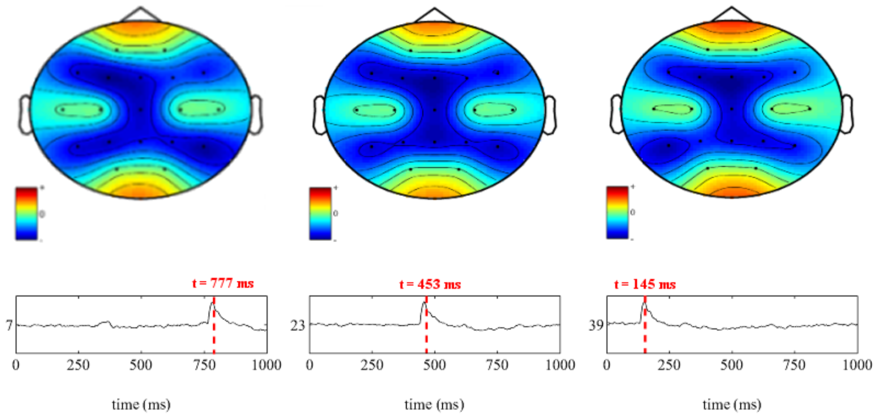


Fig. 4. The dominant components DC7, DC23, DC39 and the mapping of the critical epochs. The dominant components are plotted in the bottom, the critical time point of each epoch is highlighted with a red vertical dashed line. The EEG mapping at the critical time points is shown on top.

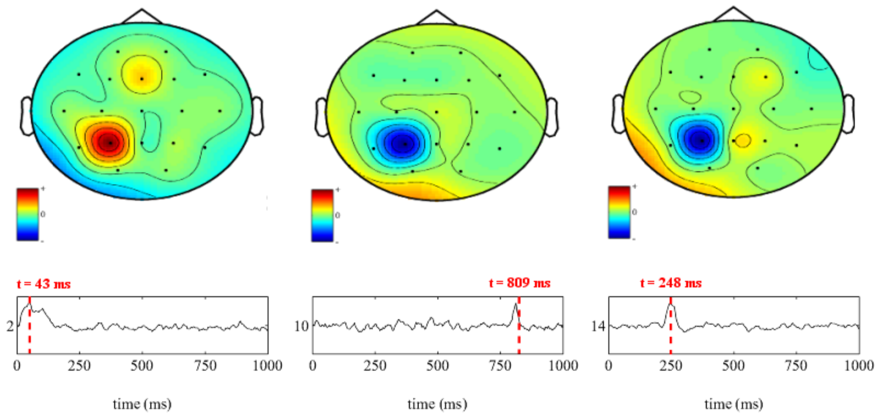


Fig. 5. The dominant components DC2, DC10, DC14 and the mapping of the critical epochs. The dominant components are plotted in the bottom, the critical time point of each epoch is highlighted with a red vertical dashed line. The EEG mapping at the critical time points is shown on top.

EP23, EP39 are critical according to both the measures. The dominant components associated to the epochs EP7, EP23 and EP39 are shown in Figure 4 whereas the dominant components associated to the epochs EP2, EP10 and EP14 are shown in Figure 5. These are the epochs which is worth inspecting very carefully in order to figure out whether the detected waves are associated to brain activity or to artifacts. The critical time point of each epoch is highlighted with a red vertical dashed line. Looking at the mapping of the EEG at the critical time points (Figures 4 and 5), we realized that there are two kind of critical behaviour of the EEG signals: the first one is characterized by a relative high amplitude in the frontal and occipital region and a low amplitude in the central, fronto-temporal and parietal regions. The second kind of behaviour is characterized by a concentration of the activity in the electrode P3.

5 Conclusions

This paper proposed a method, based on the joint use of PCA, ICA and higher order statistics, for the automatic detection of critical epochs in EEG recordings from patients in coma. The EEG was divided into non-overlapping epochs and the joint use of Principal Component Analysis (PCA) and Independent Component Analysis (ICA) extracted a few meaningful dominant components from each epoch. Then, some higher order statistics (kurtosis and the Renyi's entropy) were extracted from each dominant component so that each epoch was described by the set of features associated to the corresponding dominant component. Once we got the features, we selected the epochs that were likely to be critical according to a threshold. This feature extraction detected some epochs that are worth inspecting by the expert in order to assess the possible restarting of the brain activity. Future investigation will be devoted to the optimization of the statistics and to the optimization of the threshold.

References

1. La Foresta, F., Morabito, F.C., Azzerboni, B., Ipsale, M.: PCA and ICA for the Extraction of EEG Dominant Components in Cerebral Death Assessment. Proc. of IJCNN (2005) 2532–2537.
2. Tyner, F.S., Knotte, J.R., Mayer, W.B.: Fundamentals of EEG Technology, vol. 2: Clinical Correlates. Raven Press NY (1989).
3. Chratian, G.: Coma, other states of altered responsiveness, and brain death. Daly D.D., Pedley T.A., Raven Press NY (1990) 425–487.
4. Chratian, G.: Electrophysiologic evaluation of brain death: A clinical appraisal. Electrodiagnosis in Clinical Neurology, Aminoff M. J. Ed., **1** (1986) 669–736.
5. Shewmon, D.: Chronic 'brain death': meta-analysis and conceptual consequences. American Journal of Neurology **51** (1998) 1538–1541.
6. Bennet, D.R., Hughes, J.R., Korein, J., Merlis, J.K., Suter, C.: An Atlas of Electroencephalography in Coma and Cerebral Death. Raven Press NY (1976).
7. Van de Velde, M., Ghosh, I.R., Cluitmans, P.J.: Context related artefact detection in prolonged EEG recordings. Computer Methods and Programs in Biomedicine **60** (1999) 183–196.
8. Jung, T., Humphries, C., Lee, T.W., Makeig, S., McKeown, M.J., Iragui, V., Sejnowski, T.: Removing Electroencephalographic Artifacts: Comparison between ICA and PCA. Neural Networks for Signal Processing **VIII** (1998) 63–72.
9. 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** (2004) 9–21.

10. Hyvarinen, A., Karhunen, J., Oja, E.: Independent Component Analysis. John Wiley & Sons Inc. (2001).
11. Bell, A.J., Sejnowski, T.J.: An information-maximisation approach to blind separation and blind deconvolution. Neural Computation **7** (1995) 1129–1159.
12. Lee, T.W., Girolami, M., Sejnowski, T.J.: Independent Component Analysis using an extended infomax algorithm for mixed sub-Gaussian and super-Gaussian sources. Neural Computation **11(2)**, (1999) 606–633.
13. Delorme, A., Makeig, S., Sejnowski, T.J.: Automatic Artifact Rejection for EEG Data Using High-Order Statistics and Independent Component Analysis. Proc. of ICA (2001).
14. Erdogmus, D., Hild II, K.E., Principe, J.C.: Blind source separation using Renyis marginal entropies. Neurocomputing **49** (2002) 25–38.

Sparse Bump Sonification: A New Tool for Multichannel EEG Diagnosis of Mental Disorders; Application to the Detection of the Early Stage of Alzheimer's Disease

François B. Vialatte and Andrzej Cichocki

BSI RIKEN ABSP Lab 2-1 Hirosawa, Wako-Shi, Saitama-Ken, 351-0198, Japan
fvialatte@brain.riken.jp, cia@brain.riken.jp

Abstract. This paper investigates the use of sound and music as a means of representing and analyzing multichannel EEG recordings. Specific focus is given to applications in early detection and diagnosis of early stage of Alzheimer's disease. We propose here a novel approach based on multi channel sonification, with a time-frequency representation and sparsification process using bump modeling. The fundamental question explored in this paper is whether clinically valuable information, not available from the conventional graphical EEG representation, might become apparent through an audio representation. Preliminary evaluation of the obtained music score – by sample entropy, number of notes, and synchronous activity – incurs promising results.

1 Introduction

The purpose of this paper is to investigate utilization of music and multimedia technology to create a novel procedure for EEG multichannel signals analysis that would be of clinical utility to medical practitioners and researchers. Sonification is the presentation of information as non speech sounds. Vision is the most important sense for human perception of space, however audition also convey useful complementary information in the time domain. Standard visually-based methods of analysis involve sophisticated processing and filtering of the data, so that by definition some spatial multidimensional aspects are illuminated at the expense of others. This paper proposes a flexible listening sonification system, by which one can adjust sonification to represent different electrodes and time-frequency dynamics: we may for instance consider the brain as an orchestra, where brain regions would represent different musical instruments. From this point of view, analyzing multichannel EEG signals using sounds seems a natural method: using a sonification of EEG signals, we would perceive simultaneously every channel, and analyze more tractably the time dynamics of the signals – hoping to gain new insights about the brain signals. However this metaphor stops here. As a matter of fact, in order to study EEG signals via a sonification process, three constraints for generating sounds should be taken into account:

- it should not loose relevant information,
- it should keep a biologic consistency with the original EEG signals (*i.e.* it should be biologically inspired),
- it should be designed for multi-channel EEG, as it is for this use that it will prove the most worthwhile.

Because of these constraints, and because of the noises inherent to EEG signal, a direct playback of the EEG (also termed as 'audification') would give an inaccurate representation [1].

The first constraint itself leads to a preliminary question: what will we consider as meaningful within EEG signals? Usually, EEG signals can be studied through several viewpoints, the most frequently investigated being:

- EEG signal amplitude (usually in given frequency bands),
- EEG signal oscillatory rhythms (time-frequency study),
- EEG synchronization

Furthermore, in order to preserve the consistency of the sonification with the EEG original signal, the sonification process should take into account the signals origin, *i.e.* brain areas the signals are recorded from (occipital, temporal, frontal, parietal, etc.), because these area are not involved in the same functional processes (since about 50 years, we know and have had confirmations that brain areas are functionally organized to deal with several distinct functions). For multichannel EEG this constraint implies an audio representation where each electrode's contribution can be identified. Finally, as human beings have to study the audio output, a tractable representation is mandatory, the last constraint will therefore be to produce sufficiently sparse musical scores. Up to now, few possibilities of sonification for the analysis of EEG data have been addressed: 'spectral mapping sonification' in frequency domain, audio alarm for a surgical instrument [2]; synchrony study with 'distance mapping sonification' [1]; 'model based sonification' for time-frequency (TF) analysis of epileptic seizures [3]; discrete frequency transform for brain computer interface [4] (see also this paper for a review about EEG sonification) – however to the best of our knowledge, none of these sonification process are able to solve all the constraints exposed above, which are required for a satisfactory EEG analysis method. The main purpose of this paper is to propose a new sonification satisfying these conditions, based on a sparse musical transformation of multichannel EEG that we will call bump sonification (BUS).

In the course of a clinical study [5], EEG signals from elderly patients in the early stage of Alzheimer's disease (AD) who developed AD within one year and a half, and from age matched controls were recorded in a 'rest eyes-closed' condition. We will present an application of BUS as a diagnosis method for the discrimination of these two groups of patients.

2 BUS: A Sparse Time-Frequency Sonification Method

From raw EEG signals, we seek to obtain a representation allowing the listening of EEG records obtained from a multi-channel EEG device. The Bump Sonification (BUS) method (Fig. 1) follows three steps:

- preprocessing, including artifacts removal and dimensionality reduction based on Blind Source Separation (BSS) or Independent component analysis (ICA),
- sparse TF representation,
- music generation (midi files).

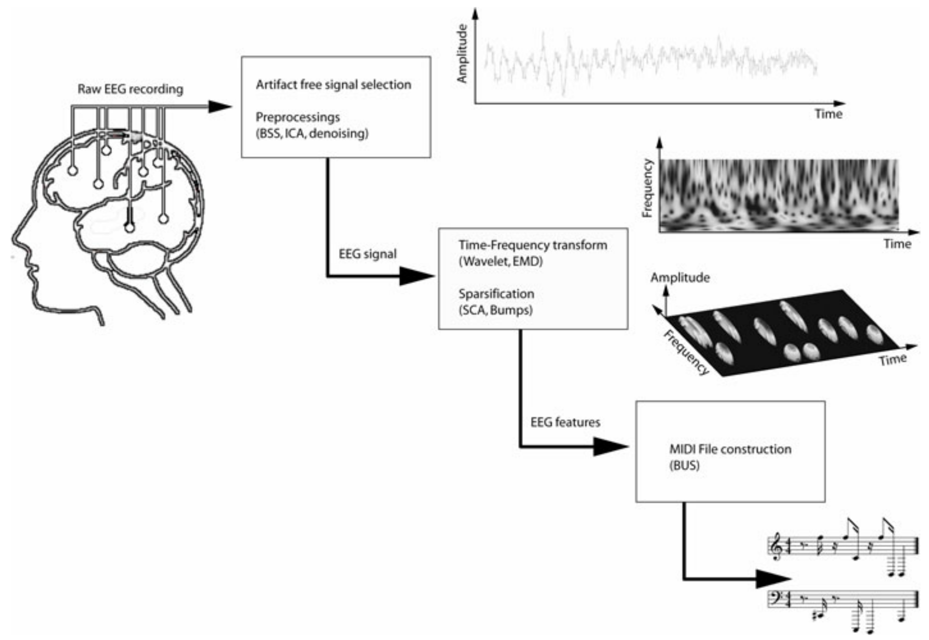

Raw EEG recording
Artifact free signal selection
Preprocessings (BSS, ICA, denoising)
Amplitude
Time
Frequency
Time
EEG signal
Time-Frequency transform (Wavelet, EMD)
Sparsification (SCA, Bumps)
Amplitude
Frequency
Time
EEG features
MIDI File construction (BUS)

Fig. 1. BUS model. From EEG signals, features are extracted using a sparsification process. TF representation of the signal is obtained, and the features extracted are used to build a MIDI (".mid") music sonification file. Illustrations on the right side are obtained from a scalp EEG record (2 sec, sampling frequency 1KHz): from top to bottom the signal, its wavelet TF representation, its bump model, and the final sonification are represented.

The TF representation is obtain using wavelet transform, with complex Morlet function (Eq. 1), which is well highly redundant but however well suited for TF analysis of electrophysiological signals [6] because of its symmetrical and smooth Gaussian shape both in time and frequency domains.

$$w(t) = A.\exp\left(-t^2 / 2\sigma_t^2\right).\exp(2i\pi ft) \tag{1}$$

where t is time, f is frequency, σ_t is the time deviation, and A is a scalar normalization factor. After the wavelet transform, c_{ft} coefficients describing time t and frequency f are obtained along all T time steps and all F frequency steps.

The main idea of the TF sparsification step (bump modeling [7],[8], Fig. 2) is to approximate a TF map with a set of elementary parameterized functions: a bump is adapted under constraints inside a TF window W, in order to minimize (using BFGS algorithm [9]) the cost function C:

$$C = \frac{1}{2} \sum_{t,f \in W} \left(z_{ft} - \beta(f,t)\right) \tag{2}$$

where the summation runs on all pixels within the window, z_{ft} are properly normalized TF coefficients at time t and frequency f, and $\beta(f,t)$ is the value of the bump function at time *t* and frequency *f*. In the present application, the TF coefficients c_{ft} where normalized to obtain z_{ft} z-score values by comparison with an average baseline, at each frequency bin *f*, from Control patients:

$$\forall t, z_{ft} = \frac{c_{ft} - \mathrm{M}_f}{\Sigma t} \tag{3}$$

where M_f is the average of baseline means m_p along time for all Control patients *p*:

$$M_f = \left\langle \overline{m_p} \right\rangle_p = \left\langle \sum_t^T \frac{c_{ft}^p}{T} \right\rangle_p \tag{4}$$

and Σ_f is the average baseline of standard deviations s_p along time for all Control patients *p*:

$$\Sigma_f = \left\langle \overline{s_p} \right\rangle_p = \left\langle \frac{1}{T-1} \sum_t^T \left(c_{ft}^p - m_p \right)^2 \right\rangle_p \tag{5}$$

This way, high normalized z_{ft} values represent patterns differing significantly from usual activity (which should convey the most significant information).

Half ellipsoid functions were found to be the most suitable bump functions:

$$\begin{cases} \beta(f,t) = a\sqrt{1-v} & \text{for } 0 \le v \le 1 \\ \beta(f,t) = 0 & \text{for } v > 1 \end{cases} \tag{6}$$

where $v = \left(e_f^2 + e_t^2\right)$ with $e_f = \left(f - \mu_f\right)/\, l_f$ and $e_t = \left(t - \mu_t\right)/\, l_t$.

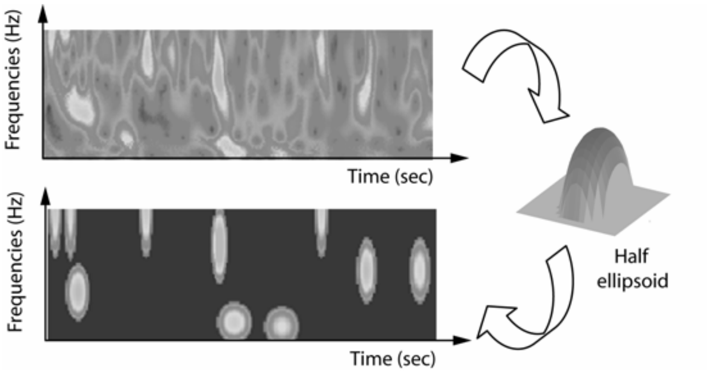


Fig. 2. Bump modeling, example. The TF map (*top*) is modeled by parameterized functions (*right*) into a more tractable model (*bottom*) highlighting the most prominent TF patterns.

μ_f and μ_t are the coordinates of the centre of the ellipsoid, l_f and l_t are the half-lengths of the principal axes, *a* is the amplitude of the function, *t* is the time and *f* the frequency.

Thereafter, the most prominent TF features from the artifact free EEG signals are obtained. This BUS model is a general scheme, which was already successfully applied for a brain computer interface application [10]. We will focus here on the application of BUS to the discrimination between PMCI and Control patients.

3 Application of BUS to Multichannel EEG Signals Investigation

3.1 General Method of Multi Channel Sonification

When more than one channel is under investigation – for instance, if one seeks to investigate global EEG dynamics, or EEG long-distance synchrony – we propose an approach (Fig. 3) inspired by the brain areas functional significance: if one is to study brain dynamics via EEG sonification, then brain areas whose functionalities are close should be represented by sounds which should be close; whereas brain areas whose functionalities are remote should be represented by easily differentiable sounds.
Therefore, for multi channel EEG representation, the music notes will be defined by the following parameters of the bumps:

- Amplitude of the bump will be converted into a velocity of the note played (valued between 40 and 127), *i.e.* how loud the note is played;
- Position of the electrode will be converted into the note pitch (in MIDI format, C4 note = pitch 60) scaled to a pentatonic scale (following pitch progressions such as 60-63-65-67-70) if there is many electrodes, depending on the electrode position on the head (close electrodes have close pitches, remote electrodes have distant pitches).
- Position and width in time of the bump will be converted into onset and duration of the note (in ticks per square).

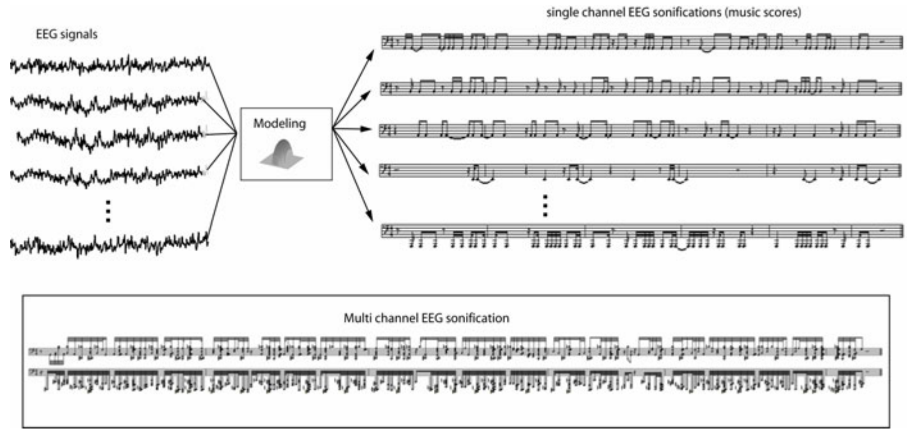

single channel EEG sonifications (music scores)
EEG signals
Modeling
Multi channel EEG sonification

Fig. 3. Multi channel EEG sonification example. This example is obtained from a 21 channel EEG lasting 20 sec, between 5 and 25 Hz. The huge amount of information is synthesized into a music score, and its listening highlights signal dynamics and synchrony.

This representation gives efficient representations of the brain dynamics. However, two caveats should be noticed:

- if the number of electrodes is too high, too much musical information may be produced: the sounds generated would become cacophonic;
- if the frequency span studied is to wide, too much musical information may be produced per channel, which leads to the same problem.

These two situations are different in type, but similar in nature. To avoid cacophony, we propose the following solutions:

- if the number of electrodes is too high, select the most representative electrodes (usually, when several electrodes are used, the signal is redundant) – this may be also obtained by a mathematical reduction (such as ICA projection), however, as we stated in the introduction, the biological realism imposes the constraint to regroup only functionally close electrodes (*i.e.* electrodes belonging to the same brain area), for instance regrouping redundant electrodes from frontal and occipital areas together would lead to a loss of synchrony information; the other solution would be to consider groups of electrodes;
- if the frequency span to study is wide, dividing it into frequency sub-bands to simplify the music scores.

3.2 Application to Early Detection of Alzheimer's Disease

Mildly impaired patients progression towards AD (PMCI) and age-matched control subjects were recorded in a 'rest eyes-closed' condition, with electrodes located on 21 sites according to the 10-20 international system. The first continuous artifact-free 20s interval of each recording were used to create two datasets: the PMCI group (n=22) and Control group (n=38). We assessed the capability of BUS to extract significant differences between dynamics of these groups.

As our previous reports [11] about Alzheimer's disease emphasized the importance of the theta range, and because this frequency band is slow (and will therefore give a more tractable representation than higher frequencies) we investigated the records from this database in the theta range (3.5-7.5 Hz), and applied the BUS method described above to generate 60 music scores (22 MCI, and 38 Control music scores).

We intended to highlight synchronization likelihoods uncovered in previous investigations in frontal and parietal areas [12], and therefore gathered the bump modeled from frontal areas (group1 = F3, F3, Fz) and parietal areas (group2=P3, P4, Pz). Low pitches (33, 35 and 37) were associated with group1, whereas high pitches (57, 60 and 63) were associated with group2, following pentatonic scales.

The model was assessed with mathematical complexity indexes as well as a perception test in which participants were asked to identify patients with PMCI (progressive mild cognitive impairment) and Control (healthy) by auditory and visual displays. The results show a high degree of accuracy in the identification of PMCI patients from control (healthy) subjects by the auditory displays (see samples available on internet [13]). We thereafter tried to analyze these rhythms divergences, by

Table 1. Results of three measures applied to the music scores from PMCI and Control groups: sample entropy, number of notes, and synchronous activity. Central columns indicates mean and standard deviations of the measures, right column indicates the Mann-Whitney p-value (testing for significant differences of median, highly significant when p<0.01). Synchronization is the most discriminative feature (bold p-value) between PMCI and Controls.

Feature	PMCI	Control	Mann-Whitney p-value[1]
Sa(2,1)	0.66±0.07	0.72±0.08	0.007
No	73.9±28	50.6±26	0.001
Sy (%)	3.52±1.04	5.10±1.96	**4e10^{-4}**

the computation of different measures of organization (Table 1): sample entropy (predictability of time series [14], Eq. 4-6), number of notes (Eq. 7), and synchronization (Eq. 8).

- Sample entropy is defined by:

$$Sa(m,r) = -\ln(A/B) \tag{7}$$

with

$$A = ((N-m-1)(N-m)/2)A^m(r) \tag{8}$$

and

$$B = ((N-m-1)(N-m)/2)B^m(r) \tag{9}$$

where N is the number of observations in the serie (here N is the total number of notes for the six electrodes), $A^m(r)$ the probability that two sequences will match for m+1 points, $B^m(r)$ the probability that two sequences will match for m points, and r is the tolerance for accepting matches. We used $Sa(2,1)$, therefore we looked upon organization along each different electrodes.

- The number of notes is simply the overall number of notes for the six electrodes.

[1] For *Sy*, standard deviations are not similar in PMCI and Control sets, as Mann-Whitney test is restricted to similarly shaped distributions we therefore log-normalized in order to obtain closest standard deviations before calculation of the p-value.

$$N_o = N \tag{10}$$

- The synchronization measure (Fig. 4) is defined by Equation 5:

$$Sy = \#V / N \tag{11}$$

where #V is the number of notes which own at least one neighbor in the following 200 msec (duplicates are withdrawn), which we deemed to be the largest biologically plausible time window for synchronous activity. We confirmed statistically significant differences between Control and MCI databases with all these measures using the Mann-Whitney statistical test for median differences, the best result being for the synchronization measure.

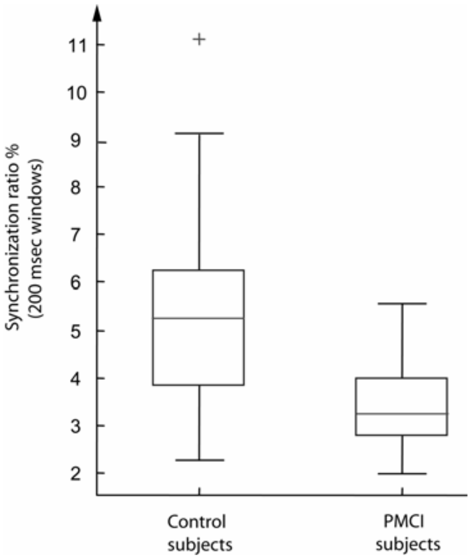


Fig. 4. Box plots of synchronization ratio in percentage, showing significant differences between Control (*left*) and PMCI (*right*) subjects

4 Discussion

We presented a biologically inspired method for multi channel EEG sonification, *i.e.* a method extracting TF components, and transforming these components into music, while keeping consistency with the EEG original signal.

This method were proven to be useful on a validation study, where two sets of data (records from patients at the early stage of Alzheimer's disease, and records from age-matched controls) are analyzed in term of musical complexity, and can be discriminated by human hearing. The results obtained concerning the AD early stage diagnosis are consistent with previous studies: brain dynamics evolution related to AD has been reported in several studies using coherence [15], mutual information [16] and synchronization likelihood [12],[17].

This sonification model can be fine-tuned for various frequency sub-bands and reflect unambiguously the oscillatory characteristics of MCI that may not be evident from a visual representation. The improvement of BUS resides in the fact that in contrast to visualization techniques, the temporal patterns extracted in the auditory domain by sonifications are usually better memorized by a trained neuroscientist than visual representations [1]. Our method merges multichannel EEG signals into a time-frequency-space representation (space for electrodes position on the scalp) and is therefore well-suited in order to carry out neurobiological investigations of brain dynamics, not only from a spectral or temporal point of view, but also through other EEG features, such as long-distance synchronization activities [18]. Since the identification of AD in early stage through EEG recordings is a current priority in neuroscience, sonification may become a valuable component in medical diagnosis.

References

1. Hermann, T., Meinicke, P., Bekel, H., Ritter, H., M¨uller H. M., Weiss, S., Sonifications for EEG data analysis. Proceedings of the 2002 International Conference on Auditory Display, Kyoto, Japan, July 2–5, 2002.
2. Jovanov, E., Starcevic, D., Karron, D., Wegner, K., Radivojevic, V., Acoustic Rendering as Support for Sustained Attention during Biomedical Procedures. International Conference on Auditory Display ICAD'89, 1998, Glasgow.
3. Baier, G., Hermann, T., The sonification of rhythms in human electroencephalogram. Proceedings of ICAD 04-Tenth Meeting of the International Conference on Auditory Display, 2004, Sydney, Australia.
4. Miranda, E. R., Brouse, A., Interfacing the Brain Directly with Musical Systems: On developing systems for making music with brain signals . Leonardo, 2005, 38(4):331-336.
5. Musha, T., Asada, T., Yamashita, F., Kinoshita, T., Chen, Z., Matsuda, H., Masatake, U., Shankle, W.R., A new EEG method for estimating cortical neuronal impairment that is sensitive to early stage Alzheimer's disease. Clinical Neurophysiology, 2002, 113(7): 1052-1058.
6. Tallon-Baudry, C., Bertrand, O., Delpuech, C., Pernier, J., Stimulus specificity of phase-locked and non-phase-locked 40 Hz visual responses in human. Journal of Neuroscience, 1996, 16:4240-4249.
7. Vialatte, F., Modélisation en bosses pour l'analyse des motifs oscillatoires reproductibles dans l'activité de populations neuronales : applications à l'apprentissage olfactif chez l'animal et à la détection précoce de la maladie d'Alzheimer. PhD Thesis, Paris VI University, Paris, 2005. [Available from http://www.neurones.espci.fr/Theses_PS/VIALATTE_F. pdf]
8. Vialatte, F.B., Martin, C., Dubois, R., Haddad, J., Quenet, B., Gervais, R., Dreyfus, G., A machine learning approach to the analysis of time-frequency maps, and its application to neural dynamics. Neural networks, in press.
9. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T., Numerical Recipes in C: The Art of Scientific Computing, 425 - 430. 1992, Cambridge Univ. Press, New York.
10. Rutkowski T.M., Vialatte F., Cichocki A., Mandic D.P., Barros A.K., Auditory Feedback for Brain Computer Interface Management - An EEG Data Sonifcation Approach. KES2006 10th International Conference on Knowledge-Based & Intelligent Information & Engineering Systems, Bournemouth, England, in press.

11. Vialatte F., Cichocki A., Dreyfus G., Musha T., Rutkowski T., Gervais R. Blind source separation and sparse bump modelling of time frequency representation of EEG signals: New tools for early detection of Alzheimer's disease. IEEE Sign. Proc. Soc. Workshop on MLSP 2005, Mystic, USA, 27-32.
12. Babiloni, C., Ferri, R., Binetti, G., Cassarino, A., Forno, G.D., Ercolani, M., Ferreri, F., Frisoni, G.B., Lanuzza, B., Miniussi, C., Nobili, F., Rodriguez, G., Rundo, F., Stam, C.J., Musha, T., Vecchio, F., Rossini, P.M., Fronto-parietal coupling of brain rhythms in mild cognitive impairment: A multicentric EEG study. Brain Research Bulletin, 2006, 69(1): 63-73.
13. Vialatte, F., MIDI multi channel EEG sonification of PMCI and Control subjects, (Fronto-Parietal multichannel sonification). Riken BSI, april 2006. [Available from http://www.bsp.brain.riken.jp/~fvialatte/data/Iconip2006_midi/sample.htm.]
14. Richaman J.S., Moorman J.R., Physiological time-series analysis using approximate entropy and sample entropy, American journal of physiology Heart and circulatory physiology, 2000, 278:H2039–H2049.
15. Adler, G., Brassen, S., Jajcevic, A., EEG coherence in Alzheimer’s dementia. Journal of Neural Transmission ,2003, 110(9):1051-1058.
16. Brinkmeyer, J., Grass-Kapanke, B., Ihl, R., EEG and the Test for the Early Detection of Dementia with Discrimination from Depression (TE4D): a validation study. International Journal of Geriatric Psyhiatry, 2004, 19:749-753.
17. Stam, C.J., Montez, T., Jones, B.F., Rombout, S.A.R.B., van der Made, Y., Pijnenburg, Y.A.L., Scheltens, Ph., Disturbed fluctuations of resting state EEG synchronization in Alzheimer’s disease. Clinical Neurophysiology, 2005, 116:708–715.
18. Varela, F., Lachaux, J.P., Rodriguez, E., Martinerie, J., The brainweb: phase synchronization and large-scale integration. Nature Reviews Neuroscience, 2001, 2(4):229-39.

Effect of Diffusion Weighting and Number of Sensitizing Directions on Fiber Tracking in DTI

Bo Zheng[1] and Jagath C. Rajapakse[1,2]

[1] BioInformatics Research Center, School of Computer Engineering
Nanyang Technological University, Singapore 639798
[2] Biological Engineering Division, Massachusetts Institute of Technology,
MA 02139, USA

Abstract. Diffusion Tensor (DT) fiber tracking techniques offer significant potential for studying anatomical connectivity of human brain *in vivo*. And the reliability and accuracy of fiber tracking results depend on the quality of estimated DT which is determined by parameters of image acquisition protocol. The aim of this paper is to investigate what echo-planar image (EPI) acquisition parameters: the number of sensitizing directions K and diffusion weighting b-value gives the best estimation of DT and shorter scan time. We carried out tracking on synthetic dataset that was artificially corrupted by various levels of Gaussian noise to study the effects of K and b-value on fiber tracking results, and to evaluate the quality of estimated DT. It was found that when K value larger than 13 and b-value larger than 800 smm^{-2} best estimated DTs. And further increments of K and b-value had no significant effect on quality of estimated DT.

Keywords: DT-MRI, DWI, DT, b-value, Gaussian noise, Fiber Tracking, fiber tracts.

1 Introduction

In recent years, Diffusion Tensor Magnetic Resonance Imaging (DT-MRI) is emerging as the only available medical image modality to non-invasively explore information about location, direction and extent of white matter (WM) fiber tracts (consisting of million of parallel nerve fibers) of human brain *in-vivo*, which is based on local principles of anisotropic water diffusion direction**.** In brain white matter (WM), water molecules diffuse the fastest in the direction parallel to fiber tract and the slowest in the direction perpendicular to the boundary of fiber tract. This is termed as *anisotropic water molecular diffusion*. Water diffusion in the presence of a strong magnetic gradient results in a loss of MR signal due to the de-phasing of spin coherence. The application of a pair of gradient to elicit differences in the water molecule diffusion among different biologic tissues is known as *diffusion weighting* [1]. DT-MRI consists of acquiring Diffusion Weighting Images (DWI)

$I_i, i = 1,2,...K;\ K \geq 6$, which reflect relative amount of diffusion along the different diffusion-sensitizing directions $g_i,\ i = 1,2,...K; K \geq 6$. In DT-MRI, diffusion tensor D that characterizes anisotropic water diffusion within a macroscopic voxel is estimated from the set of at least 6 DWIs with non-collinear and non-coplanar diffusion-sensitizing directions, which are uniformly distributed on a unit sphere surface, plus the non-diffusion weighting image I_0 (i.e. b = 0) [1]:

$$I_i = I_0 \exp(-b g_i \mathrm{D} g_i^T) \tag{1}$$

where b-value renders the amount of diffusion weighting. Since D is a symmetric 3-by-3 matrix, at least 6 DWIs are required to estimate it. However, due to artifacts caused by head motion, eddy current, etc., more than 6 DWIs are usually acquired, which allows robust estimation of diffusion tensors. It is expected that the quality of diffusion tensor data improves as the number of sensitizing direction increases.

The Stejskal-Tanner pulsed-gradient spin echo scheme (see Figure.1) is the most commonly implemented on the clinical MR scanners to acquire DWIs. The Stejskal-Tanner sequence uses a pair of gradient, systematically positioned around a 180 degree refocusing pulse allowing for controlled diffusion weighting. Here, b-value is determined according to the Equation below [1]:

$$b = \gamma^2 G^2 \delta^2 (\Delta - \delta / 3) \tag{2}$$

where γ is the gyromagnetic ratio; δ and G are the duration and strength of the sensitizing pulsed gradient, respectively; and Δ is the time interval between the two pulsed gradients. Since the intensity of MR signal increases as b-value increases, it is expected that larger b-value will lead to higher quality of estimated diffusion tensor.

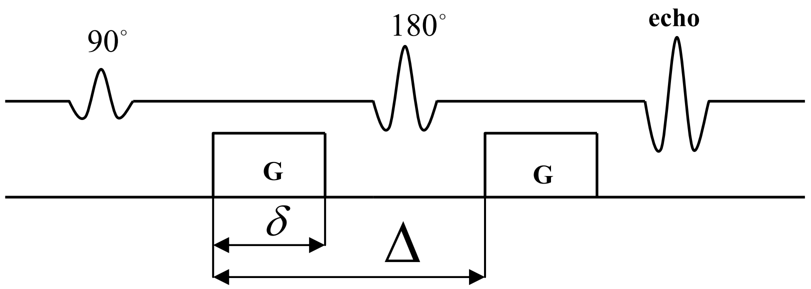


Fig. 1. The Stejskal-Tanner sequence used to acquire DW images

The diffusion tensor may be represented as an ellipsoid with the principal axes lengths related to the tensor eigenvalues ($\lambda_1 \geq \lambda_2 \geq \lambda_3$) and the directions given by the tensor eigenvectors (e_1, e_2, e_3). It is commonly assumed that the eigenvector e_1, also known as Principle Diffusion Direction (PDD), corresponding to the largest eigenvalue λ_1 is tangent to fiber tracts. However, although diffusion data provide directional information concerning microscopic tissue fiber orientation at voxel scale, it does not provide explicit connectivity between voxels. Therefore, fiber tracking algorithms have been developed to trace WM fiber tracts by diffusion tensor data. The performance of those fiber tracking algorithms relies on the quality of the estimated diffusion tensors. To evaluate the quality of estimated diffusion tensor based on different combination of K and b-value, we compared the similarity between fiber tracts reconstructed from noisy diffusion tensor data and the one reconstructed from noise-free data. We generated a synthetic diffusion tensor dataset for this purpose.

This paper is to investigate which combination of K and b-value gives the best estimation of diffusion tensor by using synthetic data. The synthetic data that simulate a helix were first built following the mathematical framework proposed by [2] and artificially corrupted by Gaussian noise. One of the most popular fiber tracking algorithms, line propagation model [3], was then chosen to be assessed. The fiber tracts were reconstructed by propagating a line from a seed point based on PDD of each voxel. Finally, a fiber similarity measure was proposed to evaluate the quality of estimated diffusion tensor.

2 Method

2.1 Synthetic Data

Synthetic dataset were generated to evaluate the effects of the number of sensitizing directions K and the b-value on the estimation of diffusion tensor. There are a number of reasons to use the synthetic data: (i) it is time-consuming to acquire DWIs with different combinations of K and b-value, especially when both factors are large, (ii) it is almost impossible to validate the fiber tracking result using real data due to lack of ground truth, and (iii) the signal to noise ratio of DWIs cannot be controlled for real data. For each combination of K and b-value, the synthetic data need to be tested at different levels of noise. Generally, the synthetic data can be built with known geometry of fiber tracts and controlled signal-to-noise ratio of DWIs.

The diffusion tensor D is a symmetric 3-by-3 matrix and eigenvalue decomposition gives:

$$D=VPV^{-1} \tag{3}$$

where V is the matrix whose columns are the eigenvectors (e_1, e_2, e_3); and P is diagonal matrix whose diagonal elements are the corresponding eigenvalues ($\lambda_1, \lambda_2, \lambda_3$). Therefore, to build the synthetic diffusion tensor data, the eigenvalues and the corresponding eigenvectors should be provided for each voxel. However, this is very exhaustive (e.g. 40x40x40 3D scan consists of 6,4000 voxels). A mathematical framework for the construction of synthetic diffusion tensor data proposed in [2] was used here. Instead of specifying the information for every voxel, the framework requires specifications only on points that lie on the skeleton of fiber tract; since this framework constructs the synthetic data in continuous coordinate system point is used here. Then, 3 eigenvectors and corresponding eigenvalues of other points can be computed based on the given information. Furthermore, this framework facilitates to build cylindrically symmetric fiber tract which is close to the physical structure of real fiber tract. This leads to a fiber tract that has the fastest water diffusion in the center and slower water diffusion as moving to its boundary (see Figure 2 (a)).

The skeleton of fiber tract is represented as a piecewise continuous 3D space curve. The helical geometry $r(t) = (k_1 \sin(2\pi t), k_1 \cos(2\pi t), k_2 t)$, $t \in [0,1]$ is sampled at t_i; $i = 1,\ldots,N$, to construct the skeleton. In general, PDD e_1 at a point is given by weighted sum of PDD of those points that lie on the skeleton and are the closest to the given point. And the weight is the distance between two points. The other eigenvector e_2 and e_3 can be found by solving the equations $\langle e_1, e_2 \rangle = 0$ and $e_1 \times e_2 = e_3$. The eigenvalues are computed in the similar way.

To build noisy synthetic diffusion tensor data, the noise-free DWIs are first generated using Eq. (1) with given K and b-value. In particular, non-diffusion weighting image, I_0, is estimated from fractional anisotropy (FA) (Eq. (4)). Then, Gaussian noise was added to DWIs. Here, the signal-to-noise ratio is defined as ratio between the mean intensity of non-diffusion weighting image and the standard deviation of Gaussian noise. Finally, the noisy synthetic diffusion tensor data can be re-estimated from noisy DWIs. Figure 2(b) shows one slice of noisy synthetic diffusion tensor data. Obvious difference can be observed compared to Figure 2(a) which shows one slice of noise-free synthetic tensor data.

$$FA = \sqrt{1.5}\frac{\sqrt{(\lambda_1 - \bar{\lambda})^2 + (\lambda_2 - \bar{\lambda})^2 + (\lambda_3 - \bar{\lambda})^2}}{\sqrt{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}} \tag{4}$$

with $\bar{\lambda} = \dfrac{\lambda_1 + \lambda_2 + \lambda_3}{3}$

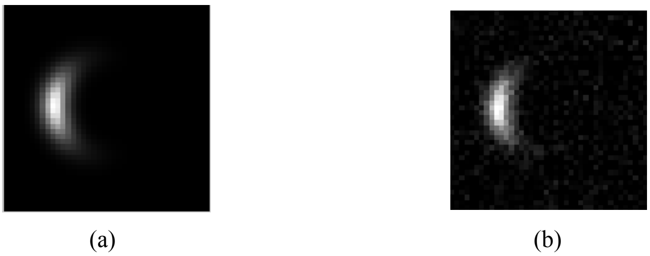


Fig. 2. Horizontal slice of synthetic diffusion tensor data showing fractional anisotropy map, (a) noise-free and (b) with noise; the brightness represents FA.

2.2 Fiber Tracking

The fiber tracking algorithm is a linear propagation approach called Fiber Assignment by Continuous Tracking (FACT) [3]. In brief, starting from user-defined seed points, fiber tracts are reconstructed from the diffusion tensor by propagating forward and backward, following the PDD. As given in Eq. (5), suppose the current point is p_i, the next point p_{i+1} along the path is calculated by adding the normalized PDD u_i

(u_i depends on which voxel p_i is inside) multiplied by the step size α. The tracking process is terminated when fractional anisotropy (FA), or the angles between two consecutive PDD, *C* in Eq. (6), or occurrence of sudden transition on the fiber orientation, *R* in Eq. (7), is smaller than a certain threshold value. Note that, interpolation of diffusion tensor is not used because it will reduce the noise effect on the fiber tracking result.

$$p_{i+1} = p_i + \alpha u_i \tag{5}$$

$$C = \langle u_i, u_{i-1} \rangle \tag{6}$$

$$R = \frac{2}{s(s-1)} \sum_{i=0}^{s-2} \sum_{j=i+1}^{s-1} \langle u_i, u_j \rangle \tag{7}$$

where u_i is PDD of current voxel and s is the number of voxels in the neighborhood.

2.3 Fiber Similarity

The reconstructed fiber tract can be represented as piecewise continuous 3D space curve. There are two factors that affect the similarity between two such curves. The first factor is the length of corresponding portions of two curves. The points located on corresponding portion of one curve have symmetric one-to-one relationship with those points located on corresponding portion of another curve. The second factor is the average distance between corresponding portions. It is expected that similarity between two curves increases as the first factor increases or the second factor decreases.

The most difficult problem in measuring fiber similarity between two tracts is the determination of corresponding portions of two curves. A weak symmetric one-to-one relationship is defined here. Suppose point x_1 from curve 1 has a corresponding point y from curve 2. And y has a corresponding point x_2 from curve 1. If difference $\|x_1 - x_2\|$ is very small, then y and x_1 are considered having symmetric one-to-one relationship. Then we can find out all pairs of point from two curves that satisfy the weak symmetric one-to-one relationship. The following explains how we define the similarity measure for two fiber tracts. The similarity between the reconstructed fiber using noise-free synthetic data, $t_s : (x_i : i = 1, \ldots M)$ and the one using noisy synthetic data, $t_e : (y_j : j = 1, \ldots, N)$ is here defined as:

$$S(t_s, t_e) = \frac{L_{cp}}{L_{t_s} + L_{t_e} - L_{cp}} \exp\left\{ -C \frac{\sum_{\forall (x_i, y_i) \in P} \|x_i - y_i\|}{|P|} \right\} \tag{8}$$

$$P = \{(x,y) \mid x \in t_s \wedge y \in t_e \wedge (y = N_x(x) \vee x = N_y(y)) \wedge$$

$$\|x - N_y(N_x(x))\| \le \varepsilon \wedge \|y - N_X(N_y(y))\| \le \varepsilon\}$$

$$L_{cp} = (\sum_{i=i\min}^{i\max-1} \|x_{i+1} - x_i\| + \sum_{j=j\min}^{j\max-1} \|y_{j+1} - y_j\|)/2$$

where the function $y = N_x(x)$ returns the point y from t_e which is the closest to the point *x* from t_s, and similarly for $x = N_y(y)$. *P* is the set of pairs of point from two tracts which have weak symmetric one-to-one relationship; L_{t_s} and L_{t_e} are the length of two tracts; *imin* and *imax* are the minimum and maximum index of x in *P*; *jmin* and *jmax* are the minimum and maximum index of y in *P*; and *C* and ε are constant. Similarity measure ranges between 0 and 1.

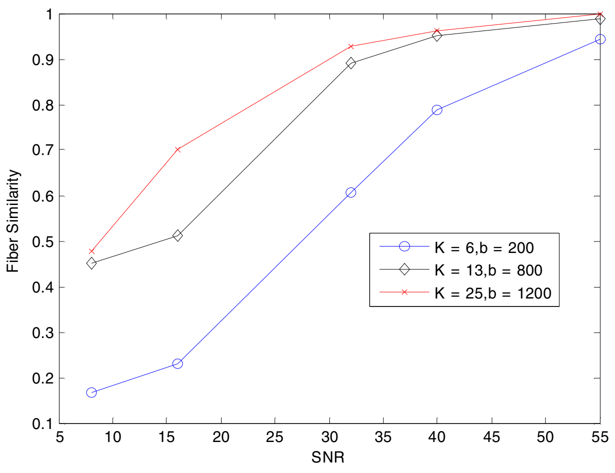
3 Results

Two noise-free 40x40x40 synthetic diffusion tensor dataset were built with the same skeleton of fiber tract but with different eigenvalues specified for points on the skeleton. Typically, for white matter, $\lambda_1 > \lambda_2 = \lambda_3$ and the anisotropic ratio λ_1 / λ_2 is in the range $[2.0, 10.0]$ [4]. For dataset 1, $\lambda_1 = 3\lambda_2 = 3\lambda_3 = 1260 \mu m^2 s^{-1}$; for dataset 2, $\lambda_1 = 6\lambda_2 = 6\lambda_3 = 1260 \mu m^2 s^{-1}$. The noisy synthetic diffusion tensor dataset were built for each noise-free dataset according to different combinations of K, b-value and signal-to-noise ratios. The following combinations were studied:

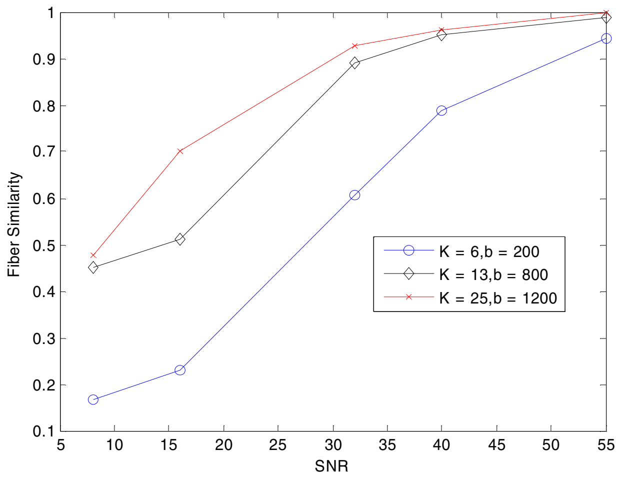
1. SNR = 8, 16, 32, 40, 55
2. K = 6, 13, 18, 25
3. b-value = 200, 500, 800, 1000, 1200 smm^{-2}

For each combination 10 fiber similarity measurements were taken. Then the mean and standard deviation of the measurements were calculated. The higher the fiber similarity, the better diffusion tensor estimation is.

The experimental results, shown in Figure 3, confirm the previous expectation that the quality of estimated diffusion tensor improves, i.e., fiber similarity measurement increases, as K and b-value increases whatever SNR and anisotropic ratio of dataset are. However, the amount of improvement decreases when SNR becomes larger. For example (see Figure 3(a)), when SNR = 8, the fiber similarity increases from 0.1681 ± 0.0168 to 0.4770 ± 0.0564. And when SNR = 55, the fiber similarity increases from 0.9439 ± 0.0324 to 0.9994 ± 0.0001. This implies that the effect



(a) $\lambda_1 / \lambda_2 = 6$



(b) $\lambda_1 / \lambda_2 = 3$

Fig. 3. Experimental results for three different combinations of K and b-value using (a) the synthetic diffusion dataset whose anisotropic ratio is 6, and (b) the synthetic diffusion dataset whose anisotropic ratio is 3. The mean values of 10 fiber similarity measurements are shown.

of K and b-value on the quality of estimated diffusion tensor is negligible when SNR of DWIs is above certain value. It can be observed from Figure 3 that, once SNR is larger than 40, K and b-value could be set to the possible smallest value to save the scan time. Given a SNR, the amount of improvement on quality of diffusion tensor also reduces as K and b-value increase. It was also found that significant improvement is achieved until K = 13 and b-value = $800\,\mathrm{smm}^{-2}$ for the two datasets whose SNR is below 40.

4 Conclusion

We studied the effect of the number of sensitizing directions K and diffusion weighting b-value for best estimation of diffusion tensor as well as the shortest scan time. Due to lack of ground-truth for real data, a synthetic diffusion tensor data was used with helical geometry and tested when corrupted by certain amount of Gaussian noise by using various combinations of K and b-values. The fiber tracking was then performed on noise-free and noisy synthetic diffusion datasets. And a new measure was introduced to determine the similarity between the fiber tracts reconstructed from noisy data and the one reconstructed from noise-free data. Such fiber similarity was used to evaluate the quality of estimated diffusion tensor. Finally, it was concluded that considering acquisition time as well as quality of estimated diffusion tensor, the best combination is K = 13 and b-value = $800\,\mathrm{smm}^{-2}$. In addition, if the SNR of real DWIs is larger than 40, effect of K and b-value are negligible.

References

1. Elias R. Melhem, Susumu Mori, Govind Mukundan, and etc: Diffusion Tensor MR Imaging of the Brain and White Matter Tractography. AJR (2002). Vol. 178,3-16
2. A.Leemans, J. Sijbers, M. Verhoye, A.Van der Linden, and D. Van Dyck.: Mathematical Framework for Simulating Diffusion Tensor MR Neural Fiber Bundles. Magnetic Resonance in Medicine (2005). Vol. 53,944-953
3. Mori S, Crain BJ, Chacko VP, and van Zijl PCM: Three dimensional tracking of axonal projections in the brain by magnetic resonance imaging. Ann Neurol (1999). Vol. 45, 265-269
4. Sinisa Pajevic, Akram Aldroubi, and Peter J. Basser: A Continous Tensor Field Approximation of Discrete DT-MRI Data for extracting Microstructural and Architectural Features of Tissue. Journal of Magnetic Resonance (2002). Vol. 154,85-100

3-D Reconstruction of Blood Vessels Skeleton Based on Neural Network

Zhiguo Cao and Bo Peng

Institute of Image Processing and Artificial Intelligence Huazhong University of Sci&Tec
State Key Lab for Image Processing and Intelligent Control Wuhan 430074, Hubei, China
zgcao@hust.edu.cn, pbo820722@yahoo.com.cn

Abstract. For the 3-D reconstruction of blood vessels skeleton from biplane angiography system, an efficient 3-D reconstruction method based on neural network(NN) is proposed in this paper. First, we find a set of 2-D corresponding points on the vessels' skeleton in the matched image pair. Secondly, NN is utilized to build the relationships between the 3-D points and their projective 2-D points. Thirdly, by feeding the corresponding points we found into the NN, the 3D coordinates of the points on vessels can be obtained. At last we employ B-spline interpolation to improve reconstruction performance. Experiment results demonstrate the efficiency of the new method.

Keywords: biplane angiography, 3-D reconstruction, neural network, B-spline.

1 Introduction

The recovery of 3-D blood vessels is an important application in medical image processing. In traditional way of realizing conversion from 2-D to 3-D points, a precise calibration is required to determine the relationships between the two cameras and the global coordinate system[1]. And 3D vessels reconstruction techniques follow a bottom up approach based on image feature extraction and reconstruction by interactively indicating corresponding point projections[2][3][4]. However, because the methods are based on optical and perspective geometry, the relationships between the 3-D points and their projective 2-D points are complex. And the realization of the algorithm is complicated. Moreover, its results cannot represent the blood vessels continuously and correctly because of the difficulties in precisely determining corresponding points. So we can't use all of the points as processing cells.

The purpose of this paper is to propose an efficient method for 3D reconstruction of blood vessels in order to avoid the problems mentioned above. Instead of finding a great deal of corresponding points, we only pick up parts of them on the vessels' skeletons in the image pair and employ neural network (NN) to build the relationships between the 3-D points and their projective 2-D points . After feeding the image corresponding points which we have chosen into the input layer, the 3D position of these points can be computed by the NN we designed. Compared with the traditional way, the advantage of this method is that relationships between 2-D points and 3-D points can be established automatically by NN without explicitly deducing the exact functions explicitly. Moreover, employing NN is not only a process of reconstruction but also an optimization technique. At last, through these points reconstructed, the 3-D skeleton of the vessel is presented by a 3rd order interpolated B-spline[5]. We use rational B-spline preserved by perspective transformation, in order to keep the correspondence between the 3D curve and its two 2-D projections. Experiment results show that satisfying reconstruction performance can be obtained more efficiently.

2 Description of Problem

The Biplane angiography system is presented as Fig. 1.

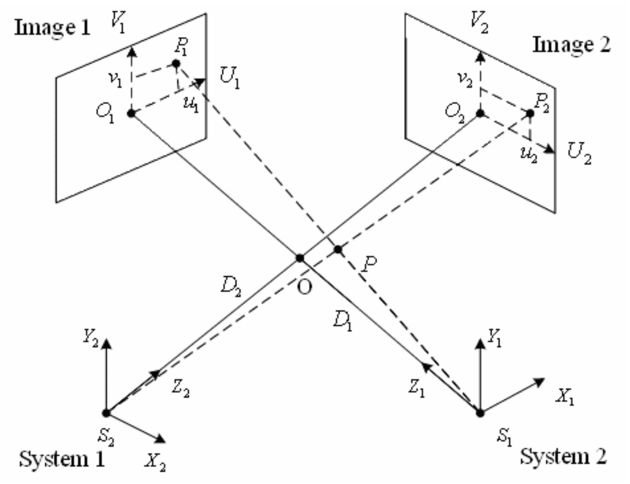


Fig. 1. Biplane angiography system

S_i is the focal-spot in system i . D_i is the distance (S_iO_i) from focal-spot to imaging-plane(SID).A scene point P is expressed in system 1 as (x_1, y_1, z_1) .The corresponding point (x_2, y_2, z_2) can be obtained by translation and then rotation as shown in Equ(1):

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \mathbf{R} \times \left\{ \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} - \mathbf{t} \right\}, \mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} ; \mathbf{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} . \quad (1)$$

Assuming that we have obtained two images from the system and known the projection angles LAO/RAO, CRAN/CAUD and $S_1\mathrm{O}, S_2\mathrm{O}$, then we can get the rotation matrix $\mathbf{R}$ and translation vector $\mathbf{t}$ [6] .

According to perspective geometry, a projection of a given 3-D point onto 2-D point can be represented by:

$$\xi_i^j = \frac{u_i^j}{D_i} = \frac{x_i^j}{z_i^j}\,,\ \eta_i^j = \frac{v_i^j}{D_i} = \frac{y_i^j}{z_i^j}\ ,\ i = 1,2\ ;\ j = 1,2\ldots n\,. \tag{2}$$

Here we assume there are n points. $\left(x_i^j, y_i^j, z_i^j\right)$ denotes the coordinate of the jth point on the vessels in system i . $\left(u_i^j, v_i^j\right)$ is its corresponding 2-D coordinate in the projection plane i, it is located at the position where a ray from the focal spot through the object intersects the image plane.

Then we obtain Equ.(3) from Equ.(1) and Equ.(2):

$$\begin{bmatrix} r_{11} - r_{31}\xi_i^j & r_{12} - r_{32}\xi_i^j & r_{13} - r_{33}\xi_i^j \\ r_{21} - r_{31}\eta_i^j & r_{22} - r_{32}\eta_i^j & r_{23} - r_{33}\eta_i^j \\ 1 & 0 & -\xi_i^j \\ 0 & 1 & -\ \eta_i^j \end{bmatrix} \times \begin{bmatrix} x_i^j \\ y_i^j \\ z_i^j \end{bmatrix} = \begin{bmatrix} \mathbf{a}\cdot\mathbf{t} \\ \mathbf{b}\cdot\mathbf{t} \\ 0 \\ 0 \end{bmatrix}. \tag{3}$$

$$\mathbf{a} = \begin{bmatrix} r_{11} - r_{31}\xi_i^j \\ r_{12} - r_{32}\xi_i^j \\ r_{13} - r_{33}\xi_i^j \end{bmatrix}\ ;\ \mathbf{b} = \begin{bmatrix} r_{21} - r_{31}\eta_i^j \\ r_{22} - r_{32}\eta_i^j \\ r_{23} - r_{33}\eta_i^j \end{bmatrix}\ .$$

After the geometry of the biplane imaging system ($\mathbf{R}$ and $\mathbf{t}$) has been found, given the coordinates of a pair of matched points $\left(u_1^j, v_1^j\right)$, $\left(u_2^j, v_2^j\right)$ in both image planes (the Epipolar geometry [7] is used to find them), their corresponding 3-D position $\left(x_i^j, y_i^j, z_i^j\right)$ can be computed by employing Equ. (3).

The process to get Equ.(3) from Equ.(1),(2) is not convenient because the knowledge based on optical and perspective geometry is rather complex. And Equ.(3) should return a least-squares solution.

3 Reconstruction Based on Neural Network

In order to Reconstruct the vessel tree, epipolar geometry algorithm[7] is adopted for explicitly matching each point on the tree axis. However, epipolar constraint is very sensitive if the perspective viewing angle is small, especially when the image is very noisy. So it is impractical to find all the corresponding points on vessels exactly. Otherwise, the result is rough and discontinuous.

To avoid this, we find parts of corresponding points in the image pair to represent vessels' skeleton. After that, in order to find the depth information conveniently, instead of the traditional way, we employ the NN system. The advantage is that the relationships between 2-D and 3-D points can be established automatically by NN without explicitly deducing the exact functions and NN can do reconstruction and optimize the result simultaneously.

Here we construct a three-layers NN to perform the operation. The input layer has 4 nodes corresponding to the values of the matched points $\left(u_1^j, v_1^j\right)$, $\left(u_2^j, v_2^j\right)$ on the image pair. The number of neurons for simulating the complex relationship between 2-D image and 3-D space in the hidden layer is determined based on experiment. The output layer has 1 node corresponding to the 3-D points depth information z_i^j. The architecture of the NN is given as Fig.2(a). Fig.2(b) shows a typical neuron in NN. P_i ($i = 1 \ldots 4$) are the inputs, $w(i,1) \ldots w(i,4)$ are weights for ith neuron. f_i is the transition function . O_i is the output.

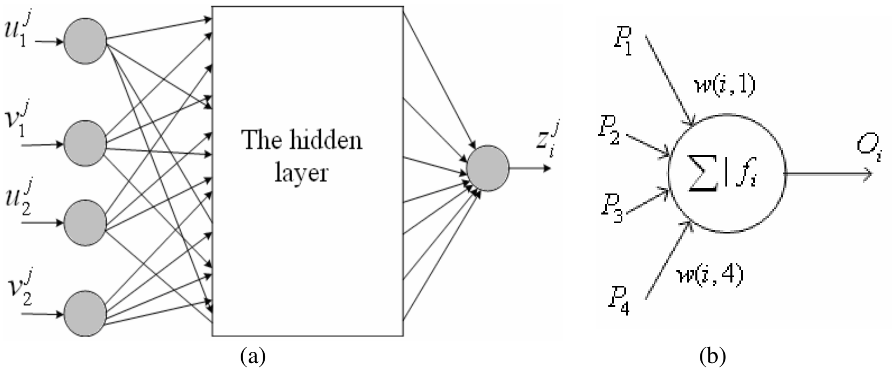


Fig. 2. (a)The architecture of neural network for 2D to 3D point conversion . (b) Construction of a neuron.

In order to recover the vessels in system 1(Fig.1), network needs to be trained. First, we produce enough 3-D sample points between focal spot and image plane in system1. Secondly, according to the known projection angles LAO/RAO, CRAN/CAUD and distances S_1O, S_2O, $\mathbf{R}$ and $\mathbf{t}$ which can rotate and translate the points in system 1 to system 2 can be deduced[6] . After rotating and translating these points, we can get their corresponding 3-D positions in system 2. Next step, if D_i (Fig.1) is known, projective 2-D points $\left(u_i^j, v_i^j\right)$ of these sample 3-D points in each image plane i are obtained by Equ.(1). At the same time, the depth information z_1^j of the 3-D sample points in system1 and the coordinates of their corresponding 2-D points are saved.

After training, feeding the 2-D points on vessels we have chosen into the input layer, the NN can output the true depth information z_1^j of 3-D points of the object in system 1.

In Section 5, we test our method with different neural networks.

4 B-spline Interpolation

After reconstruction, the initial 3-D curve representing the vessel is built using a 3rd order interpolated B-spline[5] through these points. Since B-spline is continuous and smooth, a better visualization can be achieved. We use rational B-spline which is preserved by perspective transformation, in order to keep the correspondence between the 3D curve and its two 2-D projections.

5 Experiment

We have tested the approach presented in this paper with a set of angiography images. Here we give an example of our experimental results. Fig.3(a) shows the DSA (Digital Subtraction Angiography) images of a vessel tree which are acquired with the angulations of LAO= 30°, CRAN= 45°, RAO= 60°, CAUD= 45° and the SID of each image system is the same. Resolution of images is 0.35 mm/pixel.

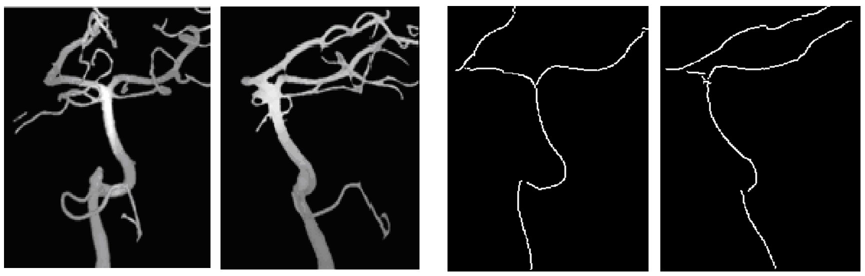


Fig. 3.(a) Two DSA images from different projection angles of a vessel tree. (b) Their extracted centerlines.

Fig.3(b) shows the skeleton images extracted from each view[8]. Only main vessels are focused on.

We test our algorithm by BP net, RBF net and GRNNs net, with the structure described in section 3. In training, the number of sample points produced in 3D space is 200. The sum square error (SSE) expected is 0.5.

For BP net, considering the time cost and reconstruction performance, we choose 50 as the number of neurons in hidden layer. Levenberg-Marquardt algorithm is utilized to train the BP net.

For RBF net, we don't give the number of neurons in hidden layer. In training, the number will increase automatically until the net satisfies the SSE. The number of neurons during training is shown in Table 1.

For GRNNs net, we only give the SSE. Comparison of the results using different neural networks are shown later.

Table 1. The increasing number of neurons in RBF and corresponding training error(SSE)

Neurons	0	25	50	75	100
SSE	7.54813	5.60639	3.05885	2.10325	1.0891

Here we compare the reconstruction results by BP net with traditional way. Fig.4(a) is the recovered vessel by BP net. Fig.4(b),(c) show the projections on the two original images. Fig.5(a),(b),(c) show the vessel reconstructed in traditional way.

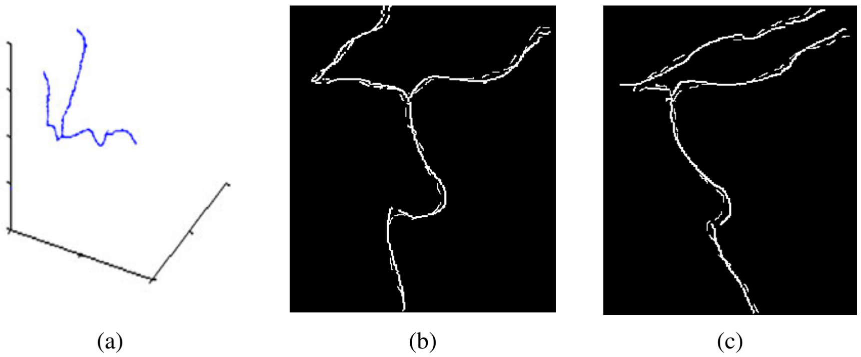


Fig. 4. (a) Reconstruction result using BP network. (b),(c) Projections on the two original image planes: True position of the vessel (*real lines*) and projections of the reconstruction result (*dash lines*).

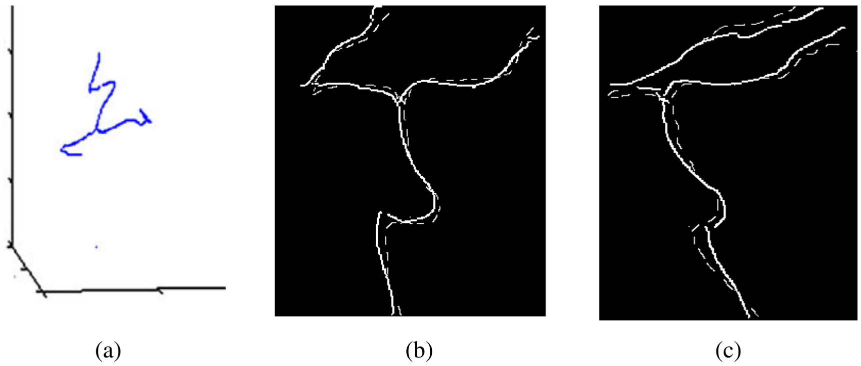


Fig. 5. (a) Reconstruction result in traditional way. (b),(c) Projections on the two original image planes: True position of the vessel (*real lines*) and projections of the reconstruction result (*dash lines*).

It can be observed from Fig.4 and Fig.5 that result by BP net has been improved significantly, compared with traditional way.

Fig.6(a) shows one of the recovered vessels which employs all the points on the vessel. (b) shows the result when the vessel is reconstructed by parts of corresponding points and we improve the result by B-spline interpolation. We can see that the result in (a) isn't as smooth and continuous as in (b).

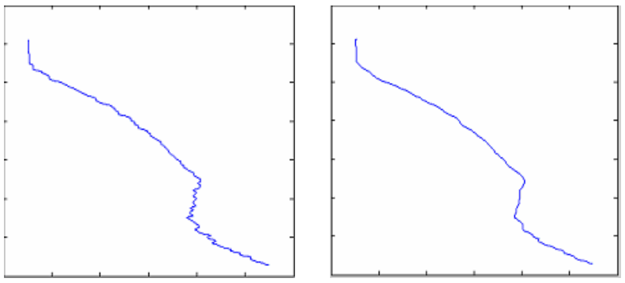


Fig. 6. (a)One of the recovered vessels presented without B-spline.(b)One of the recovered vessels presented by B-spline.

We define reconstruction errors in terms of the squares of Euclidean distances between the 2D input data and the projections of the calculated 3D data points onto the two image planes.

$$\varepsilon(P_i) = (p'_{1i} - p_{1i})^2 + (p'_{2i} - p_{2i})^2 . \tag{4}$$

Equ.(4) denotes the square of the distance between the input image data and the position of the calculated 3D position of the ith point after it is projected onto the the two image planes 1,2.Where P_i denotes the sets of 3D object position vectors for i=1,2…n.

Table 2 shows the differences between traditional way and the methods proposed in this paper.

Table 2. Reconstruction errors compared between traditional way and neural networks with B-spline

	Traditional Way	BP	RBF	GRNNs
Maximal error(mm)	6.45	4.88	4.75	6.0
Minimal error(mm)	0	0	0	0
Average error(mm)	0.86	0.45	0.41	0.62
Standard deviation(mm)	2.842	1.075	1.045	2.237
Training time(s)		8	18	5

From Table 2, we can see that NN performs better than traditional way and BP net and RBF net yield better reconstruction performance than GRNNs method does. However, RBF net needs more neurons in hidden layer and more training time.

We have tested our methods on a set of angiography images and get almost the same conclusion.

6 Conclusion

The proposed reconstruction algorithm based on neural network(NN) has several advantages over other traditional geometric methods. First, unlike geometric approaches, the relationships between 2-D points and 3-D points can be established automatically by NN without explicitly deducing the exact functions. Another advantage is that NN can do reconstruction and optimize the result simultaneously.

We test our method with a set of angiography images by Bp net, RBF net and GRNNs net. Compared with traditional way, all of them can improve reconstruction results. BP net and RBF net yield higher reconstruction accuracy than GRNNs method does. Otherwise, RBF net requires more neurons in hidden layer and more training time than BP net and GRNNs net.

Another contribution of our method is that it needs only parts of the corresponding points to represent the vessel and B-spline is employed to improve the reconstruction performance. So we don't need to find all the corresponding points on vessels.

Moreover, this model is especially efficient in condition that a large number of points need to be recovered. The application of our algorithm in other systems for reconstruction from 2-D to 3-D is to be addressed in future work.

References

1. Milan, S., Vaclav, H., Roger, B.: Image Processing, Analysis, and Machine Vision(Second Edition). Thoms Brooks/Code: People's Post and Telecom Press (2002) 469-471.
2. Dumay, A.C.M., Reiber, J.H.C. Gerbrands, J.J.: Determination of Optimal Angiographic Viewing Angles: Basic Principles and Evaluation Study. Med. Imaging. Vol.13 (1) (1994) 13-24.
3. Wahle, A., Oswald, H., Schulze ,GA., Beier , J., Fleck, E.: 3D Reconstruction, Modelling and Viewing of Coronary Vessels. Computer Assisted Radiology(1991) 669-676.
4. Wahle, A., Oswald, H., Fleck, E.: 3D Heart-Vessel Reconstruction from Biplane Angiograms. IEEE Computer Graphics and Applications Vol.16(1) (1996) 65-73
5. Milan, S., Vaclav, H., Roger, B.: Image Processing, Analysis, and Machine Vision Thoms Brooks/Code: People's Post and Telecom Press (2002) 245-248.
6. Daoyin,Y., Jiaxiang, H.: Study on theoretical models for 3D reconstruction of coronary arterial tree. Journal Of Engineering Graphics. No.4 (2003) 83-89.
7. Milan, S., Vaclav, H., Roger, B.: Image Processing, Analysis, and Machine Vision(Second Edition). Thoms Brooks/Code: People's Post and Telecom Press (2002) 458-459.
8. Xiao-xiao, L., Zhiguo, C., Baopu, L.: Automated Centerline Extraction of Angiogram Vessels Based on Multiscale Gabor Filters. Journal of Image and Graphics. Vol.10(12) (2005) 1542-1547

Design of a Fuzzy Takagi-Sugeno Controller to Vary the Joint Knee Angle of Paraplegic Patients

Marcelo C.M. Teixeira[1], Grace S. Deaecto[1], Ruberlei Gaino[2], Edvaldo Assunção[1], Aparecido A. Carvalho[1], and Uender C. Farias[3]

[1] UNESP-São Paulo State University, Department of Electrical Engineering, Campus of Ilha Solteira, 15385-000, Av. Brasil 56, Ilha Solteira, São Paulo, Brazil
[2] Londrina State University, Department of Electrical Engineering, Campus of Londrina, 86051-990, Londrina, Paraná, Brazil
[3] UNIDERP-NEAC, R. Ceará 333, 79040-000, Campo Grande, Mato Grosso do Sul, Brazil
marcelo@dee.feis.unesp.br, grace_2227@yahoo.com.br, rgaino@aluno.feis.unesp.br, edvaldo@dee.feis.unesp.br, aac@dee.feis.unesp.br, uendercf@yahoo.com.br

Abstract. The papers shows, through theoretical studies and simulations, that using the description of the plant by Takagi-Sugeno (T-S), it is possible to design a nonlinear controller to control the position of the leg of a paraplegic patient. The control system was designed to change the angle of the joint knee of 60°. This is the first study that describes the application of Takagi-Sugeno (T-S) models in this kind of problem.

Keywords: Takagi-Sugeno Fuzzy Models, Nonlinear Control, Paraplegia, Rehabilitation Engineering, Functional Electrical Stimulation.

1 Introduction

Several researchers have been used Functional Electrical Stimulation (FES) to restore some motion activities of persons with the spinal cord injured [1]. However, FES is not yet a regular clinical method because the amount of effort involved in using actual stimulation systems still outweighs the functional benefits they provide. One serious problem using FES is that artificially activated muscles fatigue at a faster rate than those activated by the natural physiological processes. Due these problems a considerable effort has been directed toward developing FES systems based on closed-loop-control. The movement is measured in real time with several types of sensors and the stimulation pattern is modulated accordingly [1].

The dynamics of the lower limb were represented by a nonlinear second order model, which took account of the gravitational and inertial characteristics of the anatomical segment as into well as the damping and stiffness properties of the knee joint.

In this paper we present a Takagi-Sugeno nonlinear system with the aim of controlling the position of the leg of a paraplegic patient. The controller was designed in order to change the angle of the joint knee from 0° to 60° when electrical stimulation is applied in the quadriceps muscle. We considered the leg mathematical model proposed by [1]. The authors showed that for the conditions considered in their experiments, a simple one-pole transfer function was able to model the relationship between stimulus pulse width and active muscle torque.

2 Takagi-Sugeno Fuzzy Control

In this section, we present a systematic procedure to design fuzzy control systems that involves the neuro-fuzzy model construction for nonlinear systems [2]. Muscle is a highly complex nonlinear system capable of producing the same output for a variety of inputs. A property exploited by physiologically activated muscle is its effort to the minimize fatigue [3]. Considering that when the quadriceps is electrically stimulated is present a nonlinear response, we used T-S fuzzy models in order to design a controller to vary the angle of the knee. T-S fuzzy models have been used to represent fuzzy-neural-linear control systems, for stability analysis [4], [5].

2.1 Takagi-Sugeno Fuzzy Model

The T-S fuzzy model is given in (2), and more details can be found in [6]. Let z the premise vector gives by. Now define

$$z \in R^{px1},\ \alpha_i\left(z(t)\right) \geq 0 \ and \ \sum_{i=1}^{r} \alpha_i\left(z(t)\right) = 1,\ \alpha = \left[\alpha_1, \ldots, \alpha_r\right]^T \in R^{1xr} \tag{1}$$

A T-S continuous time fuzzy model is defined as follows.

$$\begin{aligned} \dot{x}(t) &= \left(\sum_{i=1}^{r} \alpha_i\left(z(t)\right) A_i\right) x(t) + \left(\sum_{i=1}^{r} \alpha_i\left(z(t)\right) B_i\right) u(t) \\ &= A(\alpha) x(t) + B(\alpha) u(t), \quad i = 1, 2, \ldots, r, \end{aligned} \tag{2}$$

where $x(t) \in R^n$ is the state vector and $u(t) \in R^m$ is the input vector. An analytical approach to obtain local models for a class of nonlinear systems is described in [6].

2.2 Dynamic Model Used in the Control of the Joint Knee Angle of Paraplegic Patients

In this work, the mathematical model of the lower limb is the same proposed by [1]. This model relates the width of the applied pulse with the torque generated around the articulation of the knee. In [1] the authors considered the lower limb as an open kinematics chain composed of two rigid segments: the thighs, and the shank-foot complex, in showed Fig. 1. The equation that describes the dynamic of the junction of the knee is:

$$J\ddot{\theta}_v = -mglsen\left(\theta_v\right) - M_s - B\dot{\theta} + M_a, \tag{3}$$

where: J is the inertial moment of shank-foot complex, θ is the knee joint angle (angle between shank and thigh in the sagittal plane), $\dot{\theta}$ is the knee joint angular velocity, θ_v is the shank angle (angle between shank and the vertical direction in the sagittal plane), $\ddot{\theta}_v$ is the angular acceleration of the shank, *m* is the mass of shank-foot complex, *g* is the gravitational acceleration, *l* is the distance between knee and center of mass of shank foot complex, *B* is the viscous coefficient, M_s is the torque due to joint stiffness component, M_a is the active knee torque produced by electrical stimulation, M_d is the torque due damping (component that depend respectively on the knee angle and angular velocity), M_i is the torque due inertial , M_g is the torque gravitational, τ is the time control of the pole, *G* is the static gain.

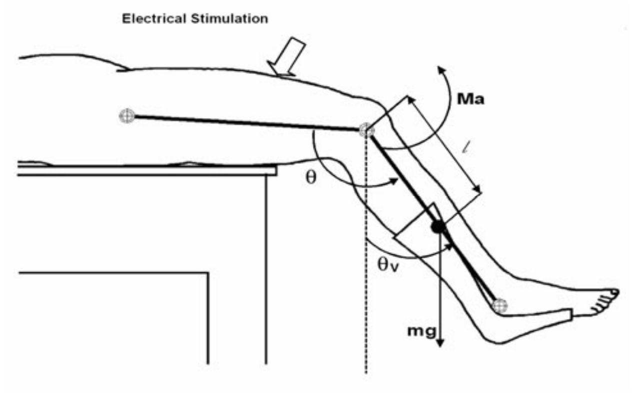


Fig. 1. The lower limb parameters

The dynamic equilibrium of these components around knee joint was represented by the following equation, [1], [7]:

$$M_i = M_g + M_s + M_d + M_a, \tag{4}$$

$$M_s = -\lambda e^{-E\theta}(\theta - \omega). \tag{5}$$

where λ and *E* are the coefficients of the exponential terms and ω is the resting elastic knee angle. According [1], the torque that the muscle will be subject (M_a) and the width of the pulses of the to electric stimulation (*P*) can be appropriately related by the following transfer function:

$$H(s) = \frac{M_a(s)}{P(s)} = \frac{G}{1+s\tau}. \tag{6}$$

The authors consider the following values of τ and *G*, obtained graphically in [1], as well as the other parameters of interest:

$$\begin{gathered}\tau = 0.951s,\ B = 0.27[N.m.s/rad],\ \omega = 2.918[rad] \\ \lambda = 41.208[N.m/rad], E = 2.024[1/rad],\ G = 42500[N.m/\mu s].\end{gathered} \tag{7}$$

Substituting M_s from (5) in (3), and considering from Fig. 1 that $\theta = \theta_v + \pi/2$.,we have

$$\ddot{\theta}_v = \frac{1}{J}\left[-mgl\sin\theta_v - \lambda e^{-E\theta_v} e^{-E\frac{\pi}{2}}\left(\theta_v + \frac{\pi}{2} - \omega\right) - B\dot{\theta}_v + M_a \right] \tag{8}$$

In the operation point, θ_{v0}=60º, considering (3) and (5), we have

$$M_{ao} = mglsen\left(\theta v_0\right) + \lambda e^{-E\left(\theta v_0 + \frac{\pi}{2}\right)}\left(\theta v_0 + \frac{\pi}{2} - \omega\right) \tag{9a}$$
$$= 8.76525 Nm,$$

$$P_0 = \frac{M_{ao}}{G} = 2.06241.10^{-4}. \tag{9b}$$

For convenience, we make a change of variables so that (8) and (9) are rewritten in terms of ΔM_a and $\Delta\theta_{v,}$. The exact T-S fuzzy model was described using these new variables. Defining,

$$\begin{aligned} &\Delta\theta_v = \theta_v - \theta_{vo},\ \theta_v = \Delta\theta_v + \theta_{vo},\ \dot{\theta}_v = \Delta\dot{\theta}_v, \\ &\ddot{\theta}_v = \Delta\ddot{\theta}_v,\ \Delta M_a = M_a - M_{ao}. \end{aligned} \tag{10}$$

We can rewriter the equation (8):

$$J\Delta\ddot{\theta}_v = \left[\frac{-mglsen\left(\Delta\theta_v + \theta_{vo}\right) - \lambda e^{-E\left(\Delta\theta_v + \theta_{vo} + \frac{\pi}{2}\right)}\left(\Delta\theta_v + \theta_{vo} + \frac{\pi}{2} - \omega\right) + M_{ao}}{\Delta\theta_v} \right]\Delta\theta_v - B\Delta\dot{\theta}_v + \Delta M_a. \tag{11}$$

Then from (11), the state space variables are.

$$\Delta\theta_v = x_1,\ \Delta\dot{\theta}_v = \dot{x}_1 = x_2,\ \Delta M_a = x_3. \tag{12}$$

From the equations (6), (9) and (10), we have.

$$\tau\dot{x}_3 = -x_3 + GP_N,\ P_N = P - \frac{M_{ao}}{G}. \tag{13}$$

Rewriting (11) and (13) in the form of state variables, we can obtain:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ \tilde{f}_{21}(x_1) & \frac{-B}{J} & \frac{1}{J} \\ 0 & 0 & \frac{-1}{\tau} \end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\end{bmatrix} + \begin{bmatrix}0\\ 0\\ \frac{G}{\tau}\end{bmatrix}P_N. \tag{14}$$

In (13) $\tilde{f}_{21}(x_1)$ is given by the nonlinear equation:

$$\tilde{f}_{21}(x_1) = \frac{1}{Jx_1}\left[-mglsin(x_1+\theta_{vo}) - \lambda e^{-E\left(x_1+\theta_{vo}+\frac{\pi}{2}\right)}\left(x_1+\theta_{vo}+\frac{\pi}{2}-\omega\right)+M_{a0}\right]. \quad (15)$$

In (15) we can observe that when the value of the state x_1 is equal to zero, there is a problem in the determination of $\tilde{f}_{21}(x_1)$, because the denominator of the function tends to zero. This fact hinders the determination of the pertinences functions used in the description of the plant through exact Takagi-Sugeno fuzzy models [2]. To solve this problems we expanded (15) in Taylor series, since that the substitution of (9a) in (15) allows the cancellation of the sine and exponential terms when x_1 is equal to zero. Expanding (15) in Taylor series, we eliminate the term in the denominator, avoiding the implementation problem in $x_1=0$. Fig. 2 shows that the Taylor series of eleventh order is very close to the exact curve of the nonlinear function in the interval of [-1 1], corresponding to range of -60° to 60°. It is possible to obtain a more exact representation, increasing the order of the Taylor series.

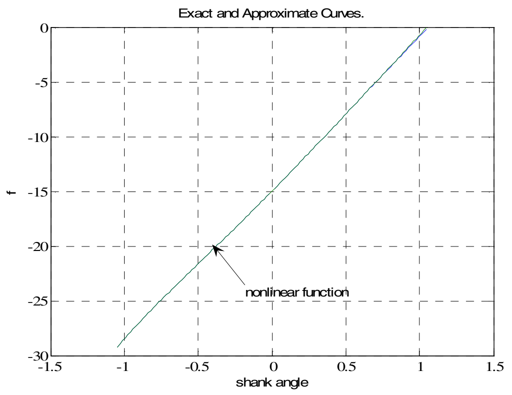


Fig. 2. Exact and approximate curves of the nonlinear function $\tilde{f}_{21}(x_1)$

3 Modelling and Controller Designs Using the Exact Model Fuzzy Takagi-Sugeno

To determine the local models, using the exact modeling [2], the following class of nonlinear systems was considered:

$$\dot{x}_i = \sum_{j=1}^{n} \tilde{f}_{ij}(x(t))x_j(t) + \sum_{k=1}^{m} g_{ik}(x(t))u_k(t),$$
$$i=1,2,\dots,n, \; x(t)=\left[x_1(t)\dots x_n(t)\right]^T, \quad (16)$$

where n and m are the number of states and inputs, respectively. To obtain the widespread form, the following variables are considered:

$$\begin{aligned} a_{ij1} &\equiv \max_{x(t)}\left\{\tilde{f}_{ij}\left(x(t)\right)\right\},\ a_{ij2} \equiv \min_{x(t)}\left\{\tilde{f}_{ij}\left(x(t)\right)\right\}, \\ b_{ij1} &\equiv \max_{x(t)}\left\{g_{ik}\left(x(t)\right)\right\},\ b_{ij2} \equiv \min_{x(t)}\left\{g_{ik}\left(x(t)\right)\right\}. \end{aligned} \tag{17}$$

In [1] it is demonstrated that, to represent the original system (17), through T-S fuzzy models in the form given in (2) and with the widespread form we need necessary 2^s local models, where s is the number nonlinearities in the systems.

3.1 Fuzzy Controller Design

The concept of Parallel Distributed Compensation PDC [8] was used to design the fuzzy regulators in order to stabilize nonlinear systems described by T-S fuzzy models. The idea is to design a regulator for each rule of the fuzzy model. The global fuzzy is a fuzzy combination of each regulator. PDC offers a systematic procedure to design regulators. The local regulator design has the following form:

Rule i:
IF $z_1(t)\ is\ M_1^i\ E\ \ldots\ E\ z_p(t)\ é\ M_p^i$,
THEN $u(t) = -F_i x(t)$.

The fuzzy regulator is given by.

$$u(t) = -\frac{\sum_{i=1}^{r}\omega^i\left(z(t)F_i x(t)\right)}{\sum_{i=1}^{r}\omega^i\left(z(t)\right)} = -\sum_{i=1}^{r}\alpha_i\left(z(t)F_i x(t)\right) = -F(\alpha)x(t), \tag{18}$$
$$\alpha = \left[\alpha_1, \ldots, \alpha_r\right].$$

The objective of the design of fuzzy regulator is to find the local feedback controllers [8], given by (17), such that controlled systems below is stable and presents a suitable performance.

$$\begin{aligned} \dot{x}(t) &= \sum_{i=1}^{r}\sum_{j=1}^{r}\alpha_i(x(t))\alpha_j(x(t))\left\{A_i - B_i F_j\right\}x(t) = \sum_{i=1}^{r}\alpha_i^2(x(t))G_{ii}(x(t)) \\ &\quad + 2\sum_{i<j}^{r}(\alpha_i(x(t))\alpha_j(x(t)))\left\{\frac{G_{ij}(x(t)) + G_{ji}(x(t))}{2}\right\}x(t), \end{aligned} \tag{19}$$

where, $G_{ij} = A_i - B_i F_j$ and $\sum_{i<j}^{3} d_{ij} = d_{12} + d_{13} + d_{23}$.

Theorem 1: The PDC Controller [9] that simultaneously considers the stability and the decay rate controller design β_t for the controlled system (19), can be designed by solving the following Linear Matrix Inequalities (LMI's):

To maximize β_t

X, M_1, M_2, Y_0

Subject to: $X>0,\ Y_0 \geq 0$

$$\hat{S}_{ii}+(s-1)Y_1<0,\ \hat{T}_{ij}-2Y_2<0,\ i<j, \textit{such that}\ \alpha_i(x_1)\cap\alpha_j(x_1)\neq 0,$$

$$L(A_i,B_i,X,M_i)=XA_i^T+A_iX-B_iM_i-M_i^TB_i^T,\ L(A_j,B_j,X,M_i)=XA_j^T+A_jX-B_jM_i-M_i^TB_j^T,$$

$$\hat{S}_{ii}=L(A_i,B_i,X,M_i)+2\beta_t,\ \hat{T}_{ij}=L(A_i,B_i,X,M_j)+L(A_j,B_j,X,M_i)+4\beta_tX,$$

$$Y_1=Y_0,\ Y_2=Y_0,\ \ i=1,...,r,\ s>1$$

where s is the maximum number of rules that can are simultaneously different from zero when these LMI's are feasible the controller gains are given by $F_i=M_iX^{-1}$, $i=1,2,\ldots,r$.

4 Results of Simulations

The software MATLAB, with the LMI control toolbox, solves LMIs when they are feasible [10]. The model of the paraplegic problem is nonlinear, and folowing with the exact modeling method proposed in [2], it has two local models. The minimum and maximum values of range are -60° to 60°. For the equation (16), the minimum and maximum value of the defined nonlinear function were determined:

$$\begin{aligned} a_{211} &= \max\{\tilde{f}_{21}(x_1(t))\} = \text{-}0.033294, \\ a_{212} &= \min\{\tilde{f}_{21}(x_1(t))\} = \text{-}29.21650. \end{aligned} \tag{20}$$

Therefore we found two local models starting from the maximum and minimum values of the equation (15) following the method given in [2]:

$$\tilde{f}_{21}(x_1)=\alpha_1(x_1(t))a_{211}+\alpha_2(x_1(t))a_{212}. \tag{21}$$

For the method proposed in [2],

$$\begin{aligned} \alpha_1(x_1(t)) &= \frac{\tilde{f}_{21}(x_1)-a_{212}}{a_{211}-a_{212}}, \\ \alpha_2(x_1(t)) &= \frac{\tilde{f}_{21}(x_1)-a_{211}}{a_{212}-a_{211}}. \end{aligned} \tag{22}$$

The local models considering (14) are equal to [2].

$$A_1 = \begin{bmatrix} 0 & 1 & 0 \\ a_{211} & -B/J & 1/J \\ 0 & 0 & -1/\tau \end{bmatrix};\ A_2 = \begin{bmatrix} 0 & 1 & 0 \\ a_{212} & -B/J & 1/J \\ 0 & 0 & -1/\tau \end{bmatrix},\ B_1 = B_2 = \begin{bmatrix} 0 & 0 & \dfrac{G}{\tau} \end{bmatrix}^T. \tag{23}$$

The T-S fuzzy model is composed by,

$$\dot{x}(t) = \sum_{i=1}^{2} \alpha_i\left(x_1(t)\right)\left(A_i x(t) + B_i u(t)\right). \tag{24}$$

The feedback gain (18), for the system vary from -60° to 60°, obtained by the method of the Theorem 1, with β_t =0, were

$$\begin{aligned} F_1 &= M_1 X^{-1} = \begin{bmatrix} 0.00116 & 0.00057 & 0.00010 \end{bmatrix} \\ F_2 &= M_2 X^{-2} = \begin{bmatrix} 0.00151 & 0.00068 & 0.00012 \end{bmatrix} \end{aligned} \tag{25}$$

To illustrate the validity of the designed control, that assures the stability of the system, the simulation of the controlled system was made. Fig. 3 shows that the results.

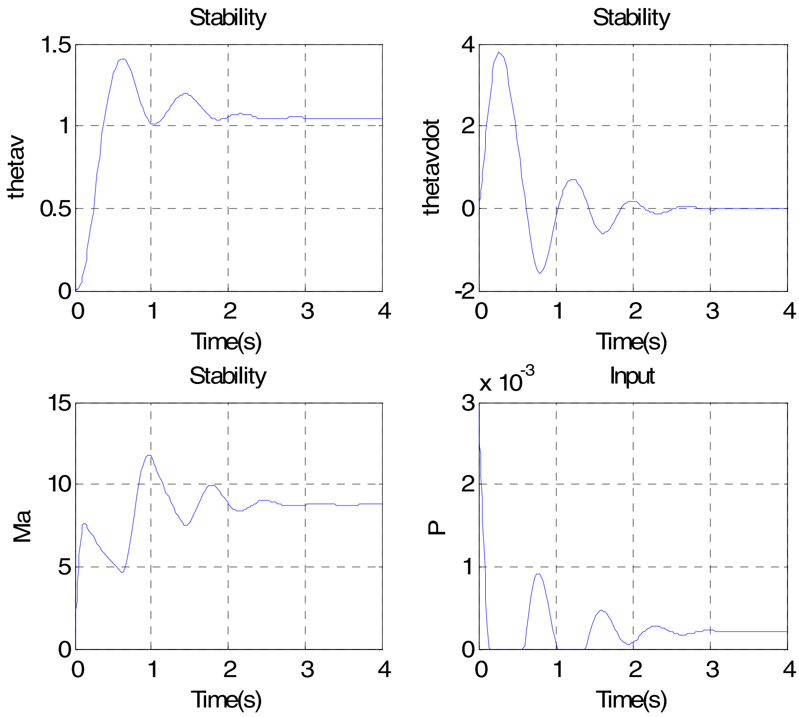


Fig. 3. Answer for the controller design fuzzy

The simulation results shown that the θ_v=60°, that is the desired value. The setting time was approximate equal to 3 seconds e can be adequate. Note the input saturation between 0.5 and 1.0 seconds, because the pulse P≥0, [11].

5 Conclusion

The control system considers all nonlinearities of the plant and satisfies the specified project restrictions: positive and limited pulse width, warranty of stability (the project uses Lyapunov functions and speed of the transient response. The design was based on LMIs, then it is possible to still add other specifications in the design of the control system, for instance, constrains in the input and output signals and the consideration of uncertainties in the parameters of the plant.

Acknowledgments. The authors acknowledge the financial support by Fapesp and CNPq.

References

1. Ferrarim, M., Pedotti, A.: The Relationship Between Electrical Stimulus and Joint Torque: a Dynamic Model. IEEE Transactions on Rehabilitation Engineering, Vol. 8 (2000) 342-352
2. Taniguchi, T., Tanaka, K. Ohatake, H. Wang, H. O.: Model Construction, Rule Reduction, and Robust Compensation for Generalized Form of Takagi-Sugeno Fuzzy Systems. IEEE Transactions on Fuzzy Systems, Vol. 9 (2001) 525-537
3. Dorgan, S. J., O Malley, M. J.: A Mathematical Model for Skeletal Muscle Activated by N Let Pulse Trains. IEEE Transaction on Rehabilition Engineering, Vol. 6 (1998) 286-299
4. K. Tanaka.: Stability and Stabilizability of Fuzzy-Neural-Linear Control-Systems. IEEE Transactions on Fuzzy Systems, Vol. 3 (1995) 438-444
5. K. Tanaka.: An Approach to Stability Criteria of Neural-Network Control Systems. IEEE Transactions on Neural Network, Vol. 7 (1996) 629-642
6. Teixeira, M. C. M., Zak, S. H.: Stabilizing Controller Design for Uncertain Nonlinear Systems Using Fuzzy Models. IEEE Transaction on Fuzzy Systems, Vol. 7 (1992) 133-142
7. Deaecto, G. S. ,Teixeira, M. C. M.: Projetos e Simulações de Sistemas de Controle Utilizando Modelos Fuzzy e LMI. Scientific Report (in Portuguese), UNESP, Ilha Solteira (2005)
8. Tanaka, K., Wang H. O.: Fuzzy Control Systems Design and Control Analysis. Jonh Wiley & Sons, New York Chichester Weinheim Brisbane Singapore Toronto (2001).
9. Tanaka, K., Ikeda, T., Wang, H. O.: Fuzzy Regulators and Fuzzy Observers: Relaxed Stability Conditions and LMI-Based Designs. IEEE Transactions on Fuzzy Systems, Vol. 6 (1998) 250-265
10. Boyd, S., Ghaoui, L. E Feron, E. Balakrishan.: Linear Matrix Inequalities Systems and Control Theory. SIAM Studies in Apllied Mathematics, (1994)
11. Farias, U. C., Carvalho A. A.: Implementação de Um Sistema de Geração de Marcha Para Pacientes com Lesões Medulares. Phd Thesis (in Portuguese), UNESP, Ilha Solteira (2006)

Characterization of Breast Abnormality Patterns in Digital Mammograms Using Auto-associator Neural Network

Rinku Panchal and Brijesh Verma

School of Information Technology, Faculty of Business & Informatics
Central Queensland University, Rockhampton, QLD 4702, Australia
{r.panchal, b.verma}@cqu.edu.au

Abstract. Presence of mass in breast tissues is highly indicative of breast cancer. The research work investigates the significance of neural-association of mass type of breast abnormality patterns for benign and malignant class characterization using auto-associator neural network and original features. The characterized patterns are finally classified into benign and malignant classes using a classifier neural network. Grey-level based statistical features, BI-RADS features, patient age feature and subtlety value feature have been used in proposed research work. The proposed research technique attained a 94% testing classification rate with a 100% training classification rate on digital mammograms taken from the DDSM benchmark database.

Keywords: Digital mammography, neural networks, auto-associator network, classifier.

1 Introduction

Practice of computational intelligence techniques in medical field have strengthened and improved the current medical decision support system. With advances in technology, digital mammography has taken over conventional mammography which opened up a new door for breast cancer treatment by facilitating computer aided detection and diagnosis (CAD) of cancerous breast tissues. Currently digital mammography is the most reliable and appealing screening technique for early breast cancer detection in women. A space occupying lesion mass is a localized sign of breast cancer [1]. Masses are described by their shape and margin characteristics, and present with wide range of shapes, sizes and contrasts. Masses having irregular shapes and spiculated or indistinct margins suggest a higher possibility of malignancy compared with circumscribed oval and round shaped masses [2,3] . Resultant detected suspicious abnormalities of mammographic screening are further classified into benign and malignant either via ultra sound or biopsy.

Early detection of breast cancer increases the treatment options and the average survival rate in sufferers. Interpretation of mammograms for early stage suspicious abnormalities is repetitive and fatiguing task often leading to human errors either missing malignant cases or more benign biopsies. Practical studies have proved that CAD has efficiently detected early stage breast abnormalities, which were missed by radiologists in first pass mammographic interpretation [4, 5]. The advantages and challenges of early detection of breast cancer motivates researchers to discover an intelligent system which can interpret mammograms repetitively with accuracy and uniformity as well as, and conceivably better than, the current human process. Along with skilled radiologists, a computer supported detection and classification technique can effectively improve and accelerate the overall interpretation process.

Literature shows many research methods have been proposed and investigated by researchers for computerized detection and classification of breast abnormalities in digital mammograms. The sole aim of computer aided classification systems is to assist decision making by radiologists of the detected suspicious lesion as malignant or benign. Statistical methods, fuzzy logic, wavelet transforms, and artificial neural networks (ANNs) are the most common methods that have been used and explored by researchers for reliable classification of breast abnormalities.

Baker et al. [6] investigated ANN classifier based on BI-RADS features. Wei et al. [7] investigated the use of global and local multi resolution texture features with linear discriminant analysis (LDA) for identifying masses in digital mammograms. Sahiner et al. [8] characterized mammographic masses using texture features derived from spatial grey level dependency (SGLD) matrices and Run-Length Statistics (RLS) matrices and LDA classifier. Bovis et al. [9,10] used SGLD texture features for classification of masses and normal breast tissue on mammograms of MIAS database. They evaluate the performance of Back-Propagation Neural Network (BPNN) and Radial Basis Function (RBFNN) for classification. Christoyianni et al. [11] presented a novel method for detection of circumscribed masses in mammograms using radial basis function neural network (RBFNN) with set of decision criteria. They used MiniMIAS database consists of 22 mammograms containing circumscribed lesions. The lesion sizes vary from 18 pixels- 198 pixels in radius. Edwards et al. [12] proposed a Bayesian artificial neural network (BANN) technique to estimate the decision variables for classifying computer detected benign and malignant mass lesions and false positive computer detections. The BANN differs from a conventional ANN in that the error function used in training includes a regularization term, equivalent to a Bayesian prior probability on the neural network weight values, to panelize solutions which are more complicated that the training data justify [13]. The attempt is made to regularize training to improve the robustness of the classifier.

Wide variation in abnormality structure and appearance in breast tissues and lack of individuality in abnormality patterns (malignant and benign) makes their classification more difficult. In practice, performance of a single classifier may not be satisfactory for reliable classification of breast abnormality because of ambiguity in data. Considering the vagueness of breast abnormality patterns researchers have researched and proposed various ensemble network models in an attempt to achieve a more accurate result. Wu et al. [14] used non-generative neural network ensembles for identifying breast masses. They used two fusion algorithms; weighted average (WA) and perceptron average (PA) to evaluate the performance of their ensemble network.

The main objective of this paper's research work is to investigate the neural network based pattern characterization and classification technique for reliable classification of mass type of breast cancer abnormalities into benign and malignant classes. In the proposed research work characterization of breast abnormality patterns implies to more detailed and finer representation of benign and malignant patterns which enable classifier network to produce maximum classification. The proposed technique uses pre-processed digital mammogram images from Digital Database of Screening Mammography (DDSM). Grey level based statistical features, radiologists' interpretation in terms of BI-RADS lexicon features, patient age feature and subtlety value feature are used to characterize benign and malignant class patterns using auto-associator neural network. Characterized patterns are finally classified into benign and malignant classes using feed forward neural network classifier.

The remainder of this paper is broken down into 4 sections. Section 2 gives a detailed explanation of the proposed research methodology. Experimental results obtained with extracted features using the proposed research technique are presented in section 3. Section 4 covers a detailed analysis on the experimental results. In Section 5, conclusions are drawn.

2 Research Methodology

The proposed research methodology comprises four parts; acquisition of a digital mammograms from DDSM database, extraction of grey level based features from suspicious areas and BI-RADS features, patient age feature and subtlety value feature from associated files supplied along with mammogram in each DDSM case, neural-association of extracted features to characterise benign and malignant patterns, and final classification of characterized (neural associative) patterns into malignant and benign.

2.1 Acquisition of Digital Mammograms

Digital mammograms in this research work have been taken from Digital database for screening mammography (DDSM) benchmark database [15]. The DDSM database includes both calcification and mass types of abnormalities. Mammograms containing mass type of abnormalities have been used in this research work. Each case study includes two mammograms (mediolateral oblique (MLO) and cranio-caudal (CC) views) of each breast from a screening exam. Only CC view mammograms of right and left breast were used. The mammograms of selected cases were digitized using HAWTEK digitizer at 50 microns and 12-bit depth pixels (grey level range 0~4096). The malignant cases of training and testing dataset were selected as defined by the DDSM. The benign cases were selected randomly allowing inclusion of all kind of breast abnormality structures.

2.2 Feature Extraction

Features discussed below are used in the proposed research methodology to characterize breast abnormality class patterns in to malignant and benign classes. Figure 1 outlines the procedure of feature extraction of proposed research technique.

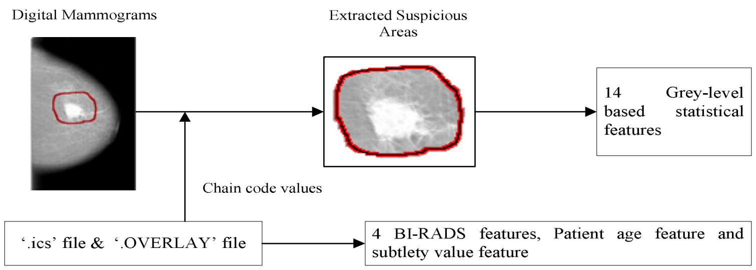

Digital Mammograms
Extracted Suspicious Areas
14 Grey-level based statistical features
Chain code values
'.ics' file & '.OVERLAY' file
4 BI-RADS features, Patient age feature and subtlety value feature

Fig. 1. Overview of feature extraction process

Grey-level based statistical features
The Grey-level based statistical features extraction is accomplished in two steps: extract suspicious (abnormality) areas from mammograms, and then extract grey level based features from extracted suspicious areas employing statistical formulas. Firstly all suspicious areas are derived using the outline marked by an expert radiologist in all digital mammograms [15]. Pixel level 'ground truth' information such as locations of suspicious areas, chain code values to define the boundary pixels of suspicious area along with starting pixel position is also provided in each DDSM case (figure 1). Grey level values of each suspicious area and the respective surrounding boundary area are extracted to calculate the feature values using statistical formulas. The extracted total 14 grey level based statistical features includes Number of pixels, histogram, average grey, average boundary grey, contrast, difference, energy, modified energy, entropy, modified entropy, standard deviation, modified standard deviation, skew, and modified skew [16].

BI-RADS features
The density of breast tissues and abnormality shape/type and its distribution/outline/margin inside breast tissues are the key factors radiologists consider when judging the likelihood of cancer being present. Abnormality assessment rank suggests the severity of abnormality. Considering their importance in the human interpretation process BI-RADS features have been used along with grey-level features to achieve reliable classification of mass type breast abnormalities in digital mammograms.

The DDSM case studies include associate patient information and abnormality interpretation results using the BI-RADS™ (ACR 1998) lexicon [15] from expert radiologists along with mammograms. This information is stored in '*.ics*' file and '*OVERLAY*' file of each case (Figure 1). Interpretation results include ACR breast density rating, an abnormality description and abnormality assessment rank.

The morphological descriptors of mass abnormalities are the shape of abnormality and its margin. Hence the total four BI-RADS features are density, mass shape, mass margin, and abnormality assessment rank [6, 17]. Morphological descriptions of breast abnormality are encoded into numeric values to get real feature values. For example numeric value 1 is set to encrypt 'round' mass shape and 2 for 'Oval' shape and so on. The same process is used to encode mass margins.

Patient Age feature
Each case in the DDSM contains information such as the patient age at time of the mammogram was taken. Various medical findings and past breast cancer statistics show that the risk of breast cancer increase with age. This makes patient age a very useful feature for medical practitioners to consider in diagnosis of breast cancer. Considering this patient age feature is used for neural classification of mass abnormality.

Subtlety Value feature
A subjective impression of the subtlety of an abnormality rated from 1 to 5 by an expert radiologist is considered as a subtlety value feature.

2.3 Characterization (Neural-Association) of Breast Abnormality Class Patterns

Capturing underlying distribution of data points is very important task in pattern recognition. Auto-associator neural network (AANN) can be used effectively for capturing the distribution of data. A three layer linear AANN performs Principle component analysis (PCA) on input features [18]. The AANN performs auto-associative (identity) mapping, in which it tries to map the input vectors onto themselves at output layers as it has same input and output. Wide range of AANN structure can be found in literature based on the problem being solved. The basic Auto-Associator neural network (AANN) structure consists of the equal size of input and output layer with usually one or more hidden layers of smaller size than the input and output layer.

In general, Auto-association targets the compression of input feature space. It encourages the network to get rid of redundancies in a set of input patterns and discover any structure present in the input so the input can be represented more abstractly and compactly. The pattern of hidden neuron activations must be efficient and concise if the hidden neurons are to successfully regenerate the input pattern on the output layer. However training of AANN remains a complicated issue as AANN's ability to capture the distribution of training data is highly dependent on the size and structure of the AANN and the shape of activation function used for training [19].

The proposed technique focuses on characterization of breast abnormality patterns into benign and malignant classes using AANN. The dimension of hidden layer (no of hidden neurons) is not restricted to be of smaller size than input and output layer in an attempt to exploit the original extracted features. It aimed to regenerate the composite of its learned patterns most consistent with new information disregarding the compression of input data, and thus the regenerated patterns are capable of characterizing each class patterns in a given feature set. These characterized patterns further explored with classifier NN to obtain higher classification accuracy.

Single hidden layer perceptron is used for the neural-association of breast abnormality patterns. An error-back propagation (EBP) learning algorithm with sigmoid transfer function is used to evaluate AANN training. Combination of all features are used as input and output for AANN training. Network parameters such as learning rate, momentum, number of hidden neurons and iterations are adjusted during AANN training to obtain unique patterns for benign and malignant classes which enable the classifier NN to produce maximum classification. The detailed description and architecture of AANN used in this research work can be found in our previous work [20].

2.4 Classification of Characterized Breast Abnormality Class Patterns

Another single hidden layer Perceptron with EBP learning algorithm is used to further explore the charectrised abnormality patterns for their final classfication into benign and malignant classes [20] . It has two output nodes in output layer, which represents each class. The desired output is set as (0.9, 0.1) for malignant class and (0.1, 0.9) for benign class. The classifier network is trained using the neural associative patterns (hidden neurons values) obtained from trained AANN. Classifier network is trained extensively trying various combination of network parameters to attain maximum classfication on unknow test breast abnormality patterns.

3 Experimental Results

The experimental dataset contains a total of 200 suspicious areas of mass type of abnormality, 100 (50 malignant, and 50 benign) for training and the same for testing. The proposed technique has been evaluted using 14 grey level features, 4 BI-RADS features, patient age feature and subtlety value feature. It is an attempt to evaluate the combination of human (abnormality interpretation in terms of BI-RADS lexicon) and non-human (grey level based statistical features) expertise for reliable classification of breast abnormalities in digital mammogram. It is included as the main objective of CAD is to work along with radiologists to offer reliable and error-free reading of mammograms.

Table 1 shows the results obtained with all features using proposed technique. Both networks AANN and classifier network, have been extensively trained on feature vector comprises of all features using various network configurations. The network configurations comprise different sets of hidden neurons (from 4 to 30 in even order) and iterations (from 5000 to 100000), and various learning rate and momentum values (0.1 to 0.9) to achieve reliable classification of breast abnormalities into malignant and benign classes. In result tables, *'HN'* is used to represent the total number of hidden neurons and *'IT'* represents the number of iterations, *'LR'* represents the learning rate value and *'MN'* represents the momentum values used in training of the particular NN.

Table 1. Classification results with all features

AANN				Classifier NN				Training	Testing
LR	MN	HN	IT	LR	MN	HN	IT	(%)	(%)
0.1	0.1	16	30000	0.1	0.1	28	40000	100	91
0.1	0.1	16	30000	0.1	0.1	28	45000	100	90
0.1	0.1	24	60000	0.1	0.2	10	30000	100	92
0.1	0.1	14	80000	0.2	0.3	20	60000	100	91
0.1	0.1	14	80000	0.2	0.3	20	80000	100	90
0.1	0.1	24	60000	0.2	0.3	20	70000	100	92
0.1	0.1	24	60000	0.2	0.3	20	80000	100	91
0.2	0.3	28	80000	0.1	0.1	30	40000	100	92
0.2	0.3	28	80000	0.1	0.1	30	50000	100	93
0.2	0.3	28	80000	0.1	0.1	30	60000	100	94
0.2	0.3	28	80000	0.1	0.1	30	65000	100	94
0.2	0.3	28	80000	0.1	0.1	30	70000	100	93

The highest testing classification rate 94% is observed with the corresponding 100% training classification rate with different NN configurations. Neural associative patterns, obtained from AANN trained on all features using 0.2 learning rate and 0.3 momentum value attained 94% testing classification rate. AANN trained with 28 hidden neurons for 80000 iterations. The classifier NN trained using 0.1 learning rate and 0.1 momentum value with 30 hidden neurons for 60000 and 65000 iterations attained a high testing classification rate. 100% training classification rate was observed with many network configurations.

4 Analysis of Experimental Results

Lack of individuality in breast abnormality patterns makes their characterization challenging. In proposed technique AANN is used to characterize benign and malignant patterns using grey-level based statistical features, BI-RADS features, patient age feature and subtlety value feature. Figure 2 shows the characterized (neural associative) patterns obtained from AANN which attained the highest testing classification rate 94% with all features. The AANN is trained using 28 hidden neurons for 80000 iterations at 0.2 learning rate and 0.3 momentum value. Data values of hidden neuron no 20 to 28 is shown in figure-graph. While carefully looking at Figure 2 and Figure 3, the graphical presentation of neural associative patterns shows the distinct range at hidden neuron number 25 for all benign patterns on given feature scale.

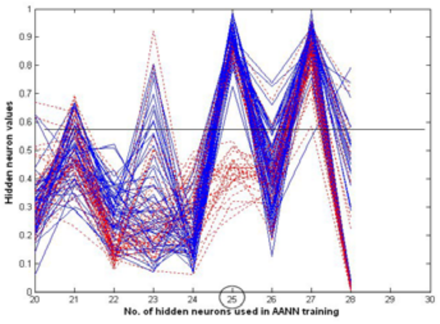


Fig. 2. Neural associative malignant (solid line) and benign (dashed line) patterns distribution

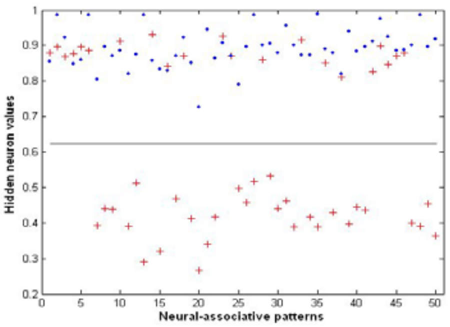


Fig. 3. Hidden neuron no. 25 values of malignant (+) and benign (.) patterns

Figure 3 shows the detailed view of feature space of hidden neuron no. 25 of neural associative patterns attained using all features shown in figure 2. Malignant is forming a wide region on feature scale either falling in to benign region (overlapping) or making it a part of it (sub region). During analysis of overlapped malignant patterns, it has been observed that the same number of patterns have been falling in to benign region in many sets of experiments with different network configurations. This may be because of the quite similarity in benign and malignant type abnormalities.

The classification accuracy of proposed research technique to classify benign and malignant class patterns has been analysed. Table 2 shows the sensitivity and specificity results obtained with proposed technique. In result table, '*Sn*' represents the sensitivity and '*Sp*' represents the specificity obtained with particular network. Figure 4 shows the ROC curves obtained with the experimental results listed in table 2. The highest 98% sensitivity observed with very low specificity 86%, which is not recommended. With the highest A_Z value 0.951, the system with 96% sensitivity and 92% sensitivity is more reliable as it is sensitive to malignant patterns and specific towards benign patterns. Benign patterns are forming a distinct region on feature scale; however their overall classification is lesser than malignant. The reason may be a malignant is forming a wide region on feature scale so network is not able to recognise benign patterns which have similar features values as malignant. Although benign classification is consistent with malignant, no big difference has been observed in experiential results.

Quantitative comparison of pattern classification systems in digital mammography is a difficult task due to development and testing of such systems is usually done using different datasets and different kind of features. The results obtained with the proposed research methodology are compared with the other researchers. Bovis et al. [10] attained 77% classification accuracy and 0.74 Az value with MLP on 161 breast images of MIAS dataset. Wu et al. [14] used dataset containing 500 masses from the China Society for Industrial and Applied Mathematics. They reported highest 88.27% accuracy with the Perceptron Average (PA) fusion algorithm on imbalanced input

Table 2. Sensitivity and Specificity results

#	AANN				Classifier NN				Testing (%)		
	LR	MN	HN	IT	LR	MN	HN	IT	Sn	Sp	Total
1	0.1	0.1	24	60000	0.2	0.3	20	70000	98	86	92
2	0.2	0.3	28	80000	0.1	0.1	30	40000	92	92	92
3	0.2	0.3	28	80000	0.1	0.1	30	50000	94	92	93
4	0.2	0.3	28	80000	0.1	0.1	30	60000	96	92	94

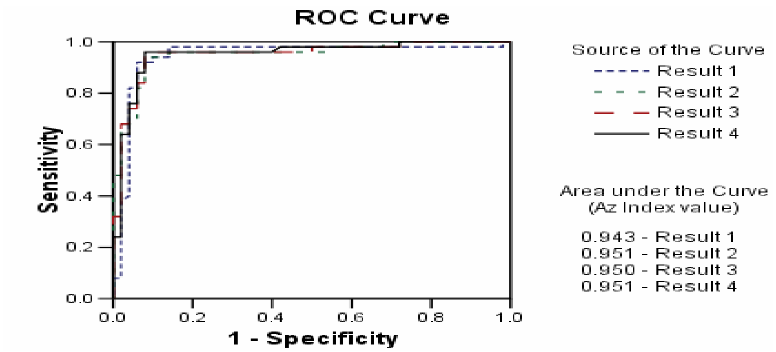


Fig. 4. ROC curves for experimental results of Table 2

patterns. In comparison our proposed technique attained 94% testing classification accuracy on a mass dataset comprise of 100 suspicious areas for training and the same for testing (Table 1). This shows the proposed technique is able to classify malignant and benign mass type of abnormalities in digital mammograms.

5 Conclusions

The proposed research technique investigates significance of auto-association in characterization of benign and malignant breast abnormality patterns. The highest 94% classification accuracy with 96% sensitivity and 92% specificity on the mass test dataset is attained using the proposed technique. It is clear from experimental results and graphical presentation of benign and malignant neural associative patterns, that auto-association of breast abnormality patterns improves the characteristic of benign and malignant patterns and their classification accuracy.

References

1. Tourassi, G.D., Current Status of Computerized Decision Support Systems in Mammography, Studies in Fuzziness and Soft Computing, 2005, 184: p. 173-208.
2. Bassett, L.W., 30F Imaging the Breast: Section 9 Principles of Imaging, in Cancer Medicine, 5th Edition, M.D. Robert C. Bast Jr., et al., Editors. 2000, the American Cancer Society and BC Decker, Inc. p. 420-427.
3. Sampat, M., Markey M., & Bovik A., Computer-Aided Detection and Diagnosis in Mammography, in Handbook of Image and Video Processing. 2005, p. 1195-1217.
4. Bird, R.E., Wallace, T.W., & Yankaskas, B.C., Analysis of Cancers Missed at Screening Mammography, Radiology, 1992, 184(3): p. 613-617.
5. Birdwell, R.L., Ikeda, D.M., O'Shaughnessy, K.F. & Sickles E.A., Mammographic Characteristics of 115 Missed Cancers Later Detected with Screening Mammography and the Potential Utility of Computer-aided Detection, Radiology, 2001, 219(1): p. 192-202.
6. Baker, J.A., Kornguth, P.J., Lo, J.Y., Williford, M.E., & Floyd, C.E., Jr., Breast Cancer: Prediction with Artificial Neural Network based on BI-RADS Standardized Lexicon, Radiology, 1995, 196(3): p. 817-822.
7. Wei, D., Chan, H-P, Petrick, N., Sahiner, B., Helvie, M.A., Adler, D.D., & Goodsitt, M.M., False-Positive Reduction Technique for Detection of Masses on Digital Mammograms: Global and Local Multiresolution Texture Analysis, Medical Physics, 1997, 24(6): p. 903-914.
8. Sahiner, B., Chan, H-P, Petrick, N., Helvie, M.A., & Goodsitt, M.M., Computerized Characterization of Masses on Mammograms: The Rubber Band Straightening Transform and Texture Analysis, Medical Physics, 1998, 25(4): p. 516-526.
9. Bovis, K. & Singh, S., Detection of Masses in Mammograms using Texture Features, proc. of the 15th International Conference on Pattern Recognition, 2000.
10. Bovis, K., Singh, S., Fieldsend, J., & Pinder, C., Identification of Masses in Digital Mammograms with MLP and RBF Nets, proc. of the IEEE International Joint Conference on Neural Networks (IEEE-INNS-ENNS'2000). 2000.
11. Christoyianni, I., Dermatas, E. & Kokkinakis, G., Fast Detection of Masses in Computer-Aided Mammography, Signal Processing Magazine, IEEE, 2000, 17(1): p. 54-64.

12. Edwards, D.C., Lan, L., Metz, C.E., Giger, M.L. & Nishikawa, R.M., Estimating Three-Class Ideal Observer Decision Variables for Computerized Detection and Classification of Mammographic Mass Lesions, Medical Physics, 2004, 31(1): p. 81-90.
13. Kupinski, M.A., Lan, L., Metz, C.E., Giger, M.L., & Nishikawa, R.M., Ideal Observer Approximation using Bayesian Classification Neural Networks, Medical Imaging, IEEE Transactions on, 2001, 20(9): p. 886-899.
14. Wu, Y., He, J., Man, Y., & Arribas, J.I., Neural Network Fusion Strategies for Identifying Breast Masses, proc. of the IEEE International Joint Conference on Neural Networks (IEEE-IJCNN'2004), 2004.
15. Heath, M., Bowyer, K., Kopans, D., Moore, R., & Jr. Kegelmeyer, P., The Digital Database for Screening Mammography, proc. of the Digital Mammography: IWDM 2000, 5th International Workshop on Digital Mammography, 2001, Medical Physics Publishing.
16. Verma, B. and Zakos J., A computer-aided diagnosis system for digital mammograms based on fuzzy-neural and feature extraction techniques, Information Technology in Biomedicine, IEEE Transactions on, 2001, 5(1): p. 46-54.
17. Lo, J.Y., Gavrielides, M.A., Markey, M.K., & Jesneck, J.L., Computer-Aided Classification of Breast Microcalcification Clusters: Merging of Features from Image Processing and Radiologists, proc. of the SPIE Medical Imaging 2003, Image Processing, 2003.
18. Bishop, C.M., Neural Networks for Pattern Recognition, 1995, Oxford – Clarendon Press.
19. Iversen, A., Taylor, N.K., and Brown, K.E., Classification and Verification through the Combination of the Multi-Layer Perceptron and Auto-Association Neural Networks, proc. of the IEEE International Joint Conference on Neural Networks (IEEE-IJCNN'2005), 2005.
20. Panchal, R. and Verma B., A Fusion of Neural Network Based Auto-associator and Classifier for the Classification of Microcalcification Patterns, proc. of the 11th International Conference on Neural Information Processing (ICONIP'2004), 2004.

Evolving Hierarchical RBF Neural Networks for Breast Cancer Detection

Yuehui Chen, Yan Wang, and Bo Yang

School of Information Science and Engineering
Jinan University, Jinan 250022, P.R. China
yhchen@ujn.edu.cn

Abstract. Hierarchical RBF networks consist of multiple RBF networks assembled in different level or cascade architecture. In this paper, an evolved hierarchical RBF network was employed to detect the breast cancel. For evolving a hierarchical RBF network model, Extended Compact Genetic Programming (ECGP), a tree-structure based evolutionary algorithm and the Differential Evolution (DE) are used to find an optimal detection model. The performance of proposed method was then compared with Flexible Neural Tree (FNT), Neural Network (NN), and RBF Neural Network (RBF-NN) by using the same breast cancer data set. Simulation results show that the obtained hierarchical RBF network model has a fewer number of variables with reduced number of input features and with the high detection accuracy.

1 Introduction

Breast cancer is the most common cancer in women in many countries. Most breast cancers are detected as a lump/mass on the breast, or through self-examination or mammography [1]. Screening mammography is the best tool available for detecting cancerous lesions before clinical symptoms appear [7]. Surgery through a biopsy or lumpectomy have been also been the most common methods of removal. Fine needle aspiration (FNA) of breast masses is a cost-effective, non-traumatic, and mostly invasive diagnostic test that obtains information needed to evaluate malignancy. Recently, a new less invasive technique, which uses super-cooled nitrogen to freeze and shrink a non-cancerous tumor and destroy the blood vessels feeding the growth of the tumour, has been developed [2] in the USA.

Various artificial intelligence techniques have been used to improve the diagnoses procedures and to aid the physician's efforts [3][4][5][6]. In our previous studies, the performance of Flexible Neural Tree (FNT) [11], Neural Network (NN), Wavelet Neural Network (WNN) and an ensemble method to detect breast-cancer have been evaluated [12].

Hierarchical RBF networks (HRBF) consist of multiple RBF networks assembled in different level or cascade architecture in which a problem was divided and solved in more than one step. Mat Isa et al. used Hierarchical Radial Basis Function (HiRBF) to increase RBF performance in diagnosing cervical cancer [14].

The HiRBF cascaded together two RBF networks, where both networks have different structure but using the same learning algorithms. The first network classifies all data and performs a filtering process to ensure that only certain attributes to be fed to the second network. The study shows that the HiRBF performs better compared to single RBF. Hierarchical RBF network has been proved effective in the reconstruction of smooth surfaces from sparse noisy data points [15]. In order to improve the model generalization performance, a selective combination of multiple neural networks by using Bayesian method was proposed in [16].

In this paper, an automatic method for constructing HRBF networks is proposed. Based on a pre-defined instruction/operator set, the HRBF network can be created and evolved. The HRBF network allows input variables selection. In our previous studies, in order to optimize the Flexible Neural Tree (FNT) and the hierarchical TS fuzzy model (H-TS-FS), the hierarchical structure of FNT and H-TS-FS was evolved using Probabilistic Incremental Program Evolution algorithm (PIPE) [10][11] and Ant Programming [13] with specific instructions. In this research, the hierarchical structure is evolved using the Extended Compact Genetic Programming (ECGP), a tree-structure based evolutionary algorithm. The fine tuning of the parameters encoded in the structure is accomplished using the DE algorithm. The proposed method interleaves both optimizations. The novelty of this paper is in the usage of hierarchical RBF network model for selecting the important input variables and for breast cancel detection.

The paper is organized as follows. The RBF network is introduced in Section 2. An optimal design method for constructing the HRBF networks is described in Section 3. Section 4 gives the simulation results. Finally in Section 5 we present some concluding remarks.

2 The RBF Network

An RBF network is a feed-forward neural network with one hidden layer of RBF units and a linear output layer. By an RBF unit we mean a neuron with multiple real inputs $\boldsymbol{x} = (x_1, \ldots, x_n)$ and one output y computed as:

$$y = \varphi(\xi); \quad \xi = \frac{\|\boldsymbol{x} - \boldsymbol{c}\|_C}{b} \tag{1}$$

where $\varphi : R \to R$ is a suitable activation function, let us consider Gaussian radial basis function $\varphi(z) = e^{-z^2}$. The center $\boldsymbol{c} \in R^n$, the width $b \in R$ and an $n \times n$ real matrix $\boldsymbol{C}$ are a unit's parameters, $|| \cdot ||_C$ denotes a weighted norm defined as $\|\boldsymbol{x}\|_C^2 = (\boldsymbol{C}\boldsymbol{x})^T(\boldsymbol{C}\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{C}^T\boldsymbol{C}\boldsymbol{x}$.

Thus, the network represents the following real function $\boldsymbol{f} : R^n \to R^m$:

$$f_s(\boldsymbol{x}) = \sum_{j=1}^{h} w_{js} e^{-(\frac{\|\boldsymbol{x}-\boldsymbol{c}\|_C}{b})^2}, \qquad s = 1, \ldots, m, \tag{2}$$

where $w_{js} \in R$ are weights of s-th output unit and f_s is the s-th network output.

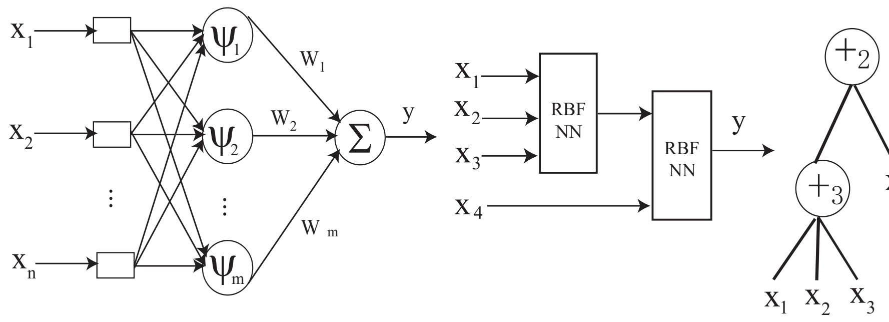


Fig. 1. A RBF neural network (left), an example of hierarchical RBF network (middle), and a tree-structural representation of the HRBF network (right)

The goal of an RBF network learning is to find suitable values of RBF units' parameters and the output layer's weights, so that the RBF network function approximates a function given by a set of examples of inputs and desired outputs $T = \{\boldsymbol{x}(t), \boldsymbol{d}(t); t = 1, \ldots, k\}$, called a *training set*. The quality of the learned RBF network is measured by the *error function*:

$$E = \frac{1}{2}\sum_{t=1}^{k}\sum_{j=1}^{m} e_j^2(t), \qquad e_j(t) = d_j(t) - f_j(t). \tag{3}$$

3 The Hierarchical RBF Network

3.1 Encode and Calculation

A function set F and terminal instruction set T used for generating a HRBF network model are described as $S = F \bigcup T = \{+_2, +_3, \ldots, +_N\} \bigcup \{x_1, \ldots, x_n\}$, where $+_i (i = 2, 3, \ldots, N)$ denote non-leaf nodes' instructions and taking i arguments. x_1,x_2,...,x_n are leaf nodes' instructions and taking no arguments. The output of a non-leaf node is calculated as a HRBF network model (see Fig.1). From this point of view, the instruction $+_i$ is also called a basis function operator with i inputs.

In this research, Gaussian radial basis function is used and the number of radial basis functions used in hidden layer of the network is same with the number of inputs, that is, $m = n$.

In the creation process of HRBF network tree, if a nonterminal instruction, i.e., $+_i (i = 2, 3, 4, \ldots, N)$ is selected, i real values are randomly generated and used for representing the connection strength between the node $+_i$ and its children. In addition, $2 \times n^2$ adjustable parameters a_i and b_i are randomly created as radial basis function parameters. The output of the node $+_i$ can be calculated by using Eqn.(1) and Eqn.(2). The overall output of HRBF network tree can be computed from left to right by depth-first method, recursively. Finding an optimal or near-optimal HRBF network structure is formulated as a product of evolution. In our previously studies, the Genetic Programming (GP), Probabilistic Incremental Program Evolution (PIPE) have been explored for structure optimization of the FNT. In this paper, the ECGP is employed to find an optimal or near-optimal structure of HRBF networks.

3.2 Tree Structure Optimization by ECGP

Finding an optimal or near-optimal HRBF is formulated as a product of evolution. In our previously studies, the Genetic Programming (GP), Probabilistic Incremental Program Evolution (PIPE) have been explored for structure optimization of the FNT [10][11]. In this paper, the Extended Compact Genetic Programming (ECGP) [17] is employed to find an optimal or near-optimal HRBF structure.

ECGP is a direct extension of ECGA to the tree representation which is based on the PIPE prototype tree. In ECGA, Marginal Product Models (MPMs) are used to model the interaction among genes, represented as random variables, given a population of Genetic Algorithm individuals. MPMs are represented as measures of marginal distributions on partitions of random variables. ECGP is based on the PIPE prototype tree, and thus each node in the prototype tree is a random variable. ECGP decomposes or partitions the prototype tree into sub-trees, and the MPM factorises the joint probability of all nodes of the prototype tree, to a product of marginal distributions on a partition of its sub-trees. A greedy search heuristic is used to find an optimal MPM mode under the framework of minimum encoding inference. ECGP can represent the probability distribution for more than one node at a time. Thus, it extends PIPE in that the interactions among multiple nodes are considered.

3.3 Parameter Optimization with DE Algorithm

The DE algorithm was first introduced by Storn and Price in 1995 [8]. It resembles the structure of an evolutionary algorithm (EA), but differs from traditional EAs in its generation of new candidate solutions and by its use of a 'greedy' selection scheme. DE works as follows: First, all individuals are randomly initialized and evaluated using the fitness function provided. Afterwards, the following process will be executed as long as the termination condition is not fulfilled: For each individual in the population, an offspring is created using the weighted difference of parent solutions. The offspring replaces the parent if it is fitter. Otherwise, the parent survives and is passed on to the next iteration of the algorithm. In generation k, we denote the population members by x_1^k, x_2^k, $\ldots$, x_N^k. The DE algorithm is given as follows [9]:

S1 Set $k = 0$, and randomly generate N points x_1^0, x_2^0, $\ldots$, x_N^0 from search space to form an initial population;

S2 For each point $x_i^k (1 \leq i \leq N)$, execute the DE offspring generation scheme to generate an offspring $x_i^(k + 1)$;

S3 If the given stop criteria is not met, set $k = k + 1$, goto step S2.

The DE Offspring Generation approach used is given as follows,

S1 Choose one point x_d randomly such that $f(x_d) \quad f(x_i^k)$, another two points x_b, x_c randomly from the current population and a subset $S = \{j_1, \ldots, j_m\}$ of the index set $\{1, \ldots, n\}$, while $m < n$ and all j_i mutually different;
S2 Generate a trial point $u = (u_1, u_2, \ldots, u_n)$ as follows:
DE Mutation. Generate a temporary point z as follows,

$$z = (F + 0.5)x_d + (F - 0.5)x_i + F(x_b - x_c); \tag{4}$$

Where F is a give control parameter;
DE Crossover. for $j \in S$, u_j is chosen to be z_j; otherwise u_j is chosen a to be $(x_i^k)_j$;
S3 If $f(u) \leq f(x_i^k)$, set $x_i^{k+1} = u$; otherwise, set $x_i^{k+1} = x_i^k$.

3.4 Procedure of the General Learning Algorithm

The general learning procedure for constructing the HRBF network can be described as follows.

S1 Create an initial population randomly (HRBF network trees and its corresponding parameters);
S2 Structure optimization is achieved by using ECGP algorithm;
S3 If a better structure is found, then go to step S4, otherwise go to step S2;
S4 Parameter optimization is achieved by DE algorithm. In this stage, the architecture of HRBF network model is fixed, and it is the best tree developed during the end of run of the structure search;
S5 If the maximum number of local search is reached, or no better parameter vector is found for a significantly long time then go to step S6; otherwise go to step S4;
S6 If satisfactory solution is found, then the algorithm is stopped; otherwise go to step S2.

3.5 Variable Selection Using HRBF Network Paradigms

It is often a difficult task to select important variables for a classification or regression problem, especially when the feature space is large. Conventional RBF neural network usually cannot do this. In the perspective of HRBF network framework, the nature of model construction procedure allows the HRBF network to identify important input features in building a HRBF network model that is computationally efficient and effective. The mechanisms of input selection in the HRBF network constructing procedure are as follows. (1) Initially the input variables are selected to formulate the HRBF network model with same probabilities; (2) The variables which have more contribution to the objective function will be enhanced and have high opportunity to survive in the next generation by an evolutionary procedure; (3) The evolutionary operators i.e., crossover and mutation, provide a input selection method by which the HRBF network should select appropriate variables automatically.

Table 1. Comparative results of the FNT, NN, RBF [12] and the proposed HRBF network classification methods for the detection of breast cancer

Cancer type	FNT(%)	NN(%)	RBF-NN(%)	HRBF(%)
Benign	93.31	94.01	94.12	96.83
Malignant	93.45	95.42	93.21	96.83

Table 2. The important features selected by the HRBF network

x_0, x_1, x_2, x_3, x_6, x_7, x_9, x_{18}, x_{20}, x_{25}, x_{27}, x_{29}

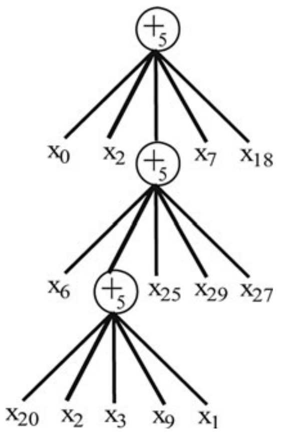


Fig. 2. The optimized HRBF network for breast cancel detection

4 Simulations

As a preliminary study, we made use of the Wisconsin breast cancer data set from the UCI machine-learning database repository [18]. This data set has 30 attributes (30 real valued input features) and 569 instances of which 357 are of benign and 212 are of malignant type. The data set is randomly divided into a training data set and a test data set. The first 285 data is used for training and the remaining 284 data is used for testing the performance of the different models.

All the models were trained and tested with the same set of data. The instruction sets used to create an optimal HRBF network classifier is $S = F \bigcup T = \{+_2, \ldots, +_5\} \bigcup \{x_0, x_1, \ldots, x_{29}\}$. Where $x_i (i = 0, 1, \ldots, 29)$ denotes the 30 input features. The optimal hierarchical HRBF network for breast cancel detection problem is shown in Figure 2. The classification results for testing data set are shown in Table 1. For comparison purpose, the detection performances of the FNT, NN and RBF-NN are also shown in Table 1 (for details, see [12]). The important features for constructing the HRBF network models are shown in Table 2. It

Table 3. Comparison of false positive rate (fp) and true positive rate (tp) for FNT, NN, RBF-NN [12] and hierarchical HRBF network

Cancer	FNT		NN		RBF-NN		HRBF	
Type	fp(%)	tp(%)	fp(%)	tp(%)	fp(%)	tp(%)	fp(%)	tp(%)
Benign	3.88	91.71	4.85	93.37	6.6	97.14	2.91	96.69
Malignant	2.76	86.41	4.97	96.12	9.2	96.87	3.31	97.09

should be noted that the obtained HRBF network classifier has smaller size and reduced features and with high accuracy in breast cancel detection. Receiver Operating Characteristics (ROC) analysis of the FNT, NN, RBF-NN and the HRBF network model is shown in Table 3.

5 Conclusion

In this paper, we presented an optimized HRBF network for the detection of breast cancel and compared the results with some advanced artificial intelligence techniques, i.e., FNT, NN and RBF-NN. As depicted in Table 1, the preliminary results are very encouraging. The best accuracy was offered by the HRBF network method followed by the RBF neural network for detecting benign types and PSO trained neural network for detecting the malignant type of cancer. An important advantage of the HRBF network model is the ability to reduce the number of input variables as presented in Table 2. ROC analysis (Table 3) illustrates that RBF neural network has the highest false positive rate and the HRBF network model has the lowest false positive rates for detecting benign and malignant cancer. The time required to construct these models are not very much and hope these tools would assist the physician's effort to improve the currently available automated ways to diagnose breast cancer.

Acknowledgment

This research was partially supported the Natural Science Foundation of China under contract number 60573065, and The Provincial Science and Technology Development Program of Shandong under contract number SDSP2004-0720-03.

References

1. DeSilva, C.J.S. et al., Artificial Neural networks and Breast Cancer Prognosis, The Australian Computer Journal, 26, pp. 78-81, 1994.
2. The Weekend Australia, Health Section, pp. 7. July, 13-14, 2002.
3. David B. Fogel , Eugene C. Wasson , Edward M. Boughton and Vincent W. Porto, A step toward computer-assisted mammography using evolutionary programming and neural networks, Cancer Letters, Volume 119, Issue 1, pp. 93-97, 1997.

4. Charles E. Kahn, Jr , Linda M. Roberts, Katherine A. Shaffer and Peter Haddawy, Construction of a Bayesian network for mammographic diagnosis of breast cancer, Computers in Biology and Medicine, Volume 27, Issue 1, pp. 19-29, 1997.
5. Shinsuke Morio , Satoru Kawahara , Naoyuki Okamoto, Tadao Suzuki, Takashi Okamoto , Masatoshi Haradas and Akio Shimizu, An expert system for early detection of cancer of the breast, Computers in Biology and Medicine, Volume 19, Issue 5, pp. 295-305, 1989.
6. Barbara S. Hulka and Patricia G. Moorman, Breast Cancer: Hormones and Other Risk Factors, Maturitas, Volume 38, Issue 1, pp. 103-113, 2001.
7. Jain, R. and Abraham, A., A Comparative Study of Fuzzy Classifiers on Breast Cancer Data, Australiasian Physical And Engineering Sciences in Medicine, Australia, Volume 27, No.4, pp. 147-152, 2004.
8. R. Storn, and K. Price, "Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces," Technical report, International Computer Science Institute, Berkley, 1995.
9. K. Price, "Differential Evolution vs. the Functions of the 2nd ICEO," In proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC'97), Indianapolis, USA, pp. 153-157, 1997.
10. Chen, Y., Yang, B., Dong, J., Nonlinear systems modelling via optimal design of neural trees, International Journal of Neural ystems, **14**, pp. 125-138, 2004.
11. Chen, Y., Yang, B., Dong, J., Abraham A., Time-series forcasting using flexible neural tree model, Information Science, Vol.174, Issues 3/4, pp.219-235, 2005.
12. Chen, Y., Abraham, A., Yang, B., Hybrid Neurocomputing for Breast Cancer Detection, The Fourth International Workshop on Soft Computing as Transdisciplinary Science and Technology (WSTST'05), pp.884-892, Springer, 2005.
13. Chen, Y., Yang, B. and Dong, J., Evolving Flexible Neural Networks using Ant Programming and PSO algorithm, International Symposium on Neural Networks (ISNN'04), LNCS 3173, pp. 211-216, 2004.
14. N. A. Mat Isa, Mashor, M. Y., and Othman, N. H., "Diagnosis of Cervical Cancer using Hierarchical Radial Basis Function (HiRBF) Network," In Sazali Yaacob, R. Nagarajan, Ali Chekima (Eds.), Proceedings of the International Conference on Artificial Intelligence in Engineering and Technology, pp. 458-463, 2002.
15. S. Ferrari, I. Frosio, V. Piuri, and N. Alberto Borghese, "Automatic Multiscale Meshing Through HRBF Networks," IEEE Trans. on Instrumentation and Measurment, vol.54, no.4, pp. 1463-1470, 2005.
16. Z. Ahmad, J. Zhang, "Bayesian selective combination of multiple neural networks for improving long-range predictions in nonlinear process modelling," Neural Comput & Applic. Vol. 14. pp. 78C87, 2005.
17. K. Sastry and D. E. Goldberg. "Probabilistic model building and competent genetic programming", In R. L. Riolo and B. Worzel, editors, Genetic Programming Theory and Practise, chapter 13, pp. 205-220. Kluwer, 2003.
18. Merz J., and Murphy, P.M., UCI repository of machine learning databases, http://www.ics.uci.edu/- learn/MLRepository.html, 1996.

Ovarian Cancer Prognosis by Hemostasis and Complementary Learning

T.Z. Tan[1], G.S. Ng[1], C. Quek[1], and Stephen C.L. Koh[2]

[1] Center for Computational Intelligence, School of Computer Engineering, Nanyang Technological University, Nanyang Avenue, Singapore
[2] Department of Obstetrics and Gynaecology, Yong Loo Lin School of Medicine, National University of Singapore, Lower Kent Ridge Road, Singapore

Abstract. Ovarian cancer is a major cause of deaths worldwide. As a result, women are not diagnosed until the cancer has advanced to later stages. Accurate prognosis is required to determine the suitable therapeutic decision. Since abnormalities of hemostasis and increased risk of thrombosis are observed in cancer patient, assay involving hemostatic parameters can be potential prognosis tool. Thus a biological brain-inspired *Complementary Learning Fuzzy Neural Network* (CLFNN) is proposed, to complement the hemostasis in ovarian cancer prognosis. Experimental results that demonstrate the confluence of hemostasis and CLFNN offers a promising prognosis tool. Apart from superior performance, CLFNN provides interpretable rules to facilitate validation and justification of the system. Besides, CLFNN can be used as a concept validation tool for ovarian cancer prognosis.

1 Introduction

Ovarian cancer is a major cause of cancer deaths worldwide. It ranks fourth as the cause of cancer deaths in US. A total of 20,180 new cases and 15,310 cancer deaths are anticipated in US alone in 2006 [1]. The similar observations can be found in Singapore, where ovarian cancer accounts for 4.4% of the cancer deaths in women, and it is the fourth most frequent cancer (5.4%) [2]. In the period from 1998-2002, ovarian cancer recorded a 385 cancer deaths, and 1,055 new cases [2]. Thus, many endeavors have been made; yet, the number of cancer deaths due to ovarian cancer is still high, and the 5-year survival rate for all stages remains at 30%-45% [3]. This high morbidity rate is because the ovarian cancer is not diagnosed until it has advanced to stages III and IV. Since the therapeutic decision is determined by the response of the disease, accurate prognosis is needed to aid in making the right therapy decision.

Abnormalities of hemostasis (stoppage of blood flow) and increased risk of *thrombosis* (presence of blood clot within blood vessel) have been reported in cancer patients (50% of all patients, and up to 95% of metastatic disease patients [4]). Thus, hemostatic parameters are potential indicators of disease free interval and long-term survival, as well as tumor growth and tumor angiogenesis [3]. Moreover, *thromboembolism* is the second most important cause of death in cancer patients [5]. This happens because tumor cells can activate systemic coagulation, induce tissue factor production, and cause endothelial cells to express procoagulants, and hence, stimulate the extrinsic coagulation pathway [5]. These hemostasis pathways are often altered in patient with different types of malignancies, especially for gynecologic cancer [6].

Prognosis by means of hemostatic parameters is suitable for ovarian cancer because in advanced stage it shows hypercoagulation, increased platelets and enhanced fibrinolysis with further enhanced thrombin generation [3]. Furthermore, ovarian tumor demonstrates pronounced aberration of the coagulation and fibrinolysis activation markers such as D-Dimer and fibrinogen levels.

Hence, the hemostasis is a promising prognostic tool to fight ovarian cancer. Unfortunately, there is high uncertainty in prognosis when hemostatic assay is used alone. This is because, the blood samples have high risk of contamination, and the threshold values to determine whether one is normal or cancerous varies from individual to individuals, and from populations to populations. In addition, some of them are less specific [4]. Realizing this, hemostatic assay is used with *computational intelligence* methods to provide accurate prognosis of ovarian cancer.

Among the computational intelligence methods, *Fuzzy Neural Network* (FNN) is more suitable as ovarian cancer *Clinical Decision Support System* (CDSS): in contrast to statistical methods, it is autonomous and does not need the manual construction of knowledge. On the other hand, different from *Artificial Neural Network* (ANN), FNN has the capacity of offering interpretation and justification for their output. Of these FNN, biological brain-inspired FNN is more desirable because aside from the advantageous traits mentioned, they provide human-like operations, and allow human to analyze or understand the system in their familiar terms.

Thus, *Complementary Learning Fuzzy Neural Network* (CLFNN) [7] is proposed to complement hemostasis in ovarian cancer prognosis. Complementary learning refers to the learning from positive and negative samples, as well as the exploitation of the lateral inhibition between positive and negative classes. The lateral inhibition means: given a positive sample, only positive rules are activated, and at the same time, negative rules are inhibited, leading to a positive and correct decision. Such complementary learning can be observed in the human brain during pattern recognition task. Furthermore, CLFNN has been previously applied in ovarian cancer diagnosis and is found to perform competently. Hence, in this work, CLFNN is used with hemostatic assays in ovarian cancer prognosis. A representative of the CLFNN named *Hierarchical Complementary Learning* (HCL) is introduced. It births forth from the biological inspirations of complementary learning and the hierarchical category learning [8]. Since HCL does not take into account the relationship between the input features, it is suitable in this task as the relationship between the hemostatic parameters is not well understood.

The paper is organized as follows: Section 2 describes the dataset used in this study, the CLFNN (HCL) employed, as well as the experimental setup. Experimental results and analyses are given in Section 3. Section 4 concludes the work and discusses possible future work.

2 Materials and Methods

2.1 Dataset

The dataset is obtained from the Department of Obstetrics and Gynaecology, Yong Loo Lin School of Medicine, National University of Singapore. (http://www.nuh.edu.sg). There are a total of 149 subjects, and they are divided into cancer and control groups. The control group includes the normal patients and patients with benign cyst (88), whereas the cancer group contains patients from FIGO stages I-IV.

The blood sampling was performed mainly in the morning, where nine parts of blood from a clean venupuncture were mixed with a part of 0.129 mol/L trisodium citrate contining 0.21 mol/L hydroxylehylpiperazine-ethnesulfonic acid in cold plastic tube. Subsequently, the blood was spun at 2000g for 15 minutes in a refrigerated centrifuge [5]. The biological assays used are:

1. Plasma fibrinogen
2. D-Dimer
3. von Willebrand factor (vWF)
4. Antithrombin III complex (TAT)
5. Antithrombin III and plasminogen activities (ATIII)
6. Prothrombin Fragment 1+2 (F_{1+2})
7. Factor VII
8. Tissue factor pathway inhibitor (TFPI)

These assays were chosen because there are evidences that these assays are indicative of ovarian cancer or survival outcome [9]. Evidence of tumor-associated activation of coagulation and fibrinolysis can be indicated by plasma fibrin split products. One example of this is *D-Dimer* [6], which is a stable end product of cross-linked fibrin degradation by plasmin [4]. D-Dimer is a good marker of ongoing fibrin formation and degradation in different disease and malignancies [6].

Von Willebrand factor (vWF) is directly involved in the mechanism of platelet thrombus formation. Although it has no direct role in coagulation, it is required for normal hemostasis, and elevated levels may cause thrombotic complications [3].

Both prothrombin fragment F_{1+2} and TAT are markers of *in vivo* hemostasis activation [10]. TAT is generated by the reaction between thrombin and ATIII, where fragment F_{1+2} and the active thrombin are formed when prothrombin is cleaved — an important step in the coagulation cascade [10].

Tissue factor that binds factor VII and its inhibitor TFPI are required to initialize blood coagulation by activating both factor X and factor IX. Hence, they are indicative of the hemostasis as well [11].

The statistics of the data is summarized in Table 1. As shown in Table 1, the difference between control and cancer groups is significant for most of the assays. For example, the age of the control group is significantly younger than the cancer group, and the fibrinogen level of the control group is significantly lower than that of the cancer group. In other words, this suggests that performing prognosis using hemostasis, improved results can potentially be achieved.

Table 1. Statistics of the attributes

Group		**Age**	**Fi** g/L	**D-D** ng/ml	**vWF** iu/ml	**TAT** µg/ml	**ATIII** iu/ml	**F_{1+2}** nmol/L	**FVII** %	**TFPI** ng/ml
Cancer	µ	51.6	5	214K	1.9	13.3	0.9	1.5	111.8	75.1
	σ	14.1	2	656K	1.1	13.9	0.2	0.9	33.9	48.4
	R	16-81	1.2-10.4	96K-486K	0.4-5.4	2-68	0.5-1.3	0.11-5	43-236	20-352
Benign	µ	41.7	3.8	788	1.3	6.8	1	1.1	103.2	65
	σ	14	1.3	1855	0.6	9.9	0.2	1.1	27.6	120
	R	16-74	1.7-7.3	50-153	0-3.5	1-48	0.5-1.3	0.3-6.6	33-166	0-999
p-value		<0.001	<0.001	<0.02	<0.001	<0.01	<0.01	0.02	0.1	>0.1

µ: Mean; σ: Standard deviation; R: Range; Fi: Fibrinogen; D-D: D-Dimer; FVII: Factor VII

2.2 Hierarchical Complementary Learning

The *Hierarchical Complementary Learning* (HCL) integrates the complementary learning and hierarchical organization. By functionally model the complementary and hierarchical category learning, better recognition performance may be obtained as the former exploits the characteristics of both positive and negative concepts, as well as the lateral inhibition between the two. On the other hand, the breaking down of complex concepts into simpler ones through the later, allows a simpler and closer representation of the problem dynamic. Hence, putting together complementary and hierarchical learning is beneficial. The resulting hybrid, HCL, can be described by the followings:

Given dataset $D = D^+ \cup D^-$, where $D^+ = \{X^+, Y^+\}$ and $D^- = \{X^-, Y^-\}$ are the positive and negative samples, respectively. HCL creates a rulebase R according to the dataset. Again, the rulebase R is made up of positive and negative rules, i.e. $R = R^+ \cup R^-$, where $R^+ = \bigcup_{k=1}^{K^+} r_k$ and $R^- = \bigcup_{k=1}^{K^-} r_k$, $k \in [1, \mathrm{K}]$ is the index for the rules, and K is the total number of rules, K = $|R| = |R^+| + |R^-|$ = $K^+ + K^-$. Each rule r_k is in the form delineated in Eq. 1.

$$r_k\text{: IF } x_1 \text{ is } \mathbf{A}_{1k},\ldots, x_i \text{ is } \mathbf{A}_{ik},\ldots, x_I \text{ is } \mathbf{A}_{Ik}, \text{ THEN } y_1 \text{ is } \mathbf{B}_{(l1)k}, \ \ , y_m \text{ is } \mathbf{B}_{(lm)k},\ldots, y_M \text{ is } \mathbf{B}_{(lM)k}. \tag{1}$$

Thus, $R: \mathbf{A} \rightarrow \mathbf{B}$; **A** and **B** are the input and output fuzzy sets respectively. They defined the input and output linguistic terms. x is the input, and $i \in [1, \mathrm{I}]$ is the index of input dimension. x_i refers to the input from the i^{th} dimension. Likewise, y is the output, $m \in [1, \mathrm{M}]$ is the output dimension, and y_m is the output from m^{th} dimension. $l \in [1, \mathrm{L}]$ is the index for output linguistic terms **B**. $\mathbf{A}_{ik}$ is the input linguistic term that links the input from i^{th} dimension to the k^{th} rule, similarly for output linguistic terms. Every linguistic term (represented by fuzzy set) characterizes a concept. The HCL infers based on the difference in firing strength between the positive and negative, as described in Eqs. 2 and 3.

$$Y = \begin{cases} Y^{+}, & \text{if } \mu_{R^{+}}(\mathbf{x}) > \mu_{R^{-}}(\mathbf{x}) \\ Y^{-}, & \text{if } \mu_{R^{+}}(\mathbf{x}) < \mu_{R^{-}}(\mathbf{x}) \end{cases} \tag{2}$$

$$\mu_{R^{*}}(\mathbf{x}) = \max_{k \in \mathrm{K}^{*}} \left\{ \frac{1}{\mathrm{I}} \sum_{i=1}^{\mathrm{I}} \mu_{\mathbf{A}_{ik}^{*}}(x_i) \right\} \text{ or } \frac{1}{\mathrm{K}} \sum_{k=1}^{\mathrm{K}^{*}} \left\{ \frac{1}{\mathrm{I}} \sum_{i=1}^{\mathrm{I}} \mu_{\mathbf{A}_{ik}^{*}}(x_i) \right\} \tag{3}$$

* = + for positive, - for negative.

Thus, when a sample $\mathbf{x} \in X^{+}$ is presented to HCL, the positive rules will have greater firing strengths than the negative rules, i.e. $\mu_{R^{+}}(\mathbf{x}) > \mu_{R^{-}}(\mathbf{x})$. Consequently, the positive rules are activated, and concurrently, negative rules are inhibited, leading to a positive decision, $y \in Y^{+}$. This complementary learning mechanism exploits the lateral inhibition between positive and negative classes, which avoids possible confusion in the inference process, and hence may promise better recognition performance. Note that this inference is done in a hierarchical manner, i.e., from the most relevant feature to the least relevant feature.

In order to equip HCL an easy rule/knowledge extraction, HCL is designed as a five-layer FNN, with each layer corresponds to the elements of the fuzzy rule shown in Eq. 1. The architecture of HCL is given in Fig. 1.

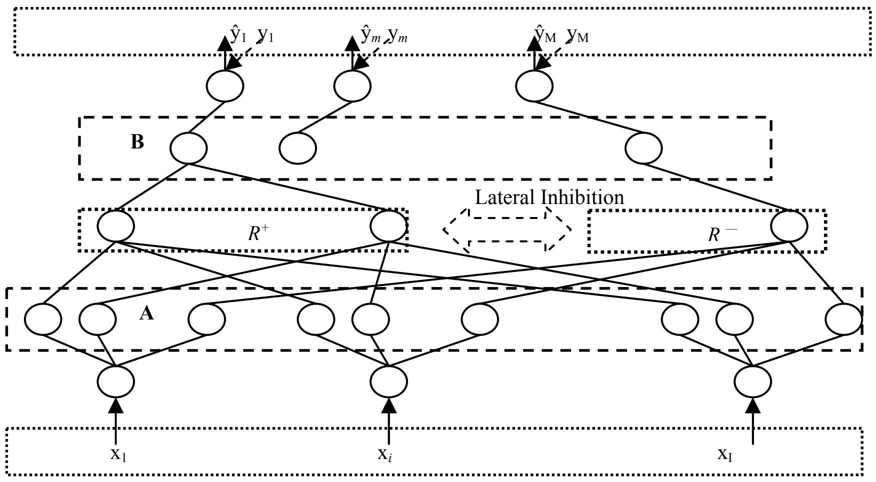


Fig. 1. Architecture of hierarchical complementary learning fuzzy neural network. Each layer corresponds to the elements of fuzzy rules: (1) input linguistic labels, *X*; (2) input linguistic terms, **A**; (3) rules, *R*; (4) output linguistic terms, **B**; (5) output linguistic labels, *Y*.

The construction of the structure is autonomous. The HCL learning algorithm is outlined as follows:

Given a training dataset $D = \left\{ (\mathbf{x},\mathbf{y})^{1}, \ldots, (\mathbf{x},\mathbf{y})^{t}, \ldots, (\mathbf{x},\mathbf{y})^{\mathrm{T}} \right\}$:

Step 1: Segregate the dataset D into D^{+} and D^{-}. This is to functionally model and to facilitate the implementation of the segregation of positive and negative knowledge in human brain, where different brain areas are registered for recognition of different objects.

Step 2: Rank the features, $f_i \in [1, \mathrm{I}]$, and put into F_{rank} in descending order. In this work, *Augmented Variance Ratio* (AVR) is used (Eq. 4).

$$AVR = S_b \Big/ \frac{1}{\mathrm{M}} \sum_{m=1}^{\mathrm{M}} \frac{S_m}{\min\limits_{m \neq m^*} |\mu_m - \mu_{m^*}|} \tag{4}$$

where M = number of classes. S_m = within-class variance for mth class, S_b = between-class variance, μ_m = mean for mth class. AVR is the ratio between inter-class variance and intra-class variance of the feature. The greater the AVR, the greater is the discriminative power of the feature. The features are added into a list F_{rank} according to their ranking (most relevant feature first).

Step 3: Select the feature/dimension f_i, with the highest AVR score inside F_{rank}, and then remove it from F_{rank}. HCL classifies the data features by features. Thus, HCL maintains a feature list, F to store the features to be considered when performing the classification. If there is no existing rules, create rule with current sample, go to Step 5. HCL uses trapezoidal fuzzy set as its membership function. They are centered on the sample x_i initially, and subsequently adjusted according to the training sample. If there are existing rules, go to Step 4.

Step 4: Select the rule that is nearest to this sample. If there is a perfect match, then no learning is required, go to Step 8. Else, determine if there is any sample belonging to other class in between the kernel of this nearest rule and the current sample. If not, then expand the kernel of the fuzzy set to incorporate this sample. Otherwise, create a new rule based on this sample. This is a functional model of how the chunking mechanism [12] is performed in human learning. Note that, Steps 3 and 4 are carried out separately and concurrently for positive and negative rules. Go to Step 5.

Step 5: Combine the rules by chunking the fuzzy sets, as well as to remove redundant rule. This step is to ensure no redundant or irrelevant rules, as well as to decrease the number of rules in the system. After the chunking process, go to Step 6. The chunking procedure is summarized in Eq. 5, it essentially chunks two pieces of similar information/fuzzy sets into a single unit [12].

$$\begin{bmatrix} \overset{+}{u}_{ik}(t+1), \\ \overset{+}{v}_{ik}(t+1) \end{bmatrix} = \begin{cases} \left[\overset{+}{u}_{ik^*}(t), \overset{+}{v}_{ik^*}(t)\right], \text{ if } \overset{+}{u}_{ik^*}(t) < \overset{+}{u}_{ik}(t), \overset{+}{v}_{ik^*}(t) > \overset{+}{v}_{ik}(t) \\ \left[\overset{+}{u}_{ik}(t), \overset{+}{v}_{ik}(t)\right], \text{ if } \overset{+}{u}_{ik^*}(t) > \overset{+}{u}_{ik}(t), \overset{+}{v}_{ik^*}(t) < \overset{+}{v}_{ik}(t) \\ \left[\overset{+}{u}_{ik}(t), \overset{+}{v}_{ik^*}(t)\right], \text{ if } \overset{+}{u}_{ik^*}(t) > \overset{+}{v}_{ik}(t) \cap \left\{\left(\overset{-}{u}_{ik'}, \overset{-}{v}_{ik'}\right) \cap [\overset{+}{v}_{ik}(t), \overset{+}{u}_{ik^*}(t)] = \phi\right\} \\ \left[\overset{+}{u}_{ik^*}(t), \overset{+}{v}_{ik}(t)\right], \text{ if } \overset{+}{v}_{ik^*}(t) < \overset{+}{u}_{ik}(t) \cap \left\{\left(\overset{-}{u}_{ik'}, \overset{-}{v}_{ik'}\right) \cap [\overset{+}{v}_{ik^*}(t), \overset{+}{u}_{ik}(t)] = \phi\right\} \end{cases} \tag{5}$$

where $\left[u_{ik*}^{+}(t), v_{ik*}^{+}(t)\right]$ is the kernel of the nearest positive rule, and $k^* \neq k, k' \neq k$.

Step 6: Ensure the distinguishability of rules by resetting the kernel and support of the fuzzy sets. The strategy is that, given a fuzzy set, the system checks for the nearest two fuzzy sets (from the same class), and then adjust the support according to the kernel of the two nearest fuzzy sets (Eq. 6). Go to Step 7.

$$\left[Min_{ik}^{+}(t+1), Max_{ik}^{+}(t+1)\right] = \left\{ u_{ik}^{+}(t) - \left[u_{ik}^{+}(t) - v_{ik_{Left}^{*}}^{+}(t)\right] \Big/ 2, \; v_{ik}^{+}(t) + \left[u_{ik_{Right}^{*}}^{+}(t) - v_{ik}^{+}(t)\right] \Big/ 2 \right\} \tag{6}$$

where $\left(u_{ik_{Left}^{*}}^{+}(t), v_{ik_{Left}^{*}}^{+}(t)\right)$ and $\left(u_{ik_{Right}^{*}}^{+}(t), v_{ik_{Right}^{*}}^{+}(t)\right)$ are the kernel of the nearest rule smaller and greater than $\left(u_{ik}^{+}(t), v_{ik}^{+}(t)\right)$, respectively, and $k^* \neq k$.

Step 7: Assess the rules. This requires the definition of the rule fitness. The rule fitness is set to how many samples the rule correctly classifies in the training set. If the rule fitness is lower than a pre-defined threshold, the rule is removed. Go to Step 8.

Step 8: Test if the training termination condition is met. In this work, the termination is determined by the user parameter, i.e. maximum number of features. Unfortunately, there is no systematic way of determining this parameter yet. If the termination condition is not met, go to Step 3 (Eq. 7).

$$\text{HCL learning} = \begin{cases} \text{terminate, if } |F(t)| > \text{maximum number of features} \\ f_{i'} = \arg\max_{i \in \mathrm{I}} \{F_{rank}\} \text{and go to Step 3, Otherwise} \end{cases} \tag{7}$$

2.3 Experimental Setup

Ten-fold cross-validation is used to assess the performance of the system; that is, 90% of the data are used for training, and 10% of the data are used for testing. The procedure is repeated for ten times, where each sample is used for testing exactly once. The task is to predict whether the cancer patients will survive a year after. All hemostatic assays in Table 1 (except age) are employed.

The performance of the system is benchmarked against some popular decision support systems [13] such as C4.5, *Multilayer Perceptron* (MLP), Naïve Bayesian, *Linear Discriminant Analysis* (LDA). Note that the same experimental settings are used for all the methods presented. All simulations are run on Intel Pentium 4, 2 GHz machine with 1 GB RAM. The metrics used are (1) Recall – accuracy on training set; (2) Predict – accuracy on testing set; (3) sensitivity and specificity, as described in Eqs. 8 and 9, respectively. In prognosis task, the positive refers to patients who do not survive, whereas the negative refers to the patients who are still alive.

$$\text{Sensitivity} = \frac{\text{Positive samples correctly classified}}{\text{Total number of positive samples}} . \quad (8)$$

$$\text{Specificity} = \frac{\text{Negative samples correctly classified}}{\text{Total number of negative samples}} . \quad (9)$$

3 Experiments and Results

The ten-fold cross-validation performances of ovarian cancer prognosis using hemostasis-HCL, as well as other methods, are summarized in Table 2. As can be seen from Table 2, the HCL outperforms others in ovarian cancer prognosis, attaining high sensitivity and specificity. HCL demonstrates competent recall and generalization abilities. This is because HCL considers positive and negative knowledge, as well as the difference of positive and negative in its decision making. Thus, through the exploitation of the lateral inhibition, it gives better performance over other methods. This also suggests there are significant differences between positive and negative samples, affirming the information shown in Table 1. The training duration of HCL seems longer than the others, because it analyzes the data feature by feature; this does not necessarily imply that the complexity of HCL is higher than the others.

Table 2. Performance on ovarian cancer prognosis (desired value in bold)

Method	**Recall (%)**	**Sensitivity (%)**	**Specificity (%)**	**Predict (%)**	**Training time (s)**
C4.5	88.89	79.4	9.1	62.22	0.05
MLP	91.11	88.2	45.5	**77.78**	0.44
Naïve Bayesian	75.56	67.6	54.5	64.44	**0.01**
LDA	75.56	80.55	10	65.22	0.03
HCL	**99.6**	**90**	**65**	77.5	2.6

One of HCL significant strengths is that, it provides fuzzy rules to explain its computation. In contrast to MLP, LDA, or Naïve Bayesian, HCL offers means for user to understand the system. These fuzzy rules allow user to validate the system using their familiar terms. In comparison to the crisp rule generated by decision tree (C4.5), fuzzy rule is able to capture uncertainties that exists in the data, and hence, allows HCL to give a more stable solution than C4.5. In other words, a small perturbation in data would not cause a totally different HCL. In this task, the averaged number of rules generated by HCL and C4.5 are five and four respectively, suggesting that the rulebase of both are compact. Table 3 displays the example of the positive and negative rules generated by HCL and C4.5.

As shown in Table 3, the HCL rule is as intuitive as the rules provided by decision tree. Yet another merit of fuzzy rule is that it allows encapsulation of unnecessary details from the user. Apart from that, fuzzy rule generated by HCL is more expressive, as it allows the use of linguistic hedges such as “very”, “quite”, etc. These are described by the fuzzy sets/membership functions autonomously constructed by HCL

Table 3. Rules generated for ovarian cancer prognosis

	HCL (Fuzzy rule)	C4.5 (Crisp rule)
Positive	IF *ATIII* is low, *D-Dimer* is quite high, *Factor VII* is very low, THEN die within a year.	IF *vWF* > 2.36, *Factor VII* > 113, *ATIII* <= 0.81, THEN die within a year.
Negative	IF *ATIII* is medium, *D-Dimer* is medium, *Factor VII* is quite low, THEN survive.	IF *vWF* > 2.36, *Factor VII* > 113, *ATIII* > 0.81, THEN survive.

during learning. An example is given in Fig. 2, illustrating the fact that the fuzzy sets generated by HCL is highly distinguishable; this enables the unambiguous assignment of linguistic terms, and hence, improves the interpretability of the system.

In addition, the rules generated by HCL can be used as a concept validation tool. In Table 3, it is implied that, when ATIII decreased, D-Dimer increased, and Factor VII is low, then the patient would likely not survive after the first 12 months. This supports the findings that there exists significant associations between preoperative hemostatic levels of ATIII, and D-Dimer and survival outcome by 12 months [3]; The rules also affirms that elevated D-Dimer levels were seen in those who died within 13 months from the disease, and a decreased ATIII level was seen in those patients who died at 24 months compared to those patients still alive [5].

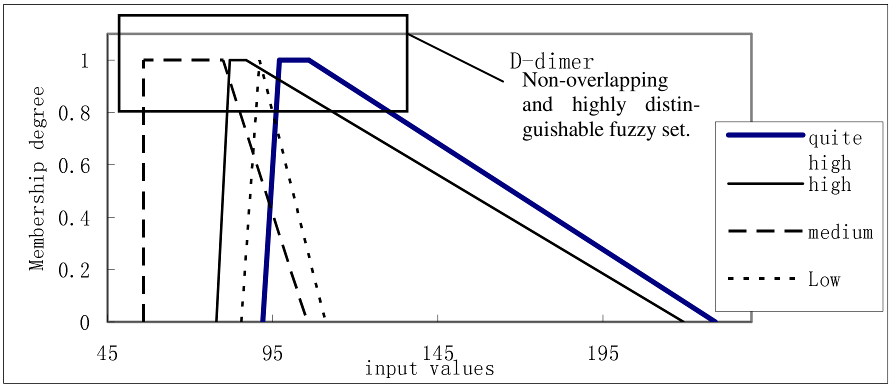


Fig. 2. Membership function of D-Dimer (Dotted lines are negative rules, solid lines are positive rules)

4 Conclusion

The key to improved survival and patient outcome is the early detection and treatment. Therefore, the right therapeutic decision is paramount and this can be obtained through the help of good prognosis system. As abnormal hemostasis is observed in ovarian cancer patients, hemostatic parameters are good indicators for ovarian cancer. However, hemostatic parameters vary across individuals and populations. Hence, CLFNN (HCL) is proposed to assist the hemostasis with prognosis. CLFNN is a functional model of a pattern recognition mechanism in human brain. CLFNN uses positive, negative knowledge, as well as the lateral inhibition between the two classes of knowledge. By exploiting the lateral inhibition, superior performance in pattern recognition can be achieved, as affirmed by the experimental result. Furthermore, CLFNN offers intuitive rules autonomously, enabling user to validate and understand the system, in contrast to black-box CDSS such as MLP. Thus, the confluence of CLFNN and hemostasis displays a promising tool to fight ovarian cancer. However, the present HCL totally ignores the inter-feature relationship, which could be useful in recognition. In future, clustering on the feature and sample space will be implemented, so that the inter-feature relationship can also be used for classification.

References

1. American Cancer Society: Cancer Facts & Figures 2006. American Cancer Society, Inc. 2006
2. A. Seow, W. P. Koh, K. S. Chia, L. Shi, H. P. Lee: Trends in cancer incidence in Singapore 1968-2002. Singapore Cancer Registry. 2004.
3. S. C. L. Koh, R. Khalil, F. –K. Lim, A. Ilancheran, M. Choolani: The association between fibrinogen, von Willebrand Factor, Antithrombin III, and D-Dimer levels and survival outcome by 36 months from ovarian cancer. Clin Appl Thrombosis/Hemostasis. 11(3) (2005) T1-T6
4. A. Gadducci, U. Baicchi, R. Marrai, M. Ferdeghini, R. Bianchi, V. Facchini: Preoperative evaluation of D-Dimer and CA 125 levels in differentiating benign from malignant ovarian masses. Gynecologic Oncology. 60 (1996) 197-202
5. S. C. L. Koh, K. –F. Tham, R. Khalil, P. –L. Oei, F. –K. Lim, A. –C. Roy, R. N. V. Prasad: Hemostatic and fibrinolytic status in patients with ovarian cancer and benign ovarian cysts: could D-Dimer and antithrombin III levels be included as prognostic markers for survival outcome? Clin Appl Thrombosis/Hemostasis. 7(2) (2001) 141-148
6. A. Gadducci, R. Marrai, U. Baicchi, M. Ferdeghini, A. Fanucchi, C. Weiss, A. R. Genazzani: Preoperative D-Dimer plasma assay is not a predictor of clinical outcome for patients with advanced ovarian cancer. Gynecologic Oncology. 66 (1997) 85-88
7. T. Z. Tan, C. Quek, G. S. Ng: Ovarian cancer diagnosis by hippocampus and neocortex-inspired learning memory structures. Neural Networks. 18(5-6) (2005) 818-825
8. K. Griffee, M. J. Dougher: Contextual control of stimulus generalization and stimulus equivalence in hierarchical categorization. Journal of the Experimental Analysis of Behavior. 78(3) (2002) 433-447
9. S. C. L. Koh, R. Khalil, M. Choolani, A. Ilancheran: Systemic prognostic haemostatic marker levels for disease outcome within five years from ovarian cancer. Journal of Thrombosis and Haemostasis. 3 (suppl. 1) (2005) P1314
10. A. Gadducci, R. Marrai, U. Baicchi, O. Gagetti, V. Facchini, A. R. Genazzni: Prothrombin fragment F1+2 and thrombin-antithrombin III complex (TAT) plasma levels in patients with gynecological cancer. Gynecologic Oncology. 61 (1996) 215-217
11. N. Mackman: Role of tissue factor in hemostasis and thrombosis. Blood Cells, Molecules, and Diseases. (2006) in press.
12. R. P. Bagozzi, M. Bergami, L. Leone: Hierarchical representation of motives in goal setting. Journal of Applied Psychology. 88(5) (2003) 915-943
13. J. Hanm, N. Kamber: Data mining: concepts and techniques. San Francisco: Morgan Kanfmann Publishers. 2001.

Multi-class Cancer Classification with OVR-Support Vector Machines Selected by Naïve Bayes Classifier

Jin-Hyuk Hong and Sung-Bae Cho

Dept. of Computer Science, Yonsei University
134 Sinchon-dong, Sudaemoon-ku
Seoul 120-749, Korea
hjinh@sclab.yonsei.ac.kr, sbcho@cs.yonsei.ac.kr

Abstract. Support vector machines (SVMs), originally designed for binary classification, have been applied for multi-class classification, where an effective fusion scheme is required for combining outputs from them and producing a final result. In this work, we propose a novel method in which the SVMs are generated with the one-vs-rest (OVR) scheme and dynamically organized by the naïve Bayes classifiers (NBs). This method might break the ties that frequently occur when working with multi-class classification systems with OVR SVMs. More specifically, we use the Pearson correlation measure to select informative genes and reduce the dimensionality of gene expression profiles when constructing the NBs. The proposed method has been validated on GCM cancer dataset consisting of 14 types of tumors with 16,063 gene expression levels and produced higher accuracy than other methods.

1 Introduction

In many works, SVMs have been used for multi-class classification that is an important task in pattern recognition. Even though SVMs show excellent performance in many applications, it is required to formulate a multi-class SVM method since they are originally designed for binary classification [1]. There are some direct approaches for developing a multi-class extension of SVMs such as Vapnik or Crammer and Singer [2], but this leads to a complex optimization problem. Instead, many researchers prefer to use several binary SVMs, in which constructing a pool of SVMs and combining the outputs of the classifiers are important [3]. One-vs-rest (OVR), pair-wise and complete codes are popular schemes to train a pool of SVMs, while majority voting, winner-takes-all, error-correcting coding are representative fusion methods.

Hsu and Lin compared various schemes such as OVR, pair-wise and DAGSVM (directed acyclic graph SVM), where one-vs-rest and DAGSVM were more suitable for practical use than pair-wise [3]. Bredensteiner and Bennett combined the linear programming and quadratic programming based on SVM for multi-class problems [1], while Angulo *et al.* proposed *K*-SVCR (*K* classes-Support Vector Classification-Regression) [4]. Sebald and Bucklew proposed *M*-ary SVM that uses only $\lceil \log_2(K) \rceil$ SVMs (*K*: # of classes) [5], Gestel *et al.* used Bayesian decoding to compose LS-SVMs (Least Squares SVMs) that repeatedly infer the posterior multi-class probabilities [6]. Crammer and Singer adopted output codes for combining the outputs of multiple classifiers [2]. In the application of bioinformatics, SVMs have been applied for multi-class cancer classification by Lee [7] or Ramaswamy *et al.*[8] where Ramaswamy *et al.* used the OVR and winner-takes-all scheme.

This paper proposes a novel multi-class classification approach integrating SVMs and NBs, especially applying to multi-class cancer classification based on gene expression. Since SVMs might manage the high dimensional data like gene expression profiles, the original training dataset is used to learn SVMs based on the OVR scheme. NBs are designed with the dataset consisting of genes selected by the Pearson-correlation measure, which organize the SVMs dynamically to classify a sample. OVR SVMs are sequentially evaluated by the probability of the classes from the NBs.

The proposed method has been validated on the GCM cancer dataset [8] with three series of experiments. In the first experiment, the NB with Pearson-correlated features achieved an accuracy rate of 72.2%. In the second experiment, SVMs learned with the original training dataset achieved an accuracy rate of 76.9%. Finally, the proposed method, which integrates NBs and SVMs, produced an accuracy rate of 79.6%. This result indicates the benefits of the proposed method that integrates NBs and SVMs effectively.

2 Background

2.1 Cancer Classification Based on Gene Expression

Microarray technology recently developed produces large volume of gene expression profiles and provides richer information on diseases. It simultaneously monitors the expression patterns of thousands of genes under a particular experimental environment. With the technology, many researchers have been studying cancer classification using gene expression profiles. Researchers in pattern recognition have developed and applied several machine learning techniques to many clinical problems by constructing classifiers or predictive models from the data, yielding promising results [9].

Especially the classification of cancers from gene expression profiles is actively investigated in bioinformatics. It commonly consists of feature selection and pattern classification as shown in Fig. 1. In advance, feature selection methods select informative features useful to categorize a sample into predefined classes from lots of gene expression profiles. Pattern classification is composed of learning a classifier with those features and categorizing samples with the classifier [9].

Gene expression profiles provide useful information to classify different forms of cancers, but the data also include useless information for classification. Therefore only relevant one for the classification of cancers should be extracted from them. It is well known that the irrelevant or redundant data degrade the accuracy of classification, so constructing an appropriate gene subset is essential to learn a good classifier. Moreover, it is also important to find a small subset of genes sufficiently informative to distinguish cancers for diagnostic purposes [10].

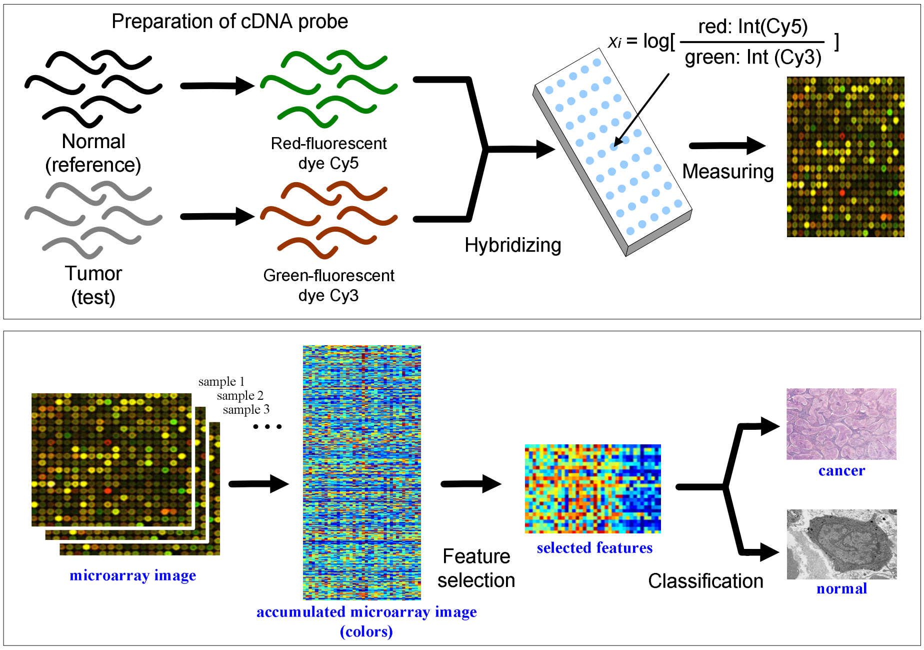


Fig. 1. Classification of gene expression profiles

2.2 Multi-class Classification Using SVMs

SVMs, well researched in statistical learning theory, have been actively investigated in pattern classification and regression. SVMs map an input sample to a high dimensional feature space and try to find an optimal hyperplane that minimizes the recognition error for the training data using the non-linear transformation function [4].

$$X : x = (x_1, \ldots, x_n) \rightarrow F : \Phi(x) = (\Phi_1(x), \ldots, \Phi_n(x)) \tag{1}$$

Let n be the number of training samples. For the sample x_i with the class-label $c_i \in \{1,-1\}$, the SVM calculates

$$f(x) = \sum_{i=1}^{n} \alpha_i c_i K(x, x_i) + b, \quad K(x, x_i) = \Phi(x) \cdot \Phi(x_i) \tag{2}$$

Coefficient α_i in Eq. (2) is non-zero when x_i is a support vector that composes the hyperplane; otherwise it is zero. The kernel function $K(x, x_i)$ can be easily computed by an inner product of the non-linear mapping function. Table 1 shows some representative kernel functions, including the linear, polynomial, Gaussian, and sigmoid functions.

Table 1. Kernel functions of SVMs

Linear	Polynomial	Gaussian	Sigmoid
$(x \cdot x_i)$	$(x \cdot x_i + \gamma)^d$	$\exp\left(-\frac{\lVert x - x_i \rVert^2}{2\sigma^2}\right)$	$\tanh(x \cdot x_i + \gamma)$

Since SVMs are basically binary classifiers, a decomposition strategy for multi-class classification, such as OVR, pairwise or complete-code methods, is required [2]. As a representative scheme, the OVR strategy trains M (# of classes) SVMs, where each SVM classifies samples into the corresponding class against all the others. The decision function $f_j(x)$ of the j^{th} SVM replaces c_i of Eq. (2) with t_i as follows:

$$t_i = \begin{cases} +1 & if \quad c_i = j \\ -1 & if \quad c_i \neq j \end{cases} \tag{3}$$

After constructing SVMs, a fusion method is required to combine the multiple outputs of them. Popular methods for combining multiple outputs include majority voting, winner-takes-all, error-correcting codes (ECCs), behavior knowledge space (BKS) and decision templates.

1) The majority voting method: For a sample, this method simply counts the votes received from the individual classifiers, and selects the class with the largest number of votes. An analytic justification may be given by the well-known Condorcet's theorem, while a theoretical study can be found in [12]. Although it is simple to achieve good performance, this method cannot handle cases where classifiers tie.
2) The winner-takes-all method: In order to resolve problems caused by majority voting, this method classifies a sample into the class that receives the highest value among the L classifiers for the M-class problem. This is often known as maximum where $ind_{i,j}(x)$ is an indicator function with 1 if the label i is the positive class of the jth SVM, -1 if it is the negative class, and 0 if it is otherwise.

$$c = \arg \max_{i=1,\ldots,M} \sum_{j=1}^{L} ind_{i,j}(x) d_j(x) \tag{4}$$

3 A Hybrid Classifier for Multi-class Cancer Classification

Contrary to conventional methods that use static classification, we propose a hybrid classifier that considers the probability of classes obtained by NBs to manage the ambiguity of OVR SVMs. Tie cases, which frequently occur when using OVR SVMs for multi-class classification, might decrease classification performance. The proposed method manages this possibility by organizing OVR SVMs based on the subsumption architecture. The subsumption architecture is a representative method used to select a proper action when there are multiple actions activated [13], while the order of the models is determined by NBs in this paper.

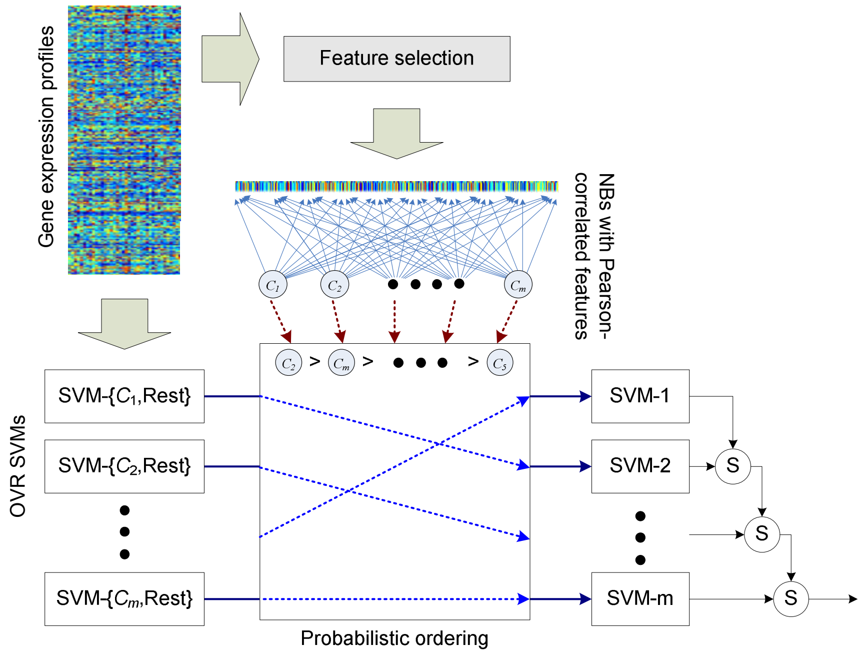


Fig. 2. Overview of the proposed method

The proposed method consists of NBs and OVR SVMs as shown in Fig. 2. NBs estimate the posterior probability for classes *prob* = $\{p_1, p_2, ..., p_m\}$ by using the training dataset that only includes Pearson-correlated genes, while OVR SVMs classify samples by using the original training dataset based on the OVR scheme as explained in Section 2. The margin of a sample *o-svm* = $\{ma_1, ma_2, ..., ma_m\}$ is produced by classifying with OVR SVMs. In order to manage ambiguity in cases of ties (multiple SVMs satisfy) and rejections (no SVM satisfies), in this paper, the proposed method sequentially selects OVR SVMs, where the evaluation order is determined by the posterior probability of each class that NBs produces. The corresponding OVR SVM of a more probable class takes precedence in the subsumption architecture over the others.

When classifying a sample, the method first estimates the probability of each class by using NBs, and then organizes OVR SVMs as the subsumption architecture according to the probability. Finally, a sample is evaluated sequentially until an OVR SVM is satisfied. When an OVR SVM is satisfied, the sample is classified into the corresponding class, while it is classified into the class of the highest probability when no OVR SVMs are satisfied. Fig. 3 shows the pseudo code for the proposed method. Ordering OVR SVMs properly for an input sample produces dynamic classification.

DNA microarray data includes the expression information of thousands or even tens of thousands of genes. Since it is hard to design NBs that includes all the genes, a subset of informative genes is selected by using the feature selection process based on Pearson correlation. Cutting down the number of features to a sufficient minimum is often required to improve classification performance [9].

```
$prob[m] = \{ p_1, p_2, ..., p_m \}$            // $prob[]$ is calculated by NBs
$order[m] = \{0, 1, 2, ..., m-1\}$
$o\text{-}svm[m] = \{ ma_1, ma_2, ..., ma_m \}$   // $o\text{-}svm[]$ is obatined by OVR SVMs

// determine the order of OVR SVMs to evaluate
for(i=0; i<$m$; i++)
for(j=i+1; j<$m$; j++)
   if($prob$[i] < $prob$[j])
   {
     int iTemp = $prob$[i];   $prob$[i] = $prob$[j];        $prob$[j] = iTemp;
     iTemp = $order$[i];      $order$[i] = $order$[j];      $order$[j] = iTemp;
   }

// classify with OVR SVMs according to the subsumption architecture
if($prob$[$order$[0]] < $r_1$)              // $r_1$ is a rejection threshold
   return reject;
for(i=0; i<$m$; i++)
{
   if($o\text{-}svm$[$order$[i]] >= $a$)            // $a$ is a threshold
   {
     if($o\text{-}svm$[$order$[i]] < $r_2$)         // $r_2$ is a rejection threshold
       return reject;
     return $order$[i];
}}
return $order$[0];
```

Fig. 3. Pseudo code for probabilistically ordering OVR SVMs

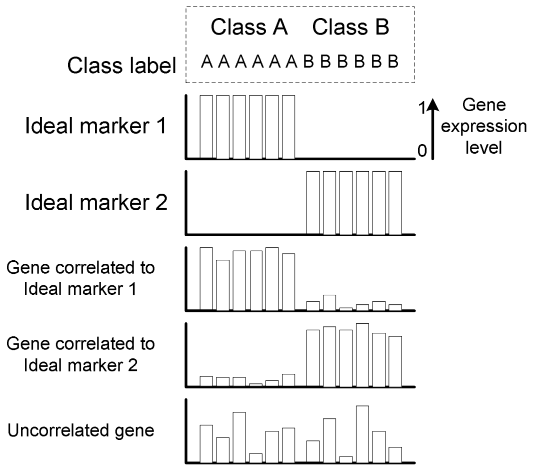


Fig. 4. Negatively correlated features

We define two ideal markers, obtained a standard of good features, and utilize the features by scoring the respective similarity with each ideal marker (as shown in Fig. 4). Two ideal markers are negatively correlated to represent two different aspects of classification boundaries. The first marker is high in class *A* and low in class *B*, and the second marker is low in class *A* and high in class *B*. The first marker is a binary vector which consists of 1 for all the samples in class *A* and 0 for all the samples in class *B*, while the second marker is another binary vector which is composed of 0 for all the samples in class *A* and 1 for all the samples in class *B*. Since the feature selection method is originally designed for binary classification, we select features based on the OVR scheme. Ten genes are selected for each class: the first 5 for the ideal marker 1 and the rest for the ideal marker 2. When there are *m* classes, total *m*×10 genes are used to construct NBs.

The similarity between an ideal marker *ideal* and a gene *g* can be regarded as a distance, while the distance represents how far they are located from one another. A gene is regarded as an informative gene if the distance is small, while the gene is regarded as an uncorrelated gene if the distance is large. Pearson correlation is used to measure the similarity as follows:

$$PC = \frac{\sum_{i=1}^{n}(ideal_i \times g_i) - \frac{\sum_{i=1}^{n} ideal_i \times \sum_{i=1}^{n} g_i}{n}}{\sqrt{\left(\sum_{i=1}^{n} ideal_i^2 - \frac{\left(\sum_{i=1}^{n} ideal_i\right)^2}{n}\right) \times \left(\sum_{i=1}^{n} g_i^2 - \frac{\left(\sum_{i=1}^{n} g_i\right)^2}{n}\right)}} \tag{5}$$

All conditional probabilities in Fig. 2 are estimated from the training set. A_i is the *i*th state of a feature *A*, and *count*(A_i) is the number of samples whose state is A_i. The conditional probability $P(A_i)$ can be estimated with equation (6).

$$P(A_i) = \frac{count(A_i)}{n_T} \tag{6}$$

If *A* has a parent node *B*, $P(A_i|B_j)$ can be estimated by the equation (7).

$$P(A_i \mid B_j) = \frac{count(A_i, B_j)}{count(B_j)} \tag{7}$$

The probability of each class is calculated by inference using the ten features as evidence as follows:

$$P(C \mid F_1, ..., F_n) \tag{8}$$

over a class *C* and $F_1 \sim F_n$. Using the Bayes theorem [20], we can rewrite it as:

$$P(C \mid F_1, ..., F_n) = \frac{P(C)P(F_1, ..., F_n \mid C)}{P(F_1, ..., F_n)} \tag{9}$$

In practice, we are only interested in the numerator of the fraction, since the denominator does not affect C. Feature F_i is conditionally independent from the other feature F_j for $j \neq i$, so the probability of a class is described by expression (10)

$$\begin{aligned} &P(C)P(F_1,...,F_n \mid C) \\ &= P(C)P(F_1 \mid C)P(F_2 \mid C)...P(F_n \mid C) \\ &= P(C)\prod_{i=1}^{n} P(F_i \mid C) \end{aligned} \tag{10}$$

4 Experimental Results

4.1 Dataset

We verify the proposed method with GCM cancer dataset consisting of 144 train samples and 54 test samples with 16,063 gene expression levels GCM [8,14], which is a popular microarray dataset for multi-class cancer classification. There are 14 different tumor categories including breast adenocarcinoma, prostate, lung adenocarcinoma, colorectal adenocarcinoma, lymphoma, bladder, melanoma, uterine adenocarcinoma, leukemia, renal cell carcinoma, pancreatic adenocarcinoma, ovarian adenocarcinoma, pleural mesothelioma, and central nervous system. Since the dataset provides only a few samples but lots of features, it is a challenging job for many machine learning researchers to construct a good classifier. Ramaswamy et al. produced an accuracy of 78% [8], while Li et al. yielded an accuracy of 63.3% [14]. We select 140 genes for learning NBs based on Pearson correlation. The linear kernel is used as a basis kernel function of SVMs. The values of all samples are normalized from 0 to 1.

4.2 Results and Analysis

Table 2 shows competitive results of the proposed method with several traditional approaches. SVMs with the winner-takes-all strategy produced 75.9% classification accuracy, while NBs with Pearson-correlated features yielded an accuracy of 72.2%. The product-based fusion of SVMs and NBs obtained an accuracy of 66.7%, whereas the proposed method produced 79.6% classification accuracy that is higher than the others.

Table 2. Classification accuracy for GCM cancer dataset

Method (feature #)	MLP (140)	SVMs (16,063)	NBs (140)	Product (SVMs+NBs)	Proposed method
Accuracy	46.5% (±2.1)	75.9%	72.2%	66.7%	79.6%

We analyzed samples classified by SVMs and NBs as shown in Table 3. Only 2 samples were failed to classify correctly by both methods, while the proposed method correctly classified 5 samples among 7 samples that are failed by SVMs. A confusion matrix for the test set is presented in Table 4. From this table, we can see that lung, colorectal, uterus, mesothelioma and CNS have been classified 100%, prostate, lymphoma and leukemia 83%.

Table 3. Analysis of classification results

(correct/incorrect): Proposed method		SVMs	
		O	X
NBs	O	38 (38/0)	7 (5/2)
	X	7 (3/4)	2 (0/2)

Table 4. Confusion matrix for the test set (0: Breast, 1: Prostate, 2: Lung, 3: Colorectal, 4: Lymphoma, 5: Bladder, 6: Melanoma, 7: Uterus__Adeno, 8: Leukemia, 9: Renal, 10: Pancreas, 11: Ovary, 12: Mesothelioma, 13: CNS)

%	0	1	2	3	4	5	6	7	8	9	10	11	12	13	n
0	75										25				4
1		83										17			6
2			100												4
3				100											4
4			17		83										6
5			33			67									3
6							50				50				2
7								100							2
8		17							83						6
9								33		67					3
10				33		33					34				3
11				25			25					50			4
12													100		3
13														100	4
n	3	6	6	6	5	3	2	3	5	2	3	3	3	4	54

5 Conclusion

Multi-class classification is a challenging task in pattern recognition, where various approaches have been investigated especially using SVMs. Since SVMs are basically binary classifiers, it is necessary to formulate a fusion method. In this work, we proposed a hybrid classifier that integrates SVMs and NBs learned based on the OVR scheme. GCM cancer dataset, a popular multi-class benchmark dataset in bioinformatics, has been used to verify the proposed method. Since the dataset has an amount of genes, we reduced the dimensionality by using a feature selection method with Pearson correlation. The proposed method showed better performance than SVMs and

NBs when working individually or combined by the product strategy. As the future work, we will demonstrate the proposed method with other popular benchmark datasets of multi-class.

Acknowledgement

This research was supported by Brain Science and Engineering Research Program sponsored by Korean Ministry of Science and Technology.

References

[1] E. Bredensteiner and K. Bennett, "Multicategory classification by support vector machines," *Computational Optimization and Applications*, vol. 12, no. 1, pp. 53-79, 1999.
[2] K. Crammer and Y. Singer, "On the learnability and design of output codes for multiclass problems," *Machine Learning*, vol. 47, no. 2-3, pp. 201-233, 2002.
[3] C. Hsu and C. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Trans. Neural Networks*, vol. 13, no. 2, pp. 415, 425, 2002.
[4] C. Angulo, et al., "K-SVCR. A support vector machine for multi-class classification," *Neurocomputing*, vol. 55, no. 1-2, pp. 57-77, 2003.
[5] D. Sebald and J. Bucklew, "Support vector machines and the multiple hypothesis test problem," *IEEE Trans. Signal Processing*, vol. 49, no. 11, pp. 2865-2872, 2001.
[6] T. Gestel, et al., "Multiclass LS-SVMs: Moderated outputs and coding-decoding schemes," *Neural Processing Letters*, vol. 15, no. 1, pp. 45-58, 2002.
[7] Y. Lee and C. Lee, "Classification of multiple cancer types by multicategory support vector machines using gene expression data," *Bioinformatics*, vol. 19, no. 9, pp. 1132-1139, 2003.
[8] S. Ramaswamy, et al., "Multiclass cancer diagnosis using tumor gene expression signatures," *Proc. National Academy of Science*, vol. 98, no. 26, pp. 15149-15154, 2001.
[9] S.-B. Cho and J. Ryu, "Classifying gene expression data of cancer using classifier ensemble with mutually exclusive features," *Proc. of the IEEE*, vol. 90, no. 11, pp. 1744-1753, 2002.
[10] I. Inza, et al., "Feature subset selection by genetic algorithm and estimation of distribution algorithms. A case study in the survival of cirrhotic patients treated with TIPS," *Artificial Intelligence in Medicine*, vol. 23, no. 2, pp. 187-205, 2001.
[11] L. Kuncheva, "A theoretical study on six classifier fusion strategies," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 281-286, 2002.
[12] R. Brooks, "A robust layered control system for a mobile robot," *IEEE J. of Robotics and Automation*, vol. 2, no. 1, pp. 14-23, 1986.
[13] T. Li, et al., "A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression," *Bioinformatics*, vol. 20, no. 15, pp. 2429-2437, 2004.

Breast Cancer Diagnosis Using Neural-Based Linear Fusion Strategies

Yunfeng Wu[1], Cong Wang[1], S.C. Ng[2], Anant Madabhushi[3], and Yixin Zhong[1]

[1] School of Information Engineering, Beijing University of Posts and Telecommunications, Xi Tu Cheng Road 10 Haidian District, Beijing 100876, China
y.wu@ieee.org
[2] School of Science and Technology, The Open University of Hong Kong, Hong Kong
[3] Department of Biomedical Engineering, Rutgers University, Piscataway, NJ 08854, USA

Abstract. Breast cancer is one of the leading causes of mortality among women, and the early diagnosis is of significant clinical importance. In this paper, we describe several linear fusion strategies, in particular the Majority Vote, Simple Average, Weighted Average, and Perceptron Average, which are used to combine a group of component multilayer perceptrons with optimal architecture for the classification of breast lesions. In our experiments, we utilize the criteria of mean squared error, absolute classification error, relative error ratio, and Receiver Operating Characteristic (ROC) curve to concretely evaluate and compare the performances of the four fusion strategies. The experimental results demonstrate that the Weighted Average and Perceptron Average strategies can achieve better diagnostic performance compared to the Majority Vote and Simple Average methods.

1 Introduction

Breast cancer is one of the leading forms of cancer diagnosed among women in the United States [9]. The latest surveillance investigation indicates this type of cancer accounts for an estimated 32% incidence rate and an estimated 15% mortality rate in 2005, ranking second only to lung carcinoma [9]. The most common and palpable signs of cancer are lumps or masses detected in the breast, and the benign masses are frequent in a majority of cases [12]. Studies have shown that early diagnosis by means of breast imaging, including digital mammography, ultrasound imaging, and magnetic resonance imaging (MRI), could help prognosis and increase therapeutic options [4]. In this paper, we are considering the binary classification problem of distinguishing benign or malignant breast lesions. In order to improve the biopsy yield ratio, techniques and systems are being developed for computer-aided diagnosis, to effectively assist radiologists and physicians in screening and diagnosis [1].

Recently, artificial neural networks have been applied to classifying mammographic masses for early-stage breast cancer detection and diagnosis [21], which would help reduce the number of unnecessary surgical biopsies. Artificial neural networks, with the properties of experience-based learning and generalization ability, are regarded as one of the emerging computational technologies for solving complex problems that might not have a tractable solution provided by traditional methods.

However, when given a complex data set, different neural classifiers typically provide diverse generalizations by determining different boundaries. The variety of performance would be dramatically influenced by a number of factors, including different network architectures, learning styles (supervised or unsupervised), network architecture (the number of layers and hidden nodes involved, type of activation functions, and degree of connectivity), training parameters (weights initialization, learning rates, and training epochs), and so forth.

Previous research showed that an ensemble of neural networks may significantly improve the generalization capability of an intelligent system [10], [18]. Other than solely toiling over the training data toward an expected generalization, a group of Component Neural Networks (CNNs) could work collectively with given fusion strategies to ameliorate the classification capability, and then hopefully solve an entire complex problem. The ensembles of neural networks can be divided into two main categories: Generative and Nongenerative methods [11]. The Generative methods generate a series of CNNs whose training sets are determined by the performance of former ones (e.g. Boosting [5]), or based on the bootstrap sampling data sets (e.g. Bagging [2]). The Nongenerative ensembles combine their well-devised CNNs to comprehend the entire problem and drive a comprehensive decision with the fusion strategies. The research focus has recently been shifted from practical applications of ensembles towards investigating why ensembles and fusion strategies may work so well and in which situations some methods may outperform the others [14], [23]. In the following sections, we will focus on the Nongenerative ensemble methods in the context of distinguishing between malignant and benign breast lesions.

The rest of this paper is organized as follows. Section 2 and Section 3 describe the optimal Multilayer Perceptron (MLP) architecture selection and several linear fusion strategies applied in our experiments. Section 4 presents the empirical results of breast cancer diagnosis. Section 5 discusses some technical details of linear fusion strategies. Conclusion and directions for the future work are presented in Section 6.

2 Optimal MLP Architecture Selection Based on Regularization

The implementation of the optimal MLP architecture selection in our work contains two steps: First, search the minimum risks associated with a series of MLP structures based on parameter regularization and cross-validation; later select the optimal MLP architecture according to the dynamics of the minimum-risk ranking.

Interpreted as a nonlinear system, a MLP maps the input features $\mathbf{x}$, $\mathbf{x} \in \Re^{P \times N}$ by following the rule: $\mathbf{O}(\mathbf{x}, \mathbf{w})$, $\mathbf{O} \in \Re^{P \times M}$. Referring to Hornik *et al.* [8], we consider the MLP with N input nodes, K hidden nodes in only one hidden layer, and M output nodes (herein denoted as (N-K-M) architecture.) Let $w_{k,m}$ be the weight between m-th output node and k-th hidden node, and $\overline{w}_{n,k}$ be the weight between the k-th hidden node and n-th input node. The MLP architecture is selected by minimizing a scalar risk function $R(\mathbf{w}, \boldsymbol{\lambda})$, which is the sum of a performance-loss function $E(\mathbf{w})$, and a complexity-cost function $C(\mathbf{w})$ parameterized by a linear regularization vector $\boldsymbol{\lambda}$, i.e.,

$$R(\mathbf{w}, \boldsymbol{\lambda}) = E(\mathbf{w}) + \boldsymbol{\lambda}^{\mathrm{T}} C(\mathbf{w}) \tag{1}$$

where the parameter $\boldsymbol{\lambda}$ represents the relative importance of the complexity-cost in respect of the performance-loss.

For regression and signal processing problems, the loss function is normally measured by mean squared errors between the expected targets $\mathbf{t}_p$ and the estimated outputs over training patterns, i.e.,

$$E(\mathbf{w}) = \frac{1}{P}\sum_{p=1}^{P}\left[\mathbf{t}_p - \hat{\mathbf{O}}(\mathbf{x}_p, \mathbf{w})\right]^2 = \frac{1}{P}\sum_{p=1}^{P}\left[\mathbf{t}_p - \hat{\mathbf{O}}(\mathbf{x}_p, \mathbf{w})\right]^{\mathrm{T}} \cdot \left[\mathbf{t}_p - \hat{\mathbf{O}}(\mathbf{x}_p, \mathbf{w})\right] = \frac{1}{P}\sum_{p=1}^{P} e_p(\mathbf{w})^{\mathrm{T}} \cdot e_p(\mathbf{w}) \tag{2}$$

where $e_p(\mathbf{w})$ denotes the error between the expected targets and estimated outputs.

There are some complexity regularization methods, well-known as Weight Decay [7] and Weight Elimination [19]. Here we only consider the Weight Decay proposed by Hinton *et al.* [7]. In the Weight Decay, the complexity cost is defined as squared norm of the synaptic weights, including the input-to-hidden and hidden-to-output weights. Thus the regularization term in the risk function is

$$\begin{aligned}\boldsymbol{\lambda}^{\mathrm{T}} \cdot C(\mathbf{w}) &= \boldsymbol{\lambda}^{\mathrm{T}} \cdot \|\mathbf{w}\|^2 = \lambda_{NK}\left\|\overline{\mathbf{w}}_{N,K}\right\|^2 + \lambda_{KM}\left\|\mathbf{w}_{K,M}\right\|^2 \\ &= [\lambda_{NK}, \lambda_{KM}] \cdot \left[\left\|\overline{\mathbf{w}}_{N,K}\right\|^2, \left\|\mathbf{w}_{K,M}\right\|^2\right]^{\mathrm{T}} = [\lambda_{NK}, \lambda_{KM}] \cdot \left[\overline{\mathbf{w}}_{N,K}^{\mathrm{T}} \cdot \overline{\mathbf{w}}_{N,K}, \mathbf{w}_{K,M}^{\mathrm{T}} \cdot \mathbf{w}_{K,M}\right]^{\mathrm{T}}\end{aligned} \tag{3}$$

For architecture selection purpose, the Cross-Validation approach [3], [16] is employed to validate the optimal network architecture with the best-performance parameter estimates. Normally, data for regression and classification problems may involve a training set and a testing set, and in the L-fold cross-validation method, all the available training set of P patterns would be randomly split into L disjoint subsets of approximately equal size, i.e. $P = \bigcup_{l=1}^{L} P_{\mathrm{V}}^{l}$ and $\forall i \neq j : P_{\mathrm{V}}^{i} \cap P_{\mathrm{V}}^{j} = \varnothing$. Training and validation are repeated for a total of L trials, in the l-th iteration using the subset $P \setminus P_{\mathrm{V}}^{l}$ for training and the other subset P_{V}^{l} for validation. The performance-loss of L-fold cross-validation is estimated by the average of validation mean squared errors:

$$\Gamma_{\mathrm{V}}(\hat{\mathbf{w}}) = \frac{1}{L}\sum_{l=1}^{L} E_{\mathrm{V}}^{l}(\hat{\mathbf{w}}^{l}) \tag{4}$$

$$E_{\mathrm{V}}^{l}(\hat{\mathbf{w}}^{l}) = \frac{1}{P_{\mathrm{V}}^{l}}\sum_{j\in P_{v}^{l}}\left[\mathbf{t}_j - \hat{\mathbf{O}}(\mathbf{x}_j^{l}, \hat{\mathbf{w}}^{l})\right]^2 = \frac{1}{P_{\mathrm{V}}^{l}}\sum_{j\in P_{v}^{l}} e_j(\hat{\mathbf{w}}^{l})^{\mathrm{T}} \cdot e_j(\hat{\mathbf{w}}^{l}) \tag{5}$$

Using the second order information during regularization [3], the parameter vector $\boldsymbol{\lambda}$ would converge through the gradient descent path of the network risk:

$$\boldsymbol{\lambda}^{(i+1)} = \boldsymbol{\lambda}^{(i)} - \eta \cdot \nabla_{\boldsymbol{\lambda}} \Gamma_{\mathrm{V}}(\hat{\mathbf{w}}) \tag{6}$$

where is $\eta > 0$ is the convergence update rate. Note that during the i-th iteration the synaptic weights $\hat{\mathbf{w}}$ (or to be explicitly written as $\hat{\mathbf{w}}(\boldsymbol{\lambda})$) is an implicit function of $\boldsymbol{\lambda}^{(i)}$, since $\boldsymbol{\lambda}$ could only be optimized after the settlement of synaptic weights $\hat{\mathbf{w}}$. In case of linear regularization, the gradient of the cross-validation error is

$$\nabla_{\boldsymbol{\lambda}} \Gamma_{\mathrm{V}}(\hat{\mathbf{w}}) = \frac{1}{L}\sum_{l=1}^{L} \frac{\partial E_{\mathrm{V}}^{l}(\hat{\mathbf{w}}^{l})}{\partial \boldsymbol{\lambda}} \tag{7}$$

Following the differential chain rule, the gradient vector of the cross-validation error can be derived:

$$\frac{\partial E_{\mathrm{V}}^{l}(\hat{\mathbf{w}}^{l})}{\partial \boldsymbol{\lambda}} = \frac{\partial \mathbf{w}^{l^{\mathrm{T}}}}{\partial \boldsymbol{\lambda}} \cdot \left(\hat{\mathbf{w}}^{l}(\boldsymbol{\lambda})\right) \cdot \frac{\partial E_{\mathrm{V}}^{l}(\hat{\mathbf{w}}^{l})}{\partial \mathbf{w}} \tag{8}$$

where $\partial \mathbf{w}^{l^{\mathrm{T}}} / \partial \boldsymbol{\lambda}$ is the derivative matrix of synaptic weights. To get this derivative matrix, we consider the Taylor expansion of scalar risk function around $\boldsymbol{\lambda}^{(i)}$:

$$\frac{\partial R(\mathbf{w}, \boldsymbol{\lambda})}{\partial \mathbf{w}} = \frac{\partial R(\mathbf{w}, \boldsymbol{\lambda}^{(i)})}{\partial \mathbf{w}} + \frac{\partial^2 R(\mathbf{w}, \boldsymbol{\lambda}^{(i)})}{\partial \mathbf{w} \partial \boldsymbol{\lambda}^{\mathrm{T}}} \cdot \left(\boldsymbol{\lambda} - \boldsymbol{\lambda}^{(i)}\right) + o\left(\left|\boldsymbol{\lambda} - \boldsymbol{\lambda}^{(i)}\right|\right) \tag{9}$$

where $o\left(\left|\boldsymbol{\lambda} - \boldsymbol{\lambda}^{(i)}\right|\right)$ represents a high-order small value which could be ignored when estimated. Note that when regularization parameter vector $\boldsymbol{\lambda}$ meets the optimal scene (i.e. both the gradient of the cross-validation error and network risk cannot be updated further), derived from (9), we have

$$\frac{\partial^2 R(\mathbf{w}, \boldsymbol{\lambda}^{(i)})}{\partial \mathbf{w} \partial \boldsymbol{\lambda}^{\mathrm{T}}} = 0 \tag{10}$$

Combining (1), (3), (4), and (7), we may develop

$$\frac{\partial \mathbf{w}^{l^{\mathrm{T}}}}{\partial \boldsymbol{\lambda}} \cdot \left(\hat{\mathbf{w}}^{l}(\boldsymbol{\lambda})\right) = -\frac{\partial C(\hat{\mathbf{w}}^{l})}{\partial \mathbf{w}^{\mathrm{T}}} \cdot \left[\frac{\partial^2 R(\mathbf{w}^{l}, \boldsymbol{\lambda})}{\partial \mathbf{w} \partial \mathbf{w}^{\mathrm{T}}}\right]^{-1} \tag{11}$$

Finally, substituting (11) into (8) gives

$$\frac{\partial E_{\mathrm{V}}^{l}(\hat{\mathbf{w}}^{l})}{\partial \boldsymbol{\lambda}} = -\frac{\partial C(\hat{\mathbf{w}}^{l})}{\partial \mathbf{w}^{\mathrm{T}}} \cdot \left[\frac{\partial^2 R(\hat{\mathbf{w}}^{l}, \boldsymbol{\lambda})}{\partial \mathbf{w} \partial \mathbf{w}^{\mathrm{T}}}\right]^{-1} \cdot \frac{\partial E_{\mathrm{V}}^{l}(\hat{\mathbf{w}}^{l})}{\partial \mathbf{w}} = -2\hat{\mathbf{w}}^{l^{\mathrm{T}}} \cdot H_{\mathrm{V}}^{-1}(\hat{\mathbf{w}}^{l}) \cdot \frac{\partial E_{\mathrm{V}}^{l}(\hat{\mathbf{w}}^{l})}{\partial \mathbf{w}} \tag{12}$$

where $H_{\mathrm{V}}(\hat{\mathbf{w}}^{l}) = \partial^2 R(\hat{\mathbf{w}}^{l}, \boldsymbol{\lambda}) / \partial \mathbf{w} \partial \mathbf{w}^{\mathrm{T}}$ is the Hessian matrix of the risk function, and $\partial E_{\mathrm{V}}^{l}(\hat{\mathbf{w}}^{l}) / \partial \mathbf{w}$ could be estimated during training over the validation subset.

3 Linear Fusion Strategies for Combining Neural Classifiers

There are several Non-generative neural networks fusion strategies that have proved to be effective in improving the classification performance [10], [22]. In general, they can be differentiated into two styles: Fixed and Trained rules [15]. Fixed rules, e.g. Majority Vote (MV) [18] and Simple Average (SA) [14], do not need any training phase in the fusion. Trained rules, on the other hand, like Weighted Average (WA) [15] and Perceptron Average (PA) [22], require a learning phase to initialize and adjust fusion parameters. For the MV fusion, the class which receives the largest number votes among the CNNs is chosen as the consensus or majority decision. For the SA and WA fusions, the CNNs are linearly combined to form an overall decision. In this investigation, we use the both fixed and trained fusion methods to effectively improve performance of multiple classifier systems for breast cancer diagnosis.

Simple Average (SA). For the SA fusion, the outputs of the independently trained CNNs are assumed to be scalar-valued and then linearly combined with the equal fusion coefficients to form an overall output. Assume a fusion combines the outputs of total K CNNs, with normalized fusion coefficients α_k, and we have

$$F(\mathbf{x}_p) = \sum_{k=1}^{K} \alpha_k \cdot \mathbf{O}_k(\mathbf{x}_p, \hat{\mathbf{w}}), \quad \alpha_k \ge 0 \tag{13}$$

$$\sum_{k=1}^{K} \alpha_k = 1, \quad k = 1,2,\ldots,K \tag{14}$$

where $\mathbf{O}_k(\mathbf{x}_p, \hat{\mathbf{w}})$ denotes the output of the k-th CNN for a given p-th input pattern vector $\mathbf{x}_p$. For the SA fusion, the fusion coefficients are $\alpha_k = 1/K$.

Weighted Average (WA). In the WA fusion [17], for a one-dimensional input x_p, the estimation of the *a posteriori* probability of the i-th class from the output of the k-th CNN is denoted as $\hat{p}_k^i(\mathbf{x}_p)$. According to Roli *et al.* [15], it can be expressed as

$$\hat{p}_k^i(\mathbf{x}_p) = p_k^i(\mathbf{x}_p) + \varepsilon_k^i(\mathbf{x}_p) \tag{15}$$

where $p_k^i(\mathbf{x}_p)$ is the *a posteriori* probability of the i-th class, and $\varepsilon_k^i(\mathbf{x}_p)$ denotes the estimation error. Assume that the class boundaries provided from the approximate *a posteriori* probabilities are close to the optimal Bayes boundaries [17]. According to Tumer *et al.* [17], if the estimation errors $\varepsilon_k^i(\mathbf{x}_p)$ on different classes are independent and identically distributed (i.i.d.) variables with zero mean and variance σ_ε^2, the expectation of the added errors (the error in addition to the Bayesian one) can be expressed as $E^{add} = \sigma_\varepsilon^2 / s$, where s is a constant term depending only on the values of probability density functions at the optimal decision boundary. Using (14) and (15), under the hypothesis that the output of the network approximates the posterior probabilities of the classes, the *a posteriori* probability of the linear fusion is

$$\hat{p}_{ave}^i(\mathbf{x}_p) = p_{ave}^i(\mathbf{x}_p) + \sum_{k=1}^{K} \alpha_k \cdot \varepsilon_k^i(\mathbf{x}_p) = p_{ave}^i(\mathbf{x}_p) + \bar{\varepsilon}^i(\mathbf{x}_p) \tag{16}$$

where $\bar{\varepsilon}^i(\mathbf{x}_p)$ denotes the estimation fusion. In the case of uncorrelated estimation errors, the expectation E_{ave}^{add} of the added error of the WA fusion is [15]

$$E_{ave}^{add} = \sum_{k=1}^{K} E_k^{add} \cdot \alpha_k^2 \tag{17}$$

Considering (13), the fusion coefficients that minimize E_{ave}^{add} are [17]

$$\alpha_j = \left(\sum_{k=1}^{K} 1/E_k^{add} \right)^{-1} \cdot \left(1/E_j^{add} \right) \tag{18}$$

In other words, the optimal fusion coefficients are inversely proportional to the expectation errors of each CNN.

Perceptron Average (PA). When the data are statistical independent Gaussian distributed, the operation of the Bayes classifier reduces to a linear classifier, which is equivalent to the perceptron having exponential family activation functions [6]. Note that the WA fusion is "parametric," because its derivation is contingent on the assumption that the underlying distributions of the estimation errors $\varepsilon_k(\mathrm{x}_p)$ on different classes are Gaussian, which may limit its area of applications. On the other hand, the perceptron convergence algorithm is "non-parametric" in the sense that it makes no assumptions concerning the form of the underlying distributions [6]. It operates by concentrating on errors that occur where the distributions overlap. It may therefore work well when the input patterns are generated by some nonlinear physical mechanisms whose distributions might be heavily skewed and non-Gaussian. With such a concept, we may utilize the perceptron convergence algorithm to train the linear fusion to obtain the optimal fusion coefficients assigned to each output of the CNNs. In the PA fusion, the bias $b^{(n)}(\mathrm{x}_p)$ over the p-th input pattern at the n-th training epoch is treated as an additional coefficient driven by a fixed input equal to +1. Let $D^{(n)}(\mathrm{x}_p)$ denote the desired fusion output at the n-th training epoch, we have:

$$D^{(n)}(\mathbf{x}_p) = \begin{cases} +1 & \text{if } \mathbf{x}_p \text{ belongs to } \textit{malignant} \\ -1 & \text{if } \mathbf{x}_p \text{ belongs to } \textit{benign} \end{cases} \tag{19}$$

Thus, the fusion coefficients and bias are updated by following the rule:

$$\alpha_k^{(n+1)} = \alpha_k^{(n)} + \left[D^{(n)}(\mathbf{x}_p) - \operatorname{sgn}\left(F^{(n)}(\mathbf{x}_p)\right)\right] \cdot \mathbf{O}_k^{(n)}(\mathbf{x}_p, \hat{\mathbf{w}}) \tag{20}$$

$$b^{(n+1)}(\mathbf{x}_p) = b^{(n)}(\mathbf{x}_p) + \left[D^{(n)}(\mathbf{x}_p) - \operatorname{sgn}\left(F^{(n)}(\mathbf{x}_p)\right)\right] \tag{21}$$

4 Experimental Results

4.1 Data Description

The data set applied in our experiments was obtained from the Wisconsin Diagnostic Breast Cancer Database described by Mangasarian *et al.* [13]. The data set contains 569 instances (357 benign cases and 212 malignant cases) with thirty real-valued input features, including the mean, standard error, and "worst" or largest (mean of the three largest values) of ten cell nucleus attributes (i.e. radius, texture, perimeter, area, smoothness, compactness, concavity, concave, points, symmetry, fractal dimension). In the experiments, we split the whole data set into two sets: training set and testing set, each involving 200 instances and 369 instances, respectively. And we divided the thirty input features into three parts: Mean, Standard Error, and Largest Deviation features of the ten cell nucleus attributes correspondingly sent to three CNNs (labelled CNN-1, CNN-2, CNN-3) which to be independently trained by the Resilient Back-propagation, Scaled Conjugate Gradient, and Levernberg-Marquardt algorithms. All the input features were normalized to zero mean and unity standard deviation in order to accelerate the backpropagation learning process. And the MLP performance was validated with the 10-fold cross validation.

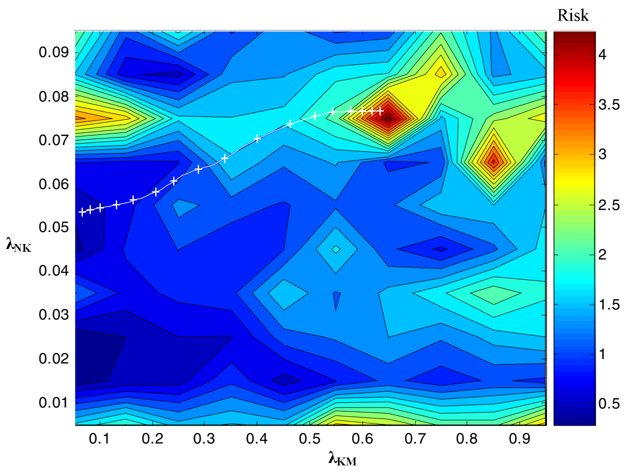


Fig. 1. The convergence of regularization parameter vector $\boldsymbol{\lambda} = [\lambda_{NK}, \lambda_{KM}]^T$ through the gradient descent path of the network risk (the track points are depicted as "+"). The current MLP architecture is (*10-4-1*) and trained by the Resilient BP algorithm.

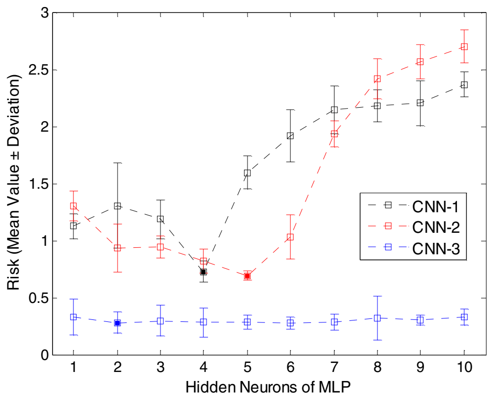


Fig. 2. MLP risk dynamics curves and MSE performance independently carried out by different algorithms for the CNNs in breast cancer diagnosis

4.2 Results of Optimal MLP Architecture Selection

Referring to (1), the risk exported from a MLP would be jointly affected by the synaptic weights and regularization parameter vector $\boldsymbol{\lambda}$. In order to achieve the optimal architecture for a particular task, we first relax the number of the hidden nodes at the range from 1 to 10, and then search the most appropriate parameter vector $\boldsymbol{\lambda}$ which minimizes the risk associated with each certain network structure. In this case, we will have a series of 10 dynamic networks which reach the minimum risks within their own structures. On comparison of the risk dynamics, the optimal MLP architecture could be found. In Fig. 1, we can observe that the regularization parameter vector λ converges through the gradient descent risk path on the network (*10-4-1*) for CNN-1. By locating the regularization parameter vector λ after convergence (referring to (6)), we will find the minimum risk on each network structure. Later, the optimal MLP architecture can be obtained according to the risk dynamics curve varying from a series of hidden nodes (see Fig. 2). The optimal structures for CNN-2 and CNN-3 are (*10-5-1*) and (*10-2-1*), respectively. Their regularization convergence situations are quite similar to the one of CNN-1, and are not illustrated again.

4.3 Results of Breast Lesion Classification Via Linear Fusion Strategies

Table 1 and Table 2 show the weighted coefficients and diagnostic results of the four fusion strategies. Note that absolute errors are the misclassification cases in percentage terms, and for relative error ratios, the averaged error of the CNNs is regarded as 1.0000, and the reported error of each fusion is in fact the ratio over that of the averaged value of the CNNs. In Table 2, we find that MV and SA are at the same degree, when WA and PA are at a lower level in terms of relative error ratio.

Table 1. Normalized fusion coefficients assigned to the CNNs in different fusion strategies

Fusion Strategy	Weighted Coefficients		
	CNN-1	CNN-2	CNN-3
MV Fusion	N/A	N/A	N/A
SA Fusion	0.3333	0.3333	0.3333
WA Fusion	0.3727	0.2037	0.4236
PA Fusion	0.3401	0.2081	0.4518

Table 2. Diagnostic performances of different fusion strategies

	MSE	Absolute Error (%)	Relative Error Ratio
CNN averaged	0.3577	8.9431	1.0000
MV Fusion	0.3111	7.7778	0.8697
SA Fusion	0.2883	7.2087	0.8061
WA Fusion	0.2060	5.1491	0.5758
PA Fusion	0.1951	4.8780	0.5454

Measures of overall error of classification as percentage provide limited indications in a medical diagnostic method. Especially in breast cancer diagnosis, a misidentification between benign mass and malignant tumor has their different costs [4]. The provision of separate correct classification rates for each class, such as Sensitivity/True Positive (TP) rate (the percentage of cancer correctly diagnosed) and Specificity (the percentage of benign lesions correctly diagnosed), can facilitate improved analysis. A Receiver Operating Characteristic (ROC) curve is a plot of operating points showing the possible tradeoff between the classifier's TP rate versus its False Positive (FP) rate

(1-Specificity) [20]. A summary measure of effectiveness of classifier is given by the ROC Area Under Curve (AUC). Here we show two zoomed ROC plots in Figures 3 (a) and (b), because if we move out the range of 0.03 to 0.3 in the horizontal axis, all four fusion ROCs tend to converge with no apparent significant differences. It is clear from Fig. 3 that the PA fusion's ROC covers a larger area (AUC = 0.9801) than the second ranking one of the SA fusion (AUC = 0.9775). It is interesting that the WA fusion strategy just covers the smallest area under the ROC, and the further discussion is provided in Section 5.

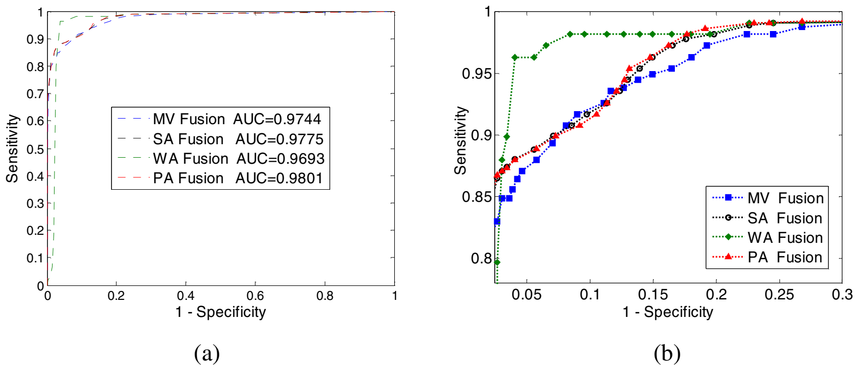


Fig. 3. ROC curves of all four fusion strategies in breast cancer diagnosis: (a) panoramic curves, and (b) zoomed curves from the range of 0.03 to 0.3 in the horizontal axis

5 Discussion

For the MV fusion, assuming that only two classes are considered, and we restrict the choice of the number of CNNs (N) to an odd number. The MV fusion will assign the wrong class to input vector $\mathbf{x}$ if at least $\frac{N+1}{2}$ CNNs incorrectly vote for it. It is therefore possible that the consensus decision might be worse than that of the best individual CNN. So the decision of MV fusion might not be always superior to all the individual CNNs.

The SA fusion is widely used due to its simplicity and effectiveness, which has been demonstrated in several experimental studies. However, it might suffer from individual classifiers whose performances are significantly diverse. In our experiments, the SA fusion was poor at fusing the individual CNNs, i.e., the relative error ratio of the SA fusion is 0.2303 and 0.2607 above those in the WA and PA fusions, respectively (see Table 2).

For the WA fusion, we note in Fig. 3 (a) that the ROC curve of the WA fusion ascends slowly (even behind the MV and SA fusions) from 0 to 0.02 in the horizontal axis. We believe that the preliminary assumption of Gaussian distributions for the estimation errors on different classes in the WA fusion results in this phenomenon in the ROC curve, especially when a casualty of training data sizes.

The PA fusion can achieve the lowest absolute error and relative error ratio in our experiments, but it is vastly inferior to the WA when a moderate FP rate is tolerable. This could be the direction for us to improve the PA fusion in the future work.

6 Conclusion

In this paper, we presented the MLP architecture selection method based on parameter regularization and cross-validation, and four linear fusion strategies for combining the component MLP classifiers. The numerical experiments reveals the pitfalls of the MV, SA, and WA fusion strategies in solving the classification of breast lesions, and also exhibits the advantages of the PA fusion strategy, which achieves the lowest absolute error and relative error ratio, and has the top ranking AUC in its ROC versus the other linear fusion strategies. The development of new adaptive weighted average algorithm and the nonlinear fusion strategies will be the next step of our work.

Acknowledgment

This work was supported in part by National Science Foundation of China under the Grant No. 60575034, and the 2005 Innovation Research Funds from Graduate School, Beijing University of Posts and Telecommunications.

References

1. Baker, J.A., Rosen, E.L., Lo, J.Y., Gimenez, E.I., Walsh, R., Soo, M.S.: Computer-aided Detection (CAD) in Screening Mammography: Sensitivity of Commercial CAD Systems for Detecting Architectural Distortion. American J. Roentgenology **181** (2003) 1083–1088
2. Breiman, L.: Bagging Predictors. Machine Learning **24** (1996) 123-140
3. Chen D., Hagan M.: Optimal Use of Regularization and Cross-Validation in Neural Network Modeling. Proc. the 1999 Int'l Joint Conf. on Neural Networks (1999) 1275–1280
4. Donegan, W.L., Spratt, J.S., Orsini, A. (ed.): Cancer of the Breast. 5th edn. Elsevier Science, Amsterdam, Netherlands (2002)
5. Freund, Y., Schapire, R.E.: Experiments with a New Boosting Algorithm. Proc. the 15th Int'l Conf. on Machine Learning (1996) 148–156
6. Haykin, S.: Neural Networks: A Comprehensive Foundation. 2nd edn. Prentice Hall PTR, Englewood Cliffs, NJ, USA (1998)
7. Hinton, G.E.: Connectionist Learning Procedures. Artificial Intelligence **40** (1989) 185-234
8. Hornik, K.M., Stinchcombe, M., White, H.: Multilayer Feedforward Networks are Universal Approximators. Neural Networks **2** (1989) 359–366
9. Jemal, A., Murray, T., Ward, E., Tiwari, R.C., Ghafoor, A., Feuer, E.J., Thun, M.J.: Cancer Statistics, 2005. CA: A Cancer Journal for Clinicians **55** (2005) 10–30
10. Kuncheva, L.I.: A Theoretical Study on Six Classifier Fusion Strategies. IEEE Transactions on Pattern Analysis and Machine Learning **24** (2002) 281–286
11. Madabhushi, A., Feldman, M., Metaxas, D., Tomasezweski, J., Chute, D.: Automated Segmentation of Prostatic Adenocarcinoma from High Resolution MR by Optimally Combining 3D Texture Features. IEEE Transactions on Medical Imaging **24** (2005) 1611–1625

12. Madabhushi, A., Metaxas, D.: Combining, Low, High and Empirical Domain Knowledge for Automated Segmentation of Ultrasonic Breast Lesions. IEEE Transactions on Medical Imaging **22** (2003) 155–169
13. Mangasarian, O.L., Street, W.N., Wolberg, W.H.: Breast Cancer Diagnosis and Prognosis via Linear Programming. Operations Research **43** (1995) 570–577
14. Roli, F., Giacinto, G.: Design of Multiple Classifier Systems. In: Bunke, H., Kandel, A. (eds.): Hybrid Methods in Pattern Recognition. World Scientific Publishing (2002)
15. Roli, F., Fumera, G., Kittler, J.: Fixed and Trained Combiners for Fusion of Unbalanced Pattern Classifiers. Proc. the 5th Int'l Conf. on Information Fusion (2002) 278–284
16. Stone M.: Cross-validatory Choice and Assessment of Statistical Predictions. Journal of Royal Statistics Society **B36** (1974) 111–133
17. Tumer, K., Ghosh, J.: Analysis of Decision Boundaries in Linearly Combined Neural Classifiers. Pattern Recognition **29** (1996) 341–348
18. Wanas, N.M., Kamel, M.S.: Decision Fusion in Neural Network Ensembles. Proc. of the 2001 Int'l Jt. Conf. on Neural Networks **4** (2001) 2952–2957
19. Weigend, A.S., Rumelhart, D.E., Huberman, B.A.: Generalization by Weight-Elimination with Application to Forecasting. Advances in Neural Information Processing Systems **3** (1991) 875–882
20. Woods, K., Bowyer, K.W.: Generating ROC Curves for Artificial Neural Networks. IEEE Transactions on Medical Imaging **16** (1997) 329–337
21. Wu, Y., He, J., Man, Y., Arribas, J.I.: Neural Network Fusion Strategies for Identifying Breast Masses. Proc. the 2004 Int'l Jt. Conf. on Neural Networks **3** (2004) 2437–2442
22. Wu, Y., Zhang, J., Wang, C., Ng, S.C.: Linear Decision Fusions in Multilayer Perceptrons for Breast Cancer Diagnosis. Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence (2005) 699–700
23. Zhou, Z.H., Wu, J, Tang, W.: Ensembling Neural Networks: Many Could be Better Than All. Artificial Intelligence **137** (2002) 239–263

A Quantitative Diagnostic Method Based on Bayesian Networks in Traditional Chinese Medicine

Huiyan Wang[1] and Jie Wang[2]

[1] College of Computer Science & Information Engineering, Zhejiang Gongshang University, 310035, Hangzhou, Zhejiang, China
cederic@mail.zjgsu.edu.cn
[2] Xiyuan Hospital, China Academy of Traditional Chinese Medicine, 100091, Beijing, China

Abstract. Traditional Chinese Medicine (TCM) is one of the most important complementary and alternative medicines. Due to the subjectivity and fuzziness of diagnosis in TCM, quantitative model or methods are needed to facilitate the popularization of TCM. In this article, a novel quantitative method for syndrome differentiation based on BNs is proposed. First the symptoms are selected by a novel mutual information based symptom selection algorithm (MISS) and then the mapping relationships between the selected symptoms and key elements are constructed. Finally, the corresponding syndromes are output by combining the key elements. The results show that the diagnostic model obtains relative reliable predictions of syndrome, and its average predictive accuracy rate reach 91.68%, which testifies that the method we proposed is feasible and effective and can be expected to be useful in the modernization of TCM.

Keywords: Traditional Chinese Medicine (TCM), Quantitative diagnosis, Bayesian networks, Symptom selection, Syndrome differentiation.

1 Introduction

Traditional Chinese Medicine (TCM) is one of the most important complementary and alternative medicines. With a history of over 23 centuries, TCM has formed an integrated medical system which diagnoses, treats, and prevents diseases. On one hand, TCM can diagnose diseases at an earlier stage and subsequently prevent the state of illness from deteriorating by adjusting the balance within the body in time. On the other hand, TCM causes little pain, no injury and treats the human body as a whole. Therefore, TCM has been accepted gradually and applied more frequently in the world.

Syndrome differentiation is a method of understanding and diagnosing disease by the theories of TCM. The overall guiding principles for clinical treatment are based on the results of syndrome differentiation. Traditional diagnosis in TCM requires long experiences and a high level of skill, and is subjective and deficient in quantitative diagnostic criteria. This seriously affects the reliability and repeatability of diagnosis and limits the popularization of TCM. So the focal problem that needs to be solved urgently is to construct quantitative methods or models to differentiate syndrome automatically. Recently, a few researchers developed some methods and systems to modernize TCM. But most of them are built incorporating totally or partially rule-based reasoning model [1], which are lack of the feasibility of implementing all possible inference by chaining rules and limits their practical applications in clinical medicines.

In TCM, fuzziness and uncertainty are inherent issues in the procedure of diagnosis. An attraction tool for managing various forms of uncertainty is Bayesian networks (BNs) [2], [3], [5], which is able to represent knowledge with uncertainty and efficiently performing reasoning tasks. Comparing with other methods that also can handle uncertainty quite adequately, such as fuzzy logic [4], belief functions [3] etc., BNs have several advantages. Firstly, it is based on a rigorous theory with a vast amount of known results, which prompted some researchers to claim that probability is the only sensible description of uncertainty and is adequate for all purposes. Secondly, BNs describe causal relationships in graphics mode, which is prone to comprehend and can be used to predict the consequences of intervention. Thirdly, it is often insensitive to imprecision in the numerical probabilities. In view of the advantages, we utilize BNs to construct a model for quantitative diagnosis in TCM.

In this article, we proposed a new quantitative method for syndrome differentiation based on BNs. Firstly the symptoms are selected by a novel mutual information based symptom selection algorithm (MISS). Secondly, the mapping relationships between the selected symptoms and key elements are constructed. Finally, the corresponding syndromes are output by combining the key elements. The performance of the method was evaluated. The results show that the methodology we proposed is feasible and effective and can be expected to be useful in the modernization of TCM.

2 United System of Syndrome Differentiation

This method is characterized by the concept called 'key elements for syndrome differentiation' [11]. The key elements were abstracted from some key words in routine TCM diagnostic theories and could be grouped into two categories: some were named as key element of disease location, which indicate the place where diseases occur, such as heart and kidney, and the others were named as key element of disease character, which reflect the pathological state of the body or possible causes that make disease break out, such as qi deficiency, blood stasis and so on.

In our method, the standard syndrome-name database was created, in which the names of those syndromes frequently appearing in practice were standardized by combing the key elements in accordance with certain principles and rules in expertise. The name of one syndrome always includes at least one key element of disease location and one key element of disease character.

The process of syndrome differentiation includes two phases. Firstly, the key elements are diagnosed by fusing the symptoms. Secondly, the corresponding syndromes that consist of these key elements are activated from the standard syndrome-name database.

3 Diagnosis Using Bayesian Networks

The architecture of the model for pulse diagnosis is given (see Fig. 1). The probabilistic reasoning module consists of discovering dependency relationship module, parameter learning and reasoning module.

The algorithms of automatic learning BNs from data can be grouped into two main categories: methods based on conditional independence tests and methods based on scoring metrics. The computational complexity of the algorithms based on independence tests increases exponentially with the number of variables and even may lead to unreliable results, unless huge amounts of data are available. The algorithms based on scoring metrics are characterized by the specific scoring metric and the search procedure. However, they may not find the optimal solutions, but the local one. Based on the algorithms of these two categories, the algorithms utilizing hybrid methodology are investigated, among which GBPS algorithm is one of the most effective hybrid learning algorithms. By modifying the search procedure algorithm, a new hybrid leaning algorithm [6] for dependency relationship discovery and parameter learning was used, which was developed by our research group and named as GBPS* algorithm. In [6], we have proved that GBPS* algorithm is more accurate and efficient than GBPS. Given the parameterized model, the reasoning module is implemented via Clique Tree Propagation algorithm (CTP) [7], which allows computation sharing among multiple queries and satisfies the requirement of syndrome differentiation.

In addition, we use Markov blanket [3] to perform causal inference. A Markov blanket is the minimum set of nodes that renders node X conditionally independent of all other nodes in the directed graph. The Markov blanket of a node X consists of the parents of X, the children of X, and the parents of the children of X. We can get rid of all nodes outside Markov blanket of X to obtain simplified BNs without influencing predictive accuracy rate (PAR).

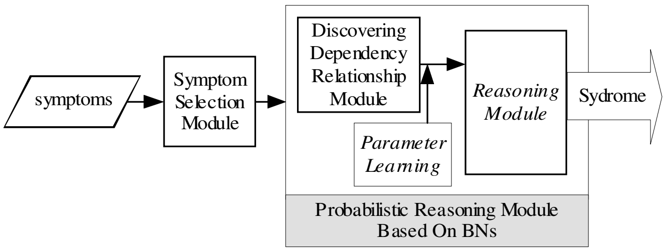

symptoms
Symptom Selection Module
Discovering Dependency Relationship Module
Parameter Learning
Reasoning Module
Sydrome
Probabilistic Reasoning Module Based On BNs

Fig. 1. The architecture of the model for quantitative syndrome differentiation

4 Symptom Selection

In TCM, there are often so many symptoms that make patient cases high dimensional. High dimension is a great obstacle for the construction of probabilistic reasoning model (PRM). For a certain type of syndrome, such as blood stasis syndrome, the numbers of symptoms that appear in clinic are more than hundreds, which makes the reasoning based on BNs intractable. In order to reduce the dimensionality and improve the prediction performance of system, variable selection is requisite. In order to achieve simplicity and scalability, many researchers use variable ranking as a baseline method [8] for variable selection, among which mutual information based feature selection (MIFS) [9] is one of the most effective algorithms. MIFS algorithm selects an informative subset to be used as input data for the model to be built on the basis of the mutual information criterion. The result of symptoms selection using MIFS algorithm are influenced directly by a parameter β , which is usually determined by testing methods. In this article, we determine β based on the expertise (prior knowledge) of TCM and propose a novel mutual information based symptom selection algorithm (we named it MISS), which is inspired by the idea of image template matching. The symptoms that are relevant to the interested syndrome are predefined by expertise and denoted as the template M_T and the symptoms selected under different β are the sequences to be matched. The procedure of MISS can be summarized five steps, as described follows:

Step 1: Assume F={initial symptom set}, $S=\{\phi\}$, the number of variables denote by V$_a$. For each $f_i \in F$, compute the mutual information $I(C;f_i)$. Choose the symptom f_i that maximize $I(C;f_i)$. Set $S \leftarrow \{f_i\}$, $F \leftarrow F \setminus \{f_i\}$

Step 2: Assume D_F and D_S are the dimensions of the set F and S. For each $\beta \in \{0,0.2,0.4,0.6,0.8,1\}$, compute $I(f_j;f_i)$, where $f_i \in S$, $f_j \in F$. Choose the symptom f_i that produces $\max\limits_{j}(I(C,f_j)-\beta / D_S \sum\limits_{i=1}^{D_S} I(f_j,f_i))$. Set $S_\beta \leftarrow S_\beta \cup \{f_i\}$, $F \leftarrow F \setminus \{f_i\}$, where $i=1,2,...,D_S$, $j=1,2,...,D_F$

Step 3: If $D_S < V_a$, turn to step 2; else output S_β as the sequence to be matched

Step 4: Number each symptoms that included in S_β and M_T with a digital code, respectively. Store these codes as characteristic gray set of S_β and M_T , denote by M_o and I_i , where $i=1,2,...,6$, respectively. Let I_i be the template characteristic sequences and M_o be the characteristic sequences to be matched. Construct the similarity measure:

$$h(I_i,M) = \sum_{a \in M_o} \min_{b \in I_i} \|a-b\| \tag{1}$$

Choose I_i , where $i=1,2,...,6$, that minimize $h(I_i,M)$ and output the corresponding set S_β containing the selected symptoms

5 Experiment Results

5.1 Sample Database

A total of 802 patient records constitute the database, which were gleaned from Xiyuan Hospital of China academy of TCM. The database consists of three parts: symptoms, key elements and syndromes. All attributes are expressed in linguistic reports. The majority of the symptoms, such as amnesia, take four values: absent, low, medium and high. Some of the attributes, for example hesitant pulse, take two values: absent and present. The rest take three values: baddish, medium and better. The diagnostic results are given by one group of clinical physicians with three members. In addition, we assign 0 to the missing values that contained in databases, so we do not consider handling missing values. The syndromes and the corresponding key elements are listed in Table 1.

Table 1. Syndromes, key elements and corresponding sample number

Syndrome	Key elements	Sample number
qi deficiency and blood stasis syndrome	qi deficiency, blood stasis	528
qi stagnation and blood stasis syndrome	qi stagnation, blood stasis	103
meridian blocking and blood stasis syndrome	meridian blocking, blood stasis	171

5.2 Evaluation of MISS Algorithm and the Syndrome Differentiation System

In the paper, a stratified k -fold cross validation technique [10] ($k = 5$, named CV-5) is used to evaluate the performance of MISS algorithm and syndrome differentiation model.

The experiments include three procedures: Firstly, use GBPS* algorithm to learn BNs, and their variants incorporating MIFS, MISS and without feature selection procedure, which are denoted by BN-M1, BN-M2 and BN-UNSELECT, respectively. Secondly, for BN-M1, BN-M2 and BN-UNSELECT, perform CTP algorithm to establish the reasoning models, the results denoted by R-M1, R-M2 and R-UNSELECT, respectively. Thirdly, validate the reasoning models using CV-5.

In the process of constructing PRM, the irrelevant attributes may produce so much interferential interdependence relationships that the practitioners and even experts cannot recognize all of them. It is a great obstacle to construct PRM based on BNs with so many attributes. As described in the section 4, we propose MISS algorithm to solve this problem. An illustrating example is given as following, in which a key element 'blood stasis' and its corresponding 112 symptoms in the database are selected. The corresponding Markov blanket for 'blood stasis' is shown as Fig. 2, from which we can see that the directed acyclic graph encodes dependency and conditional independency relationships among symptoms and between symptoms and the key element. After the procedure of symptom selection using MISS algorithm, most attributes are eliminated, such as age, gender, occupation and so on. Only 26 attributes survive. Meanwhile, some symptoms, such as tongue petechiae, hesitant pulse, cramp and so on, which have been validated to be correlative with 'blood stasis', are selected and are included in directed acyclic graph. Some attributes, such as 'vertigo' and 'amnesia', are preserved after the procedure of MISS algorithm but not included in Markov blanket of 'blood stasis', which is also accordant with the experiences of TCM. Moreover, some dependency relationships that were usually neglected by experts of TCM are also discovered. For example, 'chest pain' and 'chest distress' are also dependent on 'blood stasis' as shown in Fig. 4. It has been demonstrated by experts of TCM that most of the results are consistent with expertise, which also proves that the probabilistic reasoning models in our system are effective for discovering dependency relationship between symptoms and key elements.

In our method, the key elements whose probabilities exceed the thresholds set by experts are selected. Then the syndromes that contain these key elements are queried and output from standard syndrome-name database. Furthermore, the occurrence probability of each syndrome is computed by multiplying the probabilities of its corresponding key elements, under the assumption that these key elements are independent. Finally, the syndrome whose probability is the greatest is selected and output as the final result (see Table 3).

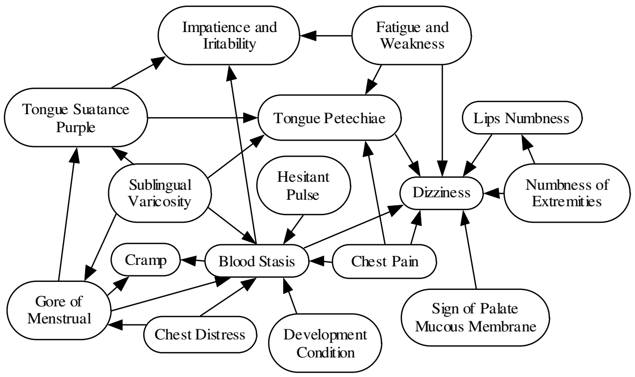


Fig. 2. The Markov blanket of the key element-blood stasis

Table 2. Predictive results of key elements

Key elements	R-UNSELECT	R-M1	R-M2
qi deficiency	0.7146 ± 0.1343	0.9033 ± 0.0279	0.9316 ± 0.0430
qi stagnation	0.6190 ± 0.0674	0.8095 ± 0.0952	0.8690 ± 0.0790
blood stasis	0.7860 ± 0.0726	0.9203 ± 0.0417	0.9594 ± 0.0272
meridian blocking	0.5786 ± 0.0474	0.8571 ± 0.1178	0.9072 ± 0.0889
Average PAR	0.6745	0.8726	0.9168

The predictive accuracies of R-M1, R-M2 and R-UNSELECT of the key elements are displayed in Table 2. It can be seen that R-M1 and R-M2 outperform R-SELECT greatly and R-M2 achieves the highest PAR, which demonstrate that MISS algorithm is efficacious for symptom selection. The average PAR of R-M2 for the four key elements is 91.68%, especially for blood stasis is 95.94%. The result also validate that the probabilistic reasoning models based on BNs are reliable and effectual for syndrome differentiation.

Table 3. The results of syndrome differentiation for three cases

	Key elements and syndromes diagnosed by expert	Key elements and their probability predicted by the system	Syndromes and their probability predicted by the system
Case 1	qi stagnation and blood stasis syndrome (qi stagnation; blood stasis)	qi deficiency : 0.37 qi stagnation: 1 meridian blocking: 0.4 blood stasis: 1	0.37 qi deficiency and blood stasis syndrome (qi deficiency; blood stasis) 0.4 meridian blocking and blood stasis syndrome (meridian blocking; blood stasis 1 qi stagnation and blood stasis syndrome (qi stagnation; blood stasis)
Case 2	qi deficiency and blood stasis syndrome (qi deficiency; blood stasis)	qi deficiency : 0.95 qi stagnation: 0.71 blood stasis: 1	0.71 qi stagnation and blood stasis syndrome (qi stagnation; blood stasis) 0.95 qi deficiency and blood stasis syndrome (qi deficiency; blood stasis)
Case 3	meridian blocking and blood stasis syndrome (meridian blocking; blood stasis)	qi deficiency : 0.33 qi stagnation: 0.78 meridian blocking: 0.96 blood stasis: 1	0.33 qi deficiency and blood stasis syndrome (qi deficiency; blood stasis) 0.78 qi stagnation and blood stasis syndrome (qi stagnation; blood stasis) 0.96 meridian blocking and blood stasis syndrome (meridian blocking; blood stasis)

6 Conclusion

TCM is one of the most important complementary and alternative medicines. But the complexity and elusiveness of diagnostic method confine its development and generalization. To facilitate the popularization of TCM, we propose a novel quantitative method based on BNs to predict syndrome automatically. The method utilizes BNs instead of using rules, which differentiates from other existing quantitative methods in TCM. Furthermore, a novel feature selection algorithm is proposed to attack the problem of large number of symptoms in the domain of TCM diagnosis. The feature that differentiates the feature selection algorithm from other methods is that it is based on prior knowledge of TCM, which can provide the diagnosis with more reasonable explanations. The experiment results validate that the methodology we proposed is effective. Note that, the feature of our model is that it can keep enhancing the predictive accuracy of syndrome by learning the experiences of experts of TCM

Acknowledgments. This study was sponsored by National Natural Science Foundation of China (No. 90209011). The authors also thank Professor Wen-Feng Zhu (Hunan College of Traditional Chinese Medicine, 410007) for his constructive suggestions and helpful discussions.

References

1. Ma, B.: Expert systems and knowledge bases in traditional Chinese medicine. Beijing, Beijing Press, In Chinese (1994)
2. Pearl J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers, Inc. San Francisco, California (1988)
3. Neapolitan R. E.: Probabilistic Reasoning in Expert Systems. John Wiley and Sons, New York, NY (1990)
4. Dubois D., Prade H., Yager R.R.: Fuzzy information engineering: A guide tour of applications. Wiley, New York (1997) 149–162
5. Cheng, J.; Hatzis, C.; Hayashi, H.; Krogel, M.-A.; Morishita, S.; Page, D.; Sese, J., KDD Cup 2001 report. SIGKDD Explor. Newsl. 2002, 3, (2), 47-64.
6. Wang X. W., Qu H. B., Liu P., Cheng Y Y.: A self-learning expert systems for diagnosis in traditional Chinese medicine. Expert Systems with Applications, Vol. 26 (2004) 557–566
7. Lauritzen S. L., Spiegelhalter D. J.: Local computations with probabilities on graphical structures and their application to expert systems. Journal of the Royal Statistical Society, Series B, Vol. 50 (1988) 157–224
8. Forman G.: An extensive empirical study of feature selection metrics for text classification. JMLR, Vol. 3 (2003) 1289–1306
9. Wong, A. K. C.; Chiu, D. K. Y., Synthesizing statistical knowledge from incomplete mixed-mode data. IEEE Trans. Pattern Anal. Mach. Intell. 1987 9 (6), 796-805
10. Mullin M. and Sukthankar R.: Complete cross-validation for nearest neighbor classifiers. Proceedings of ICML (2000): 639–646
11. Zhu W. F.: The establishment of united system of syndrome differentiation. Chinese Journal of Basic Medicine in Traditional Chinese Medicine, Vol. 4 (2001): 4–6

High-Order Markov Kernels for Network Intrusion Detection

Shengfeng Tian[1], Chuanhuan Yin[1], and Shaomin Mu[2]

School of Computer and Information Technology, Beijing Jiaotong University
Beijing, 100044, P.R. China
[1] {sftian, chhyin}@center.njtu.edu.cn, [2] msm@sdau.edu.cn

Abstract. In intrusion detection systems, sequences of system calls executed by running programs can be used as evidence to detect anomalies. Markov chain is often adopted as the model in the detection systems, in which high-order Markov chain model is well suited for the detection, but as the order of the chain increases, the number of parameters of the model increases exponentially and rapidly becomes too large to be estimated efficiently. In this paper, one-class support vector machines (SVMs) using high-order Markov kernel are adopted as the anomaly detectors. This approach solves the problem of high dimension parameter space. Experiments show that this system can produce good detection performance with low computational overhead.

1 Introduction

A lot of efforts have been put on developing intrusion detection systems (IDSs), because network intrusions are becoming top priority security issues for nowadays computer systems. There are two general approaches to detect intrusions: misuse detection and anomaly detection. Misuse detection techniques store signature patterns of known intrusions, match activities in a computer system with known intrusion patterns, and report an intrusion when finding a match. Anomaly detection techniques establish a profile to record a subject's normal activities, compare the observed subject's activities with its normal profile, and report intrusions when the observed subject's activities deviate significantly from its normal profile. Forrest and others introduced an anomaly detection method based on monitoring the system calls [1]. They scan traces of system calls in a running process and build up a database of all unique sequences of a given length k. Once a database is constructed for a given program, the database is used to monitor the ongoing behavior of the processes invoked by that program. Sequences do not occur in the normal database are considered to be anomalies.

Many other approaches have been proposed to build up the detection models from system call sequences, including data mining method [2], frequency based method [3], automation based method [4], Bayesian network [5], and Hidden Markov Model (HMM) [6].

HMM has been proved to be a good tool to model normal behaviors of privileged processes for anomaly detection using system calls. The major drawback of HMM approach is that it demands excessive computing resources in the HMM training process.

Ju et al. proposed a high-order Markov chain model for computer intrusion detection [7]. Their idea is that a simple Markov chain, where the next call depends solely on the current one, seems too crude for such an application, and so high-order model, in which the next call depends on the recent history, say of last three calls, seems more appropriate as a modeling tool. The problem is that this approach would lead to a very high dimension parameter space. Let N be the number of distinct calls and l be the index-order of the Markov chain. The parameter-space dimensionality is $N^l(N\text{-}1)$. They adopted a Mixture Transition Distribution (MTD) approach to overcome the problem.

In recent years, support vector machines (SVMs) have received considerable attention because of their superior performance in pattern recognition [8]. The major task of the SVM approach lies in the selection of its kernel. A valid kernel has to be a positive definite function, i.e., satisfying Mercer's condition. Mercer's condition tells us whether or not a kernel is actually a dot product in a given space. The well-known kernels include polynomial kernel, radial basis kernel, sigmoid kernel and so on. But these kernels are not good kernels for the recognition of sequence data, because they lack of sequence information. A kind of sequence kernel has been proposed to describe the similarities between sequence data, such as spectrum kernel [9], mismatch kernel [10], all-subsequences kernel [11] and gap weighted kernel [12]. These kernels evaluate the similarities between sequences according to the extent of sharing common subsequences.

In this paper, we discuss the SVMs using high order Markov kernel to detect anomalies in system calls, in which the similarity between sequences is evaluated according to the probabilistic properties in sequences, not the extent of sharing common subsequences. In this approach, the high order Markov chain is adopted to model normal behaviors of privileged processes for anomaly detection using system calls, but the parameter-space dimensionality is greatly reduced. First, we introduce the one-class SVMs for anomaly detection. Second, we introduce Markov chain model and Markov kernel. Finally, experimental results are presented to show the performance of the above kernel in anomaly detection.

2 Anomaly Detection with SVMs

Anomaly detection or novelty detection can be viewed as one-class classification. One strategy is to find a hyperplane in feature space such that most of the data will lie beyond that hyperplane, while at the same time the hyperplane has maximal distance to the origin [13].

Given a set of training data $x_1,x_2,\ldots,x_n \in \chi$, where $n \in R$ is the number of observations and χ is some set. Let Φ be a feature map $\chi \to F$, i.e., a map into a dot product space F such that the dot product in F can be computed by kernel $K(x,y) = \Phi(x)\cdot\Phi(y)$. To separate the data set from the origin, the task is to minimize the objective function

$$\min \tfrac{1}{2} w \cdot w + C\sum_{i=1}^{n} \xi_i - \rho \tag{1}$$

subject to $w \cdot \Phi(x_i) \geq \rho - \xi_i$, $\xi_i \geq 0$, i=1,…,n, where C is a positive constant. The dual formulation amounts to minimization of

$$W(\alpha) = \tfrac{1}{2} \sum_{i,j} \alpha_i \alpha_j K(x_i, x_j) \tag{2}$$

respect to α_i and subject to $0 \leq \alpha_i \leq C$, $i = 1,\ldots,n$, and $\sum_i \alpha_i = 1$.

After the optimal values of α_i have been found, the decision function is based on the sign of

$$f(x) = \boldsymbol{w} \cdot \Phi(x) - \rho = \textstyle\sum_i \alpha_i K(x_i, x) - \rho \tag{3}$$

where $\boldsymbol{w} = \sum_i \alpha_i \Phi(x_i)$ and ρ can be recovered by $\rho = \sum_j \alpha_j K(x_j, x_i)$ for any vector x_i with $i \in I := \{i\colon 0 < \alpha_i < C\}$. The vectors x_i corresponding to $\alpha_i > 0$ are called support vectors. In practice, many vectors x_i are not support vectors, the computation of function $f(x)$ is efficient.

In practice, the kernel value $K(x,y)$ can be interpreted as a measure of similarity between x and y. In the recognition of sequence data, the support vector anomaly detector could perform better if the kernel could reflect the sequence information.

3 Markov Chain Model and Markov Kernel

Markov chain model is a probabilistic model used in many recognition problems. In this paper, we consider the first-order Markov chain [14] and the high-order Markov chain [15] for intrusion detection.

Let X_t be a random variable taking values in the finite set $\{1,\ldots,m\}$. The first-order Markov hypothesis says that the present state at time t is only dependent on the state at time t-1. Thus we have.

$$\begin{aligned} &P(X_t = i_0 | X_{t-1} = i_1, X_{t-2} = i_2, \ldots, X_0 = i_t) \\ &= P(X_t = i_0 | X_{t-1} = i_1) = P(X_t = j | X_{t-1} = i) = p_{ij} \end{aligned} \tag{4}$$

where p_{ij} is the probability that the system is in state j at time t given the system is in state i at time t-1. Equation (4) specifies that a state transition from time t-1 to time t is independent of time. The Markov chain model can be defined by a transition probability matrix P, each of whose rows sums to 1,

$$P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ p_{m1} & p_{m2} & \cdots & p_{mm} \end{bmatrix} \tag{5}$$

and an initial probability distribution

$$Q = [\, q_1 \; q_2 \; \ldots \; q_m] \tag{6}$$

where q_i is the probability that the system is in state i at time 0, and

$$\sum_{j=1}^{m} p_{ij} = 1,\ 1 \le i \le m \tag{7}$$

To increase the model flexibility, a high-order Markov chain model is assumed, where the present state depends not only on the last single state, but on the last l observations. Thus we have

$$P(X_t = i_0 | X_{t-1} = i_1, X_{t-2} = i_2, \ldots, X_0 = i_t) = P(X_t = i_0 | X_{t-l} = i_l, \ldots, X_{t-1} = i_1) = p_{i_l \cdots i0} \tag{8}$$

For instance, the transition matrix corresponding to l=2 and m=3 is

$$P = \begin{bmatrix} p_{111} & p_{112} & p_{113} \\ p_{211} & p_{212} & p_{213} \\ p_{311} & p_{312} & p_{313} \\ p_{121} & p_{122} & p_{123} \\ p_{221} & p_{222} & p_{223} \\ p_{321} & p_{322} & p_{323} \\ p_{131} & p_{132} & p_{133} \\ p_{231} & p_{232} & p_{233} \\ p_{331} & p_{332} & p_{333} \end{bmatrix} \tag{9}$$

The number of possible combinations of l successive observations is equal to m^l. Whatever the order is, there are (m-1) independent probabilities in each row of the matrix P, since each row is a probability distribution summing to 1. The total number of independent parameters in matrix P is equal to $m^l(m\text{-}1)$. To compute these parameters, we assume

$$n_{i_l \cdots i_0} = \text{number of transitions of } X_{t\text{-}l} = i_l, \cdots, X_{t\text{-}1} = i_1, X_t = i_0 \tag{10}$$

in the observations. The maximum likelihood estimate of the corresponding transition probability is then

$$\widehat{p}_{i_l \cdots i_0} = \frac{n_{i_l \cdots i_0}}{n_{i_l \cdots i_1 +}} \tag{11}$$

where

$$n_{i_l \cdots i_1 +} = \sum_{i_0 = 1}^{m} n_{i_l \cdots i_0} \tag{12}$$

In the intrusion detection systems based on monitoring the UNIX system calls, there are about 182 system calls (i.e. m=182). The transition probability matrix of

2-order Markov chain will have 5995444 (=182^2(182-1)) parameters. Our approach to overcome the dimension problem has two key components: (a) use support vector machine to perform anomaly detection, (b) define a kernel on sequence data based on high-order Markov chain model.

Let X and Y be two random variables taking values in the finite set $\{1,\ldots,m\}$. In this study, two transition probability matrices $P=\{p_{ij}\}_{1\le i\le m^l,\ 1\le j\le m}$ and $Q=\{q_{ij}\}_{1\le i\le m^l,\ 1\le j\le m}$ of the l-order Markov chain model corresponding to X and Y are learned from training data that provide two observation sequences of system calls. The Markov kernel $K(X,Y)$ is defined as

$$K(X,Y)=\sum_{i=1}^{m^l}\sum_{j=1}^{m} p_{ij}q_{ij} \tag{13}$$

where p_{ij} and q_{ij} are computed with equations (11) and (12). Because there are few common subsequences in two system call sequences, most items $p_{ij}q_{ij}$ in above summation are 0. The kernel computation can be efficient.

Let *cstr* be the set of common subsequences in X and Y, *nx* and *ny* be arrays recording the numbers of subsequence occurrences in X and Y respectively. For an n-length sequence s, $s\_$ denotes the subsequence composed of the first n-1 elements in s. The procedure of computing the kernel $K(X,Y)$ based on l-order Markov model is as follows.

```
Procedure K(X,Y)
1. r=0,v=0,cstr={};
2. for each ($l$+1)-length subsequence sx in X, do
3.     if sx is equal to the ith member of cstr, then
4.         nx[i]++;
5.     else if (oy = number of times sx occurs in Y) > 0, then
6.         cstr = sctr ∪ sx, nx[r]=1, ny[r]=oy, r++;
7. for i=0 to r-1, do
8.     s = the ith member of cstr;
9.     a = number of times s_ occurs in X;
10.    b = number of times s_ occurs in Y;
11.    v += (nx[i]·ny[i])/(a·b);
12.K(X,Y) = v;
13.End
```

Let n be the length of sequences X and Y, the time complexity for searching subsequences in X and Y from step 2 to step 6 is $O((n\text{-}l)^2l)$. The time complexity for counting the occurrences of subsequences in X and Y from step 7 to step 11 is $O((n\text{-}l)l)$. Therefore the total time complexity of the above procedure is $O((n\text{-}l)^2l)$.

It is noticed that the Markov kernel is similar to spectrum kernel [9]. Let Σ be a finite alphabet, Σ^n the set of all finite sequences of length n from Σ. Given a number $p\ge 1$, The feature map $\Phi_p(s)$ is defined as

$$\Phi_p(s) = (\phi_u(s))_{u\in\Sigma^p}, \tag{14}$$

where $\phi_u(s)$ is the number of times u occurs in s. The p-spectrum kernel of sequence s and t is defined as

$$K_p(s,t) = \Phi_p(s) \cdot \Phi_p(t) = \sum_{u \in \Sigma^p} \phi_u(s) \cdot \phi_u(t) \tag{15}$$

The form of every item in the summation is similar to formula (11), but the denominator is removed.

Both Markov kernel and spectrum kernel express the similarity between two sequences, but they are different. The former emphasizes the probability properties of two sequences, but the later emphasizes the common subsequences in two sequences. Therefore, they suit with different applications.

4 Experiments

In the experiments, the data are taken from the UNM data sets for sendmail which are available at http://www.cs.unm.edu. The traces were obtained at UNM using Sun SPARCstations running SunOs 4.1.1 and 4.1.4. The obtained traces are broken into 100-length sequences, in which two adjacent sequences have 10 overlapped calls. We use the normal data sets “plus”, “bounce”, “bounce-1”, and “bounce-2” to build up a normal database. The test data sets consist of normal sets including “plus”, “bounce”, “bounce-1”, and “bounce-2”, and anomalous sets including “sm-280”, “sm-314”, “sm-10763”, “sm-10801” and “sm-10814”. The support vector machines are described in Section 2. The detection performance is described with detected anomaly rate, which is defined by

$$R = \frac{\text{detected number of anomalous sequences}}{\text{number of all sequences in the trace}} \tag{16}$$

The number of support vectors, SV, and detected anomaly rates for various testing traces using Markov kernel with different Markov chain orders are shown in Table 1, and those using p-spectrum kernel with different p are shown in Table 2.

The experiments show that the Markov kernel has better detection performance than the spectrum kernel, and the high-order Markov kernels have better detection performance than the first order Markov kernel. The normal database includes 970 100-length sequences, but there are only a few support vectors after training. Therefore the detection procedure is very efficient.

5 Conclusion

Markov chain model is often used in network anomaly detection. The first-order Markov chain, where the next call depends solely on the current one, seems too crude for such an application, and so high-order model, in which the next call depends on the last several calls, seems more appropriate as a modeling tool. But the approach with high-order model would lead to a very high dimension parameter space. We have presented a Markov kernel based on high-order Markov chain model and used this kernel in support vector machines to detect network intrusion. With this approach, the high efficiency in the training and detecting procedure is achieved. Further work will focus on the more efficient computation of the Markov kernel.

Table 1. The detected anomaly rates of SVMs using Markov kernel based on high-order Markov model for testing traces

Data sets	1-order (SV=30)	2-order (SV=26)	3-order (SV=32)	4-order (SV=29)	5-order (SV=29)	6-order (SV=29)
Bounce	0/8	0/8	0/8	0/8	0/8	0/8
bounce-1	0/3	0/3	0/3	0/3	0/3	1/3
bounce-2	0/8	0/8	0/8	0/8	0/8	1/8
plus	0/963	7/963	7/963	7/963	7/963	13/963
sm-280	0/16	2/16	2/16	4/16	5/16	5/16
sm-314	1/16	3/16	3/16	4/16	5/16	5/16
sm-10763	1/4	2/4	2/4	2/4	2/4	3/4
sm-10801	1/4	2/4	2/4	2/4	2/4	3/4
sm-10814	1/4	2/4	2/4	2/4	2/4	3/4

Table 2. The detected anomaly rates of SVMs using spectrum kernel for testing traces

Data sets	p=2 (SV=7)	p=3 (SV=9)	p=4 (SV=12)	p=5 (SV=16)	p=6 (SV=19)	p=7 (SV=20)
Bounce	0/8	0/8	0/8	0/8	0/8	0/8
bounce-1	0/3	0/3	0/3	1/3	0/3	0/3
bounce-2	0/8	0/8	0/8	1/8	0/8	0/8
plus	0/963	0/963	1/963	0/963	0/963	0/963
sm-280	1/16	1/16	1/16	1/16	1/16	1/16
sm-314	1/16	1/16	1/16	1/16	1/16	1/16
sm-10763	1/4	1/4	1/4	2/4	2/4	2/4
sm-10801	1/4	1/4	1/4	2/4	2/4	2/4
sm-10814	1/4	1/4	1/4	2/4	2/4	2/4

Acknowledgements

This paper is supported by the National Natural Science Foundation of China (No. 60442002) and the Science and Technology Foundation of Beijing Jiaotong University (No. 2004SM010).

References

1. Forrest, S., Hofmeyr, S.A., Somayaji, A.: Intrusion detection using sequences of system calls. Journal of Computer Security, Vol.6 (1998) 151-180
2. Lee W., Stolfo S.J., Chan P.K.: Learning patterns from UNIX process execution traces for intrusion detection. In: AAAI Workshop on AI Approaches to Fraud Detection and Risk Management. AAAI Press (1997) 50-56
3. Yeung D., Ding Y.: Host-based intrusion detection using dynamic and static behavioral models. Pattern Recognition, Vol.36 (2003) 229-243
4. Sekar R., Bendre M., Dhurjati D., Bollineni P.: A fast automation-based method for detecting anomalous program behaviors. In: Proceedings of the IEEE Symposium on Security and Privacy. (2001) 144-155
5. Feng L.,Guan X., Guo S., Gao Y.Liu P.: Predicting the intrusion intentions by observing system call sequences. Computers & Security, Vol.23 (2004) 241-252

6. Warrender S., Forrest S., Pearlmutter B.: Detecting intrusions using system calls: Alternative data models. In: Proceedings of the IEEE Symposium on Security and Privacy. (1999) 133-145
7. Ju W., Vardi Y.: A hybrid high-order Markov chain model for computer intrusion detection. Journal of Computational and Graphical Statistics. Vol.10 (2001) 277-295
8. Cristianini N., Shawe-Taylor J.: An Introduction to Support Vector machines. Cambridge, UK: Cambridge University Press (2000)
9. Leslie, C., Eskin, E. and Noble, W.S.: The spectrum kernel: a string kernel for SVM protein classification. In: Proceedings of the pacific biocomputing Symposium 7, (2002) 566-575
10. Leslie,C., Eskin,E., Weston,J. and Noble,W.S.: Mismatch string kernels for SVM protein classification. In: S. Becker, S. Thrun, K. Obermayer (eds.): Proceedings of Neural Information Processing Systems 15, Cambridge, MA: MIT Press. (2002)
11. Vishwanathan, S.V.N. and Smola, A.J.: Fast kernels for string and tree matching. In: S. Becker, S. Thrun, K. Obermayer (eds.): Proceedings of Neural Information Processing Systems 15, Cambridge, MA: MIT Press. (2002)
12. Lodhi, H., Saunders, C., Shawe-Taylor, C., Cristianini, N. and Watkins, C.: Text classification using string kernels, Journal of Machine Learning Research, 2, (2002) 419-444
13. Schölkopf, B., Platt, B. J.C., Shawe-Taylor, J., Smola, A.J.: Estimating the support of a high-dimensional distribution. Technical report MSR-TR-99-87, Microsoft Research (1999)
14. Ye N., Li X., Chen Q., Emran S.M., Xu M.: Probabilistic techniques for intrusion detection based on computer audit data. IEEE Trans. On Systems, Man, and Cybernetics – Part A: Systems and Humans, Vol.31, No.4 (2001) 266-274
15. Berchtold A., Raftery E.: The mixture transition distribution model for high-order Markov chains and non-Gaussian time series. Statistical Science, Vol.17 No.3 (2002) 328-356

Improved Realtime Intrusion Detection System*

Byung-Joo Kim[1] and Il Kon Kim[2]

[1] Youngsan University Dept. of Information and Communication Engineering
150, Junam-ri, Ungsang-eup, Yangsan-si, Kyoungnam 626-847, Korea
phone: +82-55-380-9447
bjkim@ysu.ac.kr

[2] Kyungpook National University Department of Computer Science, Korea
ikkim@knu.ac.kr

Abstract. We developed earlier version of realtime intrusion detection system using emperical kernel map combining least squares SVM(LS-SVM). I consists of two parts. One part is feature extraction by empirical kernel map and the other one is classification by LS-SVM. The main problem of earlier system is that it is not operated realtime because LS-SVM is executed in batch way. In this paper we propose an improved real time intrusion detection system incorporating earlier developed system with incremental LS-SVM. Applying the proposed system to KDD CUP 99 data, experimental results show that it has a remarkable feature feature extraction, classification performance and reducing detection time compared to earlier version of realtime ntrusion detection system.

Keywords: machine learning, IDS, feature extraction.

1 Introduction

Intrusion detection aims to detect intrusive activities while they are acting on computer network systems. Most intrusion detection systems(IDSs) are based on hand-crafted signatures that are developed by manual coding of expert knowledge. The major problem with this approach is that these IDSs fail to generalize to detect new attacks or attacks without known signatures. Recently, there has been an increased interest in data mining based approaches to building detection models for IDSs. These models generalize from both known attacks and normal behavior in order to detect unknown attacks. Several effective data mining techniques for detecting intrusions have been developed[1][2][3], many of which perform close to or better than systems engineered by domain experts. However, successful data mining techniques are themselves not enough to create effective IDSs. Despite the promise of better detection performance and generalization ability of data mining based IDSs, there are some difficulties in the implementation of the system. We can group these difficulties into three general categories: accuracy(i.e., detection performance), efficiency, and usability. In this

* This study was supported by a grant of the Korea Health 21 R&D Project, Ministry of Health & Welfare, Republic of Korea (A05-0909-A80405-05N1-00000A).

paper, we discuss the accuracy problem in developing a real-time IDS. Another issue with an IDS is that it should operate in real-time. In typical applications of data mining to intrusion detection, detection models are produced off-line because the learning algorithms must process tremendous amounts of archived audit data. An Effective IDS should work in real-time, as intrusions take place, to minimize security compromises. Feature selection therefore is an important issue in intrusion detection.

Principal Component Analysis(PCA)[4] is a powerful technique for extracting features from data sets. For reviews of the existing literature see [5][6][7]. Traditional PCA, however, has several problems. First PCA requires a batch computation step and it causes a serious problem when the data set is large. The second problem is that, in order to update the subspace of eigenvectors with another data, we have to recompute the whole eigenspace. The finial problem is that PCA only defines a linear projection of the data. It has been shown that most of the data in the real world are inherently non-symmetrical and therefore contain higher-order correlation information that could be useful[8]. For such cases, nonlinear transforms are necessary. Recently the kernel trick has been applied to PCA and is based on a formulation of PCA in terms of the dot product matrix instead of the covariance matrix[9]. Kernel PCA(KPCA), however, requires storing and finding the eigenvectors of an $N \times N$ kernel matrix where N is a number of patterns. It is an infeasible method when N is large. This fact has motivated the development of on-line way of KPCA method which does not store the kernel matrix. It is hoped that the distribution of the extracted features in the feature space has a simple distribution so that a classifier can do a proper task. But it is pointed out that features extracted by KPCA are global features for all input data and thus may not be optimal for discriminating one class from others[9]. In order to solve this problem, we developed the two-tier based realtime intrusion detection system. Proposed real time IDS is composed of two parts. The first part is used for on-line feature extraction. The second part is used for classification. Extracted features are used as input for classification. We take on-line Least Squares Support Vector Machines(LS-SVM)[10] as a classifier. This paper is composed of as follows. In Section 2 we will briefly explain the on-line feature extraction method. In Section 3 KPCA is introduced and to make KPCA on-line, empirical kernel map method is is explained. Proposed classifier combining on-line LS-SVM with the proposed feature extraction method is described in Section 4. Experimental results to evaluate the performance of the proposed system is shown in Section 5. Discussion of the proposed IDS and future work are described in Section 6.

2 On-Line Feature Extraction

In this section, we will give a brief introduction to the method of on-line PCA algorithm which overcomes the computational complexity and memory requirement of standard PCA. Before continuing, a note on notation is in order. Vectors are columns, and the size of a vector, or matrix, where it is important, is denoted with subscripts. Particular column vectors within a matrix are denoted with a superscript, while a superscript on a vector denotes a particular observation from a set of observations, so we treat observations as column vectors of a matrix. As an example, A^i_{mn} is the ith column vector in an $m \times n$ matrix. We denote a column extension to a matrix using square brackets. Thus $[A_{mn}b]$ is an $(m \times (n+1))$ matrix, with vector b appended to A_{mn} as a last column.

To explain the on-line PCA, we assume that we have already built a set of eigenvectors $U = [u_j], j = 1, \cdots, k$ after having trained the input data $\mathbf{x}_i, i = 1, \cdots, N$. The corresponding eigenvalues are Λ and $\bar{\mathbf{x}}$ is the mean of input vector. On-line building of eigenspace requires to update these eigenspace to take into account of a new input data. Here we give a brief summarization of the method which is described in [12]. First, we update the mean:

$$\overline{x}' = \frac{1}{N+1}(N\overline{x} + x_{N+1}) \tag{1}$$

We then update the set of eigenvectors to reflect the new input vector and to apply a rotational transformation to U. For doing this, it is necessary to compute the orthogonal residual vector $\hat{h} = (Ua_{N+1} + \overline{x}) - x_{N+1}$ and normalize it to obtain $h_{N+1} = \frac{h_{N+1}}{\|h_{N+1}\|_2}$ for $\| h_{N+1} \|_2 > 0$ and $h_{N+1} = 0$ otherwise. We obtain the new matrix of Eigenvectors $U^{'}$ by appending h_{N+1} to the eigenvectors U and rotating them :

$$U' = [U, h_{N+1}]R \tag{2}$$

where $R \in \mathbf{R}_{(\mathbf{k+1})\times(\mathbf{k+1})}$ is a rotation matrix. R is the solution of the eigenproblem of the following form:

$$DR = R\Lambda' \tag{3}$$

where Λ' is a diagonal matrix of new Eigenvalues. We compose $D \in \mathbf{R}_{(\mathbf{k+1})\times(\mathbf{k+1})}$ as:

$$D = \frac{N}{N+1}\begin{bmatrix} \Lambda & 0 \\ 0^T & 0 \end{bmatrix} + \frac{N}{(N+1)^2}\begin{bmatrix} aa^T & \gamma a \\ \gamma a^T & \gamma^2 \end{bmatrix} \tag{4}$$

where $\gamma = h^T_{N+1}(x_{N+1} - \bar{x})$ and $a = U^T(x_{N+1} - \bar{x})$. Though there are other ways to construct the matrix D[13][14], the only method ,however, described in [12] allows for the updating of the mean.

2.1 Eigenspace Updating Criterion

The on-line PCA represents the input data with principal components $a_{i(N)}$ and it can be approximated as follows:

$$\widehat{x}_{i(N)} = Ua_{i(N)} + \bar{x} \tag{5}$$

To update the principal components $a_{i(N)}$ for a new input x_{N+1} , computing an auxiliary vector η is necessary. η is calculated as follows:

$$\eta = \left[U\widehat{h}_{N+1}\right]^T (\overline{x} - \overline{x}') \tag{6}$$

then the computation of all principal components is

$$a_{i(N+1)} = (R')^T \begin{bmatrix} a_{i(N)} \\ 0 \end{bmatrix} + \eta, \qquad i = 1, \cdots, N+1 \tag{7}$$

The above transformation produces a representation with $k+1$ dimensions. Due to the increase of the dimensionality by one, however, more storage is required to represent the data. If we try to keep a k-dimensional eigenspace, we lose a certain amount of information. It is needed for us to set the criterion on retaining the number of eigenvectors. There is no explicit guideline for retaining a number of eigenvectors. In this paper we set our criterion on adding an Eigenvector as $\lambda'_{k+1} > 0.7\bar{\lambda}$ where $\bar{\lambda}$ is a mean of the λ. Based on this rule, we decide whether adding u'_{k+1} or not.

3 On-Line KPCA

A prerequisite of the on-line eigenspace update method is that it has to be applied on the data set. Furthermore it is restricted to apply the linear data. But in the case of KPCA this data set $\Phi(x^N)$ is high dimensional and can most of the time not even be calculated explicitly. For the case of nonlinear data set, applying feature mapping function method to on-line PCA may be one of the solutions. This is performed by so-called *kernel-trick*, which means an implicit embedding to an infinite dimensional Hilbert space[11](i.e. feature space) F.

$$K(x, y) = \Phi(x) \cdot \Phi(y) \tag{8}$$

Where K is a given kernel function in an input space. When K is semi positive definite, the existence of Φ is proven[11]. Most of the case ,however, the mapping Φ is high-dimensional and cannot be obtained explicitly. The vector in the feature space is not observable and only the inner product between vectors can be observed via a kernel function. However, for a given data set, it is possible to approximate Φ by empirical kernel map proposed by Scholkopf[15] and Tsuda[16] which is defined as $\Psi_N : \mathbf{R}^d \rightarrow \mathbf{R}^N$

$$\begin{aligned} \Psi_N(x) &= [\Phi(x_1) \cdot \Phi(x), \cdots, \Phi(x_N) \cdot \Phi(x)]^T \\ &= [K(x_1, x), \cdots, K(x_N, x)]^T \end{aligned} \tag{9}$$

A performance evaluation of empirical kernel map was shown by Tsuda. He shows that support vector machine with an empirical kernel map is identical with the conventional kernel map[17].

4 Proposed System

In previous Section 3 we proposed an on-line KPCA method for nonlinear feature extraction. It is hoped that the distribution of the mapped data in the feature space has a simple distribution so that a classifier can classify them properly.

But it is point out that extracted features by KPCA are global features for all input data and thus may not be optimal for discriminating one class from others. For classification purpose, after global features are extracted using they must be used as input data for classification.

Recently LS-SVM method developed by Suykens is computationally attractive and easier to extend than SVM. But the existed LS-SVM algorithm is trained off-line in batch way. Off-line training algorithm is not fit for the realtime IDS. In this paper we take on-line LS-SVM algorithm because proposed realtime IDS to be more realistic. Proposed real time IDS is composed of two parts. First part is used for on-line feature extraction. To extract on-line nonlinear features, we propose a new feature extraction method which overcomes the problem of memory requirement of KPCA by on-line eigenspace update method incorporating with an adaptation of kernel function. Second part is used for classification. Extracted features are used as input for classification. We take on-line Least Squares Support Vector Machines(LS-SVM)[19] as a classifier.

5 Experiment

To evaluate the classification performance of proposed realtime IDS system, we use KDD CUP 99 data[18]. The following sections present the results of experiments.

5.1 Description of Dataset

The raw training data(kddcup.data.gz) was about four gigabytes of compressed binary TCP dump data from seven weeks of network traffic. This was processed into about five million connection records. Similarly, the two weeks of test data yielded around two million connection records. Each connection is labeled as either normal, or as an attack, with exactly one specific attack type. Each connection record consists of about 100 bytes. Attacks fall into four main categories(DOS, R2L, U2R, Probing).

It is important to note that the test data(corrected.gz) is not from the same probability distribution as the training data, and it includes specific attack types not in the training data. This makes the task more realistic. The datasets contain a total of 24 training attack types, with an additional 14 types in the test data only.

5.2 Experimental Condition

To evaluate the classification performance of proposed system, we randomly split the the training data as 80% and remaining as validation data. To evaluate the classification accuracy of proposed system we compare the proposed system to SVM. Because standard LS-SVM and SVM are only capable of binary classification, we take multi-class LS-SVM and SVM. A RBF kernel has been taken and optimal hyper-parameter of multi-class SVM and LS-SVM[20] was obtained by 10-fold cross-validation procedure. In [19] it is shown that the use of 10-fold cross-validation for hyper-parameter selection of SVM and LS-SVMs consistently leads to very good results.

In experiment we will evaluate the generalization ability of proposed IDS on test data set since there are 14 additional attack types in the test data which are not included int the training set. To do this, extracted features by on-line KPCA will be used as input for multi-class on-line LS-SVM. Our results are summarized in the following sections.

5.3 Evaluate Feature Extraction and Classification Performance

Table 1 gives the result of extracted features for each class by on-line KPCA method.

Table 1. Extracted features on each class by on-line KPCA

Class	Extracted features
Normal	1,2,3,5,6,7,8,9,10,11,12,14,16,17,18,20,21,23,25,27,29,31,32,34,38,39,41
Probe	3,5,6,24,32,38
DOS	1,3,8,19,23,28,33,35,36,39,41
U2R	5,6,15,18,25,32,33,39
R2L	3,5,6,32,33,34,35

Table 2 shows the results of the classification performance by standard SVM using all features. Table 3 shows the results of the classification performance and computing time for training and testing data by proposed system using extracted features. We can see that using important features for classification gives similar accuracies compared to using all features and the training, testing time is proper enough for realtime IDS. Comparing Table 2 with Table 3, we obtain following results. The performance of using the extracted features do not show the significant differences to that of using all features. This means that proposed on-line feature extraction method has good performance in extracting features. Proposed method has another merit in memory requirement. The advantage of proposed feature extraction method is more efficient in terms of memory requirement than a batch KPCA because proposed feature extraction method do not require the whole N $\times$ N kernel matrix where N is the number of the training data. Second one is that proposed on-line feature extraction method has similar performance is comparable in performance to a batch KPCA.

5.4 Suitable for Realtime IDS

Table 3 shows that proposed system operates in a very quick manner whereas traditional batch system requires tremendous computational time when new training data is added. Furthermore classification accuracy of proposed system is similar to using all features. This makes proposed IDS suitable for realtime IDS.

Table 2. Classification Performance by SVM using all features

Class	Accuracy(%)
Normal	98.55
Probe	98.59
DOS	98.10
U2R	98.64
R2L	98.69

Table 3. Performance of proposed system using extracted features

Class	Accuracy(%)	Training Time(Sec)	Testing Time(Sec)
Normal	98.54	3.12	0.9
Probe	98.64	20.25	1.14
DOS	98.48	10.79	1.10
U2R	98.91	1.2	0.84
R2L	98.74	5.7	0.6

5.5 Comparison with Batch Way LS-SVM

Recently LS-SVM is a powerful methodology for solving problems in nonlinear classification problem. To evaluate the classification accuracy of proposed system it is desirable to compare with batch way LS-SVM.

Table 4. Performance comparison of proposed method and batch way LS-SVM. Using all features.

	Normal	Probe	DOS	U2R	R2L
batch LS-SVM	98.76	98.81	98.56	98.92	98.86
proposed system	98.67	98.84	98.48	98.86	98.82

Generally the disadvantage of incremental method is their accuracy compared to batch method even though it has the advantage of memory efficiency and computation time. According to Table 4 we can see that proposed method has better classification performance compared to batch way LS-SVM. By this result we can show that proposed realtime IDS has remarkable classification accuracy though it is worked by incremental way.

6 Conclusion and Remarks

This paper was devoted to the exposition of a new technique on realtime IDSs . Proposed on-line KPCA has following advantages. Firstly, The performance of using the extracted features do not show the significant differences to that of using all features. This means that proposed on-line feature extraction method has good performance in extracting features. Secondly, proposed method has merit in memory requirement. The advantage of proposed feature extraction method is more efficient in terms of memory requirement than a batch KPCA because proposed feature extraction method do not require the whole N $\times$ N kernel matrix where N is the number of the training data. Thirdly, proposed on-line feature extraction method has similar performance is comparable in performance to a batch KPCA though it works incrementally.

Our ongoing experiment is that applying proposed system to more realistic world data to evaluate the realtime detection performance.

References

1. Eskin, E. :Anomaly detection over noisy data using learned probability distribution. In Proceedings of the Seventeenth International Conference on Machine Learning (2000) 443-482
2. Ghosh, A. and Schwartzbard, A. :A Study in using neural networks for anomaly and misuse detection. In Proceedings of the Eighth USENIX Security Symposium, (1999) 443-482
3. Lee, W. Stolfo, S.J. and Mok, K.:A Data mining in workflow environments. :Experience in intrusion detection. In Proceedings of the 1999 Conference on Knowledge Discovery and Data Mining, (1999)
4. Tipping, M.E. and Bishop, C.M. :Mixtures of probabilistic principal component analysers. Neural Computation 11(2) (1998) 443-482
5. Kramer, M.A.:Nonlinear principal component analysis using autoassociative neural networks. AICHE Journal 37(2) (1991) 233-243
6. Diamantaras, K.I. and Kung, S.Y.:Principal Component Neural Networks: Theory and Applications. New York John Wiley & Sons, Inc (1996)
7. Kim, Byung Joo. Shim, Joo Yong. Hwang, Chang Ha. Kim, Il Kon.: On-line Feature Extraction Based on Emperical Feature Map. Foundations of Intelligent Systems, volume 2871 of Lecture Notes in Artificial Intelligence (2003) 440-444
8. Softky, W.S and Kammen, D.M.: Correlation in high dimensional or asymmetric data set: Hebbian neuronal processing. Neural Networks vol. 4, Nov. (1991) 337-348
9. Gupta, H., Agrawal, A.K., Pruthi, T., Shekhar, C., and Chellappa., R.:An Experimental Evaluation of Linear and Kernel-Based Methods for Face Recognition," accessible at http://citeseer.nj.nec.com.
10. Liu, J. Chen, J.P Jiang, S. and Cheng, J.:Online LS-SVM for function estimation and classification Journal of University of Science and Technology Beijing, vol.10, Num.5, Oct. (2003) 73-77
11. Vapnik, V. N.:Statistical learning theory. John Wiley & Sons, New York (1998)
12. Hall, P. Marshall, D. and Martin, R.: On-line eigenalysis for classification. In British Machine Vision Conference, volume 1, September (1998) 286-295
13. Winkeler, J. Manjunath, B.S. and Chandrasekaran, S.:Subset selection for active object recognition. In CVPR, volume 2, IEEE Computer Society Press, June (1999) 511-516
14. Murakami, H. Kumar.,B.V.K.V.:Efficient calculation of primary images from a set of images. IEEE PAMI, 4(5) (1982) 511-515
15. Scholkopf, B. Smola, A. and Muller, K.R.:Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation 10(5), (1998) 1299-1319
16. Tsuda, K.:Support vector classifier based on asymmetric kernel function. Proc. ESANN (1999)

17. Mika, S.:Kernel algorithms for nonlinear signal processing in feature spaces. Master's thesis, Technical University of Berlin, November (1998)
18. Accessable at http://kdd.ics.uci.edu/databases/kddcup99
19. Gestel, V. Suykens, T. J.A.K. Lanckriet, G. Lambrechts, De Moor, A. B. and Vandewalle, J.:A Bayesian Framework for Least Squares Support Vector Machine Classifiers. Internal Report 00-65, ESAT-SISTA, K.U. Leuven.
20. Suykens, J.A.K. and Vandewalle, J.:Multiclass Least Squares Support Vector Machines. In Proc. International Joint Conference on Neural Networks (IJCNN'99), Washington DC (1999)

A Distributed Neural Network Learning Algorithm for Network Intrusion Detection System

Yanheng Liu, Daxin Tian, Xuegang Yu, and Jian Wang

College of Computer Science and Technology, Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, 130012, China
lyh_lb_lk@yahoo.com.cn

Abstract. To make network intrusion detection systems can be used in Gigabit Ethernet, a distributed neural network learning algorithm (DNNL) is put forward to keep up with the increasing network throughput. The main idea of DNNL is splitting the overall traffic into subsets and several sensors learn them in parallel way. The advantage of this method is that the large data set can be split randomly thus reduce the complicacy of the splitting algorithm. The experiments are performed on the KDD'99 Data Set which is a standard intrusion detection benchmark. Comparisons with other approaches on the same benchmark show that DNNL can perform detection with high detection rate.

Keywords: Intrusion detection system; Neural network; Distributed learning.

1 Introduction

The main shortcoming of IDS is false alarm which is caused by misinterpreting normal packets as an attack or misclassifying an intrusion as normal behavior. This problem is more severe under fast Ethernet and results in that network IDS (NIDS) can't be adapted to protect Backbone network. Since network traffic bandwidth is increasing at an exponential rate, it's impossible to keep up with the speed of networks by just increase the speed of processors.

To resolve the problem and make NIDS can be used in Gigabit Ethernet, the ideal policy is using distributed and parallel detecting method. The main idea of distributed NIDS is splitting the traffic data and forwarding them to detection sensors, thus these sensors can analyze the data in parallel way. Paper [1] presents an approach which allows for meaningful slicing of the network traffic into portions of manageable size. However, their approach uses a simple Round-Robin algorithm for load balancing. The splitting algorithm of [2] ensures that a single slice contains all the evidence necessary to detect a specific attack, making sensor-to-sensor interaction unnecessary. Although the algorithm can dynamically balance the sensors' loads by choosing the sensor with the lightest load to process the new connection's packets, it still may lead to some sensor lose packet if the traffic of one connection is heavy. Paper [3] designs a flow-based dynamic load-balancing algorithm, which divides the data stream based on the current value of each analyzer's load function. The incoming data packets, which belonged to a new session, are forwarded to the analyzer that has least load currently. Paper [4] presents an active splitter architecture and three methods for improving performance. The first is early filtering/forwarding, where a fraction of the packets is processed on the splitter instead of the sensors. The second is the use of locality buffering, where the splitter reorders packets in a way that improves memory access locality on the sensors. The third is the use of cumulative acknowledgments, a method that optimizes the coordination between the traffic splitter and the sensors. The load balancer of SPANIDS[5] employs multiple levels of hashing and incorporates feedback from the sensor nodes to distribute network traffic over the sensors without overloading any of them. Although the methods of [3], [4] and [5] reduce the load on the sensors, it complicates the splitting algorithm and makes the splitter become the bottleneck of the system. This distributed learning algorithm can also be used in mobile agent [6], distributed data mining [7], distributed monitoring [8] and ensemble system [9].

In this paper a distributed neural network learning algorithm (DNNL) is presented which can be used in distributed anomaly detection system. The idea of DNNL is different from the common distributed intrusion detection system. The rest of this paper is organized as follows. Section 2 describes the main idea of DNNL and details the basic learning algorithm. The experimental results on datasets KDDCUP99 are given in Section 3, and conclusions are made in Section 4.

2 DNNL

The main idea of DNNL is: first, split the large sample data into small subnets and forward these slice to distributed sensors; then each sensor's neural network begin to be trained by the data in parallel until all of them are stable; last, a central learning is carried on the learning result of each slice data.

2.1 Competitive Learning Algorithm Based on Kernel Function

To prevent the knowledge included in different data slice is ignored, DNNL adopts the resonances mechanism of ART and adds neurons whenever the network in its current state does not sufficiently match the input. Thus the learning results of the sensors contain the integrity or part knowledge and the whole knowledge can be learned by the last training.

2.1.1 Hebb Learning

In DNNL the learning algorithm is based on the Hebbian Postulate "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased."

The learning rule for a single neuron can be derived from an energy function defined as

$$E(\mathbf{w}) = -\psi\left(\mathbf{w}^T \mathbf{x}\right) + \frac{\alpha}{2}\|\mathbf{w}\|_2^2 \tag{1}$$

where $\mathbf{w}$ is the synaptic weight vector (including a bias or threshold), $\mathbf{x}$ is the input to the neuron, $\psi(\ )$ is a differentiable function, and $\alpha \geq 0$ is the forgetting factor. Also,

$$y = \frac{\mathrm{d}\psi(v)}{dv} = f(v) \tag{2}$$

is the output of the neuron, where $v = \mathbf{w}^T\mathbf{x}$ is the activity level of the neuron. Taking the steepest descent approach to derive the continuous-time learning rule

$$\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = -\mu\nabla_{\mathbf{w}}E(\mathbf{w}) \tag{3}$$

where $\mu > 0$ is the learning rate parameter, we see that the gradient of the energy function in (1) must be computed with respect to the synaptic weight vector, that is, $\nabla_{\mathbf{w}}E(\mathbf{w}) = \partial E(\mathbf{w})/\partial\mathbf{w}$. The gradient of (1) is

$$\nabla_{\mathbf{w}}E(\mathbf{w}) = -f(v)\frac{\partial v}{\partial \mathbf{w}} + \alpha\mathbf{w} = -y\mathbf{x} + \alpha\mathbf{w} \tag{4}$$

Therefore, by using the result in (4) along with (3), the continuous-time learning rule for a single neuron is

$$\frac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = \mu[y\mathbf{x} - \alpha\mathbf{w}] \tag{5}$$

the discrete-time learning rule (in vector form) is

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \mu[y(t+1)\mathbf{x}(t+1) - \alpha\mathbf{w}(t)] \tag{6}$$

2.1.2 Competitive Mechanism Based on Kernel Function

To overcome the problem induced by traffic splitter, the inverse distance kernel function is used in Hebb learning. The basic idea is that not only the winner is rewarded but also all the losers are penalized in different rate which is calculated by the inverse distance function and its input is the dissimilarity between the sample data and neuron.

The dissimilarity measure function is Minkowski metric:

$$d_p(\mathbf{x},\mathbf{y}) = \left(\sum_{i=1}^{l} w_i |x_i - y_i|^p\right)^{1/p} \tag{7}$$

where x_i, y_i are the i th coordinates of $\mathbf{x}$ and $\mathbf{y}$, $i = 1,\cdots,l$, and $w_i \geq 0$ is the i th weight coefficient.

When the neuron is active ($y = 1$), then the learning rule of ith neuron is

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + \mu * \gamma_i[\mathbf{x}(t+1) - \mathbf{w}_i(t)] \tag{8}$$

where,

$$\gamma_i = \begin{cases} 1, & \text{winner}, i = j \\ -K(d_i), & \text{others}, i = 1, \cdots, m \text{ and } i \neq j \end{cases} \tag{9}$$

and $K(d_i)$ is the inverse distance kernel,

$$K(d_i) = \frac{1}{1 + d_i^p} \tag{10}$$

If the winner's dissimilarity measure $d < \vartheta$ (ϑ is the threshold of dissimilarity), then update the synaptic weight by learning rule (8), else add a new neuron and set the synaptic weight $\mathbf{w} = \mathbf{x}$.

2.2 Post-prune Algorithm

Neural network may also bring the overfitting problem. In decision tree learning, it uses post-prune method to prevent overfitting. DNNL also performs the post-prune whose strategy based on the distance threshold to overcome this problem. If two weights are too similar they will be substituted by a new weight. The new weight is calculated as

$$\mathbf{W}_{new} = (\mathbf{W}_{old1} \times t_1 + \mathbf{W}_{old2} \times t_2)/(t_1 + t_2) \tag{11}$$

where t_1 is the training times of $\mathbf{W}_{old1}$, t_2 is the training times of $\mathbf{W}_{old2}$.
The prune algorithm is shown below:

```
Step0: If old weights muster (oldW) is null then algo-
       rithm is over, else go on;
Step1: calculate the distance between the first weight
       (fw) and the other weights;
Step2: find the weight (sw) who is the most similar to
       fw;
Step3: if the distance between sw and fw is bigger than
       prune threshold then delete fw from oldW and add
       fw into new weights muster (newW) go to step 0;
       else go on;
Step4: get fw's training times value (ft) and sw's
       training times value (st);
Step5: calculate the new weight (nw) and nw's training
       times value (nt), nw=(fw * ft + sw *
       st)/(ft+st), nt=ft + st;
Step6: delete fw and sw from oldW, and add nw into
       newW, go to step 0.
```

2.3 Learning Algorithm of DNNL

The main learning process of DNNL is:

```
Step0: Initialize learning rate parameter $\mu$ , the
       threshold of dissimilarity $\vartheta$;
Step1: Get the first input $\mathbf{x}$ and set $\mathbf{w}_0 = \mathbf{x}$ as the ini-
       tial weight;
Step2: If training is not over, randomly take a feature
       vector $\mathbf{x}$ from the feature sample set $\mathbf{X}$ and
       compute the dissimilarity measure between $\mathbf{x}$ and
       each synaptic weight use (7);
Step3: Decide the winner neuron $j$ and tests tolerance:
       If ($d_j >= \vartheta$) add a new neuron and sets synaptic
       weight $\mathbf{w} = \mathbf{x}$, goto Step2; else continue;
Step4: Compute $\gamma_i$ by using the result of inverse dis-
       tance $K(d_i)$;
Step5: Update the synaptic weight as (8) , goto Setp2.
```

After divided the large data set into small slices, each sensor trains its neural network follow the above algorithm, then building new data set randomly under the threshold of dissimilarity ϑ (the amount of new data set is less than the original data set) and learn knowledge from them, last using post-prune algorithm to prune the similar weights.

3 Experiments

In KDDCUP 99 data set, a smaller data set consisting of the 10% the overall data set is generally used to evaluate algorithm performance. The smaller data set contains of 22 kinds of intrusion behaviors and 494,019 records among which 97,276 are normal connection records. The test set is another data set which contains 37 kinds of intrusion behaviors and 311,029 records among which 60,593 are normal records.

3.1 Performance Measures

The recording format of test results is shown in Table 1. False alarm is partitioned into False Positive (FP, normal is detected as intrusion) and False Negative (FN, intrusion is not detected). True detect is also partitioned into True False (TF, intrusion is detected rightly) and True Negative (TN, normal is detected rightly).

Definition 1. The right detection rate of ith behavior $TR = T_{ii} \Big/ \sum_{j=1}^{n} R_{ij}$, where T_{ii} is the value lies in table1's ith row and ith column.

Table 1. Recording format of test result

		Detection Results				
		Normal	**Intrusion-1**	**Intrusion-2**	**---**	**Intrusion-n**
Actual Behaviors	**Normal**	TN00	FP01	FP02	---	FP0n
	Intrusion-1	FN10	TP11	FP12	---	FP1n
	Intrusion-2	FN20	FP21	TP22	---	FP2n
	⋮	⋮	⋮	⋮	⋮	⋮
	Intrusion-n	FNn0	FPn1	FPn2	---	TPnn

Definition 2. The right prediction rate of ith behavior $PR = T_{ii} \Big/ \sum_{j=1}^{n} R_{ji}$, where T_{ii} is the value lies in table1's ith row and ith column; R_{ji} is the value lies in table1's jth row and ith column.

Definition 3. Detection rate (DR) is computed as the ratio between the number of correctly detected intrusions and the total number of intrusions. If regard Table's record as an $(n+1)\times(n+1)$ metric $\mathbf{R}$, then $DR = \sum_{i=1}^{n}\sum_{j=1}^{n} R_{ij} \Big/ \sum_{i=0}^{n}\sum_{j=0}^{n} R_{ij}$.

Definition 4. False positive rate (FPR) is computed as the ratio between the number of normal behaviors that are incorrectly classifies as intrusions and the total number of normal connections, according to the Table's record

$$FPR = \sum_{i=1}^{n} FP0i \Big/ \sum_{i=1}^{n} FP0i + TN00 .$$

3.2 Experiment Results

To test the performance of DNNL, we first divided the 494,019 records into 50 slices, each slice contains 10,000 records except the last contains 4,019 records. After the training on the distributed learning result, the knowledge is represented by 368 neurons.There are 37 kinds of intrusion behaviors in the test set. We first separate them into four kinds of attacks:

Probe: {portsweep, mscan, saint, satan, ipsweep, nmap}
DOS: {udpstorm, smurf, pod, land, processtable, warezmaster, apache2, mailbomb, Neptune, back, teardrop}
U2R: { httptunnel, ftp_write, sqlattack, xterm, multihop, buffer_overflow, perl, loadmodule, rootkit, ps}
R2L: { guess_passwd, phf, snmpguess, named, imap, snmpgetattack, xlock, sendmail, xsnoop, worm}

The test results are summarized in Table 2.

Table 2. Testing Results

		Detection Results					
		Normal	**Probe**	**DOS**	**U2R**	**R2L**	**TR**
Actual Behav-iors	**Normal**	58120	927	649	64	833	96.0%
	Probe	357	3546	174	21	118	85.1%
	DOS	256	5092	223518	52	435	97.2%
	U2R	143	39	0	23	23	10.1%
	R2L	14443	14	1	271	1460	9%
	PR	79.3%	36.9%	99.6%	5.3%	50.9%	

Comparing the result with the first winner of KDD CUP 99 we can find the TR of DNNL is almost equal to the first winner. There are two reasons lead to the low TR of U2R and R2L: first, the size of attack instance that pertained to U2R and R2L is much smaller than that of other types of attack; second, U2R and R2L are host-based attacks which exploit vulnerabilities of the operating systems, not of the network protocol. Therefore, these are very similar to the "normal" data. Table 3 shows the DR and FPR of the first and second winner of the KDD CUP 99 competition, other approaches[19] and DNNL. From the comparison we can find that DNNL provides superior performance.

Table 3. Comparison with other approaches

Performances / Algorithms	Detection Rate (DR)	False Positive Rate (FPR)
Winning Entry	91.9%	0.5%
Second Place	91.5%	0.6%
Best Linear GP - FP Rate	89.4%	0.7%
Best GEdIDS - FP Rate	91%	0.4%
DNNL	93.9%	0.4%

4 Conclusion

The bandwidth of networks increases faster than the speed of processors. It's impossible to keep up with the speed of networks by just increase the processor's speed of NIDSs. To resolve the problem, this paper presents a distributed neural network learning algorithm (DNNL) which can be used in the anomaly detection methods. The main contribution of this approach is: reducing the complexity of load balancing while still maintains the integrity of the network behavior, putting forward a dissimilarity measure method for categorical and numerical features and increasing the speed of the whole systems. In the experiments, the KDD data set is used which is the common data set used in IDS research papers. Experiment results demonstrate the performance potential of this approach. Future work includes the design and investigation of techniques to make DNNL can be easily used in large scale data (labeled or unlabeled) analysis.

References

1. Kruegel C., Valeur F., Vigna G., Kemmerer R. Stateful intrusion detection for high-speed networks. In Proceedings of the IEEE Symposium on Security and Privacy, May, 2002, pp.285-294
2. Lai H.G., Cai S.W., Huang H., Xie J.Y., Li H. A parallel intrusion detection system for high-speed networks. Applied Cryptography and Network Security: Second International Conference, ACNS 2004, Yellow Mountain, China, June 8-11, 2004, pp.439-451
3. Jiang W.B., Song H., Dai Y.Q. Real-time intrusion detection for high-speed networks. Computers & Security, Volume 24, Issue 4, June 2005, pp.287-294
4. Xinidis K., Charitakis I., Antonatos S., Anagnostakis K.G., Markatos, E.P. An active splitter architecture for intrusion detection and prevention. Dependable and Secure Computing, IEEE Transactions on Volume 3, Issue 1, Jan.-March, 2006, pp.31-44
5. Schaelicke L., Wheeler K., Freeland C. SPANIDS: a scalable network intrusion detection loadbalancer. In Proceedings of the 2nd conference on Computing frontiers, Ischia, Italy, May4-6, 2005, pp.315-322
6. Wang Y.X., Behera S.R., Wong J., Helmer G., Honavar V., Miller L., Lutz R., Slagell M. Towards the automatic generation of mobile agents for distributed intrusion detection system. The Journal of Systems and Software, Volume 79, Issue 1, 2006, pp.1–14
7. Bala J., Weng Yilin., Williams A., Gogia B.K., Lesser H.K. Applications of Distributed Mining Techniques For Knowledge Discovery in Dispersed Sensory Data. 7th Joint Conference on Information Sciences - JCIS 2003, Cary, NC, USA, Sep 26-30, 2003,
8. Kourai K., Chiba S. HyperSpector virtual distributed monitoring environments for secure intrusion detection. Proceedings of the 1st ACM/USENIX international conference on Virtual execution environments, Chicago, IL, USA, Jun. 2005, pp.197-207
9. Folino G., Pizzuti C., Spezzano G. GP ensemble for distributed intrusion detection systems. In Proceedings of the 3rd International Conference on Advanced in Pattern Recognition, Bath, UK, 22-25 August, 2005, pp.54-62

A DGC-Based Data Classification Method Used for Abnormal Network Intrusion Detection

Bo Yang[1,2], Lizhi Peng[2], Yuehui Chen[2], Hanxing Liu[1], and Runzhang Yuan[1]

[1] State Key Lab. of Advanced Technology for Materials Synthesis and Processing, Wuhan University of Science and Technology, China
[2] School of Information Science and Technology, Jinan University, Jinan, 250022, China
yangbo@ujn.edu.cn

Abstract. The data mining techniques used for extracting patterns that represent abnormal network behavior for intrusion detection is an important research area in network security. This paper introduces the concept of gravitation and gravitation field into data classification by utilizing analogical inference, and studied the method to calculate data gravitation. Based on the theoretical model of data gravitation and data gravitation field, the paper presented a new classification model called Data Gravitation based Classifier (DGC). The proposed approach was applied to an Intrusion Detection System (IDS) with 41 inputs (features). Experimental results show that the proposed method was efficient in data classification and suitable for abnormal detection using netowrk processor-based platforms.

Keywords: Data Classification, Network Intrusion Detection, Data Gravitation based Classifier, Network Processor.

1 Introduction

Intrusion detection system (IDS) is an important component of today's network security framework. Its main idea is to differentiate between normal activities of the network system and behavior that can be classified as suspicious or intrusive. IDS approaches can be divided into two main categories: misuse or anomaly detection [1]. Anomaly detection systems assume that an intrusion should deviate the system behavior from its normal pattern. This approach can be implemented using variety of data classification approaches such as statistical methods, neural networks, predictive pattern generation and association rules.

The classification model in an IDS is usually constructed according to a given training set. Once the model has been built, it can map a test data to a certain class in the given class set. Many classification techniques including decision tree [2,3], neural network (NN) [4], support vector machine (SVM) [5, 6], etc. have been proposed. Among these techniques, decision tree is simple and easy to be comprehended by human beings. It can get high classification efficiency, but its classification accuracy is usually lower than neural network. Neural network has been proved to be an approach that can get high accuracy in many classification tasks, however, its training efficiency is usually a problem. SVM is a new machine learning method developed on the Statistical Learning Theory, which is gaining popularity due to many attractive features, and promising empirical performance. But SVM is based on the hypothesis that the training samples obey a certain distribution, this restricts its application scope. Rough set [7] is also been applied to data classification in recent years, it was used for feature selection [8] or hybridize with other classification methods [9,10,11,12].

Y. Shi et al. presented a novel data preprocessing technique called *shrinking* [13]. This technique optimizes the inner structure of data inspired by the Newton's Universal Law of Gravitation. In [14], a dimension deduction approach for multi-dimensional data analysis was also presented according to shrinking technique. A spatial clustering algorithm called *GRAVIclust* was proposed in [15]. This algorithm uses a heuristic to pick the initial cluster centers and utilizes centre of cluster gravity calculations in order to arrive at the optimal clustering solution. Although both of the two former approaches are focus on the clustering problem, they both had been inspired by the concepts of physical gravitation. The natural principles of gravity was further applied to another research area in [16].

This paper introduces the concept of gravitation and gravitation field into data classification by utilizing analogical inference, and studied the method to calculate data gravitation. Based on the theoretical model of data gravitation and data gravitation field, the paper presented a new classification model called Data Gravitation based Classifier (DGC). The proposed approach was applied to the intrusion detection problem with 41 features. Experimental results show that the proposed method is efficient in data classification for network intrusion systems.

2 Data Gravitation and Data Gravitation Field [17]

Definition 1. **(Data Particle):** Data particle is defined as a kind of data unit that has "Data Mass". Data particle is made up of a group of data elements in data space that have a certain relationship between them. The "mass" of a data particle is the number of data elements that the data particle contains. An data particle composed by only one data element is called "atomic data article". The data mass of an atomic data particle is 1.

Definition 2. **(Data Centroid):** Suppose $x_1, x_2, \ldots, x_m$ ($x_i =< x_{i1}, x_{i2}, \ldots, x_{in} >, i = 1, 2, \ldots, m$) are a group of data elements in n-dimensional data space S, P is a data article built up by $x_1, x_2, \ldots, x_m$. Therefore, the data centroid of P, $x_i =< x_{01}, x_{02}, \ldots, x_{0n} >$ is the geometrical center of $x_1, x_2, \ldots, x_m$. It can be described with following formula:

$$x_{0j} = \frac{\sum_{i=1}^{m} x_{ij}}{m}, \quad i=0, 1, \ldots, m; \quad j=0, 1, \ldots n \tag{1}$$

Since data particle has data mass and data centroid, so data particle can be described by a pair expression $< m, \boldsymbol{x} >$, after the class information (feature y) has been added, it can be described as a triple expression $< m, \boldsymbol{x}, y >$ where m is the data mass of the data particle and $\boldsymbol{x}$ is the data centroid.

Definition 3. **(The Law of Data Gravitation):** Data gravitation is defined as the similarity between data. It is a kind of scalar quality without direction. The data gravitation can be described as:

$$F = \frac{m_1 m_2}{r^2} \tag{2}$$

Where F is Gravitation between two data particles; m_1 is Data mass of data particle 1; m_2 is Data mass of data particle 2; r is The Euclidian distance between the two data particle in data space.

The data gravitations from data particles in the same class also obey the superposition principle:

Lemma 1. (**Superposition Principle**): Suppose $P_1, P_2, \ldots, P_m$ are m data particles in the same data class, the gravitations they act on another data element are $F_1, F_2, \ldots, F_m$, and then the composition of gravitations is: $F = \sum_{i=1}^{m} F_i$.

Definition 4. **(Data Gravitation Field):** Data particles act on each other by data gravitation, and form a field that congests the whole data space. This field is named as data gravitation field. Because data gravitation can belong to different data classes, when data gravitation field is discussed, it refers to the field that is formed by the same kind of data gravitations.

Field strength is a key factor of data gravitation field. Field strength of an appointed point equals the composition of data gravitations that all data elements belong to the same data class act on an atomic data particle on the appointed point. Similar to the isopiestic surface in physical field, all points in data gravitation field that have equivalent field strength form a surface in data space, and this surface is called isopiestic surface in data gravitation field.

3 Data Classification Based on Gravitation

Based on the data gravitation and data gravitation field, a new classification scheme can be given. The main ideas of this classification scheme are:

1) A training data particle set is formed according to the training data set. The calculation of data particles obeys some certain principles.
2) All test data in the test set are treated as atomic data particles. And any data particle in training data particle set has data gravitation on any test atomic data particle.
3) Gravitations between training data particles and test atomic data particles obey the Law of Data Gravitation.
4) Once training data particle set has been built, the data gravitation field in the data space has also been built and data gravitation field strength on any position in the data space can be calculated.
5) The degree of a test data element belongs to a data class is determined by the data gravitation field strength on the test data's position, and the gravitation field refers to the field produced by the before-mentioned data class.

3.1 Principle of Classification

Lemma 2. Suppose c_1, c_2 are two data classes in training data set. For a given test data element P, the gravitation c_1 acts on P is F_1, and F_2 is the gravitation c_2 acts on P. If $F_1 > F_2$, then the degree of P belongs to c_1 is stronger than that to c_2.

Fig.1 describes the principle of classification.

Suppose $T=\{<\boldsymbol{x}_1, y_1>, <\boldsymbol{x}_2, y_2>, \ldots, <\boldsymbol{x}_l, y_l>\}$ is a training set in n-dimensional data space, $y \in \{c_1, c_2, \ldots, c_k\}$, c_i represents data class i, k is the number of data classes, l is the number of training samples. A new set of training data particles is created from the original training set. The new training data particle set is $T'=\{<m_1, \boldsymbol{x}_1', y_1>, < m_2, \boldsymbol{x}_2', y_2>, \ldots, < m_l, \boldsymbol{x}_l', y_l>\}$, where l' is the number of data particles, $l' \leq l$, $\boldsymbol{x}_i'$ is the centroid of data particle i, m_i is the data mass of data particle i.

After the training data particle set has been built, the strength of data gravitation field on any position in data space can be calculated. So when a test data element is given, which data class it belong to can be determined by the field strength of the data class.

Suppose $c_1, c_2, \ldots, c_k$ are the data classes in training set, they have $l_1, l_2, \ldots, l_k$ samples (data elements), the training data particle set created from training set has $l_1' + l_2' + \ldots + l_k'$ data particles, where l_i' is the number of data particles which belong to data class i. A given test data can be treated as an atomic data particle P, the centroid is its position $\boldsymbol{x}$.

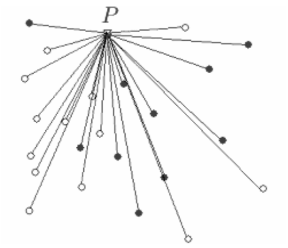


Fig. 1. Classification Based on Data Gravitation. The strength of gravitation determines which class a test data element belongs to. The black dots denote data particles in class c_1. The circles denote data particles in class c_2 .

The gravitation that data class i act on it is:

$$F_i = \sum_{j=1}^{l_i} \frac{m_{ij}}{|x_{ij} - x|^2} \tag{3}$$

Where m_{ij} is the data mass of data particle j in data class i, $\boldsymbol{x}_{ij}$ is its centroid.

If $F_{i'}=max\{F_1, F_2, \ldots, F_k\}$, then according to lemma 2, the test data element belongs to data class i'.

3.2 Principle to Create Data Particle

The simplest method to create data particle is to treat a single data element as one data particle. By means of this, a training sample in training data set can create a data particle. This method is simple and easy to realize, but the shortage of the method is also obvious: The calculation will grow up tremendously with the expanding of the training data set and the efficiency of classification will be reduced smartly.

Another method to create data particle is the Maximum Distance Principle (MDP), the algorithms can be found at [17].

4 Data Classification in Intrusion Detection

The data for our experiments was prepared by the 1998 DARPA intrusion detection evaluation program by MIT Lincoln Lab[19]. This data set has 41 features and five different classes named Normal, Probe, DoS, U2R and R2L. The training and test set comprises of 5092 and 6890 records respectively. As the data set has five different classes we performed a 5-class binary classification. The normal data belongs to class 1, Probe belongs to class 2, DoS belongs to class 3, U2R belongs to class 4 and R2L belongs to class 5.

4.1 Feature Selection

Feature selection is very important in data mining because the quality of data is an important factor which can affect the success of data mining algorithms on a given task. According to the lemma and method in CLIQUE clustering algorithm, we studied a effective feature selection principle by utilizing the Lemma of Monotonicity[20].Table 1 gives the result of feature selection using this algorithm.

Table 1. The feature selection result

CLASS	IMPORTANT VARIABLES (FEATURES)
Class 1	3,10,23…26, 29,30,32,33,34,35, 38…41
Class 2	3,23,24,25,27,29, 30,32,33,34,35,36,38,40
Class 3	1,3,5,6,10,11,22…41
Class 4	3,23,24,33
Class 5	2,3,23,24,33

4.2 Classification Algorithm and Experiment Results

Suppose the number of data elements in the test data set is *m*, the whole detection algorithm can be described as follows[17]:

1). Select important features using the Lemma of Monotonicity;
2). Build up the training data particle set;

```
3). for $i=1$ to $m$
        Calculate the gravitations that normal training set acts on the test data $t_i$;
        Calculate the composition of normal gravitations $F_n$;
        Calculate the gravitations that anomaly training set acts on the test data $t_i$;
        Calculate the composition of normal gravitations $F_a$;
        if $F_a > F_n$ then
                $t_i$ is an anomaly data.
        else
                $t_i$ is a normal data.
        end if
end for
```

For comparison purpose, two other classic classification methods named ID3 [2] and C4.5 [21] were applied in the experiment. A neural network classifier trained by MA with flexible bipolar sigmoid activation functions was constructed using the same training data sets and then the neural network classifier was used on the test data set to detect the different types of attacks. All the input variables were used for the experiments.

Table 2 depicts the detection performance of the NN by using the original 41 variable data set, the table also shows the detection results using C4.5 decision tree. The data from the table shows that DGC can get higher detection performance than NN and C4.5 do, except U2R and R2L attacks.

Table 2. Detection accuracy using DGC, NN,and C4.5 classification models

ATTACK CLASS	DGC	NN	C4.5
Normal	**99.93%**	96.82%	82.32%
Probe	**97.69%**	95.00%	94.83%
DOS	**97.92%**	88.40%	77.10%
U2R	99.59%	99.79%	**99.83%**
R2L	98.59%	**98.92%**	94.33%

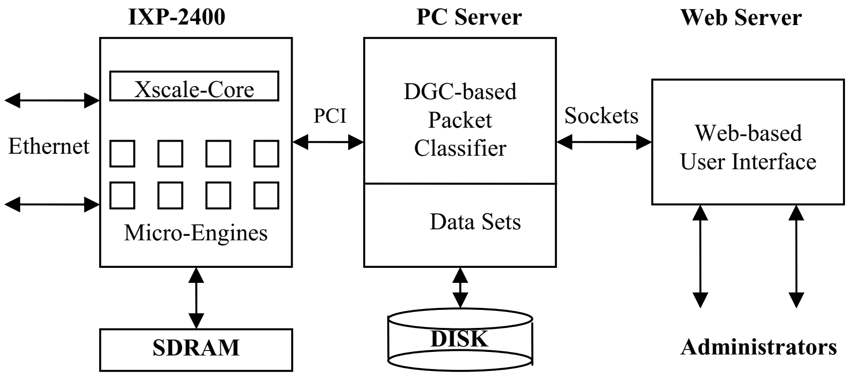


Fig. 2. Prototype Platform based on IXP-2400 Network Processor

4.3 A Prototype Platform Based on Network Processor

To test the classification method under a real IDS environment, we are constructing a prototype platform using a PCI-based Radisys ENP-2611 network processor developing board. In the test platform, an IXP-2400 network processor with eight micro-engines is adopted to give fast processing of the Ethernet packets. The data sets gathered are processed by the XScale embedded processor and a more powerful PC server with ENP-2611 attached. The feature collected from the network processor will be tested via the DGC module running on the PC server to determine if the network activity is normal or deviant. The micro-engines in the network processor can also discard the deviant packets with the same feature in the future according the decision of the DGC module. A web-based user interface is used for the administrative purpose in the system. Figure 2 shows the architecture of the platform.

5 Conclusion and Future Works

The experiment results on the intrusion detection data set proved that the DGC model is very effective. As evident from Table 2, the DGC model gave the best accuracy for most of the data classes (except U2R and R2L). As demonstrated in the paper, the feature selection and the choice of training data set may influence the accuracy of classification. Now an improved method named WDGC (Weighted-feature DGC) is studied, which proposed the concept of weighted feature. By weighting every feature of target classification problem, the degree of importance of every feature can be obtained by its weight. Besides of this, a test prototype of Intelligent Intrusion Detection System which adopts the IXP-2400 network processor (NP) is also under development.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (No.69902005); the National High Technology Development Program of China (No.2002AA4Z3240); and the Science and Technology Development Project of Shandong Province (No.2004GG1101023).

Referencess

1. J. Allen, A. Christie, W. Fithen, J. McHugh, J. Pickel, and E. Stoner. *State of the Practice of Intrusion Detection Technologies*. CMU/SEI-99-TR-028. Carnegie Mellon Software Engineering Institute. 2000.
2. J. R. Qinlan, Introduction of decision trees. *Machine Learning*, 1986;1, pp 86-106.
3. Freund Y, Boosting a weak learning algorithm by majority, *Information Computation*, 1995;121, pp 256-285.
4. Lu Hongjun, Setiono Rudy, Liu Huan, Effect data mining using neural networks, *IEEE Transaction on knowledge and data engineering*, 1996; 8(6), pp 957-961
5. B. E. Boser, I. M. Guyon, V. N. Vapnik, "A trining algorithm for optimal margin classifiers", *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, ACM Press,1992, pp 144-152.
6. V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer Verlag, 1995.
7. Pawlak A, *Rough sets ,theoretical aspects of reasoning about data*, Dordrecht :Kluwer Academic Publishers , 1991.

8. Xiaohua Hu, Nick Cercone, Data Mining via Discretization, Generalization and Rough Set Feature Selection, *Knowl. Inf. Syst.* 1(1), pp 33-60 (1999).
9. Minz S, Jain R, Rough Set based Decision Tree Model for Classification, *In Proc of 5th Intl. Conference*, DaWaK 03, Prague, Czech Republic, Springer, LNCS 2737, September (2003), pp 172-181.
10. Minz S, Jain R, Hybridized Rough Set Framework for Classification: An Experimental View. *HIS 2003*, pp 631-640
11. Peters, J. F., Skowron A, A Rough Set Approach to Knowledge Discovery, *17'th IJIS*, 2002, pp 109-112.
12. Xiaohua Hu, Using Rough Sets Theory and Database Operations to Construct a Good Ensemble of Classifiers for Data Mining Applications, *ICDM 2001*, pp 233-240.
13. Yong Shi, Yuqing Song and Aidong Zhang, A shrinking-based approach for multi-dimensional data analysis, *In the 29th VLDB conference*, Berlin, Germany, September 2003, pp 440-451.
14. Yong Shi and Aidong Zhang, A Shrinking-Based Dimension Reduction Approach for Multi-Dimensional Data Analysis, *The 16th International Conference on Scientific and Statistical Database Management*, Santorini Island, Greece, June 2004.
15. M. Indulska, M. E. Orlowska, Gravity Based Spatial Clustering, *ACM Symposium on GIS*, 2002.
16. Barry Webster, Philip J, Bernhard, A Local Search Optimization Algorithm Based on Natural Principles of Gravitation, *IKE'03*, Las Vegas, Nevada, USA, June 2003, Volume 1, CSREA Press 2003, pp 255-261.
17. Lizhi Peng, Yuehui Chen, Bo Yang, Zhenxiang Chen, "A Novel Classification Method Based on Data Gravitation", In *Proc. of International Conference on Neural Networks and Brain (ICNN&B)*, 2005,pp.667-672.
18. J. MacQueen, Some methods for classification and analysis of multivariate observations, *Proceedings Fifth Berkeley Symposium on Math. Stat. and Prob*, University of California Press, 1967, pp 281-297.
19. KDD cup 99 Intrusion Detection Data Set, online at http://kdd.ics.uci.edu/database/kddcup99/kddcup.data_10_percent.gz
20. R. Agrawal, J. Gehrke, D. Gunopulos, P. Raghavan, Automatic Subspace Clustering of High dimensional Data for Data Mining Applications. *Proc, ACM SIGMOD'98 Int.Conf. on Management of Data*, Seattle, WA (1998), pp 94-105.
21. Quinlan J R, *C4.5: Program for Machine Learning*, Morgan Kauffman Publishers, 1993.

Intrusion Alert Analysis Based on PCA and the LVQ Neural Network

Jing-Xin Wang, Zhi-Ying Wang, and Kui-Dai

Computer School, National University of Defense Technology, Changsha, China
wjx_eagle@163.com

Abstract. We present a PCA-LVQ method and a balanced-training method for efficient intrusion alert clustering. For the network connection records in the rough 1999 DARPA intrusion dataset, we firstly get a purified and dimension-reduced dataset through Principal Component Analysis (PCA). Then, we use the Learning Vector Quantization (LVQ) neural network to perform intrusion alert clustering on the purified intrusion dataset. To our best knowledge, this is the first attempt of using the LVQ neural network and the PCA-LVQ model on intrusion alert clustering. The experiment results show that the PCA-LVQ model and the balanced-training method are effective: the time costs can be shortened about by three times, and the accuracy of detection can be elevated to a higher level, especially, the clustering accuracy rate of the U2R and R2L alerts can be increased dramatically.

1 Introduction

Most current intrusion detection systems suffer from several deficiencies such as alert overload and the high false alarm rate. To overcome these shortcomings, the better intrusion detection method and the post-analysis of the intrusion alerts are both important. We focus on the intrusion alert analysis. In this research area, the 1999 DARPA intrusion dataset is the de factor standard for the comparison of the different methods, and we also use this intrusion dataset for the comparability between our method and other methods. Most of other research [1,2,3,4] are directly done on the original 1999 DARPA intrusion dataset. However, from the viewpoint of information theory, the original intrusion dataset includes large volume of redundant and inessential information. The existence of these insignificant information will result in some adverse scenarios. On one hand, the system performance cost will be increased. Obviously, the processor cost and the memory requirement will be both increased with the existence of the overmuch information to be processed, thus, the whole system performance cost will be increased. On the other hand, the existence of inessential information will lower the accuracy of detection. So, in our opinion, to operate directly on the rough intrusion dataset is not an optimal method.

Unlike other research, we propose a new hybrid method for intrusion alert analysis. We call it PCA-LVQ method. Concretely speaking, we first preprocess the rough intrusion dataset by performing principal component analysis. By doing so, the redundant and inessential information in the rough intrusion dataset will be eliminated, and a purified intrusion dataset with the least essential information will be produced. Then, we will use the learning vector quantization neural network to analyze the dimension-reduced network connection record (which is also called intrusion alert in this context) in the purified intrusion dataset. By this PCA-LVQ method, we aim at obtaining the higher accuracy of detection with the lowered system performance costs than before. We will verify this method through some experiments.

The remaining of this paper is organized as follows: Section 2 discusses the 1999 DARPA intrusion dataset and its preprocessing by principal component analysis. In Section 3, we present a LVQ clustering algorithm and describe the algorithm in detail. The experiment results and the corresponding analysis are presented in Section 4. Finally, in Section 5, we conclude the paper.

2 Intrusion Dataset and Principal Component Analysis

The 1999 DARPA intrusion dataset is a well known dataset using for evaluating the intrusion detection systems. It is also called kddcup.data on its site [5]. This dataset was prepared by the DARPA Intrusion Detection Evaluation program operated by MIT Lincoln Laboratory. In the project, Lincoln labs acquired nine weeks of raw TCPdump data. The raw data were processed into network connection records, totaled about 5 million connection records. Each network connection record represents an attack type or normal network traffic.

Here, it is important to note that the test data is not from the same probability distribution as the training data, and it includes some specific unknown attack types not in the training dataset. The difference between the training data and the test data accords with the real network environment because unknown attack types are numerous and more dangerous in the real network environment. In fact, the ability to detect unknown attack types is very important. Concretely speaking, the training dataset includes 22 attack types, and the test dataset includes additional 17 new attack types besides these 22 attack types. All of these 39 attack types may be classified to categorized into 4 classes, i.e, R2L, DOS, U2R and Probing.

These attack types and the normal network traffic are all represented by the network connection records, and each network connection record consists of 41 features. However, not all the information will be useful for analysis. Of all these features, which are the most useful, which are less important, and which may be inessential? The problem is significant because the elimination of useless features has at least two advantages. Firstly, the computation speed can be increased because the information to be processed has decreased along with the elimination of the inessential features. Secondly, the accuracy of detection can be improved. As discussed in our previous work[6], the overmuch features will make the alert threshold much higher, this is because the intrusion detection system can give an alarm only when all these features are present. The higher alert threshold will result in the increase of false negative rate, that is to say, some attack will be misclassified as normal. By eliminating the inessential features, the alert threshold can be maintained at an acceptable level.

According to the analysis above, we consider that identifying key features is necessary and beneficial. Unlike literature [7], in which Support Vector Machine

(SVM) is used for feature ranking, and for each attack type, one special SVM model is used for the feature extraction. Although SVM has its virtues, it doesn't meet our requirements here. So, we adopt another useful method called Principal Component Analysis (PCA) to fulfill the task of feature ranking and extraction.

Principal component analysis has proven to be an especially useful technique for dimension reduction and multivariate analysis. Its many application areas include data compression, image analysis, visualization, pattern recognition and time series analysis. An important virtue of PCA is that the extracted components are statistically orthogonal to each other. Orthogonality of components results in speedup in training and robust convergence as shown in [8]. So, our PCA-LVQ method is reasonable and feasible, we expect that our learning vector quantization neural network can work much better based on the production of PCA.

According to the literature [9], the most common definition of PCA is that, for a set of observed vectors $\{v_i\}, i \in \{1,2,...,N\}$, the q principle axes $\{w_j\}$, $j \in \{1,2,..,q\}$ are those orthonormal axes onto which the retained variance under projection is maximal. It can be shown that the vectors w_j are given by the q dominent eigenvectors (i.e. those with largest associated eigenvalues) of the covariance matrix $C = \sum_i \frac{(v_i - \bar{v})(v_i - \bar{v})^T}{N}$ such that $Cw_j = \lambda_i w_j$, where $\bar{v}$ is the simple mean. The vector $u_i = W^T(v_i - \bar{v})$, where $W = (w_1, w_2, ..., w_q)$, is thus a q-dimensional reduced representation of the observed vector $\{v_i\}$.

For the connection records in the 1999 DARPA intrusion dataset, the purpose of performing principal component analysis is to find the principal components of the connection records, i.e., the feature vector that can describe the connection records exactly and sufficiently, but not redundantly. In mathematical terms, we wish to find the principal components of the distribution of the connection records, or the eigenvectors of the covariance matrix of the set of the connection records [9]. Through PCA, the dimension of the feature vector should be smaller than the dimension of the raw connection record. The reduction of vector dimension will not only reduce the processor cost and the memory requirement, but also can categorize the connection records with higher accuracy. The feature vector can be thought of as a set of features which together characterize the difference between the connection records. Then, the difference between the different connection records will be learned by the LVQ neural network. After learning, the LVQ neural network can distinguish the connection records from each other correctly.

By virtue of principal component analysis and security domain knowledge, 12 principal features are selected for representing the connection records. So, the connection record can be represented by a 12-dimension feature vector *CR*, as follows:

CR=(src_bytes,dst_bytes,flag,service,root_shell,su_attempted,num_failed_logins, is_guest_login,count,srv_count,serror_rate,rerror_rate)

These 12 features include the inbeing property of a certain connection record, the statistical property of the connection records within a time window and some features which are security conscious. So, theoretically speaking, the feature vector should represent the connection record well. We will verify this through some experiments.

3 LVQ Clustering Algorithm for Intrusion Alert Analysis

Closely related to SOM but not identical, LVQ is a supervised competitive neural network model[10,11]. It uses pre-assigned cluster labels to the training samples, to maximize correct data classification. Unlike SOM, where clusters are generated automatically based on samples similarity, here the clusters are predefined. In our case the cluster labels are the attack types.

We select LVQ for several reasons. Firstly, LVQ has higher classification accuracy than SOM. As a supervised competitive ANN model, LVQ uses pre-assigned cluster labels to training samples, which can minimize the average expected misclassification probability. Secondly, it is well known that SVM is not applicable to the scenarios that involve large number of samples, so SVM is not very applicable to the real production environment. However, LVQ has no obstacle in dealing with the large dataset. We know that alert overload is very pervasive, and LVQ has the potential to be used in the real production environment. Thirdly, the optimized LVQ algorithm can learn faster than the BP model which often suffers from the slow learning process and the difficulty in reaching convergence. In view of the reasons above, we think that applying LVQ on intrusion alert analysis is feasible and helpful, and we will verify this through some experiments.

We formally describe the LVQ clustering algorithm as follows.

LVQ Clustering Algorithm
INPUT: Sample pattern vectors $X = (X_0, X_1, ..., X_{n-1})$
OUTPUT: Codebook $Y = (Y_0, Y_1, ..., Y_C)$

Step1: Initialization and normalization.

Initialize Y_i with c sample pattern vectors randomly selected. Normalize X_i.

Step2: Find the winner *w**.

For X_i and $\forall\ Y_k \in Y$, $\|X_i - Y_k\|$ are computed, and *w** will be judged as the winner *iff* the following equation is correct:

$$\|X_i - Y_w\| = \min_k\{\|X_i - Y_k\|\} \tag{1}$$

Step3: Weight vector update with the Prize-Punishment learning algorithm.

$$Y_l(t+1) = Y_l(t), l \neq w \tag{2}$$

$$Y_w(t+1) = [1 - s(t)\eta(t)]Y_w(t) + s(t)\eta(t)X(t) \tag{3}$$

where $s(t)$ is the prize-punishment factor, $X(t)$ is the input vector at time *t*, and $\eta(t)$ is the learning rate .

Step4: Repeat Step2 and Step3 until convergence.

END

By equation 1, the target cluster of the input vector X_i has been decided as the codeword Y_w, then the corresponding weights will be updated. By equation 2 and equation 3, we can see that only the winner's weights should be updated. Also by equation 3, the Prize-Punishment learning procedure has been embodied by the prize-punishment factor $s(t)$ clearly, which can be assigned to 1 if the classification is correct and -1 if the classification is wrong. The learning rate $\eta(t)$ is a descending function of time t to stipulate faster convergence. In our case, we define $\eta(t)$ as the following recursion, and the initial value was set as 0.35 (Selected through some experiments).

$$\eta(t) = \frac{\eta(t-1)}{1+\eta(t-1)} \tag{4}$$

By equation 4, we stipulate that $\eta(t)$ is descending monotonously. In comparison with literature[10], in which some additional measures must be adopted to prevent $\eta(t)$ from increasing beyond the original value, our method is more succinct and helpful to faster convergence.

In our case, we use LVQ to categorize the connection records into different clusters correctly. As discussed in section 2, we know that the network connection records can be divided into five different classes, i.e., Normal, Probing, DOS, U2R and R2L. So, after performing LVQ clustering algorithm on the network connection records, five cluster centers should be created. That is to say, ideally, each connection record should only belong to one of the five classes and congregate into a certain cluster center only. For this purpose, there needs exist two phases, namely training phase and clustering phase.

During the training phase, by using some connection records as the training data of the supervised competitive LVQ neural network model, the Euclidean space R_{CR}^{12} will be "forcefully" divided into five subspace (five clustering centers), and each subspace consists of one type of connection records. In another words, the training phase is also the partition phase of the Euclidean space R_{CR}^{12} based on the Euclidean distance similarities. After training, the following equations should be correct:

$$R_{CR}^{12} / \phi = \left\{R_{Normal}^{12}, R_{\Pr obing}^{12}, R_{DOS}^{12}, R_{U2R}^{12}, R_{R2L}^{12}\right\} \tag{2}$$

$$R_{Normal}^{12} \wedge R_{\Pr obing}^{12} \wedge R_{DOS}^{12} \wedge R_{U2R}^{12} \wedge R_{R2L}^{12} = \phi \tag{3}$$

The equation 5 and equation 6 together stipulate that the ideal training phase should divide the Euclidean space R_{CR}^{12} into five subspaces which will not overlap each other. After training, the knowledge on adscription of the connection records has been stored by the weights.

During the clustering phase, the validity of the LVQ clustering algorithm will be tested. Firstly, the 12 features computed by PCA will be extracted from some network connection records, then, the connection records can be described as a 12-dimension feature vector $\overline{X}$. As input, $\overline{X}$ will be submitted to the LVQ neural network and processed, finally clustered into one item of the finite set $\{R^{12}_{Normal}, R^{12}_{\Pr obing}, R^{12}_{DOS}, R^{12}_{U2R}, R^{12}_{R2L}\}$. More formally, we describe the problem as follows:

Given codebook $C = \{Y_{Normal}, Y_{\Pr obing}, Y_{DOS}, Y_{U2R}, Y_{R2L} | Y \in R^{12}\}$ and the input vector $\overline{X}$ to be clustered, LVQ will look for a certain codeword Y from C and stipulate the correctness of the following equation:

$$\{Y \in C \wedge \forall Y' \in C \Rightarrow \|Y - \overline{X}\| \leq \|Y' - \overline{X}\|\} \tag{4}$$

By equation 7, for any connection records X to be clustered, the codeword Y, which has the least Euclidean distance with X will be computed. Then the connection record X will be clustered to the cluster center represented by codeword Y.

4 Experiment Results and Analysis

In several previous research, especially in the literature[3], there exists an obvious problem, namely, the detection accuracy of the U2R alerts and the R2L alerts is very low (usually less than 5%), but the accuracy for other alert types is high(often more than 90%). We think that one possible reason for this difference is that the alert types distribution in the 1999 DARPA intrusion dataset is not very balanced. For example, in the training dataset, the network connection records labeled as “Normal” take up about 20%, but “U2R” and “R2L” take up only 0.01% and 0.23% respectively. In view of this, we adopt a new method in selecting the training samples during the training phase. Concretely speaking, we select 50 connection records as the training samples of the alert type “U2R”(this alert type has only 52 samples in the training dataset), and we select 200 network connection records as the training samples for other alert types respectively. By doing so, the basic balance of the alert types distribution can be reached, and we think that these training samples are enough to establish an effective clustering model.

In order to verify our methods, we have two tasks. Firstly, we use the LVQ model without PCA in the experiment 1 to test whether the LVQ model is useful for intrusion alert analysis. Then, we use the PCA-LVQ model in the experiment 2 to find the advantages of integrating PCA with LVQ. To describe the experiment results more clearly, we define a measure criterion. If M alerts should belong to $cluster_i$ and our method put N alerts of M into other clusters, then we define the clustering accuracy rate (CAR) as (M-N)/M.

For each alert type (we consider “Normal” as a special “alert type” here for convenience), we select 200 connection records from the test dataset respectively as the test samples. Each group of 200 connection records consists of two parts: 50 known-type alerts which appear in both the training dataset and the test dataset, and 150 unknown-type alerts which appear only in the test dataset. Here, the application of the unknown-type alerts is to test the generalization ability of the LVQ neural network. After performing the corresponding experiments, we get the experiment results as follows.

Table 1. LVQ model without PCA

clustered as / *actually as*	C_{Normal}	$C_{\mathrm{Pr}\,obingl}$	C_{DOS}	C_{R2L}	C_{U2R}	**measure criterion**	CAR
Normal	192		5	3			96%
Probing	2	178		17	3		89%
DOS	7		193				96.5%
R2L	7		16	164	13		82%
U2R	12	14		21	153		76.5%

From Table 1, we can see that the clustering accuracy rate for the R2L alerts and U2R alerts are increased at a large extent, respectively as 82% and 76.5%. The results show that our balanced-training method is effective. For other alert types, namely Probing, DOS and Normal, the clustering accuracy rate is neither below the accuracy rate shown in other research

Table 2. PCA-LVQ model

clustered as / *actually as*	C_{Normal}	$C_{\mathrm{Pr}\,obing}$	C_{DOS}	C_{R2L}	C_{U2R}	**measure**	CAR
Normal	197		2		1		98.5%
Probing	3	191		6			95.5%
DOS	1		199				99.5%
R2L	37		16	143	4		71.5%
U2R	44		25	2	129		64.5%

According to Table 2, in comparison with the LVQ model, the PCA-LVQ model can get higher clustering accuracy rate for the connection records which represent DOS, Probing and Normal respectively. This shows that the PCA-LVQ model is effective. On the other hand, the clustering accuracy rate of R2L alerts and U2R alerts in the PCA-LVQ model is a little lower than that in the PCA model. The part reason of this decrease is that the detection of R2L attack and U2R attack depends heavily on the content features within the connection records, but the 12 features extracted include only few content features. Although, the clustering accuracy rate of the R2L and U2R alerts in the PCA-LVQ model is still acceptable and much higher than previous 5%, this also shows that our balanced-training method is feasible and effective

Besides, with the dimension reduction and the betterment of the original LVQ algorithm, the time cost for both the training phase and the clustering phase are shortened, as shown in Table 3.

Table 3. Time costs comparison

time costs / *model*	training time	clustering time
LVQ	~1min37sec	~45sec
PCA-LVQ	~35sec	~14sec

5 Conclusions

In this paper, we present a PCA-LVQ model for intrusion alert clustering and verify the model through some experiments. Altogether, the results are encouraging. Firstly, the time costs for both the training phase and the clustering phase are shortened attributed to the dimension reduction through principal component analysis. Secondly, the balanced-training method results in the much higher clustering accuracy rate of the alert type R2L plus U2R than before. Thirdly, the overall clustering accuracy rate has been increased to a satisfactory level.

One shortage of our research is that the experiments are operated on the offline intrusion dataset. Although the standard 1999 DARPA intrusion dataset has verified the effectiveness of the PCA-LVQ model, it is not yet an applied tool that can detect intrusions in real time. So, our future research is to develop a real-time intrusion alert analysis system based on the research in this paper.

References

[1] M. Mahoney: A Machine Learning Approach to Detecting Attacks by Identifying Anomalies in Network Traffic, Ph.D. dissertation, Florida Institute of Technology,2003.

[2] Eleazar Eskin, Andrew Arnold, Michael Prerau, and Salvatore Stolfo: A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data, Applications of Data Mining in Computer Security, 2002.

[3] Y. Bouzida, S. Gombault: EigenConnections to Intrusion Detection, Proceedings of the 19th IFIP International Information Security Conference. Kluwer Academic, August, 2004.

[4] Manikantan Ramadas. Detecting Anomalous Network Traffic with Self-Organizing Maps. Master's thesis, Ohio University, Mar 2003.

[5] kddcup.data, available at http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[6] Wang jing-xin: Feature selection for the intrusion detection system. Proceedings of the sixth conference on computer application and security, China, 2002.

[7] Srinivas Mukkamala1, Andrew H. Sung: Identifying Significant Features for Network Forensic Analysis Using Artificial Intelligent Techniques, International Journal of Digital Evidence, Winter 2003, Volume 1, Issue 4.

[8] E. Oja: Neural Networks, principal components, and subspaces, International Journal of Neural Systems, vol.1, no.1, pp 61-68, 1989 .

[9] Jolliffe, I. T: Principal Component Analysis, Springer Verlag, New York, NY, third edition, 2002.

[10] Teuvo Kohonen, Jussi Hynninen, and Jari Kangas. LVQ_PAK: The Learning Vector Quantization Program Package. Techinical report, 1996.

[11] Martin T.Hagan, Howard B.Demuth, and Nark H. Beale: Neural network design, China Machine Press, 2002,8

A Novel Color Image Watermarking Method Based on Genetic Algorithm and Neural Networks*

Jialing Han[1,2], Jun Kong[1,2,**], Yinghua Lu[1], Yulong Yang[1], and Gang Hou[1,3]

[1] Computer School, Northeast Normal University, Changchun, Jilin Province, China
[2] Key Laboratory for Applied Statistics of MOE, China
[3] College of Humanities and Science, Northeast Normal University, Changchun, China
{hanjl147, kongjun, luyh}@nenu.edu.cn

Abstract. In the past a few years, many watermarking approaches have been proposed for solving the copyright protection problems, most of the watermarking schemes employ gray-level images to embed the watermarks, whereas the application to color images is scarce and usually works on the luminous or individual color channel. In this paper, a novel intensity adaptive color image watermarking algorithm based on genetic algorithm (CIWGA) is presented. The adaptive embedding scheme in color image's three component sub-images' wavelet coefficients, which belong to texture-active regions, not only improves image quality, but also furthest enhances security and robustness of the watermarked image. Then a novel watermark recovering method is proposed based on neural networks, which enhance the performance of watermark system successfully. The experimental results show that our method is more flexible than traditional methods and successfully fulfills the compromise between robustness and image quality.

1 Introduction

With the widespread use of digital multimedia and the development in computer industry, digital multimedia contents suffer from infringing upon the copyrights with the digital nature of unlimited duplication, easy modification and quick transfer over the Internet [1]. As a result, copyright protection has become a serious issue. Hence, in order to solve this problem, digital watermarking technique has become an active research area [2] [4].

In the past a few years, most of the watermarking schemes employ gray-level images to embed the watermarks, whereas their application to color images is scarce and usually works on the luminous or individual color channel. Fleet [3] embedded watermarks into the yellow-blue channel's frequency domain. Kutter et al. [5] proposed another color image watermarking scheme that embedded the watermark into the blue-channel of each pixel by modifying its pixel value. But they didn't notice that the capacity of hiding information in different color channel is varied with the image changing. In this

* This work is supported by science foundation for young teachers of Northeast Normal University, No. 20061002, China.
** Corresponding author.

paper, a novel watermarking embedding method based on genetic algorithm (GA) is proposed. GA is applied to analyze the influence on original image when embedding and the capacity of resisting attacks in every channel. Then the optimized intensity is selected for every color channel. Using GA can improve image quality and furthest enhance security and robustness of the watermarked image simultaneously. This algorithm fulfills an optimal compromise between the robustness and image quality.

This paper is organized as follows: the watermark embedding algorithm and extraction algorithm are described in Section 2 and Section 3, respectively. Experimental results are presented in Section 4. Section 5 depicts the watermark recovering method. Finally, conclusions are given in Section 6.

2 The Embedding Algorithm

2.1 The Host Image Analyzing

Every host image has its own gray information and characteristics. Based on human visual system's characteristic, the human eyes have different sensitivity to noise in areas with different luminance and texture. So we consider analyzing the host image before watermark embedding to ensure the imperceptibility of the proposed watermarking scheme.

For the purpose of well presenting the feature of the host image, we propose a method that fully makes use of the characteristics of singular value and variance. Variance measures the relative contrast and smoothness of the intensity in a region while singular values, which are obtained from the SVD, stand for the luminance of the region. We employ these characteristics to analyze the host image. The method is described as follows:

1. Separate three component sub-images Red, Green and Blue from host image *I*.
2. Divide each component sub-image into un-overlapped 8×8 sub-blocks in spatial domain.
3. Compute each image sub-block's variance *V* and find the maximum value *V_max* among them. Define the normalized variance as $A1 = V / V\_\max$.
4. Perform SVD on each sub-block and then computer *E* of the obtained *S* matrix.

$$\begin{aligned} [U\ S\ V] &= SVD(I), \\ E &= S(1,1)/S(2,2) \end{aligned} \tag{1}$$

   where *U*, *S* and *V* are the three result components of SVD transformation. S(1,1) and S(2,2) denote the two maximum values in the diagonal matrix *S*.
5. Find the maximum value of *E* marked as *E_max* and define the normalized *E* as $A2 = E / E\_\max$.
6. Calculate activity factor of each sub-block as $A = \gamma 1 \times A1 + \gamma 2 \times A2$, where $\gamma 1$ and $\gamma 2$ are the weights of *A1* and *A2* with $\gamma 1 + \gamma 2 = 1$.
7. Sort all the sub-blocks' activity factors in decreasing order. The sub-blocks, which have large magnitude of activity factors have complex texture and median luminance and are suitable for embedding watermark with imperceptibility.

After analyzing the host image using our method, we have obtained the activity factors of all the sub-blocks. According to the activity factors, we can select the best sub-blocks which have good visual masking effect to embed watermark.

2.2 Intensity Optimizing Using GA

For the selected sub-blocks, the discrete wavelet decomposition is adopted in frequency domain to embed watermarks. The multi-resolution feature and compatibility to JPEG-2000 compression standard [7] of wavelet transform make the embedded watermark robust to compression operation. Intensity optimal selecting algorithm is described as follows:

1. Transform the selected sub-blocks using discrete wavelet transform. Select coefficients to embed watermark *W*.
2. Insert watermark signal at coefficients called w_co using additive modulation. Every component sub-image of the color image has its own embedding intensity as $\alpha(i)$. w_co^w denotes the wavelet coefficients after embedding.

$$w\_co^w = w\_co + \alpha(i) \times W \quad i \in \{1, 2, 3\}. \tag{2}$$

3. Perform the inverse discrete wavelet transform on w_co^w.
4. Embed the watermarked sub-images back into the original host image to get the watermarked color image *I'*.
5. Apply the attacking schemes on *I'*, and then adopt the GA training process to search for the optimal intensity for each channel.

The flowchart for illustrating intensity optimal selecting algorithm using GA is shown in Fig. 1.

Not all watermarking applications require robustness to all possible signal processing operations. In addition, the watermarked image after attacking needs to be worthy of using or transmitting. Therefore, some attacks like image-cropping is not employed in our GA training procedure [8]. In this paper, three major attacking schemes are employed, namely, additive noise attack, median filtering attack, and JPEG attack with quality factor of 50%. The quality of watermark extracted from embedded image *I'* is measured by the normalized correlation (NC). The NC between the embedded watermark $W(i, j)$ and the extracted watermark $W'(i, j)$ is defined as,

$$NC = \frac{\sum_{i=1}^{H} \sum_{j=1}^{L} W(i, j) \times W'(i, j)}{\sum_{i=1}^{H} \sum_{j=1}^{L} [W(i, j)]^2} . \tag{3}$$

The watermarked image's quality is represented by the peak signal-to-noise ratio (PSNR) between the original color image *I* and watermarked image *I'*, as follows,

$$PSNR = 10 \times \log_{10} \left(\frac{M \times N \times \max(I^2(i, j))}{\sum_{i=1}^{M} \sum_{j=1}^{N} [I(i, j) - I'(i, j)]^2} \right) . \tag{4}$$

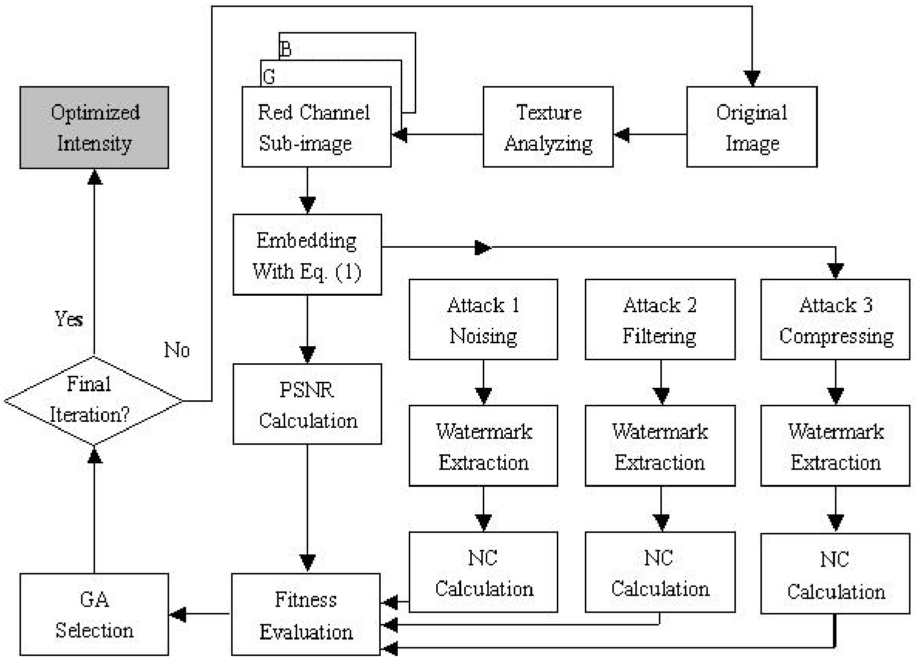


Fig. 1. The flowchart of intensity optimizing algorithm

After obtaining the PSNR of the watermarked image and the three NC values after attacking, we are ready to adopt the GA training process. The fitness function in the *m*th iteration is defined as:

$$f_m = -(PSNR_m + \lambda \sum_{i=1}^{3} NC_{m,i}), \tag{5}$$

where f_m is fitness value, λ is the weighting factor for the NC values. Because the PSNR values are dozens of times larger than the NC values in the GA fitness function, the NC values are magnified with the weighting factors λ in the fitness function to balance the influences caused by both the imperceptibility and robustness requirements.

2.3 Watermark Embedding

The first five steps of watermark embedding algorithm are the same as intensity optimal selecting algorithm, and then the obtained optimal intensity is used to form watermarked image. Fig. 2 is the block-diagram of the embedding algorithm.

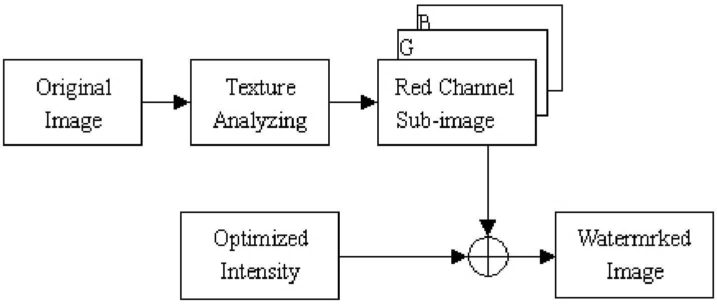


Fig. 2. The block-diagram of embedding algorithm

3 Watermark Extracting

Watermark extraction algorithm is the exact inverse process of embedding algorithm. The watermark can be extracted just when we get the optimal intensity as the secret keys.

4 Experimental Results

The performance of digital watermarking system can be characterized by the following aspects: imperceptibility, security and robustness. All these aspects are evaluated by experimental results respectively in our study. In our simulation, 'Lena' image and 'Baboon' image with the size of 256×256 are taken as test images and watermark with size of 64×64 is shown in Fig. 6(d). The result images of test image 'Lena' and 'Baboon' are shown in Fig. 3(b) and Fig. 4(b).

When free of any attacks, the PSNR of the watermarked image 'Lena' is 35.8487, NC is 1 and the PSNR of the watermarked image 'Baboon' is 36.3028 and NC is 1.

In the GA training process, ten individuals are chosen for every iteration. The crossover operation is selected as scattered function in the MATLAB Genetic

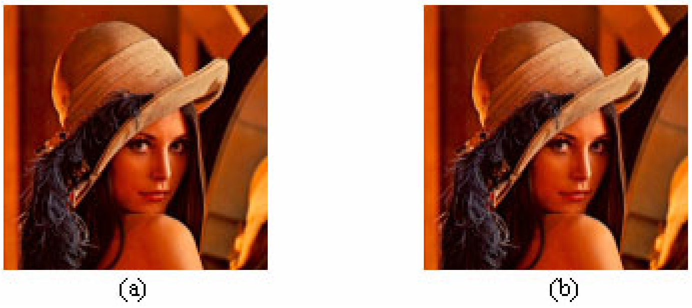


Fig. 3. (a) Original host image 'Lena', (b) Result image watermarked

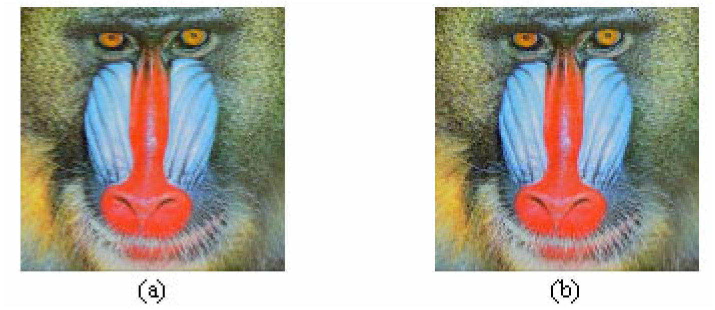


Fig. 4. (a) Original host image 'Baboon', (b) Result image watermarked

Algorithm Toolbox. The selection operation is selected as stochastic uniform function and the mutation operation is Gaussian function with the scale value 1.0 and the shrink value 1.0. The training iterations are set to 200. The fitness values converge after 200 iterations, and the optimized intensity with the optimal fitness value is 62, 64, and 94 for R, G and B channel respectively.

The result images under different attacks and the watermarks exacted are depicted in Fig. 5. Seen from Table 1, the conclusion can be drawn that our algorithm is robust to attacks encountered always in image processing and transmission.

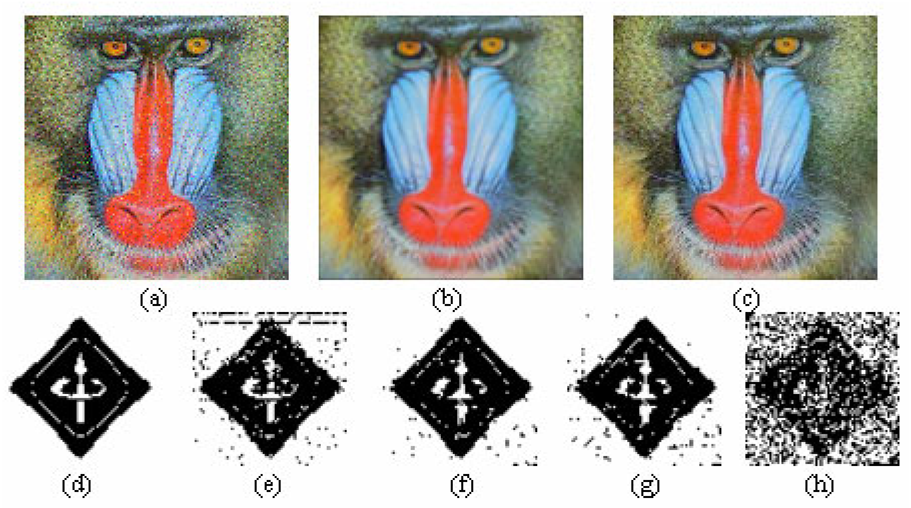


Fig. 5. (a) Result image of watermarked 'Baboon' under additive noising attack, (b) Watermarked image under filtering attack, (c) Watermarked image under compressing attack, (d) Original watermark, (e-g) Extracted watermarks from (a-c) using our method, respectively. (g) Extracted watermark from (c) using Kutter's method.

Table 1. Experimental results under different attacks of our scheme (measured by NC)

Attack Type	Baboon	Lena	Airplane
Attack-free	1	1	1
Additive noising	0.9137	0.9139	0.9479
Filtering	0.9320	0.9536	0.9139
JPEG QF=80	0.9957	0.9830	0.9957
JPEG QF=50	0.9801	0.9547	0.9861
JPEG QF=30	0.9639	0.9390	0.9752

To evaluate the robustness of the proposed watermarking scheme, Kutter's algorithm is simulated as comparison. The results under several attacks of Kutter's algorithm are shown in Table 2.

Table 2. Experimental results under different attacks of Kutter's scheme (measured by NC)

Attack-free	Noising	Filtering	JPEG QF=80	JPEG QF=50	JPEG QF=30
0.9684	0.9546	0.9362	0.6386	0.5925	0.5071

Compared with Table 1, it can be concluded that our algorithm is more robust than Kutter's, especially in resisting additive nosing and JPEG compressing.

5 Watermark Recovering

From the experimental results above, the conclusion can be drawn that our algorithm is robust to many kinds of attacks. In order to get better capability, a watermarking recovering method is proposed based on neural networks. Neural networks can distinguish the prototype of extracting watermark even when it is not clear under sharp attacking. Back-propagation (BP) neural networks are employed to identify the characteristic of extracting watermark, which is extracted using principal components analysis (PCA).

5.1 Neural Network Training

Attacking original watermark with different kinds of attacks to get a set as the training set of the neural networks. Then PCA is employed to analyze watermarks, and the above ten eigenvalues are chosen as the input in training pattern. Every node in output layer represents different watermarks. The eigenvalue as input vector is defined by $P_k = (a_1, a_2, \cdots, a_n)$, and the desired output for the neural networks corresponding to the input is defined by $T_k = (y_1, y_2, \cdots, y_q)$, where *n* and *q* stand for the number of input and output nodes respectively, and *k* is the number of training patterns. The structure of the neural networks is depicted in Fig. 6. The neural networks include an input layer with ten nodes, a hidden layer with twenty hidden nodes, and an output layer with q nodes, which can be varied with the number of watermarks' type.

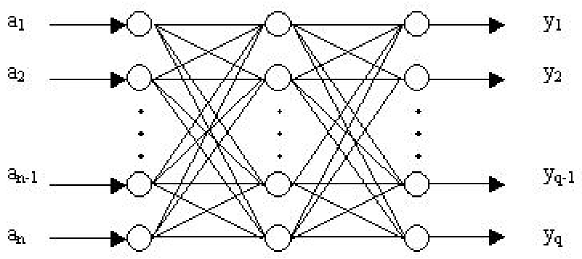


Fig. 6. The structure of the neural networks used in our watermark techniques

5.2 Watermark Recovering Based on the Trained Neural Networks

The trained neural networks perform a highly adaptive capacity in identifying watermarks. Taking an extracted watermark from an embedding image after attacking as a test image, the above ten eigenvalues are chosen as the input, the prototype of attacked watermark can be recognized clearly using neural networks. In the experiments, even NC of the test watermark is 0.3628, which is shown in Fig. 7(a), the trained neural networks also can associate its original watermark, as shown in Fig. 7(b). So it can conclude that the proposed watermark recovering method is an important complementation in watermark system.

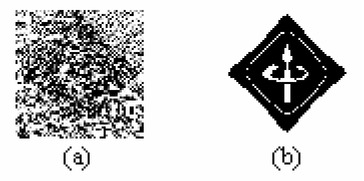


Fig. 7. (a) Test watermark, (b) Original watermark of (a)

6 Conclusion

A novel embedding intensity adaptive CIWGA is proposed in this paper. A color image is divided into three channels firstly. Then genetic algorithm is applied to analyze the influence on the original image when embedding and the capacity of resisting attacks in every channel. At last, the watermark is embedded in R, G and B channels respectively. Using genetic algorithm is not only able to improve image quality, but also furthest enhance security and robustness of the watermarked image. Using neural networks, the proposed watermark recovering method enhances the performance of watermarking technique successfully. This algorithm fulfills an optimal compromise between the robustness and image quality.

References

1. W. N. Cheung: Digital Image Watermarking in the Spatial and Transform Domains. ProCeedings of TENCON'2000, Sept. 24-27,2000,3
2. Xu-Dong Zhang, Jian Feng, Kwok-Tung Lo: Image Watermarking using Tree-based Spatial-frequency Feature of Wavelet Transform. Journal of Visual Communication and Image Representation 14(2003) 474-491
3. D. Fleet, D. Heeger: Embedding invisible information in color images. Proc. 4th IEEE International conference on Image Processing, Santa Barbara, USA, 1(1997) 532-535
4. Jun Kong, Wei Wang, Yinghua Lu, Jialing Han, Gang Hou: Joint spatial and frequency domains watermarking algorithm based on wavelet packets transform.The 18th Australian Joint Conference on Artificial Intelligence ,2005
5. M. Kutter, F. Jordan, F. Bossen: Digital watermarking of color images using amplitude modulation. J. Electron. Imaging 7(2) (1998) 1064-1087
6. M. Gen, R. Cheng: Genetic Algorithms and Engineering Design. Wiley, New York, NY, 1997
7. M. A. Suhail, M. S. Obaidat, S. S. Ipson, B. Sadoun: A Comparative Study of Digital Watermarking in JPEG and JPEG 2000 Environments. Information Sciences 151(2003) 93-105
8. Chin-Shiuh Shieh, Hsiang-Cheh Huang, Feng-Hsing Wang, Jeng-Shyang Pan: Genetic watermarking based on transform-domain techniques. Pattern Recognition 37(2004) 555-565
9. Pao-Ta Yu, Hung-Hsu Ysai, Jyh-Shyan Lin: Digital watermarking based on neural networks for color images. Signal Processing 81(2001) 663-671
10. I. J. Cox, J. Kilian, F.T. Leighton, T. Shamoon: Secure spread spectrum watermaking for multimedia. IEEE Trans. Image Process 6(12) (1997) 1673-1687
11. E. Ganic, A.M. Eskicioglu: Robust DWT-SVD domain image watermarking: embedding data in all frequencies. Proceedings of 2004 Multimedia and Security Workshop on Multimedia and Security, (2004) 166-174
12. Piyu Tsai, Yu-Chen Hu, Chin-Chen Chang: A color image watermarking scheme based on color quantization. Signal Processing 84(2004) 95-106
13. Joong-Jae Lee, Won Kim, Na-Yong Lee: A new incremental watermarking based on dual-tree complex wavelet transform. The journal of supercomputing, 33, 133-140, 2005
14. Chin-Chen Chang, Piyu Tsai, Chia-Chen Lin: SVD-based digital image watermarking scheme. Pattern Recognition Letters 26(2005) 1577-1586
15. Frank Y. Shih, Scott Y.T. Wu: Combinational image watermarking in the spatial and frequency domains. Pattern Recognition 36(2003) 969-975

Color Image Watermarking Algorithm Using BPN Neural Networks

Cheng-Ri Piao[1], Sehyeong Cho[2], and Seung-Soo Han[3]

[1, 3] Department of Information Engineering & NPT Center,
[2] Department of Computer Software Engineering
Myongji University
Yongin, Kyunggi, 449-728, South Korea
[3] shan@mju.ac.kr

Abstract. This paper proposes a new watermarking scheme in which a logo watermark is embedded into the discrete wavelet transform (DWT) domain of the color image using Back-Propagation Neural networks (BPN). In order to strengthen the imperceptibility and robustness, the original image is transformed from RGB color space to brightness and chroma space (YCrCb). After transformation, the watermark is embedded into DWT coefficient of chroma component, CrCb. A secret key determines the locations in the image where the watermark is embedded. This process prevents possible pirates from removing the watermark easily. BPN will learn the characteristics of the color image, and then watermark is embedded and extracted by using the trained neural network. Experimental results show that the proposed method has good imperceptibility and high robustness to common image processing attacks.

1 Introduction

With the rapid development of computer and communication networks, the digital multimedia reproduction and distribution are becoming extremely easier and faster. However, these advances also afford unprecedented opportunities to pirate copyrighted digital multimedia products. As a result, the watermarking technique, which embeds a watermark into digital multimedia products for detecting and tracing copyright violations, has recently become a very active area of multimedia security [1]. The watermarking techniques can be classified into two classes depending on the domain of watermark embedding, i.e. a spatial domain [2] and a frequency domain [3], [4]. Kutter, M., *et al.* [2] presented to embed the watermark by modifying a selected set of pixels in the blue channel because the human eye is less sensitive to changes in this color channel. The blue channel, however, is the frailest to JPEG compression among the three-color channels, so the hided watermark information is easy to be lost. Mei R.-M., *et al.* [3] proposed a scheme of hiding watermark information in the low frequency coefficients of 8×8 discrete cosine transform (DCT) blocks of the red, green and blue components. Bami, M., *et al.* [4] proposed a scheme that the red, green and blue channels were transformed by full-frame DCT and selected the low frequency coefficients to hide watermark. Because the loss of the energy of the red and blue components is very high when the RGB true color image has been JPEG compressed, the watermark information embedded in the red and blue components is easy to be lost, and difficult to be extracted. Moreover, the DCT transform has blocking phenomena. Recently, quantization index modulation (QIM)[5], [6], [7], [8] technique is widely used in watermarking technique, and this technique is very robust to various attacks such as JPEG compression, SPIHT compression [9], noise insertion, low-pass filtering, resize and so on.

In this paper, a new watermark embedding/extracting algorithm using error back-propagation neural network (BPN) is introduced. DWT is used to overcome the problems in DCT, and quantization method is adopted to increase robustness against attacks. First, the original color image is transformed from RGB color space to YCrCb color space, where the luminance channel is defined as Y and the chrominance channel is defined as Cr and Cb. According to human visual system (HVS), luminance channel is more sensitive than chrominance channel. With this reason, watermark is embedded into the chrominance channel. To embed the watermark, 4-level DWT transformation is performed on Cr and Cb, and watermark is embedded into LL4 low frequency band. When embedding watermark, a secret key is used to determine the watermark embedding position, and after that, embed and extract the watermark by using the trained BPN.

The experimental results show that the watermarked image has at least 38dB in peak signal-to-noise ratio (PSNR). Also, the performance characteristics are compared with other algorithms and the proposed algorithm shows a good result.

2 Related Theories

2.1 Discrete Wavelet Transform (DWT)

With 2-D signals such as images, the DWT is typically applied in a separable fashion to each dimension. This may also be represented as a four-channel perfect reconstruction four-subband, as shown in Fig. 1. In this paper, the linear-phase 2/2 biorthogonal filters are selected, and for robustness, watermark is embedded into LL4 subband that is low frequency components.

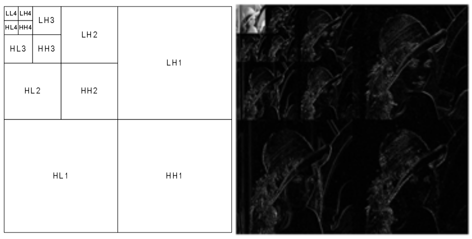


Fig. 1. Four-level DWT

2.2 The Error Back-Propagation Neural Network (BPN)

The BPN is a kind of supervised learning neural network. It is one of the most frequently used learning techniques in neural networks. The principle behind the BPN involves using the steepest gradient descent method to reach a small approximation. A general model of the BPN has an architecture like that depicted in Fig. 2. There are three layers including input layer, hidden layer, and output layer. Two nodes of each adjacent layer are directly connected to one another, which is called a link. Each link has a weighted value, which represents the relational degree between two nodes. A training process described by the following equations updates these weighted values:

$$\begin{aligned} net_j(t) &= \sum_i \alpha_{i,j} o_i(t) - \theta_j \\ o_j(t+1) &= f_{act}(net_j(t)) \end{aligned} \tag{1}$$

where $net_j(t)$is the activation value of the node j in iteration t , $o_j(t+1)$is output of the node j in iteration $t+1$, $f_{act}(x)$ is called the activation function of a node, which usually is a sigmoid function in hidden layers and a pureline function in output layer. Generally, all initial weight values $\alpha_{i,j}$ are assigned using random values. In each iteration process, all $\alpha_{i,j}$ are modified using the delta rule according to the learning data. The trained neural network can memorize the characteristics of the learning data, and predict a new output due to the adaptive capability of it. BPN will be used to learn the characteristics of image for improving the performance of the proposed watermarking scheme in the section 3.

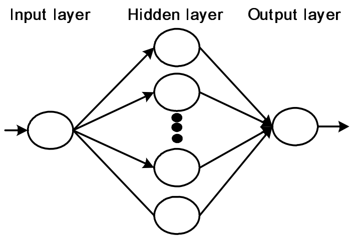


Fig. 2. Architecture of a BPN

2.3 The BPN Neural Network Training Procedures

The BPN neural network training procedures are as Fig.3 below.

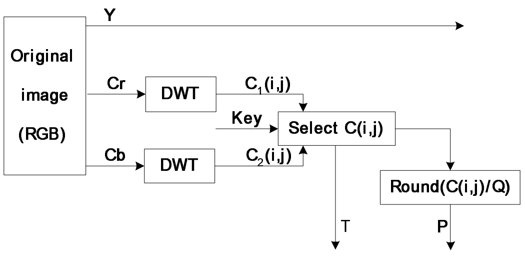


Fig. 3. BPN training procedures

In the figure, $C_1(i,j)$ and $C_2(i,j)$ is the LL4 coefficient when DWT transform is performed on *Cr* and *Cb* respectively. The coefficient *C(i,j)* is selected using random sequence with a special secret key. *Q* is the Quantization value. The output of the equation *Round(C(i,j)/Q),* which is represented as *P*, is used as an input value for the BPN, and *T*, which is the selected original value of *C(i,j)*, is used as a desired output value for the BPN neural network. The structure of BPN network presented in the paper is 1-65-1, which is obtained as an optimal neural network structure from many experiments. The hidden layer uses sigmoid function, and the output layer uses pureline function. The training method is Levenberg-Marquardt rule. The training error is set to 0.005 and the number of maximum learning iteration is set to be 5000. The training is finished when either training error is smaller than 0.005 or the iteration is reached to the maximum iteration number. Fig.4 shows the training error in each step when the BPN is trained using a 512×512 size Lena color image. The BPN trained by this method will be used to embed and extract the watermark.

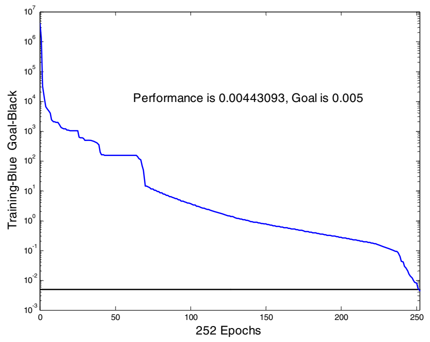


Fig. 4. The training process of BNP

3 Watermark Embedding and Extracting

Generally, a watermark algorithm includes 3 steps: watermark generation, embedding, and extraction. In this paper, a logo image, which can be easily distinguished by eyes, is embedded as a watermark. The proposed watermark technique is a kind of blind watermarking algorithm, which embed/extract the watermark into/from the DWT domain of chrominance component of the color image.

3.1 Watermark Embedding

Fig.5 below is the block diagram of the watermark embedding procedure. The trained BPN in the figure is discussed in section 2.3.

The embedding procedures are as follows:

Step1: Transform an original color image from RGB color space to YCrCb color space and perform the 4-level DWT transform on the chrominance components, *Cr* and *Cb*. In the figure above, $C_1(i,j)$ and $C_2(i,j)$ is the LL4 subband when DWT transform is performed on *Cr* and *Cb* respectively.

Step2: Select the position of watermark embedding coefficient *C(i,j)* (*$C_1(i,j)$* or *$C_2(i,j)$*) using random sequence with a secret key.

Step3: Quantize the DWT coefficient *C(i,j)* by *Q*, and use that value as the input of BPN then get the output *T'*.

Step4: Embed the watermark according to the equation 2 below which uses the output value *T'* and the *Q*.

$$C'(i,j) = \begin{cases} T' + \frac{1}{4} * Q & if \quad w_{m,n} = 1 \\ \\ T' - \frac{1}{4} * Q & if \quad w_{m,n} = 0 \end{cases} \tag{2}$$

where w is the watermark, and Q is a quantization value. A $\frac{1}{4} * Q$ should be given to each coefficient to provide the tolerance for $Round(C(i,j)/Q)$ not to be changed after watermark embedding. This also provides the tolerance for $Round(C(i,j)/Q)$ not to be changed when extracting watermark. The value, $\frac{1}{4} * Q$, is determined to be most robust against attacks.

Step5: To get *Cr'* and *Cb'* , perform the IDWT on *C_1'* and *C_2'* respectively where watermark is embedded. The transformation of YCr'Cb' color space into R'G'B' color space results in the watermarked image.

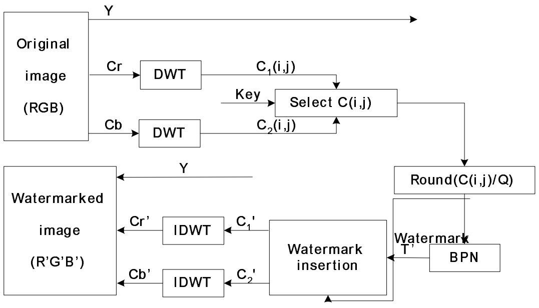


Fig. 5. Watermark embedding procedure

3.2 Watermark Extracting

The watermark extracting procedures are the converse procedures of watermark embedding, shown in Fig.6. The BPN here is trained neural network, which is discussed in section 2.3, and *Q* is the quantization value.

Step1: Transform the watermarked color image from R'G'B' color space to YCrCb color space and then perform a 4-level DWT transform on the chrominance components, *Cr* and *Cb*. In the figure above, $C_1(i,j)$ and $C_2(i,j)$ is the LL4 subband when DWT transform is performed on *Cr* and *Cb* respectively.

Step2: Select the position of coefficient $C(i,j)$ ($C_1(i,j)$ or $C_2(i,j)$), where watermark is embedded using random sequence with the same secret key, which is used in watermark embedding sequence.

Step3: Quantize the DWT coefficient $C(i,j)$ by Q, and use it as input value of the trained BPN to get the output T'.

Step4: Extract the watermark using the equation 3 below, using the output T' and coefficient $C(i,j)$.

$$w'_{m,n} = \begin{cases} 1 & if \quad C(i,j) > T' \\ 0 & else \end{cases} \tag{3}$$

Step5: The correlation between the original watermark and the extracted watermark is calculated to detect the existence of the watermark. The similarity between the original watermark *w* and the extracted watermark *w'* is quantitatively measured by the bit correlation ratio (BCR), defined as follows:

$$BCR = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\overline{(w_{i,j} \otimes w'_{i,j})}}{\sum_{i=1}^{m}\sum_{j=1}^{n}\overline{(w_{i,j} \otimes w_{i,j})}} \times 100\% \tag{4}$$

where $w_{i,j}$ is the original watermark bit, $w'_{i,j}$ is the extracted watermark bit, and $\otimes$ is the exclusive OR.

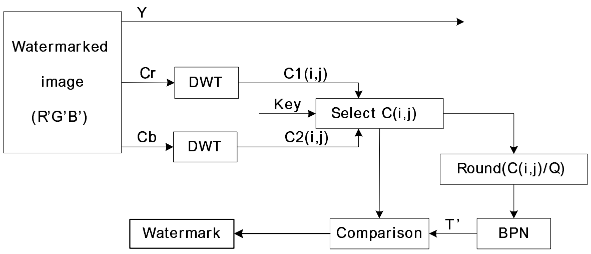


Fig. 6. Watermark extraction procedure

4 Experimental Results

In this paper, 24 bits in the RGB space 512×512×3 size Lena, baboon and Flowers color images are used as cover images, and a 32×32 size logo image, which can be easily distinguished by eyes, is used as watermark. Fig.7 shows the images used in this experiments and the watermark in use.

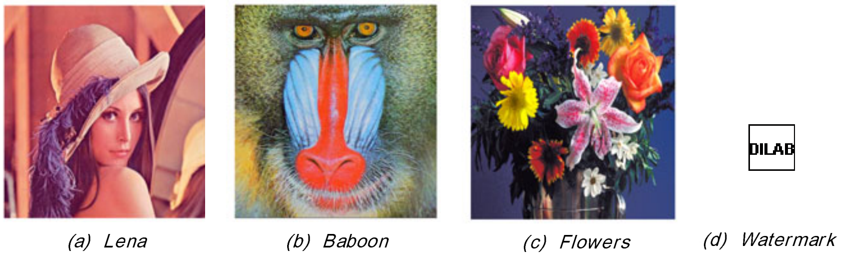


Fig. 7. Experiment images and watermark

The image quality metric is based on the PSNR. PSNR is defined as

$$PSNR = -10 * \log_{10} \frac{\frac{1}{m*n}\sum_{k=1}^{3}\sum_{n=0}^{n-1}\sum_{m=0}^{m-1}[f_k(m,n)-g_k(m,n)]^2}{3*255^2} \tag{5}$$

where *f* is the original image, *g* is the watermarked image, *m*n* is the image size.

Fig.8 shows the relationship between the embedding quantization step-size *Q* and PSNR. The PSNR of the watermarked image is decreasing with increasing *Q*, but the PSNR is bigger than 38dB in any case. This proves that the watermarked image has a good PSNR.

To test the robustness of the proposed method, experimental images are attacked by JPEG/SPIHT compressions, noise addition, low-pass filtering, and resize. The watermark was extracted from several JPEG compressed watermarked images with various compression quality factors and their corresponding BCR values are listed in Table 1. Table 1 shows that the extracted watermark is still recognizable when the compression quality factor reaches 30, which means the proposed method has good robustness to JPEG compression.

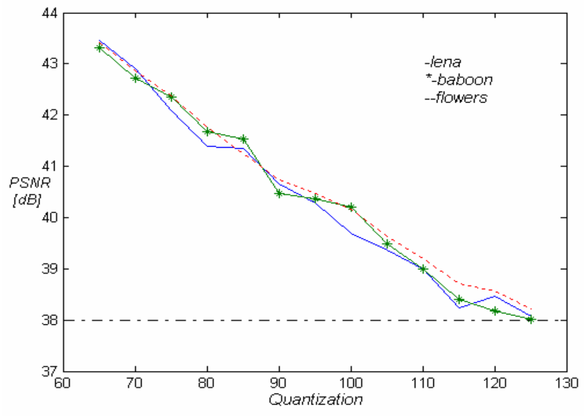


Fig. 8. The relationship between *Q* and PSNR

Finally, the performance comparison was performed between the proposed method and one proposed in QIM method. In QIM method, watermark is embedded into DWT coefficient of chroma component, CrCb. The two methods have the same test conditions including the same test color image "baboon", the same amount of embedding information (1024 bits i.e. a binary pattern watermark 32×32). Comparison results are summarized in Table 2. According to this table, the BCR values of the extracted watermarks by the proposed method are always higher than QIM method. These results prove that the proposed method has superior performance.

Table 1. BCR values of the extracted watermarks (%)

Image / JPEG Quality factor	Lena	Baboon	Flowers
	PSNR=38.46 [dB]	PSNR=39.02[dB]	PSNR= 38.57[dB]
100	99.95	99.93	98.80
90	99.54	99.70	96.79
80	98.92	99.51	96.43
70	97.26	99.31	96.19
60	98.73	98.72	94.83
50	96.28	98.41	94.24
40	95.31	95.24	94.23
30	89.81	93.62	91.05

Table 2. Comparison(BCR) results between QIM method and presented method

Method / Attacks	QIM method	The proposed method
	PSNR=40.07 [dB]	PSNR=40.21[dB]
No attack	100	99.98
JPEG (Quality factor=40)	91.53	93.02
SPIHT (1:60)	86.28	89.28
Noise (Gaussian)	85.84	87.32
Low-pass filtering	96.14	97.75
Resize (384×384)	89.49	91.60

5 Conclusions

A new watermarking method for color image has been proposed. The proposed method embeds a logo watermark into the DWT coefficient of chroma component CrCb of the color image. A secret key determines the locations in the image where the watermark is embedded. This process prevents possible pirates from easily removing the watermark. The embedding scheme has good quality of the watermarked image in terms of PSNR. This method also utilizes back-propagation neural networks (BPN), which is used to learn the characteristics of the original image. Due to the learning and adaptive capabilities of the BPN, the embedding/extracting strategy can greatly improve robustness to various attacks. Experimental results illustrate that the performance of the proposed technique is superior to other methods in the literature.

Acknowledgement

This work was supported by the ERC program of MOST/KOSEF (Next-generation Power Technology Center) and the Ubiquitous Autonomic Computing and Network Project, the Ministry of Information and Communication (MIC) 21st Century Frontier R&D Program in Korea.

References

1. Hartung, F., Kutter, M.: Multimedia watermarking techniques. Proceedings of the IEEE, (1999) 1079-1094
2. Kutter, M., Jordan, F., Bossen, F.: Digital Signature of Color Images Using Amplitude Modulation. In Proc. SPIE Electronic Imaging 97, Storage and Retrieval for image and video databases V, San Jose, CA, Feb, (1997) 518-526
3. Mei, R.-M., Lu, Z.-Q., Xie, W-X.: An adaptive watermarking algorithm for color images. Joumal of Shenzhen university (science & engineering), **18(3)**, (2001) 19-27
4. Bami, M., Bartolini, F., Piva, A.: Multichannel watermarking of color images. IEEE Transactions on circuits and systems for video technology, **12(3)**, (2002) 142-156
5. Chen, B., Wornell, G. W.: Quantization index modulation: a class of provably good methods for digital watermarking and information embedding. Information Theory, IEEE Transactions on, **47(4)**, (2001)1423 - 1443
6. Jang, Y., Kim, I., Kang, H., Kim, K., Han, S.: Blind watermarking Algorithm Using complex block selection method. Lecture Notes in computer science, **2195**, (2001) 996-1001
7. Chou, C.-H., Wu, T.-L.: Embedding color watermarks in color images. Multimedia Signal Processing, IEEE Fourth Workshop (2001) 327 - 332
8. Chou, C.-H., Liu, K.-C.: Color image watermarking based on a color visual model. Multimedia Signal Processing, IEEE Workshop (2002) 367 – 370
9. Said, A., Pearlman, W.A.: A new, fast, and efficient image codec based on set partitioning in hierarchical trees. Circuits and Systems for Video Technology, IEEE Transactions on, **6(3)**, (1996) 243 – 250

A Novel Blind Digital Watermark Algorithm Based on Neural Network and Chaotic Map

Pengcheng Wei[1,2], Wei Zhang[1,2], Huaqian Yang[1,2], and Degang Yang[2]

[1] Department of Computer and Modern Education Technology, Chongqing Education College, Chongqing, 400067, China
[2] Department of Computer Science and Engineering, Chongqing University, Chongqing, 400044, China
wpc75@163.com

Abstract. In order to enhance robustness and security of the embedded watermark, proposed a novel blind digital watermark algorithm based on neural network and chaotic map Firstly, a better chaotic sequence is generated by Cellular Neural Network (CNN) and Chebyschev map, using the chaotic sequence encrypted the watermark and its spectrum is spread. Then, BPN is trained to memorize the relationship among pixels of each sub-block image. Furthermore, the adaptive embedding algorithm is adopted to enhance the characters of the watermarking system. Simulation results are given which show that this scheme is practical, secure and robust.

Keywords: watermark, chaotic map, neural network.

1 Introduction

With the fast and steady development of network and multimedia technology, digital media grows also rapidly and achieves exciting success. However, how to protect the copyright of digital products has become a great challenge. Toward this goal, many techniques have been proposed in the last decade, in which digital watermarking is quite efficient and promising [1],[2],[3].

There are a number of desirable characteristics that a digital watermarking technique can offer, including security, imperceptibility, and robustness. Roughly speaking, the digital watermarking technology includes spatial-domain methods and transform-domain methods. Embedding the watermark into the transform-domain generally helps to increase the imperceptibility, security, and robustness. Therefore, this approach is very common, where DFT, DCT, DWT are three main transform methods used[4],[5].

Watermarking methods that do not require the original signal for watermark detection are called oblivious or blind methods [5]. In this paper, a blind robust watermarking scheme over the spatial domain is proposed. First, the watermark is encrypted and its spectrum is spread using a chaotic sequence. Then, the host image is divided distinct into 3×3 blocks, and a neural network is constructed to memorize the relationship among the pixels in a sub-block image. Finally, embed the masked binary watermark into the host image through changing the value of central pixel in each sub-block. The extraction is the inverse processing.

2 The Chaotic Watermarks Generating

2.1 Cellular Neural Network (CNN)

The researches have found that Cellular Neural Network (CNN) has a complex Dynamic behavior [6]. It is extensively applied in fields of pattern recognition, image processing and the solving partial differential equation. CNN is easy realized by using VLSI circuit because of its regular structure and each cell only coupled with adjacent cells.

Here, we select a three-order CNN dynamic model with full connection:

$$\frac{dx_j}{dt} = -x_j + a_j y_j + \sum_{k=1,k\neq j}^{3} a_{jk} y_k + \sum_{k=1}^{3} S_{jk} x_k + i_j \qquad j = 1,2,3 \tag{1}$$

Where x_j is a states variable, y_j is a cell output. y_j is related to x_j and defined as

$$y_j = 0.5(|x_j + 1| - |x_j - 1|) \quad j = 1,2,3 \; . \tag{2}$$

Let three cell-template parameters are [7] :

$$a_{12} = a_{13} = a_2 = a_{23} = a_{32} = a_3 = a_{21} = a_{31} = 0; .$$

$$S_{13} = S_{31} = S_{22} = 0 \; ; i_1 = i_2 = i_3 = 0 \; ; S_{21} = S_{23} = 1 \quad .$$

Then, model (1) is abbreviated as

$$\begin{cases} \dfrac{dx_1}{dt} = -x_1 + a_1 y_1 + S_{11} x_1 + S_{12} x_2 \\ \dfrac{dx_2}{dt} = -x_2 + x_1 + x_3 \\ \dfrac{dx_3}{dt} = -x_3 + S_{32} x_2 + x_3 \end{cases} \quad . \tag{3}$$

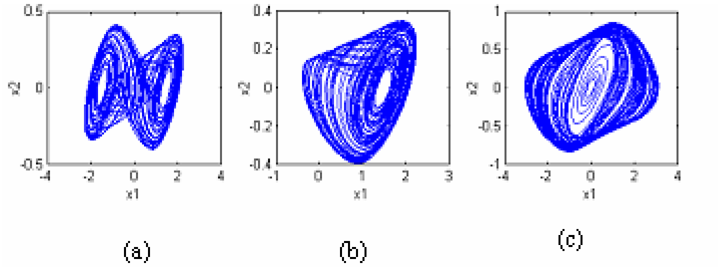


Fig. 1. Different chaotic attractors of 3-order CNN model. (a) a1 = 3.86, s11 = -1.55, s12 = 8.98, s32 = -14.25; (b) a1 = 3.85, s11 = -1.55, s12 = 8.76, s32 = -14.35; (c) a1 = -4.198, s11 = 2.365, s12 = 7.45, s32 = -10.98.

In Fig.1., we can obtain chaotic attractor with different property by assigning different values to parameters a_1 , S_{11} , S_{12} and S_{32} . These chaotic attractors can be generated in a large scale, the complexity of output sequence in subsequent encryption system is enhanced, and the systematic security is evidently enhanced as well.

2.2 Chebyschev Map

A particularly interesting candidate for chaotic sequences generators is the family of Chebyschev polynomials, whose chaoticity can be verified easily with many other properties are accessible to rigorous mathematical analysis. The independent binary sequences generated by a chaotic Chebyshev map [8],[9] were shown to be not significantly different from random binary sequences. For this reason, a kth-order Chebyshev map [9] is employed for the design of the intended image encryption scheme. This map is defined by

$$f(x_{n+1}) = \cos(k \arccos(x_n)), -1 \le x_n \le 1, n = 1,2,3\cdots \ . \tag{4}$$

Here, the map is chaotic for $k \ge 2$ and we use $k = 4$ in this study. Fig.3. shows two time series of this map, with initial values differed only by 10^{-5} ; indicating that the map can generate good chaotic (pseudorandom) sequences satisfying a basic requirement of a cryptosystem that demands such randomness. Fig. 2. further shows its statistical correlation curves.

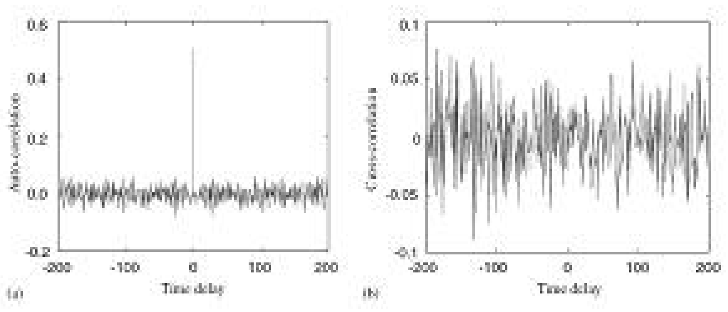


Fig. 2. The statistical correlation curves of a chaotic Chebyshev sequence: (a) Auto-correlation curve of the chaotic sequence when the initial value of 0.60000; (b) Cross-correlation curve of two chaotic sequence when their initial values are 0.60000 and 0.60001, respectively

2.3 Chaotic Sequence Generated

In order to improve the complexity and the period of the chaotic Sequence under the finite-precision circumstances, the chaotic sequence is generated combining three-order CNN and Chebyshev map, Fig. 4. is the structure of chaotic sequence generator.

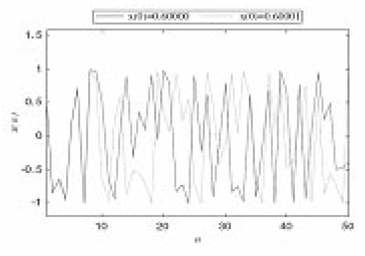


Fig. 3. Two time series of the Chebyshev

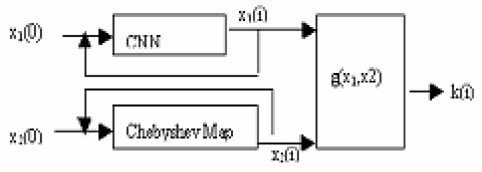


Fig. 4. Chaotic sequence generator with slightly different initial values

In Fig. 4., the chaotic sequence is defined as:

$$k(i) = g(x_1(i), x_2(i)) = \begin{cases} 1\ , & x_1(i) > x_2(i) \\ NULL\ , & x_1(i) = x_2(i) \\ 0\ , & x_1(i) < x_2(i) \end{cases} . \tag{5}$$

We can obtain a binary sequence $\{\theta_t(f^n(x))\}_{n=0}^{\infty}$ from Eq.(5).

We use the binary chaotic sequence $\theta_t(x)$ to mask the watermark image [10], and then using Eq.(6) creates the masked watermarking bits

$$w' = w \oplus \theta_t(x) \quad . \tag{6}$$

3 Neural Network Training

We establish the relationship among the contiguous pixels in a sub-block image using the BPN model [11]. For a selected pixel $I_{i,j}$, the network is trained with its 3×3 neighbors , i.e. let $I_{i-1,j-1}$, $I_{i-1,j}$, $I_{i-1,j+1}$, $I_{i,j-1}$, $I_{i,j+1}$, $I_{i+1,j-1}$, $I_{i+1,j}$, $I_{i-1,j+1}$ as input vector and the value of pixel $I_{i,j}$ as output value. We construct three layer BPN with 8, 10 and 1 neurons in the input, hidden and output layer respectively, and the sigmoid, pure-linear function are used for recognition. It states that there is tight correlation among the pixels in a sub-block image and the BPN can approach the relationship greatly.

4 Decision of Watermarking Strength

In [12], describes several methods of deciding watermarking strength based on human visual system and neural network. To achieve maximal watermarking while remaining invisible to the human eyes, we using a adaptive approach based on signal-noise-ratio(SNR) [13]:

$$SNR = 10\log_{10} \frac{\sum_x \sum_y p_{x,y}^2}{\sum_x \sum_y (p_{x,y} - p'_{x,y})^2} \ . \tag{7}$$

Where $p_{x,y}$, $p'_{x,y}$ is the pixel value of original image and watermarked image, respectively. For a given image and the value of SNR, we can obtain the strength of superposition correspondingly.

Due to the fact of difference of pixel value, there will be visible for human eyes in some square when a fixed embedding strength on the whole image is used, but in some other square the embedding strength will be too small to preserve the robust properties. So, in our scheme, the original image is divided into 3×3 blocks, and for each sub-block the strength of embedding is calculated as follows:

$$\delta = \left| p_{x,y} - p'_{x,y} \right| = \sqrt{\sum_x \sum_y p_{x,y}^2 \times 10^{\frac{-SNR}{10}}} \ . \tag{8}$$

5 The Proposed Algorithm

5.1 Watermark Embedding

Step 1. Encrypting the original watermark with chaotic sequence as mentioned in Section 2.

Step 2. Establish and training the BPN model for the host image as mentioned in Section 3.

Step 3. Embedding the watermark: if the watermark $w_{i,j}$ is corresponding to the pixel $I_{i,j}$, changing the value of $I_{i,j}$ to $I'_{i,j}$, such that $I'_{i,j} > B_{i,j}$ or $I'_{i,j} > I_{i,j+\delta}$, where $B_{i,j}$ is the output of BPN in sub-block $I_{i,j}$ and δ is defined in Eq.(9) and Eq.(10). Otherwise, for embedding $w_{i,j}=0$, let the $I'_{i,j} < B_{i,j}$ or $I'_{i,j} < I_{i,j-\delta}$, i.e.:

$$I'_{i,j} := \begin{cases} \max\{I_{i,j}, \sigma_1 + \delta\} & \text{if} \quad w_{i,j} = 1 \\ \min\{I_{i,j}, \sigma_0 - \delta\} & \text{if} \quad w_{i,j} = 0 \end{cases} . \tag{9}$$

Where

$$\sigma_1 = \begin{cases} I_{i,j} & \text{if} \quad B_{i,j} - I_{i,j} > \delta \\ B_{i,j} & \text{if} \quad B_{i,j} - I_{i,j} \le \delta \end{cases} . \tag{10}$$

5.2 Watermark Extracting

The watermark extracting is inverse process of embedding. According to the model of BPN and the secret key, the masked watermark can be retrieved as follows:

$$\tilde{w} = \begin{cases} 1 & \text{if} \quad I'_{i,j} > B'_{i,j} \\ 0 & \text{otherwise} \end{cases} . \tag{11}$$

Where $I'_{i,j}$, $B'_{i,j}$ is the watermarked image and the output of BPN. Performing as Eq.(9), the original watermark w' can be obtained approximately.

6 Simulation Results

In order to obtain a quantitative measure to the watermarked results, some measures are defined as follows. In this paper, the similarity between detected watermark and embedded watermark are used as the standard, it is defined as:

$$sim(W^*, W) = \frac{\sum_i \sum_j W(i,j) \times W^*(i,j)}{\sum_{i,j} W^*(i,j)^2} . \qquad (12)$$

Where W is the original watermark, W^* is the extracted watermark. If $W = W^*$, then sim=1.The experimental results are shown as Fig. 5.

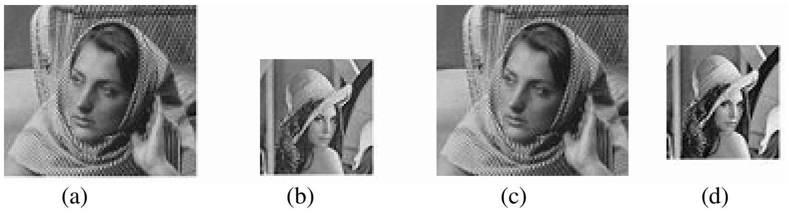


Fig. 5. (a) Original image; (b) Original watermark; (c) Watermarked image; (d) Detected watermark from (c)

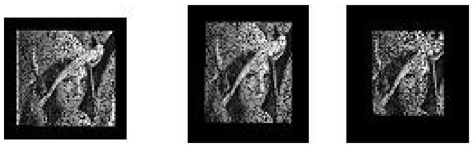


Fig. 6. Cropped for Fig.5(c) (32pixels; 64pixels; 128pixels)

Table 1. Detector watermark from cropped image

Cropped length	40	350	250
Value of sim	0.867	0.746	0.608

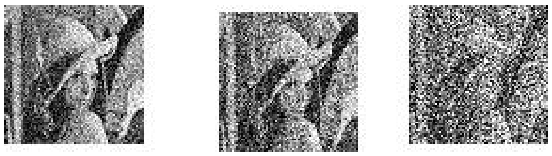


Fig. 7. Detected watermark from Fig. 5(c). added Gaussian noise(variance are2,4,8)

Table 2. Detected watermark from image added Gaussian noise

Gaussian noise	2	4	8
Value of sim	0.852	0.686	0.611

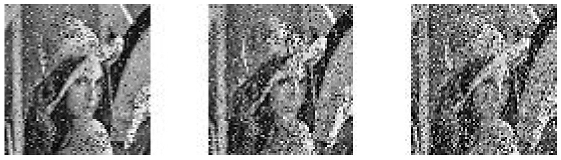


Fig. 8. Detected watermark from Fig.5(c). compressed with JPEG(80/70/60)

Table 3. Detected watermark from image compressed with JPEG

Compression ratio	80	70	60
Value of sim	0.867	0.685	0.568

7 Conclusion

A blind watermarking scheme based on neural network and chaotic map has proposed in this paper. Due to the neural network memorizing the characteristics of the relations between the original image and the watermarked image, our scheme doesn't require that the original image and watermark in the extraction process. From value of $sim(x)$ we have discussed above, the scheme is strongly resistant to attack such as low-pass filter, noise and JPEG compression.

References

1. Barni, M., Bartolini, F.: Improved Wavelet-based Watermarking through Pixel Wise Masking. IEEE Trans. Image Processing (2001) 783-791
2. Kuribayash, M., Tanaka, H.: A New Digital Watermarking Scheme Applying Locally Wavelet Transform. IEEE Trans. Fundam (2001) 2500-2507
3. Schmitz, R.: Use of Chaotic Dynamical Systems in Cryptography. J. Franklin Inst . (2001) 429-441
4. Lin, C. Y., Wu, M., Bloom, J. I.: Rotation, Scale, and Translation Resilient Watermarking for Images. IEEE Trans. Image Processing , Vol. 10 (2001) 767–782
5. Wang, Y., Doherty, J., Van Dyck, R.: A Wavelet-based Watermarking Algorithm for Ownership Verification of Digital Images. IEEE Trans.Image Processing, Vol.11 (2002) 77–88
6. Chua, L. O., Yang, T.: Cellular Neural Network: Theory. IEEE Trans. Circuits System, Vol. 35 (1988) 1257-1272
7. He, Z.Y., Zhang, Y,F., Lu, H.T.: The Dynamic Character of Cellular Neural Network with Application to Secure Communication. Journal of China Institute of Communications, Vol. 20 (1999) 59-67

8. Kohda, T., Tsuneda, A.: Statistics of Chaotic Binary Sequences. IEEE Trans. Information Theory (1997) 104-112
9. Kohda, T., Tsuneda, A.: Pseudonoise Sequences by Chaotic Nonlinear and their Correlation Properties. IEICE Trans. on Communications (1993) 855-862
10. Tefas, A., Nikolaidis, A., Nikolaidis, N., Solachidis V., Tsekeridou, S., Pitas, I. : Performance Analysis of Correlation-Based Watermarking Schemes Employing Markov Chaotic Sequences. IEEE Trans. Signal Processing Vol. 7 (2003) 1979-1994
11. Fukushhima, K., Miyake, S., Ito, T.: Neocognitron: A Neural-network Model for a Mechanism of Visual Pattern Recognition. J. A.Anderson and E. Rosenfeld Eds. Cambridge, M. A. , MIT press (1988)
12. Podilchuk,C. I., Zeng, W. J.: Image-adaptive Watermarking Using Visual Model. IEEE Journal on Selected Areas in Communications , Vol.4 (1998) 525-539

Stimulus Related Data Analysis by Structured Neural Networks

Bernd Brückner

Leibniz Institute for Neurobiology, Magdeburg, Germany
brueckner@ifn-magdeburg.de

Abstract. In the analysis of biological data artificial neural networks are a useful alternative to conventional statistical methods. Because of its advantage in analyzing time courses the Multilevel Hypermap Architecture (MHA) is used for analysis of stimulus related data, exemplified by fMRI studies with auditory stimuli. Results from investigations with the MHA show an improvement of discrimination in comparison to statistical methods. With an interface to the well known BrainVoyager software and with a GUI for MATLAB an easy usability of the MHA and a good visualization of the results is possible.

The MHA is an extension of the Hypermap introduced by Kohonen. By means of the MHA it is possible to analyze structured or hierarchical data (data with priorities, data with context, time series, data with varying exactness), which is difficult or impossible to do with known self-organizing maps so far.

1 Introduction

The analysis of stimulus related data is a common investigation method in neuroscience. Any improvement of analytical methods can be helpful for better understanding the complexity of the brain. In this sense also alternatives to common statistical methods, like the proposed structured neural networks, should be investigated, especially their usefulness in respect to the stimulus related data analysis.

One approach to explore stimulus related functional maps in the brain is the use of functional Magnetic Resonance Imaging (fMRI). Besides statistical methods ([1]) recently artificial neural networks are used for analysis of fMRI data sets ([2,3,4,5]). For our experiments with auditory stimuli the following structured neural network has to be introduced. The Multilevel Hypermap Architecture (MHA [6]) is classified under self-organizing neural networks and is an extension of the Hypermap introduced by Kohonen. Instead of the two levels in the Hypermap, the data and context level, the MHA supports multi-level data vectors. Structured or hierarchical data can be analyzed with the MHA, that is:

- data with priorities, e.g. representation of hierarchical data structures in databases
- data with context (databases, associative memories)
- time series, e.g. language or image scenes

Support for both the classification of data and the projection of the structure in a common map is a benefit of MHA. This results in a hierarchy with redundancy, as in biological systems. An overview of our last works about MHA gives [7].

One type of Learning Vector Quantization (LVQ) is the Hypermap principle introduced by Kohonen [8]. This principle can be applied to both LVQ and SOM algorithms. In the Hypermap the input pattern is recognized in several separate phases: the recognition of the context around the pattern to select a subset of nodes (cluster) is followed by a restricted recognition in this subset. This architecture speeds up searching in very large maps and may carry out stabilizing effects, especially if different inputs have very different dynamic ranges and time constants [9]. One advantage of the MHA is the storage of hierarchical relationships of data.

The modification and extension of the Hypermap, the Multilevel Hypermap Architecture (MHA), are described in [6,10,11].

In addition to older experiments ([12]) a new training strategy for the MHA is introduced. The BrainVoyager software ([13]) is used for a better visualization of the results.

2 Materials and Methods

Even it is already published ([6,10,11]), it will be useful for understanding, to explain the learning algorithm of the MHA in more detail, before the fMRI environment is explained.

2.1 The Multilevel Hypermap Architecture

The system model of the MHA is shown in Fig. 1. Instead of two levels proposed in the Hypermap [8,14], the data and the context level, the MHA supports several levels of data relationship and therefore the input vector consists also of an arbitrary number of levels. In the MHA there is the same number of levels in the weight vector of each unit and these levels are related to the corresponding levels of the input vector. A varying number of levels for the units of the map is supported.

The MHA is trained with the different levels of the input vector, whose representation is a hierarchy of encapsulated subsets of units, the so called clusters and subclusters, which define different generalized stages of classification.

In order to make the learning algorithm and the classification behavior of the MHA more clear, the following simple example should be used. Suppose stimulus related parts of an input vector have a significant structure, this example is also a demonstration of the ability of the MHA to deal with the multiple presence of such parts in the input vector and to project their relationship in its hierarchical structure.

In our example the MHA has a dimension of 50 by 50 units and 3 levels. The dimension of the map is oversized to better demonstrate the discrimination properties of the MHA. The 3 levels should be used for a classification of triples of the binary digits 0 and L to illustrate the ability of hierarchical clustering. Therefore the input vector has 3 parts, in which the binary digits are realized as ramp signals to be useful for the LVQ based learning algorithm.

Suppose the values in one level of the input vector $y_i = f(i)$, which represent one binary digit, are $y_i = c_0 i$ for the digit L and $y_i = c_0(N - i)$ for the digit 0, whereby $i = 0, .., N$ and $c_0 > 0$ constant. Therefore in each level are $N + 1$ values which form

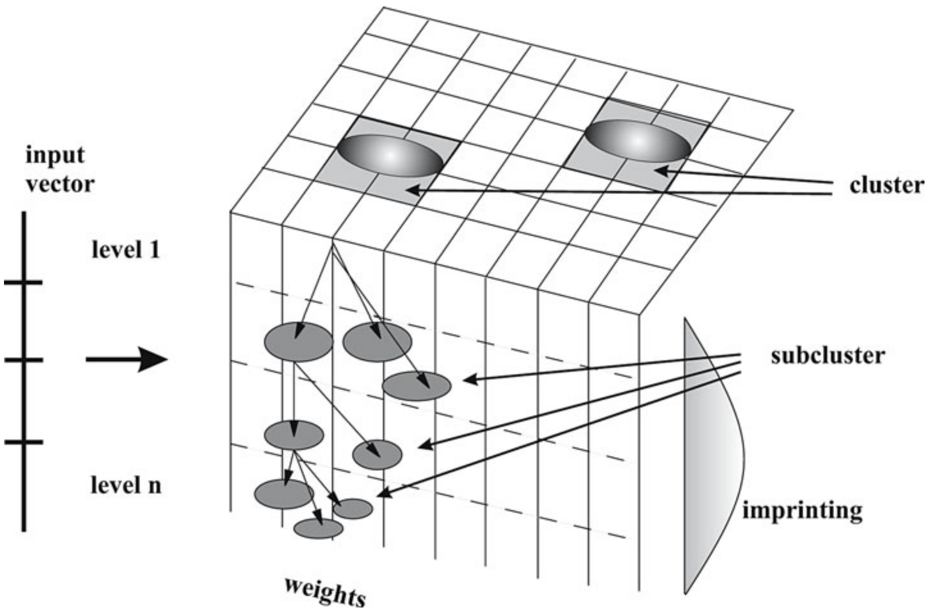


Fig. 1. Multilevel Hypermap Architecture

ramp signals as illustrated by altered colours in Fig. 2, increasingly for the binary digit L and decreasingly for 0.

After beginning of training, on the first level all input vectors with the digit L and all with 0 are separated in one cluster each, that means, on the first level of the MHA only two clusters are formed. When the training process reaches the next level, this separation procedure will continue for each cluster. Therefore, there are two subclusters in each cluster related to the digits L and 0 respectively.

The formation of only two subclusters for each 2. level subcluster takes also place on the 3. level of the MHA in the training process. The result will be a hierarchical clustering of the input data. A computational result of the classification by the MHA learning algorithm for our sample data is shown in Fig. 2.

To achieve the hierarchical relationship of clusters and subclusters in that way the so called imprinting coefficient is crucial. Without imprinting there will be only independent data relationships in respect to the levels of the MHA and no hierarchical structure. In the case of our example this would mean that only two clusters on each level of the MHA are found and the lower subclusters in the hierarchy will have no topological relationship to the upper ones.

On the other hand this feature can be useful to train a multiple set of unrelated data in one training process, each data set related to one level of the MHA.

2.2 The Learning Algorithm

The learning algorithm of the MHA as known from literature [6,10,11] is as follows.

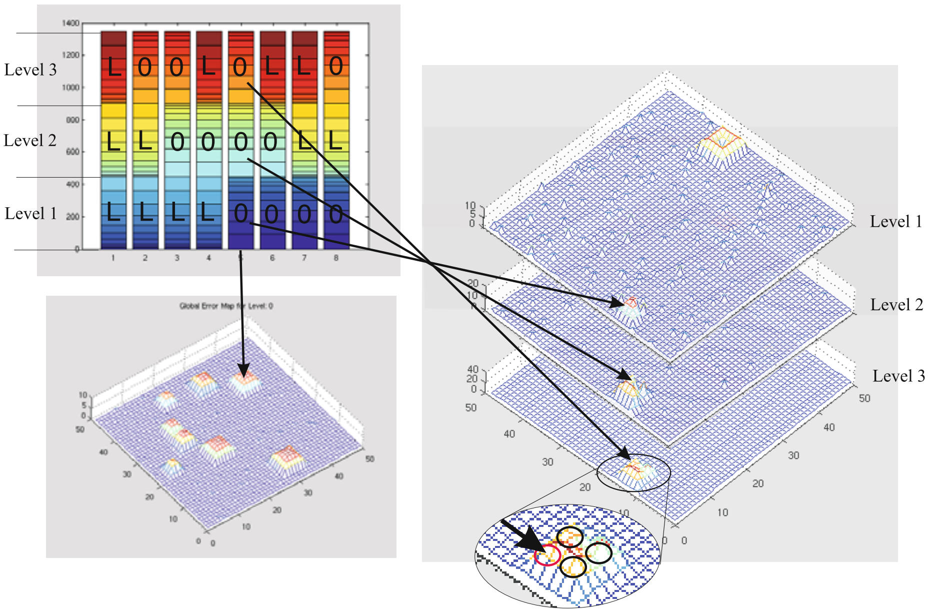


Fig. 2. Classification of a simple binary data set. Top left the input data represented by 8 vectors with 3 levels each is shown. The binary content of each level is coded by 2 different colour sets (ramp signal). The binary input triples are classified by the MHA in a hierarchical order (right). On the first level there are only two best matching nodes for the digits 0 and L respectively. All vectors starting with the same digit are kept in a cluster, as shown for the 4 vectors starting with 0 on the third level. In the error surface the triple 000 is indicated for illustration. In contrast to the 3-level MHA, a training without levels results in a map, where each vector is in a separate cluster (see error map on the left).

Let the input vector of one level l_j be $\mathbf{x}_{l_j}$ and one processing unit $\mathbf{m}_{i,l_j}$ then, in a first phase, one has to find a first level with a subset S_j of nodes (cluster) for which

$$\|\mathbf{x}_{l_j} - \mathbf{m}_{i,l_j}\| \leq \delta_j, \tag{1}$$

with δ_j being the threshold of that level. Then it is necessary to find the best match $\mathbf{m}_c$ for all nodes in the subset and to adapt the weights accordingly.

In the normal case (input learning) the adaptation of the weights is done by

$$\mathbf{m}_{c,l_j}(t+1) = \mathbf{m}_{c,l_j}(t) + \alpha'(l_j)\alpha(t)[\mathbf{x}_{l_j}(t) - \mathbf{m}_{c,l_j}(t)], \tag{2}$$

where

$$c = arg\min_i\{\|\mathbf{x}_{l_j} - \mathbf{m}_{i,l_j}\|\}, \tag{3}$$

$$\alpha'(l_j) = e^{-\|l_i - l_j\|},\ the\ \text{"}imprinting\text{"}\ coefficient, \tag{4}$$

and

$$\alpha(t) = c_0 e^{-D(t)},\ D(t)\ distance\ function \tag{5}$$

The sizes of the thresholds δ_j should be decreased according to the order of the levels to obtain encapsulated subsets S_j (subcluster). This behavior is mainly supported by the "imprinting" coefficient $\alpha'(l_j)$. Therefore the "imprinting" coefficient is responsible for the topological order of the subclusters in the MHA. Classification is achieved by finding the best matching node for each level of the hierarchy and by determining the square mean error of matching. In principle the algorithm handles different numbers of levels in the input vector.

To give more variability to the training data it is possible to mask parts of the input vector. The masked parts are ignored in the above algorithm and therefore don't influence the result of the learning process. To hold the learning algorithm for performance reasons as easy as possible all negative values of $\mathbf{x}_{l_j}(t)$ in (2) are interpreted as masked values, i.e. only positive values of $\mathbf{x}_{l_j}(t)$ are processed in that adaptation step of the learning algorithm.

Important for the occurrence of the hierarchical structure of clusters and subclusters in that way is the so called imprinting coefficient (4). Without imprinting on the lower level(s) the hierarchical relationship would disappear and there will be independent data relationships on all levels of the MHA. On the other hand this feature can be useful to train a multiple set of unrelated data in one training process; each data set related to one level of the MHA. In contrast to the MHA it is impossible to detect the hierarchical order using a conventional SOM or LVQ network. The classification of data with hierarchies is of course the main advantage of the MHA learning algorithm.

2.3 fMRI Data Analysis

In the auditory cortex of awake animals and humans responses to the same repetitive auditory stimulus will strongly habituate, dishabituate and change with general alertness, context, cognitive performances and learning. These non-stationarities are further complicated by the fact that the representation of a given stimulus in an auditory cortex field is not topographically stable over time. Several different acoustic stimuli (potpourri of various sounds, series of tones with shifting frequency, tone pairs with different frequencies) were used for the experiments with normal-hearing subjects. Subjects were scanned in a Bruker Biospec 3T/60 cm system. For principal approach and details of these experiments see also [15], [16].

Because the MHA supports several levels of data relationship and a hierarchical unsupervised clustering it is an ideal candidate for the analysis of stimulus related data, and so also for fMRI time series. The data was preprocessed with the well known BrainVoyager software [13] and VOI's (volume of interest) were defined. Only the data of these VOI's, which represent the results of the usual statistical methods, was processed by the MHA. In a first step of our analysis of acoustic stimulated fMRI data the MHA was trained to learn the stimulus structure. With this pre-trained Hypermap the learning of the VOI-based data was continued in order to build hierarchical clusters of periodically similar data. Finally we compared our results with statistical tests (Pearson's cross-correlation).

3 Main Results

As was expected and known from older experiments ([12]), the classification of the fMRI data with the MHA shows similar results in comparison to the statistical tests. But in average we get an improvement of discrimination of about 15 percent. Especially by masking the stimulus relevant regions in time course the improvement can be achieved. Furthermore it is necessary to eliminate any artifacts and make a normalization (baseline, amplitude) of the data before processing by the MHA.

In Fig. 3 results from tests with fMRI data are shown. In these tests both the number of levels of the MHA and the masked parts of the input vector were varied. In our investigations best results were realized with a 2-level MHA (the second level is marked by a green rectangle in Fig. 3) and by masking all not stimulus related regions in the input vector (marked by red rectangles in Fig. 3).

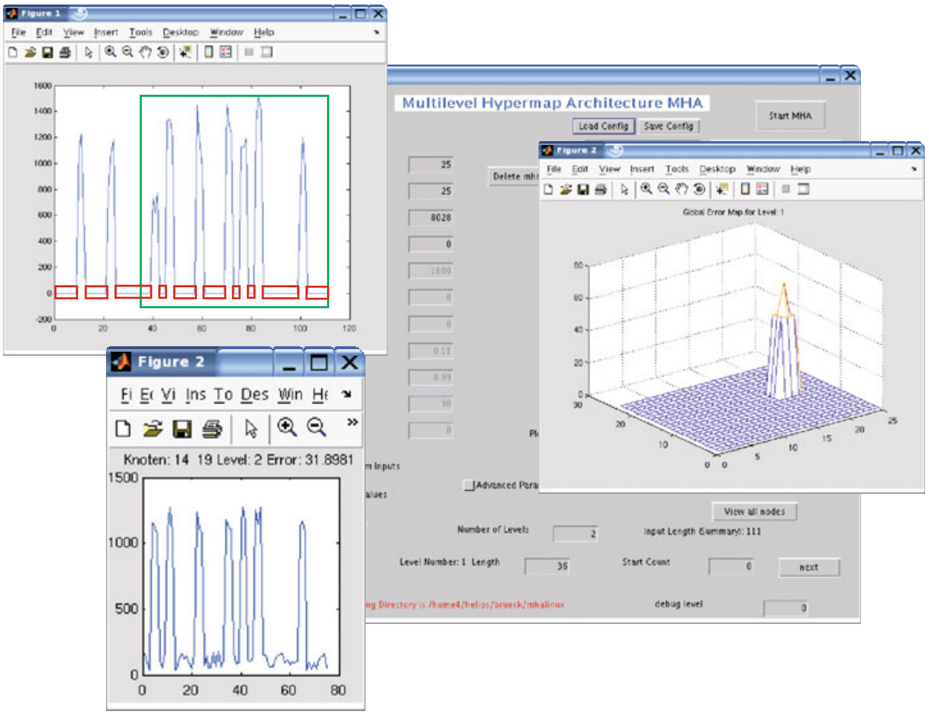


Figure 1
File Edit View Insert Tools Desktop Window Help
Multilevel Hypermap Architecture MHA
Load Config
Save Config
Start MHA
Delete mh
Figure 2
File Edit View Insert Tools Desktop Window Help
Global Error Map for Level: 1
Figure 2
Fi E(Vi Ins To Des Win He
m Inputs
Advanced Para
alues
View all nodes
next

Knoten: 14 19 Level: 2 Error: 31.8981

Number of Levels 2
Input Length Summary: 111
Level Number: 1 Length 36
Start Count 0
debug level 0
ng Directory is /home4/helios/brueck/mhalinux

Fig. 3. Results from fMRI data analysis with MHA. Top left window shows one of selected input vectors. Red rectangles mark parts of the input vector which are masked, the green rectangle is the part related to the level 2 of the MHA. The other left window shows the pre-trained values of the MHA for that input vector and that level respectively. On the right the global error map of the MHA after pre-training, in the background parts of the GUI for MATLAB are shown.

Even if not comparable directly because of the absence of a common evaluation criterion, it is necessary to find a procedure for comparing the results of the used statistical methods with the MHA. This is done by creating a VOI from the results of the statistical test, which represents the part of the brain where the activity correlates with the stimulus. With the data of this VOI the MHA is processed and the results from this process are transfered into another VOI. Comparing the mean time courses of both VOI's leads to the already stated result of an improvement of discrimination and a better selectivity of the MHA method in contrast to our statistical tests. (see Fig. 4).

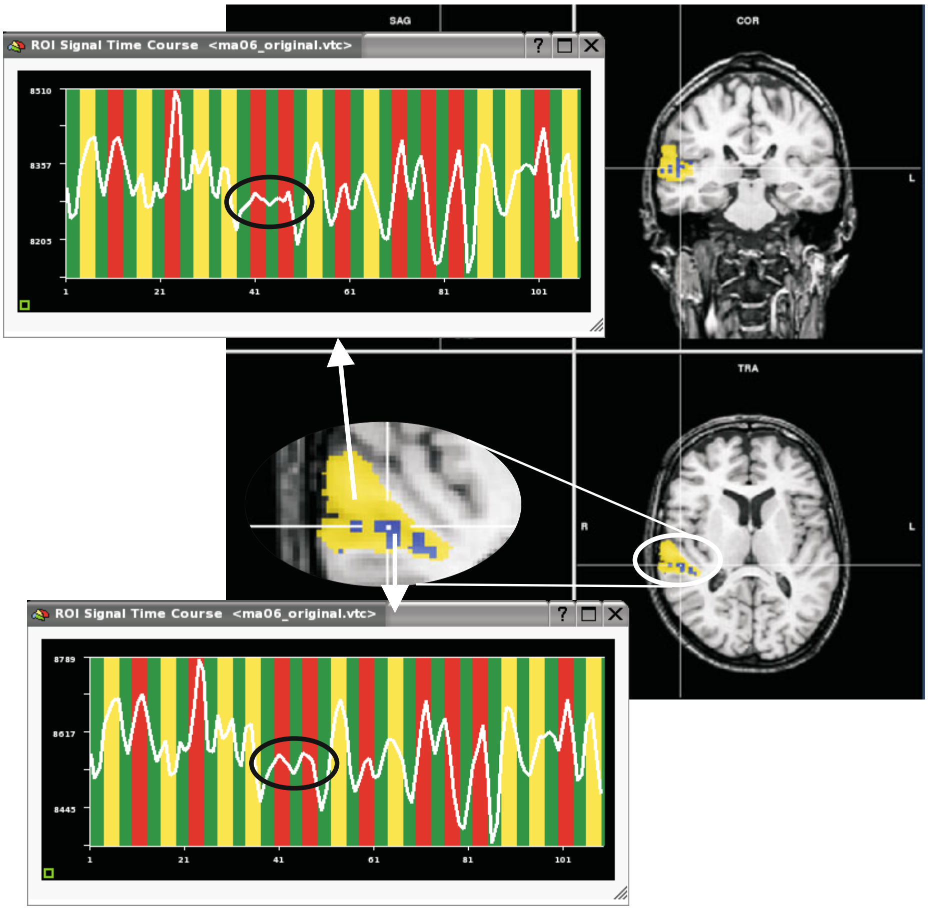

ROI Signal Time Course <ma06_original.vtc>
8510
8357
8205
1
21
41
61
81
101
SAG
COR
TRA
R
L
ROI Signal Time Course <ma06_original.vtc>
8789
8617
8445
1
21
41
61
81
101

Fig. 4. Presenting results in BrainVoyager. VOI (in blue) represents stimulus corresponding voxels from MHA data analysis inside auditory cortex region T3 (VOI in yellow) of right hemisphere. VOI signal courses are shown for both VOI's, top (T3) and down (MHA) respectively, whereby stimulus signals are in red regions. The black circles mark one of the regions, where the improvement belonging to the stimulus discrimination is visible.

With the implementation of the MHA algorithm in MATLAB and the integration of an interface to BrainVoyager it is possible to visualize the classified data in VOI's. Because the stimuli were presented periodically and these periods can be built up in the multi-level structure of the MHA, the non-stationarities in these periods were detected in the hierarchy of clusters found in the MHA after training.

4 Conclusions

The results of our analysis of fMRI data sets by means of the MHA show, that it is possible to analyze such periodically structured and hierarchical data, what is difficult or impossible to do with other known self-organizing maps so far. Furthermore is the MHA an useful complement to conventional statistical methods in this field. Because of its character like a simultaneous auto-correlation and cross-correlation to the stimulus it has a higher selectivity and discrimination than statistical methods. Even it seems to be comparable in results, the MHA has in addition to standard ICA algorithms, which, by definition, are not able to estimate statistically dependent sources ([17]), the power (e.g. by parameterization, masking of data, a-priori-knowledge) to project these dependencies in its hierarchical structure. The combination of masking the input vector with the projection of the input vector to the hierarchical level based structure of the MHA seems to be a powerful tool for analyzing stimulus related data, which was not proved so far and which is actually the novelty contribution of the presented work. Therefore the MHA should be also a preferred tool for the analysis of other kinds of stimulus related data, like event related potentials (ERP).

One advantage of the MHA is the support for both, the classification of data and the projection of the structure in one unified map. The resulting hierarchy has some redundancy like in biological systems. In the previous years some real world applications using the MHA were reported in the literature. Beside our fMRI investigations a system for speech processing and recognition [18] and an application which deals with an implementation of the Modified Hypermap Architecture for classification of image objects within moving scenes [19] are carried out.

It should be pointed out that the aim of the investigations was to test the MHA for that kind of data in principle. To find new cortical mechanisms and relevant neuronal structures with fMRI by using structured neural networks like MHA will belong to further investigations.

Acknowledgment. I like to thank colleagues from special lab non-invasive brain imaging of the Leibniz Institute for Neurobiology Magdeburg, especially André Brechmann, for their helpful hints. The development of the MHA was partly supported by LSA grant 1441B/0083.

References

1. Smith, S.M.: Overview of fmri analysis. Br J Radiol **77 Spec No 2** (2004) S167–S175
2. Ngan, S.C., Yacoub, E.S., Auffermann, W.F., Hu, X.: Node merging in Kohonen's self-organizing mapping of fMRI data. Artif Intell Med **25**(1) (2002) 19–33
3. Ylipaavalniemi, J., Vigario, R.: Analysis of auditory fmri recordings via ica: A study on consistency. In: Proceedings of the 2004 International Joint Conference on Neural Networks (IJCNN 2004). Volume volume 1., Budapest, Hungary (2004) 249254

4. Meyer-Baese, A., Wismueller, A., Lange, O.: Comparison of two exploratory data analysis methods for fMRI: unsupervised clustering versus independent component analysis. IEEE Trans Inf Technol Biomed **8**(3) (2004) 387–398
5. Voultsidou, M., Dodel, S., Herrmann, J.M.: Neural networks approach to clustering of activity in fMRI data. IEEE Trans Med Imaging **24**(8) (2005) 987–996
6. Brückner, B., Franz, M., Richter, A.: A modified hypermap architecture for classification of biological signals. In Aleksander, I., Taylor, J., eds.: Artificial Neural Networks,2, Amsterdam, Elsevier Science Publishers (1992) 1167–1170
7. Brückner, B., Wesarg, T.: Modeling speech processing and recognition in the auditory system using the multilevel hypermap architecture. In Seiffert, U., Jain, L., eds.: Self-Organizing Neural Networks. Recent Advances and Applications. Volume 78 of Springer Series on Studies in Fuzziness and Soft Computing., Heidelberg, Springer-Verlag (2001) 145–164
8. Kohonen, T.: The hypermap architecture. In Kohonen, T., Mäkisara, K., Simula, O., Kangas, J., eds.: Artificial Neural Networks, Helsinki, Elsevier Science Publishers (1991) 1357–1360
9. Kohonen, T.: What generalizations of the self-organizing map make sense? In Marinaro, M., Morasso, P., eds.: ICANN'94, London, Springer Verlag (1994) 292–297
10. Brückner, B., Wesarg, T., Blumenstein, C.: Improvements of the modified hypermap architecture for speech recognition. In: Proc. Inter. Conf. Neural Networks. Volume 5., Perth, Australia (1995) 2891–2895
11. Brückner, B.: Improvements in the analysis of structured data with the multilevel hypermap architecture. In et.al., K., ed.: Progress in Connectionist-Based Information Systems, Proceedings of the ICONIP97. Volume 1., Singapore, Springer-Verlag (1997) 342–345
12. Brückner, B., Gaschler-Markefski, B., Hofmeister, H., Scheich, H.: Detection of non-stationarities in functional mri data sets using the multilevel hypermap architecture. In: Proceedings of the IJCNN'99, Washington D.C. (1999)
13. (www.brainvoyager.com)
14. Kohonen, T.: Self-Organization and Associative Memory. Springer-Verlag, New York (1988)
15. Baumgart, F., Gaschler-Markefski, B., Woldorff, M.G., Heinze, H.J., Scheich, H.: A movement-sensitive area in auditory cortex. Nature **400**(6746) (1999) 724–726
16. Scheich, H., Ohl, F., Schulze, H., Hess, A., Brechmann, A.: What is reflected in auditory cortex: Properties of sound stimuli or what the brain does with them? In König, R., Heil, P., Budinger, E., Scheich, H., eds.: The Auditory Cortex A Synthesis of Human and Animal Research, Hillsdale , NJ, Lawrence Erlbaum Associates (2005) 383–408
17. Cichocki, A., Amari, S.I.: Adaptive Blind Signal and Image Processing. John Wiley and Sons (2002)
18. Wesarg, T., Brückner, B., Schauer, C.: Modelling speech processing and recognition in the auditory system with a three-stage architecture. In von der Malsburg et.al., C., ed.: Artificial Neural Networks - ICANN 96, Lecture Notes in Computer Science. Volume 1112., Berlin, Springer-Verlag (1996) 679–684
19. Blumenstein, C., Brückner, B., Mecke, R., Wesarg, T., Schauer, C.: Using a modified hypermap for analyzing moving scenes. In ichi Amari et.al., S., ed.: Progress in Neural Information Processing, Proceedings of the ICONIP96. Volume 1., Singapore, Springer-Verlag (1996) 428–431

Scalable Dynamic Self-Organising Maps for Mining Massive Textual Data

Yu Zheng Zhai, Arthur Hsu, and Saman K. Halgamuge

Dynamic System and Control Group,
Department of Mechanical and Manufacturing Engineering,
University of Melbourne, Victoria, Australia 3010
y.zhai@pgrad.unimelb.edu.au, alhsu@unimelb.edu.au,
saman@unimelb.edu.au

Abstract. Traditional text clustering methods require enormous computing resources, which make them inappropriate for processing large scale data collections. In this paper we present a clustering method based on the word category map approach using a two-level Growing Self-Organising Map (GSOM). A significant part of the clustering task is divided into separate sub-tasks that can be executed on different computers using the emergent Grid technology. Thus enabling the rapid analysis of information gathered globally. The performance of the proposed method is comparable to the traditional approaches while improves the execution time by 15 times.

1 Introduction

The ever increasing amount of textual information available on the internet has made textual data clustering algorithms more appealing and promising. However, to cluster these billions of documents will require enormous processing power and resources, which is beyond the capability of typical local computing resources. Therefore, scalability has become an important issue for large-scale applications.

The emergent technology of Grid Computing enables the sharing of distributed heterogeneous resources to solve computationally intensive and data intensive problems. Researches are now advancing to e-research, where normal research methods are augmented with internet tools to enhance efficiency and collaboration. Many projects have already been developed using the Grid in areas such as weather forecasting, finical modeling and earthquake simulation, to acquire various resources distributed at different locations, in order to satisfy their demands of huge computational exercises [3]. It is clear that text mining applications can also benefit from the Grid to improve scalability, if we can divide the clustering task into independent sub-tasks and execute them separately.

The motivation of this work arises from the Australian Research Council funded E-Research project on "Collection, Sharing, Visualisation and Analysis of locally gathered information from geographically remote areas vulnerable to tidal waves", where a proposed website will be open for public as a portal to collect substantiated reports of strange behavior in animals. These reports will then been analyzed, in conjunction with other source of information such as news articles to issue alert for further attention. The number of reports and news articles will grow larger as time passes. Therefore a more scalable text mining technique is required to discover useful information from the rapidly expanding database, while capable of taking advantage of the available Grid resources through Globus services.

In this paper we propose a document clustering approach based on the WEBSOM method using the scalable two-level Growing Self-Organising Maps (GSOM), which will provide us the prospect of utilizing Grid resources to cluster massive text data sets efficiently.

The rest of the paper is organized as follows. Section 2 presents an overview of the GSOM algorithm, the traditional WEBSOM approach and our proposed scalable textual mining using GSOM. Benchmarking results and comparisons are presented in Section 3. Conclusion and future research efforts are given in Section 4.

2 Background and Proposed Method

Self-Organising Maps (SOM) [4] is an unsupervised neural network model that maps high dimensional input into a low dimensional topology such that similar clusters are close to each other on the map. Each node of the SOM has a weight vector associated with it and is randomly initialized. Each input vector is presented to the network and the node that is closest to the input vector will be updated to follow the input. The nodes around the winning node, typically determined by a Gaussian neighborhood function, will also adapt to the input to produce a smooth mapping.

An extension of the SOM is the Dynamic Self-Organising Maps [5, 12, 15], which can grow into different shapes and sizes corresponding to the input. GSOM starts with a small number of nodes (e.g. seven nodes in case of a hexagonal map structure). The algorithm then goes through one growing phase, a rough tuning phase and a fine tuning phase. The rate of the growth and thus the final map size, is controlled by a Spread Factor (SF) vary between 0 and 1 (1 gives maximum growth) [14]. It is used to determine the growth threshold GT, which is given by:

$$GT = -D \times \ln(SF) \tag{1}$$

where D is the dimensionality of input vectors. Therefore, data analyst has the ability to choose from more abstract grouping to finer separation by using different values of SF (A detailed discussion of the effect of SF is presented later). In the growing phase, when the winning node is identified, the error value, i.e. the difference between the input vector and the weight vector, usually determined by Euclidean distance, is added to the error counter of that node. If the accumulated error for a given winning node is greater than GT, then a new node is created next to the winning node if it has free neighbour space, i.e. a boundary node. Otherwise, half of the accumulated error is distributed to all the neighbouring nodes. During both of the tuning phases, GSOM follows the update rules used by SOM.

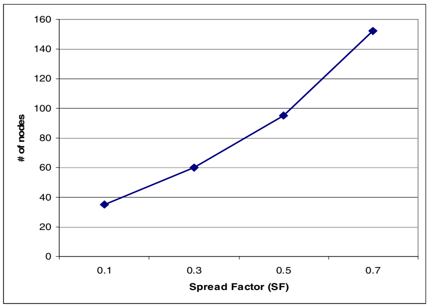
WEBSOM [6, 7] is an application of SOM that offers an alternative way to encode the text document compare to the traditional “bag of words” approach [11]. It uses the word category maps that utilize the contextual information to group similar words. Each word is given a unique random vector of size 90 and the immediate context of each word is formed by combines the vectors of the preceding and succeeding words. The average context value is calculated for all the instances of the word in the text collection. These word context vectors are then mapped onto a two-dimensional grid using the SOM. Documents are then encoded as vectors of histogram of the word clusters formed earlier. These vectors serve as input to another SOM, which produces the document map that enables the visualisation of clusters.

In the proposed work, GSOM is used instead of SOM in forming the word category maps and document map mentioned above. This is due to the fact that SOM requires the shape and size of the map to be defined in advance. Yet it is often the case that the number of classes in a data set is unknown and new information is added frequently. Therefore, SOM is less suitable in this situation as poorly defined parameters will give an unsatisfactory result. GSOM on the other hand, overcomes this problem since it only inserts a new node when required. This dynamic structure makes GSOM especially appropriate for rapidly changing document collections, as discussed in [8].

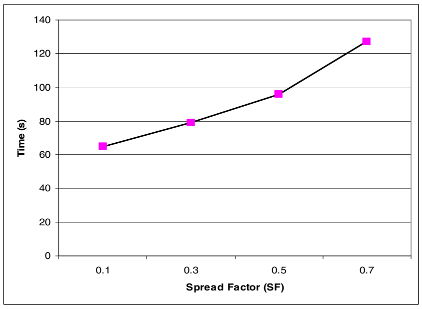
To make the clustering task scalable and efficient, we aim to achieve three goals [10]:

- Divide the original task into sub-tasks such that the processing of each sub-task is independent from the results of the others.
- Both the number of sub-task and the input data supplied to each sub-task should be great enough to compensate for the communication overheads.
- Data transfer between the central computer and each executing node should be minimized

Therefore, we propose to split the initial single growing phase in GSOM into two separate growing phases. While the first growing phase is performed as a single process, the outcome is used to initiate the second growing phase running concurrently on different computer nodes, together with the remaining two tuning phases. This is intended to first obtain an abstract and rapid grouping of the entire input data during the first growing phase by using a low spread factor with GSOM. It produces a feature map that has a few condensed nodes with high level separation between them. The data obtained within each node can then be sent to different computers and refined independently. They will be process by another GSOM with a slightly higher spread factor in order to achieve finer clustering and follows the normal procedure of one growing and two tuning phases. All the resulting outputs will be passed back to the central computer and combined together for further processing.



(a) SF vs. No. of nodes generated



(b) SF vs. Time

Fig. 1. The effect of different Spread Factor on data set two during the generation of rough word category map

The choice of a suitable SF is a critical issue. It is evident from the Figure 1 that a high spread factor will demand a longer processing time for the central computer and generate excessive nodes (hence reduce the amount of data distributed to each node) that violates our second goal. Moreover, a SF that makes the feature map excessively abstract or refined can also reduce the clustering quality (Table 2). However, the selection of the exact values of SF is currently empirical, as it depends on the required level of abstraction for a given application, as well as a tradeoff between efficiency and quality.

3 Results and Discussion

The scalable GSOM approach is first compared with the traditional SOM and GSOM method to investigate its performance on cluster quality, and then the execution time is measured to evaluate its efficiency.

3.1 Document Collections

In this experiment, two sets of text collections were used. The first set consists of a small collection of 150 news articles across 5 areas extracted from the ABC news archive[1], which were used to compare the cluster quality between the different approaches. The second set is composed of a subset of the forum articles obtained from the Usenet discussion group, which has been employed in WEBSOM studies in [1, 2]. This collection of documents was particularly interesting not only because it closely resembles the text form of the intended application, but also regarded challenging as all the documents were written in a very informal language that often hold all sorts of colloquial words and produces an even larger vocabulary. This set contains 2,000 articles in total and serves as the benchmark for the measuring the execution time.

3.2 Preprocessing

The documents were preprocessed to remove any non-textual content such as HTML tags, header information and user signatures, which results in an average text length of 200 words for the second data set. After removing from a list of 385 stop words, the vocabulary contains 26,869 words (both base and inflected forms). The words that appear less than 5 times and more than 500 times, which are regarded as less useful in distinguish between clusters, were also removed and the final vocabulary consists of 5707 words.

3.3 Cluster Evaluation Method

We used the standard F-measure as in [9], to compare the quality between different outputs using SOM, GSOM and scalable GSOM. F-measure is a function of precision (*p*) and recall (*r*) that captures how well the clusters match the pre-defined categories.

[1] http://www.abc.net.au/news/

$$p = \frac{a}{b} \tag{2}$$

where a is the number of documents belong to class X in cluster Y and b is the total number of documents in cluster Y

$$r = \frac{a}{c} \tag{3}$$

where c is the total number of documents belongs to class X

$$F = \frac{2 * p * r}{p + r} \tag{4}$$

The F value for each class over the entire collection is the maximum value for that class over all clusters. Finally, the weighted overall F value is:

$$F_{overall} = \sum_i \frac{c_i}{c_{total}} F \tag{5}$$

where i is the number of classes and c_{total} is the total number of documents in the collection.

Table 1 below shows the result of applying different approaches on the first data set.

Table 1. F-measure for ABC news archive collection

Method	F -measure
SOM	0.49
GSOM	0.51
Scalable GSOM	0.50

As can be seen from Table 1 that the quality of the resulting maps is comparable for all three methods. This gives us confidence that the scalable GSOM method will not compromise the performance of the WEBSOM approach.

Since two GSOM feature maps will be generated (one for word category map (w) and one for document map (d)), two SFs will need to be selected. Table 2 below shows the results for using different SF on both maps. It is clear that in this case, if the feature map is too abstract (SF(w) = 0.2, SF(d) = 0.2) or refined (SF (w) = 0.8, SF(d) = 0.8), the performance of GSOM deteriorates.

Table 2. Impact of SF on F-measure using GSOM

SF(d) = 0.2				
SF (w)	0.2	0.4	0.6	0.8
F-measure	0.35	0.43	0.49	0.51
SF(w) = 0.8				
SF (d)	0.2	0.4	0.6	0.8
F-measure	0.51	0.52	0.43	0.39

3.4 Comparing Execution Time

We observe that the phase of generating the word category map, which takes up more than 70% of the overall time in this case, is the bottleneck of the entire clustering process. (As the number of documents grows larger, the scalable method can also be applied to the phase of producing the document map.) Therefore, using the proposed scalable GSOM method, a rough word category map is generated first with a spread factor of 0.1, which results in 35 nodes in 65 seconds (see Figure 2). The size of the data for each node ranges from 747kB to 6,041kB. Each node is then refined separately using a GSOM again with a spread factor of 0.3 (An example is shown in Figure 3). On the other hand, the conventional GSOM uses a SF of 0.8 and runs as a single process.

Table 3 below summarizes the results of using a single GSOM and the scalable GSOM for generating the word category map. Even though ideally the scalability of 35 times should be achieved, the data is not evenly distributed to each node and the later task of producing the document map can not be realized without the results from all the nodes. Therefore, the running time shown for the scalable GSOM is the slowest time out of all 35 nodes, plus the time spent in generating the rough word map.

Table 3. Execution time for generating the word map for Usenet articles

method	Total No. of nodes	Time (s)
GSOM	402	5407
Scalable GSOM	752	347

It is clear that the time needed for the scalable method is about 15 times less than the traditional way. The time improvement is very noticeable even for such a small collection of text items, thus the huge amount of documents in real life can benefit further from this approach. However, the scalable method is only tested in a simulated situation, which does not take into consideration of other factors such as communication delays and job queuing time when actually running on the Grid. In spite of this idealization, the scalable method still gives a very promising performance improvement and ample time to accommodate for the communication overheads.

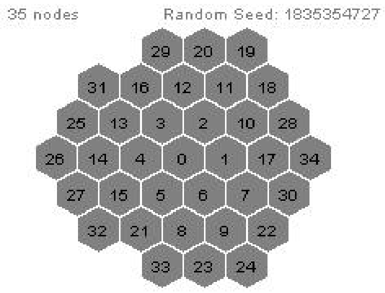
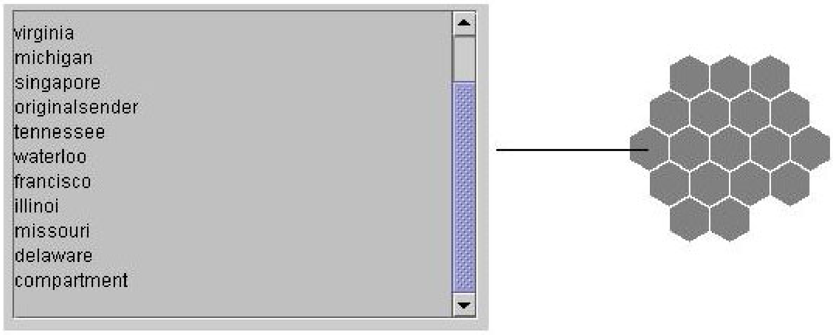


Fig. 2. The rough word category map



virginia
michigan
singapore
originalsender
tennessee
waterloo
francisco
illinoi
missouri
delaware
compartment

Fig. 3. An example of the refined node 3 from Figure 2

4 Conclusion

In this paper, we present a scalable GSOM method on text clustering that can take advantage of the Grid Computing technology. The test results indicate that the scalable method performs as well as the traditional methods while improves the execution time up to 15 times. This method is especially suitable to real-life application where the text database contains a huge collection and rapidly expanding. However, the GSOM maps separately generated by each node are not merged together to form a single map. This makes it hard to explore and visualize the final output across node boundaries. As a future development, it will be desirable to connect maps of different size and shapes to form a unified map, such that the results can be easily searched and browsed. Moreover, the method will be tested on the actual Grid to further evaluate its performance.

Acknowledgments. The work presented in this article was funded by Australian Research Council.

References

1. Honkela, T., Kaski, S., Lagus, K., Kohonen, T.: Newsgroup Exploration with WEBSOM Method and Browsing Interface, Tech. Rep. A32, Helsinki University of Technology, Laboratory of Computer and Information Science, Espoo, Finland (1996).
2. Kaski, S., Honkela, T., Lagus, K., Kohonen, T.: WEBSOM—Self-Organizing Maps of Document Collections, Neurocomputing 21 (1998) 101–117.
3. Foster, I., Kesselman, C. (eds.): The grid : blueprint for a new computing infrastructure, Amsterdam, Boston , Elsevier, 2004
4. Kohonen, T.: self-organizing maps, Springer-Verlag, Berlin, 1995
5. Alahakoon. D., Halgamuge, S.K., Srinivasan, B.: Dynamic Self-Organising Maps with Controlled Growth for Knowledge Discovery, IEEE Transactions on Neural Networks, Special Issue on Knowledge Discovery and Data Mining, vol. 11, no. 3, 2000.
6. Lagus, K., Kaski, S., Kohonen, T.: Mining Massive Document Collections by the WEBSOM Method, Information Sciences, Vol 163/1-3, pp. 135-156, 2004.

7. Honkela, T.: Self-Organizing Maps in Natural Language Processing, Ph.D. thesis, Helsinki University of Technology, Neural Networks Research Center, Espoo, Finland, 1997.
8. Nürnberger, A.: Interactive Text Retrieval Supported by Growing Self-Organizing Maps, Proc. of the International Workshop on Information Retrieval, pp. 61-70, 2001.
9. Larsen, B., Aone, C.: Fast and Effective Text Mining using Linear Time Document Clustering. Proceedings of the conference on Knowledge Discovery and Data Mining, pages 16-22, 1999.
10. Depoutovitch, A., Wainstein, A.: Building Grid Enabled Data-Mining Applications, http://www.ddj.com/184406345, 2005
11. Salton G.: Developments in Automatic Text Retrieval, Science, Vol 253, pages 974-979, 1991.
12. Hsu, A., Halgamuge, S.K.: Enhancement of Topology Preservation and Hierarchical Dynamic Self-Rrganising Maps for Data Visualisation, International Journal of Approximate Reasoning, vol. 32/2-3 pp. 259-279, Feb 2003.
13. Hsu, A., Tang, S., Halgamuge, S.K.: An Unsupervised Hierarchical Dynamic Self-Organising Approach to Class Discovery and Marker Gene Identification in Microarray Data, Bioinformatics, Oxford University Press, November 2003
14. Alahakoon, D.: Controlling the Spread of Dynamic Self Organising Maps, Neural Computing and Applications, 13(2), pp 168-174, Springer Verlag, 2004
15. Wickramasinghe, L.K., Alahakoon, L.D.: Dynamic Self Organizing Maps for Discovery and Sharing of Knowledge in Multi Agent Systems in Web Intelligence and Agent Systems: An International Journal, (IOS Press), Vol.3, No.1, 2005.

Maximum-Minimum Similarity Training for Text Extraction

Hui Fu, Xiabi Liu, and Yunde Jia

School of Computer Science and Technology
Beijing Institute of Technology, Beijing 100081, P.R. China
{pihuir, liuxiabi, jiayunde}@bit.edu.cn

Abstract. In this paper, the discriminative training criterion of maximum-minimum similarity (MMS) is used to improve the performance of text extraction based on Gaussian mixture modeling of neighbor characters. A recognizer is optimized in the MMS training through maximizing the similarities between observations and models from the same classes, and minimizing those for different classes. Based on this idea, we define the corresponding objective function for text extraction. Through minimizing the objective function by using the gradient descent method, the optimum parameters of our text extraction method are obtained. Compared with the maximum likelihood estimation (MLE) of parameters, the result trained with the MMS method makes the overall performance of text extraction improved greatly. The precision rate decreased little from 94.59% to 93.56%, but the recall rate increased a lot from 80.39% to 98.55%.

1 Introduction

Text extraction from images is becoming a hot topic in pattern recognition and with many applications including content-based image indexing, automatic video logging and OCR systems. Text extraction from images is also challengeable because texts in an image are widely varied in direction, color, arrangement and background.

Existing methods of text extraction can be classified into two categories: region-based and texture-based [1]. Region-based methods extract texts by using the difference in color or gray scales between text regions and backgrounds. These methods often work with a bottom-up strategy, where sub-structures are identified and merged to mark text regions [2-3]. Texture-based methods are based on the observation that text in images has distinct textural properties from the background, in which the textural properties of a text region can be detected by Gabor filters and Wavelet [4-5]. The most of those methods are sensitive to the variations of text in font, color, and arrangement. Furthermore, multilingual text cannot be extracted by using a single method.

Recently, the researchers introduced text extraction methods based on statistical modeling of neighboring character regions and reported promising results. Zhang and Chang [6] proposed a part-relation based approach for scene text detection, and reported their experiments on English text. Our previous work [7] presented a text extraction method based on the Gaussian Mixture modeling (GMM) of three neighboring characters, which can handle multilingual text and complicated cases such as multicolor text, text arranged in a curve. The maximum likelihood estimation (MLE) of parameters of GMM in the method was found by the Expectation-Maximization (EM) algorithm. The resultant precision rate is high, but the recall rate is imperfect. The maximum likelihood (ML) criterion belongs to informative training, in which the classifier performance is optimized indirectly. Contrarily, the discriminative training methods, such as the minimum classification error (MCE), aim to optimize the classifier performance directly, which have been applied successfully in character recognition and speech recognition [8-10].

In this paper, we use a new idea of discriminative training, the maximum-minimum similarity (MMS) training criterion [11], to revise the initial parameters estimated from the EM algorithm, and subsequently improve the performance of our text extraction method. In the MMS training method, the recognizer performance is optimized through maximizing the similarities for positive samples, and minimizing those for negative samples. Based on this approach to discriminative training, we define the corresponding objective function for our text extraction method. Through minimizing this objective function by using the gradient descent method, the optimum parameters are obtained. In the experiments of text extraction, the overall performance of our text extraction method is improved greatly by the MMS training. Compared with the ML training, the precision rate decreased little from 94.59% to 93.56%, but the recall rate increased a lot from 80.39% to 98.55%. It shows that the MMS training is effective and promising.

2 GMM Based Text Extraction

The key of text extraction is to discriminate the region of characters from that of non-characters. In this work, we use the relationship of three neighboring characters to solve this problem. The region of three neighboring characters is found to be distinguished from other regions by the following features:

x_1: the consistency of distances between centroids of characters. For most text strings in images, no matter characters are arranged in a line or in a curve, distances between adjacent characters are approximately equal.

x_2: the consistency of region areas of characters. Similar with distance between characters, region areas of neighboring characters are often approximately equal.

x_3: the region density . The region density of a character is usually different from that of other objects. We compute the mean region density of three neighboring characters as the third feature.

Then the feature vector $\mathbf{x} = \{x_1, x_2, x_3\}$ of three neighboring characters is assumed to be of the distribution of Gaussian Mixture [12]. Let C denote the case of three neighboring characters, K be the number of components in GMM, w_k, $\boldsymbol{\mu}_\mathbf{k}$ and $\boldsymbol{\Sigma}_\mathbf{k}$ be the weight, the mean vector and the covariance matrix of the k th Gaussian component respectively, d is the dimension of the feature vector $\mathbf{x}$, $\sum w_k = 1$, then we have

$$p(\mathbf{x}|C)=\sum_{k=1}^{K} w_k N(\mathbf{x}|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k), \tag{1}$$

where

$$N(\mathbf{x}|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)=(2\pi)^{-\frac{d}{2}}|\boldsymbol{\Sigma}_k|^{-\frac{1}{2}}\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\right)\cdot \tag{2}$$

$\boldsymbol{\Sigma}_\mathbf{k}$ is assumed to be diagonal for simplicity, i.e. $\boldsymbol{\Sigma}_\mathbf{k}=\left[\sigma_{kj}\right]_{j=1}^{3}$.

Using Bayes' formula, we get

$$P(C|\mathbf{x})=\frac{p(\mathbf{x}|C)P(C)}{p(\mathbf{x})}\ . \tag{3}$$

It's reasonable to use the posterior probability $P(C|\mathbf{x})$ as the similarity measure between $\mathbf{x}$ and C. In Eq.(3), $p(\mathbf{x}|C)$ can be estimated from the samples of three neighboring characters, but it's very difficult to learn $P(C)$ and $p(\mathbf{x})$ since contrary cases are too diverse to be handled. However, they are fixed for all observations of C by assuming $\mathbf{x}$ is distributed uniformly, therefore

$$P(C|\mathbf{x})\propto p(\mathbf{x}|C)\ . \tag{4}$$

It should be noted that Eq.(4) is the same as the well-known Bayes classification rule, but there are totally different meaning. In the Bayes classification rule, the relationship between $P(C|\mathbf{x})$ and $p(\mathbf{x}|C)$ is with respect to classes, where C is variable with different classes but $\mathbf{x}$ is constant. Oppositely, in Eq.(4), the relationship between $P(C|\mathbf{x})$ and $p(\mathbf{x}|C)$ is with respect to observations, where $\mathbf{x}$ is variable with different observations but C is constant.

According to Eq.(4), we can emulate $P(C|\mathbf{x})$ through embedding $p(\mathbf{x}|C)$ in a smooth, monotonically increasing function which takes value in $[0,1]$. We call the value of this kind of functions as the 'pseudo-probability', for computing which the following function is a good choice:

$$\rho(p(\mathbf{x}|C))=1-\exp(-\beta p(\mathbf{x}|C)). \tag{5}$$

where β is a positive number.

Consequently, for arbitrary $\mathbf{x}=\{x_1,x_2,x_3\}$, we compute the corresponding pseudo-probability by using Eq.(5), if the result is larger than 0.5, $\mathbf{x}$ is accepted as the case of three neighboring characters.

Based on the discrimination method above of three neighboring characters, texts in images are extracted in following steps. Firstly, the initial text regions are extracted and binarized using the method based on edge pixels clustering [13], which guarantees the integrality of extracted text regions; secondly, each character in the binary image is treated as a connected component in it. In order to make this treatment reasonable for characters with separated parts, such as Chinese characters, the morphological closing operation is performed to connect separated parts of a character. Considering diverse appearances of a character in images, several structuring elements with different directions and sizes are used in closing operations to generate several new binary images; finally, the connected components in these new binary images and the original binary image are labeled as character or non-character by using the GMM based method above. If a connected component in new binary images is labeled as character, we find the corresponding region of this connected component in the original binary image, and label all connected components in this region of original binary image as characters.

For more details about the text extraction method described above, please refer to our paper [7].

3 Maximum-Minimum Similarity Training

The unknown parameter set of the text extraction method described in Section 2 is

$$\mathbf{\Lambda} = \{w_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \beta\}, k = 1...K \ . \tag{6}$$

where the meaning of each parameter is the same as that in Eq.(2) and Eq.(5).

In previous work [7], we used the EM algorithm to find the ML estimation of parameters in GMM, i.e. $w_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$, and set K to 3 and β to 1.020 through careful experiments. The resultant text extraction algorithm achieved the precision rate of 94.59%, but the recall rate of which was only 80.39%. In this paper, we use the maximum-minimum similarity (MMS) training criterion to revise the parameters of our text extraction method and improve its effectiveness accordingly.

The MMS training is a new idea of discriminative training, which focuses on the class separability of the similarity measure of a recognizer. We can imagine a perfect recognizer in which the similarities between observations and models from the same classes approximate to 1, and those for different classes approximate to 0. This means that the class separability and subsequent recognition rate of a recognizer can be optimized by producing class models to maximize similarities between class models and their positive samples, at the same time minimize similarities for their negative samples. Here the positive samples of a class model are its observation in the training data, and other training data is corresponding with its negative samples.

More formally, let $f(\mathbf{x};\mathbf{\Lambda})$ be the similarity function for classification, in which the set of unknown parameters is $\mathbf{\Lambda}$, $\hat{\mathbf{x}}$ be arbitrary positive sample, and $\bar{\mathbf{x}}$ be arbitrary negative sample, then the class separability of $f(\mathbf{x};\mathbf{\Lambda})$ is measured by

$$F(\mathbf{\Lambda}) = \sum [f(\hat{\mathbf{x}};\mathbf{\Lambda}) - 1]^2 + \sum [f(\bar{\mathbf{x}};\mathbf{\Lambda})]^2 \ . \tag{7}$$

$F(\boldsymbol{\Lambda})=0$ means the hundred-percent class separability; the less $F(\boldsymbol{\Lambda})$ is, the more class separability is. Consequently, we can obtain the optimum parameter set $\boldsymbol{\Lambda}^*$ of the class model by minimizing $F(\boldsymbol{\Lambda})$:

$$\boldsymbol{\Lambda}^* = \arg\min_{\Lambda} F(\boldsymbol{\Lambda}). \tag{8}$$

In this paper, $f(\mathbf{x};\boldsymbol{\Lambda})$ is defined by substituting Eq.(2) into Eq.(5), i.e.

$$f(\mathbf{x};\boldsymbol{\Lambda}) = 1-\exp\left(-\beta\left(\sum_{k=1}^{K} w_k N\left(\mathbf{x}\middle|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k\right)\right)\right). \tag{9}$$

where the unknown parameter set $\boldsymbol{\Lambda}$ is given in Eq.(6). Moreover, there are only two classes in this paper: the region of three neighboring characters and other regions. So the positive samples are three neighboring connected components which are all characters, and the negative samples are three neighboring connected components in which at least one connected component is non-character. The positive samples and negative samples are also called character samples and non-character samples respectively in the following description.

In Eq.(9), some parameters must satisfy certain constraints, which are transformed to unconstrained domain for easier implementation as in the MCE training method [8]. The constraints and transformation of parameters are listed as follows.

$$\text{(1)} \because \sum w_k = 1, \quad \therefore w_k \to \tilde{w}_k : w_k = \frac{e^{\tilde{w}_k}}{\sum e^{\tilde{w}_k}}.$$

$$\text{(2)} \because \beta > 0, \quad \therefore \beta \to \tilde{\beta} : \tilde{\beta} = \ln\beta .$$

$$\text{(3)} \because \sigma_{kj} > 0, \quad \therefore \sigma_{kj} \to \tilde{\sigma}_{kj} : \tilde{\sigma}_{kj} = \ln\sigma_{kj} .$$

Finally, the optimum parameter set of our text extraction method is searched by minimizing Eq.(7) through the gradient descent method. Let α_t and $\boldsymbol{\Lambda}_t$ be the step-size and the parameter set in the tth iteration respectively, $\nabla F(\boldsymbol{\Lambda}_t)$ be the partial derivative of $F(\boldsymbol{\Lambda})$ with respect to all the parameters in $\boldsymbol{\Lambda}_t$, then

$$\boldsymbol{\Lambda}_{t+1} = \boldsymbol{\Lambda}_t - \alpha_t \nabla F(\boldsymbol{\Lambda}_t) = \boldsymbol{\Lambda}_t - \alpha_t g_t . \tag{10}$$

$\nabla F(\boldsymbol{\Lambda}_t)$ is simplified as g_t in Eq.(10) and the following description. According to Eq.(10), the MMS training procedure in our text extraction method is:

Step1: Use all character samples (positive samples) and non-character samples (negative sample) to compute the partial derivative of $F(\boldsymbol{\Lambda})$ with respect to each parameter. Let ψ be arbitrary parameter in the parameter set, then we have

$$\frac{\partial F}{\partial \psi} = 2\sum\left[((f(\hat{\mathbf{x}};\mathbf{\Lambda})-1)+f(\overline{\mathbf{x}};\mathbf{\Lambda}))\frac{\partial f(\mathbf{\Lambda})}{\partial \psi}\right], \tag{11}$$

where $\frac{\partial f(\hat{\mathbf{x}};\mathbf{\Lambda})}{\partial \psi}$ and $\frac{\partial f(\overline{\mathbf{x}};\mathbf{\Lambda})}{\partial \psi}$ are unified and simplified as $\frac{\partial f(\mathbf{\Lambda})}{\partial \psi}$, because their formulations are the same.

Step2: Compute the step-size α_t using the Armijo-Goldstein method [14].

Step3: Update the parameters according to Eq.(10).

Step4: Repeat Step 1 to Step 3 until convergence. Let ε be an infinitesimal, the convergence condition is:

$$\|g_t\| = \left(\sum\left(\frac{\partial f(\mathbf{\Lambda})}{\partial \psi}\right)^2\right)^{1/2} \le \varepsilon\ . \tag{12}$$

The partial derivatives of $F(\mathbf{\Lambda})$ with respect to each parameter in above steps, i.e. $\frac{\partial f(\mathbf{\Lambda})}{\partial \psi}$, are given below, where $N(\mathbf{x}|\boldsymbol{\mu}_k,\mathbf{\Sigma}_k)$ in Eq.(9) is simplified as $N_k(\mathbf{x})$, k is the sequence number of the Gaussian component in the GMM, $j=1,2,3$ is the sequence number of the element in the feature vector of three neighboring characters, as described in Section 2.

$$\frac{\partial f}{\partial \tilde{w}} = \beta N_k(\mathbf{x})w_k(1-w_k)e^{\left(-\beta(\sum_{k=1}^{K} w_k N_k(\mathbf{x}))\right)}. \tag{13}$$

$$\frac{\partial f}{\partial \tilde{\beta}} = \left(\sum_{k=1}^{K} w_k N_k(\mathbf{x})\right)e^{\left(-\beta(\sum_{k=1}^{K} w_k N_k(\mathbf{x}))+\tilde{\beta}\right)}. \tag{14}$$

$$\frac{\partial f}{\partial \mu_{kj}} = \beta w_k N_k(\mathbf{x})\left(\frac{x_j-\mu_{kj}}{\sigma_{kj}}\right)e^{\left(-\beta(\sum_{k=1}^{K} w_k N_k(\mathbf{x}))\right)}. \tag{15}$$

$$\frac{\partial f}{\partial \tilde{\sigma}_{kj}} = \beta w_k N_k(\mathbf{x})\left(\frac{(x_j-\mu_{kj})^2}{2\sigma_{kj}^2}\right)e^{\left(-\beta(\sum_{k=1}^{K} w_k N_k(\mathbf{x}))+\tilde{\sigma}_{kj}\right)}. \tag{16}$$

4 Experimental Results

We tested the proposed text extraction method in real Chinese and English text of images both from ICDAR 2003 [15] and our database. We obtained character samples for training from 100 English text images and 100 Chinese text images, and non-character samples for training from other 20 images. Another 30 images were used as the test set, in which 5 images without text and with cluttered background were included for testing the robustness of the proposed method in rejecting false regions.

We first get the ML estimation of parameters of GMM in our method by the EM algorithm, and set the number of components in GMM to 3 and β to 1.020 through careful experiments. Based on estimated parameters, open and closed tests of text extraction were conducted and the corresponding results are listed in Table 1. In the ML training, 427 character samples for English text and 531 character samples for Chinese text were extracted manually from training images and used as the training data.

Then the MMS training is used to revise the initial parameters. We manually extracted 5973 non-character samples from 20 images. These non-character samples are taken as negative samples, and character samples in the EM training as positive samples. Using these negative samples and positive samples, the parameters of GMM and β were updated by the MMS training. Using the updated parameters, open and closed tests of text extraction were conducted again and the corresponding results are listed in Table 1 also.

In Table 1, the recall rate and the precision rate are used to evaluate the performance of the proposed method. Let $\mathbf{L}$ be the set of connected components corresponding to true characters in the image, $\mathbf{I}$ be the set of connected components identified as characters by the text extraction process, then the recall rate R and the precision rate P were calculated as follows [16]:

$$P = \frac{|\mathbf{L} \cap \mathbf{I}|}{|\mathbf{I}|}, \qquad R = \frac{|\mathbf{L} \cap \mathbf{I}|}{|\mathbf{L}|} .$$

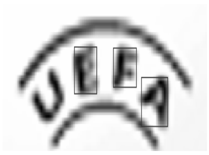
Furthermore, $|\mathbf{NT}|$ in Table 1 represents the number of connected components corresponding to non-characters.

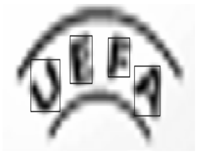
The data in Table 1 show that from the view of the overall performance, the MMS training behaved much better than the ML training in the experiments. For the training data, the precision rate is reduced from 96.08% to 95.40%, representing a 0.7% reduction. But the recall rate is increased from 81.25% to 98.09%, representing a 20.7% increase. As for the test data, the reduction in the precision rate is 1.1%, but the increase in the recall rate is 22.6%. The reasons behind these data are summarized as: (1) parameter estimation and text extraction are separated in the ML training; the performance of text extraction is optimized indirectly. Contrarily, the MMS training serves the text extraction directly; (2) only character samples are involved to estimate the distribution of the character region in the ML training, but the MMS training considers not only character samples but also non-character samples; (3) the MMS method can estimate the parameters out of the class conditional probability distribution, i.e. β in Eq.(9), but the ML method can not.

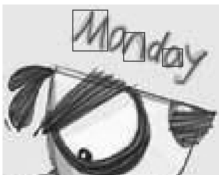
Fig.1 shows some examples of the situation that texts are omitted after the ML training but extracted after the MMS training. Fig.1(a) and Fig.1(b) shows the text extraction results after the ML training and after the MMS training respectively, where the extracted texts were indicated by rectangles. However, some non-character regions are extracted as characters after the MMS training. Fig.2 shows two examples of this situation, where extraction errors are indicated by arrowheads. We expect to overcome this shortcoming and improve the extraction precision further by using more training samples in the next research.

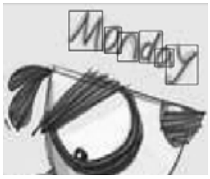
Tab1e 1. The experimental results after ML & MMS training

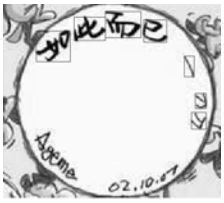
Evaluations / Training Algorithm	$\|\mathbf{L}\|$	$\|\mathbf{I}\|$	$\|\mathbf{L}\cap\mathbf{I}\|$	$\|\mathbf{NT}\|$	P	R
ML (200 training images)	1296	1096	1053	59	96.08%	81.25%
ML (30 testing images)	413	351	332	116	94.59%	80.39%
MMS (220 training images)	1417	1457	1390	152	95.40%	98.09%
MMS (30 testing images)	413	435	407	116	93.56%	98.55%

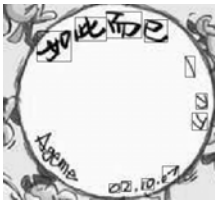

(a)

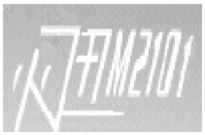

(b)


(a)


(b)


(a)


(b)


(a)

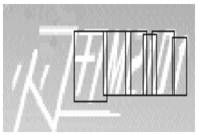

(b)

Fig. 1. The extraction results of test images indicated by rectangles, (a) the extraction results after the ML training, (b) the extraction results after the MMS training

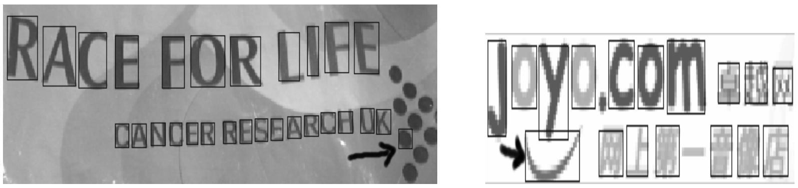


Fig. 2. The examples of false text extraction after the MMS training (false results are indicated by arrowheads)

5 Conclusions

This paper used the maximum-minimum similarity training criterion to estimate the parameters in the text extraction method based on Gaussian mixture modeling of neighbor characters. We defined the objective function for text extraction based on the MMS training, and used the gradient descent method to minimize the objective function and search the optimum parameters. The experimental results show that the parameters estimated by the MMS training are much better than those estimated by the ML training. The recall rate increased a lot and the precision rate decreased little. In the future work, we will use more training samples in the MMS training to improve the precision rate and the recall rate further.

Acknowledgements

This work was partially supported by the National Science Foundation of China (60473049).

References

1. Keechul Jung, Kwang In Kim and Anil K Jain.:Text information extraction in images and video: a survey. Pattern Recognition,Vol.37. (2004) 977-997
2. A K Jain and B Yu. Automatic text location in images and video frames. Pattern Recognition, Vol.31. (1998) 2055-2076
3. T Sato, T Kanade, E K Hughes and M A Smith, Video OCR for digital news archive. Proceedings of IEEE Workshop on Content based Access of Image and Video Databases, Bombay, India, (1998) 52-60
4. B Sin, S Kim and B Cho. Locating characters in scene images using frequency features. Proceedings of International Conference on Pattern Recognition, Vol. 3. Quebec, Canada, (2002) 489-492
5. V Wu, R Manmatha and E M Riseman.:TextFinder: an automatic system to detect and recognize text in images. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.21. (1999) 1224-1229
6. Dongqing Zhang and Shifu Chang.:Learning to Detect Scene Text Using a Higher-order MRF with Belief Propagation. IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW'04), Washington, DC, United States, (2004) 101-108

7. Hui Fu, Xiabi Liu and Yunde Jia.: Gaussian Mixture Modeling of Neighbor Characters for Multilingual Text Extraction in Images. IEEE International Conference on Image Processing 2006 (ICIP'06), Atlanta, Accepted.
8. B. H. Juang, W. Chou and C. H. Lee.: Minimum Classification Error Rate Methods for Speech Recognition. IEEE Trans. Speech and Audio Processing, Vol. 5. (1997) 257-265
9. Han Jiqing and Gao Wen.: Robust Speech Recognition Method Based on Discriminative Environment Feature Extraction. Journal of Computer Science and Technology, Vol.16. (2001) 458-464
10. Zhang Rui and Ding Xiaoqing.: Minimum Classification Error Training for Handwritten Character Recognition. 16th International Conference on Pattern Recognition,Vol.1. Auguest, (2002) 580-583
11. Xiabi Liu, Yunde Jia, Xuefeng Chen, Hui Fu and Yanjie Wang.: Maximum-Minimum Similarity Training Criterion for Pattern Recognition. Technical Report, 2006. http://www.mcislab.org.cn/reports/
12. Perry Moerland.: A comparison of mixture models for density estimation. Proceedings of the International Conference on Artificial Neural Networks (ICANN'99),Vol.1.(1999)25-30
13. Hui Fu, Xiabi Liu and Yunde Jia.: Text Area extraction Method Based on Edge-pixels Clustering. Proceedings of the 8th International Computer Scientists, Convergence of Computing Technologies in the New Era, Beijing, (2005) 446-450
14. Yaxiang Yuan and Wenyu Sun.: Optimization Theory and Methods (in Chinese). Since Press, (2003)
15. S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young.: Icdar 2003 robust reading competitions. Proceeding of the 7th International Conference on Document Analysis and Recognition, Edinburgh, UK., (2003) 682-687
16. D. Karatzas and A. Antonacopoulos.: Text Extraction from Web Images Based on A Split-and-Merge Segmentation Method Using Colour Perception. IEEE, proceedings of the 17th International Conference on Pattern Recognition (ICPR'04), Cambridge, UK, (2004) 634-637

Visualization of Depending Patterns in Metabonomics

Stefan Roeder[1], Ulrike Rolle-Kampczyk[1], and Olf Herbarth[1,2]

[1] UFZ – Centre for Environmental Research Leipzig-Halle Ltd.,
Department Human Exposure Research/Epidemiology,
D - 04318 Leipzig, Germany
{stefan.roeder, ulrike.rolle-kampczyk,
olf.herbarth}@ufz.de
[2] University of Leipzig, Faculty of Medicine,
D - 04103 Leipzig, Germany

Abstract. This paper describes an approach for visualization of patterns in large data sets. The data sets are combined from external exposure and internal stress factors on human health. For deduction of modes of action on human health, external and internal stress factors have to be combined and classified. The approach shown in this paper is based upon clustering algorithms. Relationships between cases ban be obtained by visual inspection of clustering results.

1 Introduction

During our epidemiological studies, which deal with the influence of indoor pollution on human health, we observed that several lifestyle activities such as renovation or smoking leave typical patterns in indoor air (patterns of volatile organic compounds (VOC)). Until now only single chemicals have been linked with the health effect. The problem is to judge the effect of mixtures or patterns of chemicals respectively. From that it is not clear, whether health effects are associated with these patterns. A first but key step is to find out more about this relationship. We investigated patterns of metabolites, which are traceable after passage through human body from urine samples, too. By combining of both patterns (external and internal) we try to identify combinations of compounds which hold harmful potential on human health.

This paper describes an approach visualization of patterns in large data sets which are results of classification algorithms (i.e. from self organizing maps). These patterns are used to deduce on the relationships between external and internal stress influences on human health and the processes behind.

Both stress patterns are described by a data vector for each individual case. After combining both vectors each case is represented by a data vector describing its situation. Combining is a simple case wise concatenate of the variables. Sequence is not important; it has only to be ensured that sequence is the same in all cases.

The main focus of this investigation is to classify cases into several typical clusters and subsequently to find core properties of these clusters in order to understand the processes behind them. There is no single solution for a given classification task when using self organizing maps. Therefore a possibility to assess different solutions against each other was needed.

Visualization is necessary to check the solution provided by the clustering algorithm. Because the data vectors to be classified are high dimensional and only similar but not equal, a proper method for visualization was needed.

Table 1. Nomenclature

abbreviation/term	
x_{ij}	Element of input vector
x_{ij} norm	Element of input vector normalized to a closed {0;1}-interval
I	case index
J	Variable index
$\max(x_j)$	maximum value of a variable
$\min(x_j)$	minimum value of a variable

2 Material and Methods

In general our datasets are collected by LC-MS/MS methods from human urine samples (internal load) as well data collected by GC-MS from indoor air (external stress, exposure). The dataset used in the examples for this paper is based on volatile organic compounds from indoor air samples.

For the clustering task we decided to use a method based on self organizing maps (SOM) [1-3], because they provide an efficient way to map from an n-dimensional space to a two-dimensional space and to visualize multivariate data afterwards. The self organizing map algorithm is a member of the class of the unsupervised artificial neural network algorithms. In contrast to supervised artificial neural network algorithms, self organizing maps are able to extract typical features without any prior knowledge about the structure of the data. There are shortcut methods existing, which speed up the computation [4]. In this investigation we did not apply them.

The term “property” has to be introduced for proper understanding of this paper. We use it for the entirety of variable values of a case. The property of a case is defined as a distinct combination of variable values.

3 Pattern Visualization

Pattern visualization is done to get an impression on the assignment/layout of the cases into the classes together with their properties.

Classification algorithms basing on self organizing maps do not necessarily lead to the same result. This is because the initial values in the algorithm are randomly chosen. Cases with large differences are arranged into different classes reliably, but cases which are similar to multiple cases can be arranged to one class in the first try and into another class in the second try. This raises the question whether the achieved result is suitable for the requirements which were fixed in advance. Therefore we needed a solution for visualization of classification results, so that a human is able to assess the quality of the classification.

Usually two dimensional representations are in use for this task, because they allow a precise representation of the relationships between a variable pair [5].

3.1 Two Dimensional Visualization

If only the relationship between a distinct pair of variables is of interest, then the two dimensional visualization of a distinct variable pair is suitable. The following example shows the relationship between the two chemical compounds “Undecane” and “Methylbenzoate”. Both variables are normalized to a closed {0;1}-interval.

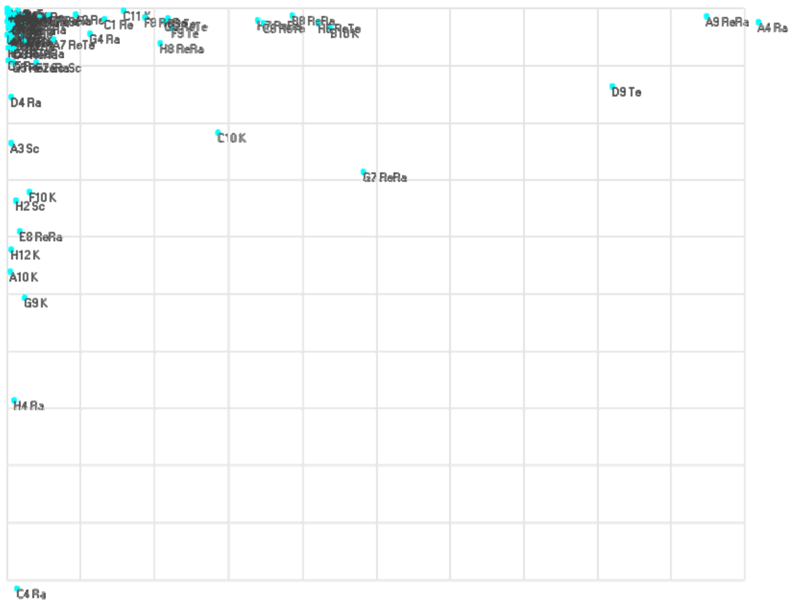


Fig. 1. Visualization of a distinct variable pair for all cases in a two dimensional view (example)

Pattern visualization can also be done as visualization of all variable values and cases in a three dimensional view and as visualization of a distinct variable pair for all cases in a two dimensional view.

3.2 Three Dimensional Visualization

The three dimensional view is useful if one needs an overview on the dataset and the relationships between the variables and cases. For a better visualization it is necessary to adopt the range of all variables to an common closed {0;1}-interval. Otherwise variables with small absolute values are nearly invisible. This is done similar to the normalization of the data set before the training of the self organizing map:

$$x_{ij}norm = \frac{x_{ij} - \min(x_j)}{\max(x_j) - \min(x_j)} \quad . \tag{1}$$

The visualization of the values can be done as surface plot (figure 1). This gives a good impression of the relative distribution of the values between classes and their cases. The visualization of missing values is difficult, because they produce holes in the surface which also affect neighbouring values and prevent from detecting the dependencies.

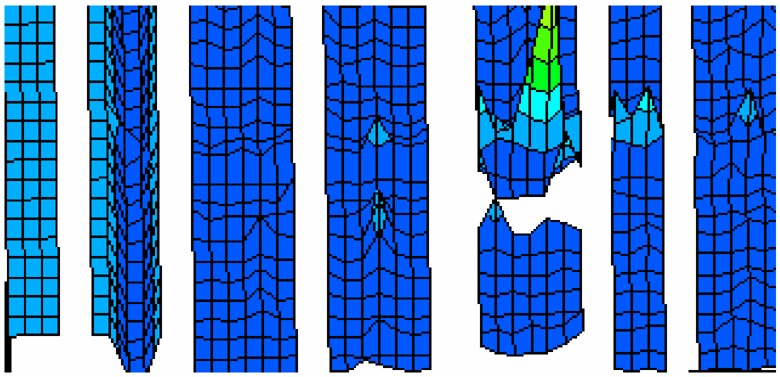


Fig. 2. Visualization of variable values as 3D-surface (example) (cases are in column, variables are in row)

Because of these problems we decided to visualize the values as bars. Missing values are clearly visible in this case (figure 2).

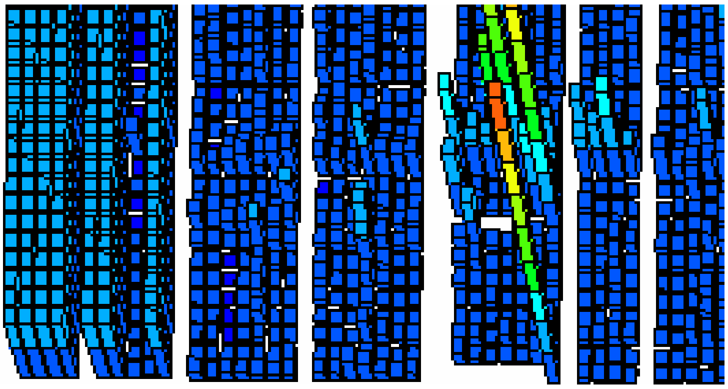


Fig. 3. Visualization of variable values and cases as 3D-bars (example) (cases are in column, variables are in row)

The ability to tilt and rotate the entire diagram is important to get an overview on the relation of the objects in three dimensional space which was projected to the two dimensional screen.

3.3 Methods of Color Coding

Color coding can be used for visualization of the values or for visualization of the classes. Color coding of different variables or cases fails, if there are a high number of variables or cases. In this case the number of available distinguishable colors is not sufficient for each different variable or case.

After several tries we decided to code the values by color, because this gives the best impression of the overall distribution. Simultaneously the different classes are visually subdivided by lines of equal small values (0.1), which are easily recognizable as borders.

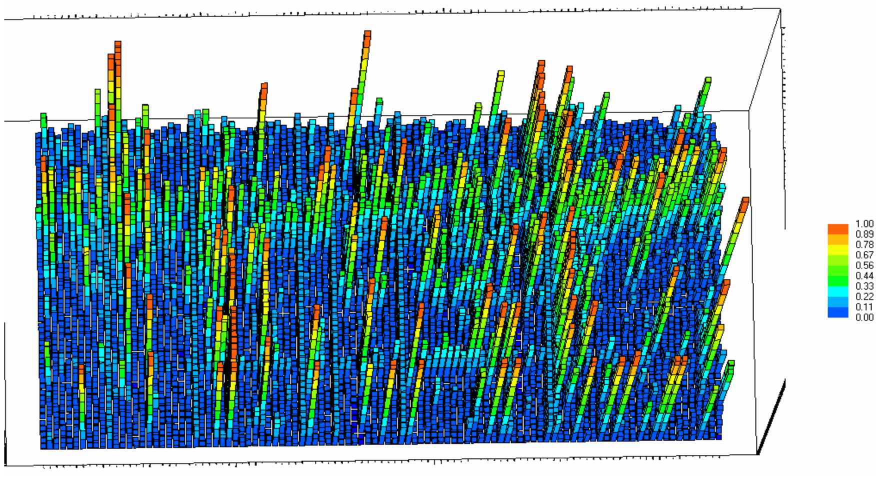


Fig. 4. Data set before training using a self organizing map (SOM)

Figure 3 shows a data set before training (cases are unordered) whereas figure 5 shows the same data set after classification using the SOM algorithm.

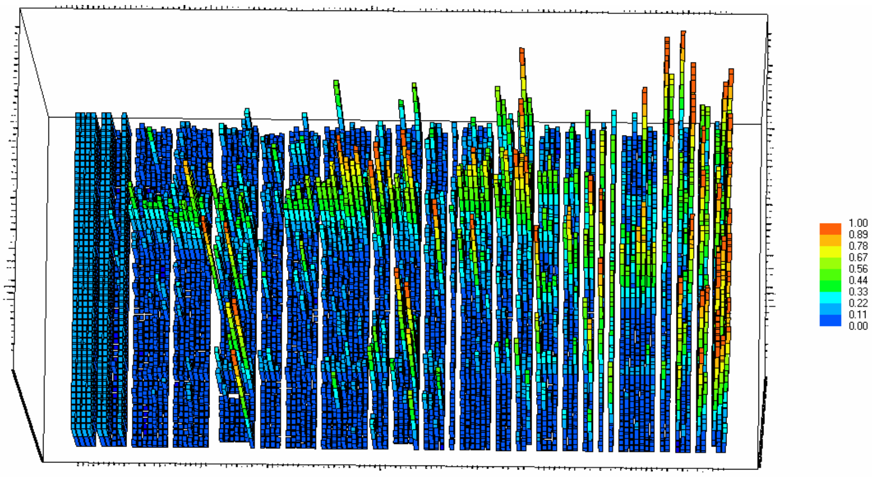


Fig. 5. Visualization of variable values and cases as 3D-bars (example) (cases are in column, variables are in row)

The cases are now assigned to classes. One can clearly see the difference between both assignments. Visual inspection allows easier recognition of possible important variables in the space.

After classification of the cases visualization can show the distribution of the cases with their variables in space as well as the most important variables together with their properties. The rows show the properties (in this example: chemical compounds) whereas the columns show the cases. On the left side there are two classes with nearly identical cases. These cases were scattered into the dataset to control the ability of the classification algorithm to concatenate these cases to classes.

The empty columns represent the borders of the classes. The visualization of these borders is important for a good visual distinction between the classes. One can clearly see that the classes share similar properties; low values with few high concentrated compounds on the left sides; high concentrations on nearly all compounds on the right side.

3.4 Calculation of Hypothetical Parameters

Stress spectra of outer and inner burden are not of the same structure. For the analysis of the results of clustering it is important to gather deeper knowledge about the variable values behind each Kohonen neuron as well as the relation of these values between all Kohonen neurons. Because of the fact that the weight vectors of the Kohonen neurons contain normalized values in a closed {0;1}-interval these values must be 'denormalized' before considering on the absolute values of a variable. Denormalization means the opposite transformation than normalization (equation (1)) and transforms the values of the weight vector in a Kohonen neuron back to the original co-domain of the underlying variable using the following equation. It can be obtained by re-arranging equation (1):

$$x_{ij} = (x_{ij}norm * (\max(x_j) - \min(x_j)) + \min(x_j) \ . \tag{2}$$

4 Discussion

A novel method for visualization of patterns has been introduced in this paper. The basic idea is to show dependencies between cases and classes as well as to verify the assignment made by a clustering algorithm in a prior step.

Using the shown technique we were able to recognize dependencies between cases and to show the core properties of each class with respect to the arrangement of the whole data set. Among different visualization methods we got best results with 3D-bars using color coding of values.

It is important not only to identify similar cases (stacking into the same class by the clustering algorithm). Rather, one should know what the reason for this similarity is. This can be achieved by visualization of the data set and inspection of the variables which are similar. As result one can find out the reason for classifying of cases into the same class.

Using this method we are able to discover the relevant variables for a given classification. Another possible method for extraction of relevant variables is Principal Component Analysis (PCA)[6;7]. Unlike the method shown here, PCA does not classify cases into classes, rather it extracts the most important variables for describing a data set. Nevertheless, PCA results may also be visualized using the methods shown here.

References

1. Kohonen, T.: Self Organizing Maps. Series in Computer Sciences ed. Heidelberg. Springer, 1997
2. Zampighi, LM., Kavanau, CL., Zampighi, CA.: The Kohonen self-organizing map: a tool for the clustering and alignment of single particles imaged using random conical tilt. Journal of Structural Biology 146 (2004) 368-380
3. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics 43 (1982) 59-69
4. Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J., Paatero, V., Saarela, A.: Self organizing of a massive document collection. IEEE Transactions on Neural Networks 11 (2000) 574-585
5. Rhee, JI., Lee, KI., Kim, CK., Yim, YS., Chung, SW., Wei, J., Bellgardt, KH.: Classification of two-dimensional fluorescence sprectra using self-organizing maps. Biochemical Engineering Journal 22 (2005) 135-144
6. Joliffe, IT.: Discarding variables in a principal component analysis I: artificial data. Applied Statistics 21 (1972) 373-374
7. Joliffe, IT.: Discarding variables in a principal component analysis II: real data. Applied Statistics 22 (1973) 21-23

A RBF Network for Chinese Text Classification Based on Concept Feature Extraction

Minghu Jiang[1,2], Lin Wang[1,3], Yinghua Lu[1], and Shasha Liao[2]

[1] School of Electronic Eng., Beijing University of Post and Telecommunication, Beijing, 100876, China
[2] Lab. of Computational Linguistics, School of Humanities and Social Sciences, Tsinghua University, Beijing 100084, China
[3] Interdisciplinary Center for Scientific Computing (IWR), University of Heidelberg, Im Neuenheimer Feld 368, 69210 Heidelberg, Germany
jiang.mh@tsinghua.edu.cn

Abstract. The feature selection is an important part in automatic text classification. In this paper, we use a Chinese semantic dictionary -- Hownet to extract the concepts from the word as the feature set, because it can better reflect the meaning of the text. We construct a combined feature set that consists of both sememes and the Chinese words, propose a CHI-MCOR weighing method according to the weighing theories and classification precision. The effectiveness of the competitive network and the Radial Basis Function (RBF) network in text classification are examined. Experimental result shows that if the words are extracted properly, not only the feature dimension is smaller but also the classification precision is higher, the RBF network outperform competitive network for automatic text classification because of the application of supervised learning. Besides its much shorter training time than the BP network's, the RBF network makes precision and recall rates that are almost at the same level as the BP network's.

Keywords: text classification, concept feature, RBF network.

1 Introduction

Automatic text classification is a process which classifies the documents to several categories by their content. With the rapid development of the online information, automatic text classification becomes one of the key techniques for handling and organizing text data. It always includes two main parts: the feature selection and the classifier. Because of the huge data of the document sets, it is difficulty for reflecting the feature vector of documents, we need to construct a proper feature set which is easily process and has a considerable classification precision. Because the concept space is much smaller than the word one, and the components are comparatively independent, the concept features are much better to reflect the content of the documents. With the semantic analysis we can get a much better vector space [1], therefore we choose semantic features as the main components of our feature set.

Many techniques can be used in feature selection to improve accuracy as well as reduce the dimension of the feature vector and thus reduce the time of computation

[2, 3]. These techniques include using concept frequency instead of original word frequency and Latent Semantic Indexing (LSI) [3]. However, after using these techniques, the feature matrix is still quite complicated and requires a robust classifier to deal with.

Many classifiers have been applied to classify texts, including Vector Space Models, K-Means, Support Vector Machine (SVM), Naïve Bayes and so on [3]. Among many methods applied, several kinds of neural networks have shown attractive abilities. The competitive networks are used in text classification, including Learning Vector Quantization (LVQ) and Self-Organizing Maps (SOM) network [4, 5]. These two are variants of the unsupervised competitive networks. Besides, the Radial Basis Function (RBF) network, which is characterized by its high speed of training, can also be used in text classification.

In this paper we make a performance comparison between the competitive network, and RBF network in text classification. For unsupervised competitive network, we use the evaluation criterion, and define positive and negative accuracies and use their average to evaluate the performance of clustering.

Table 1. Definition of positive and negative accuracy

	Clustered together	Not clustered together
In same category	*A*	C
In different category	*B*	*D*

Positive Accuracy $A_p = A/(A+C)$, and negative accuracy $A_n = D/(B+D)$. Average Accuracy $A_a = (A_p + A_n)/2$. For the supervised classification networks, the factors we take into account are time, precision, recall and $F1$. Precision and recall are two widely used criteria of evaluation in text classification and text retrieval. Let us assume that P implies precision, R implies recall, A_i is the number of documents in category i, and B_i is the number of the documents classified by the classifier to category i, then $p_i = \frac{A_i \cap B_i}{B_i}$, and $R_i = \frac{A_i \cap B_i}{A_i}$. $F1$ is a criterion that combines precision and recall, $F1 = \frac{2p \times R}{P + R}$.

2 Concept Feature Extraction

Hownet is an on-line common-sense knowledge base which unveil the inter-conceptual relations and the inter-attribute relations of concepts as connoting in lexicons of the Chinese and their English equibalents [6]. Different from Wordnet, the concept definitions in Hownet are not in a tree form in which the distance between them can be directly calculated, instead, they are in multidimensional forms. The concept definition is a description of the word semantic, and one word may have several concept definitions, reflecting several different meanings. The definitions consist of a series of sememes, which refer to the smallest basic semantic unit, and these sememes are divided to nine main categories, such as Entity and Event categories, and they have Hypernym-Hyponym relations among them.

2.1 Tree Form of Sememes and Expression Power of Sememes

As far as the sememes in Hownet have Hypernym-Hyponym relations, the father node is always abstract than the child node. The accessional tables in Hownet give the Hypernym-Hyponym relations of the 1505 sememes, and we can build several sememe trees for further process. In these trees, every node refers to one sememe, and its position shows its relation to other sememes. We take three factors into consideration for the expression power. First, the nodes in different level should have different expression power, the different value should be given when we extract them from the word. Second, if a sememe has more child nodes, its value should be more reduced, as it can be divided into several detailed concepts in Hownet, and its child nodes would be chosen to describe the word unless the concept is abstract. Third, the different sememe trees should have different expression powers, the different tree roots have the different original values. For example, the Attribute tree may has small expression powers, while Entity tree are strong in description and their root value is higher. The formula is shown as follows:

$$k(m) = Wtree_i \cdot [\log((\mathrm{Deep}_k + 1.0)/2) + \mathrm{a} + \frac{1}{Child_k + \mathrm{b}}] . \quad (1)$$

Where, $k(m)$ is the expression power of sememe m; k is the node referring to m in the sememe tree; $Wtree_i$ is the root value of tree i which the node k is in; Deep_k is the height of the node k. Because the difference among different levels would be too much if the height is used as their weighing value, therefore we use the logarithm value of the height. Meanwhile, as the range of the tree height is from 0 to 12, we divided the value by 2 in order to make the result better; $Child_k$ is the number of its child nodes, its range is from 0 to 9, and we set a mediating factor, b, to protect the calculation from invalid value; a is also a mediating factor, which avoid the result being too small. The reflection between the sememe m and the node k can be found in the corresponding table in our system, and the height and child node number are also described in an index table.

2.2 Expression Power of Concept Definition

As there are some sememes in Hownet which are abstract and occur in a lot of definitions of the words, and do not contain enough information to describe them, we need to separate the sememes into two kinds according to their expression power, the strong one and the weak one. If the definition of a certain word contains only weak ones, it means that it does not describe the word accurately and the information gain is not enough, thus we should not extract the words to their definition. To calculate the expression power of a definition, we set a threshold to filter the weak ones and reserve the word which cannot be extracted. In this way, the feature set consists of both words and sememes, the expression power of definition c can be calculated as follows:

$$\mathrm{f}(c) = \max_j k(c_j) . \quad (2)$$

Where $k(c_j)$ is the weight of sememe j in concept definition c; (2) calculates the expression power of all the sememes, and chooses the highest value as the expression power of the definition.

2.3 Threshold in Concept Extraction

To every definition, if its expression power is high enough, we can extract it from the word, otherwise, it should not be extracted. In order to decide when to extract it, we need a threshold to divide the definition into two parts, which can be extracted or cannot. When the definite value is higher than the threshold, this word is extracted and the sememes are added into the feature set. Otherwise, the original word is added instead.

2.4 Combined Weighing Method: CHI-MCOR

Analysis of feature set. When we extract the concept attribute to construct the feature set, we convert a lot of words into the concept features, and get rid of the influence of the synonymy and dependence, which makes the classification precision much higher. However, because of the mass of weak concept and the words which are not in the Hownet, some Chinese words are given a comparatively lower weight and become the middle or low occurring feature. There are still some specialty words and proprietary words which are only occur in one category and are not highly occurred in the whole documents, however these words are very important in classification. These words need a strategy to get a higher weight and contribute more in text classification, we propose the weighing methods.

The comparing result of seven weighing methods. We select seven common weighing methods and test them, and focus mainly on their selection strategy and classification precision. The table 1 gives us the experimental results. From the analysis of the selected features, we find the following facts.

Fact 1. The DF (Document Frequency), TF-IDF (Term Frequency-Inverse Document Frequency), CET (an improved method of Information Gain), CDW (Category-Discriminating Word) and CHI (χ statistics) methods prefer the high-occurred words and they are greatly related. In our experiment, CHI is the best method.

Fact 2. The MCOR (Multi-Class Odds Ratio) method mainly chooses the middle and low occurred features, so its classification precision is low when the reduction rate is high. But with the increase of the feature dimension, its precision is increased highly and when the feature dimension is above 4000, its precision is higher than CDW, CET, DF, TF-IDF and MI (Mutual Information) methods.

Fact 3. The MI method mainly selects the high and middle occurred features, it can get a good classification precision but with the increase of the feature dimension, the precision is not improved visibly.

The CHI selection method. The CHI weighing method's formula is shown as follow:

$$\chi^2(t,c) = \frac{N*(AD-CB)^2}{(A+C)(B+D)(A+B)(C+D)} . \tag{3}$$

$$\mathrm{x}^2_{\max}(t) = \max_{i=1}^{m} \chi^2(t,c_i) . \tag{4}$$

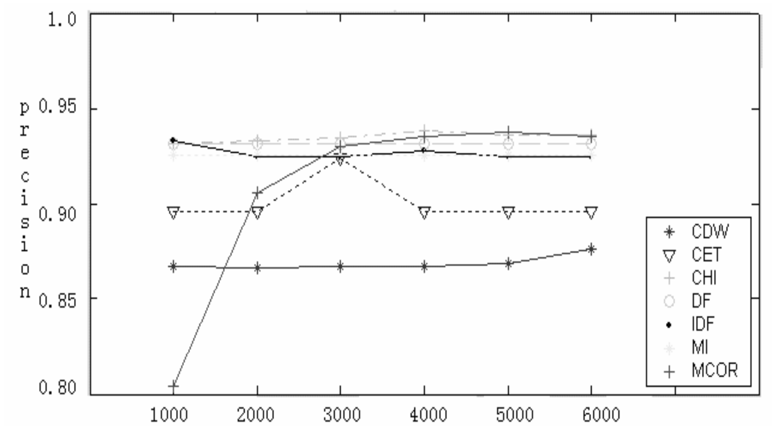


Fig. 1. The average of seven different weighing methods. Y axis is the average precision, and x axis is the feature dimension of the training set.

Where, *N* is the total document number of the training set, *c* is a certain category, *t* is a certain feature, *A* is the number of the document which belong to category *c* and contain feature *t*, *B* is those which do not belong to category *c* but contain feature *t*, *C* is those which belong to category *c* but do not contain feature *t*, *D* is those which do not belong to category *c* and do not contain feature *t*.

The CHI method is based on such hypothesis: if the feature is highly occurred in a specified category or highly occurred in other categories, it is useful for classification. Because CHI take the occurrence frequency into account, it prefers to select highly occurred words, and ignored the middle and low occurred words which maybe important in classification.

The MCOR selection method. The MCOR weighing method's formula is shown as follow [1]:

$$MCOR(t) = \sum_{i=1}^{m} P(C_i) \left| \log \frac{P(t / C_i)(1 - P(t / C_{else}))}{(1 - P(t / C_i))P(t / C_{else})} \right| . \quad (5)$$

Where, $P(C_i)$ is the occurrence probability of category C_i, $P(t / C_i)$ is the occurrence probability of the feature *t* when category *C* is occurred, $P(t / C_{else})$ is the occurrence probability of the feature *t* when category *C* is not occurred. When $P(t / C_i)$ is higher or $P(t / C_{else})$ is lower, the weight of MCOR is higher. Therefore, the MCOR selects the features which are mainly occurred in one category and nearly not occurred in other categories. Because it does not consider the occurrence frequency of the feature, it prefer to select the words which are middle or low occurred in the document while highly occurred words are always occurred in more than one categories.

The Combined Method: CHI-MCOR. Because MCOR mainly selects the words whose occurrence frequencies are middle or low, its classification precision is low when the reduction is high. But with the increase of feature dimension, its precision is improved to an appreciable level. And CHI prefers to select the words whose occurrence frequencies are high, and it is one of the best feature selection methods [1]. As a result, when we combine the both methods, we can make the advantages together and get a high classification precision [7]. Therefore, we give a combined weighing method based on CHI and MCOR, it is shown as follows [8]:

$$V(t) = \lambda V_{CHI}(t) + (1-\lambda)V_{MCOR}(t), \quad 0 < \lambda < 1. \tag{6}$$

Where, V_{CHI} is the weight of feature t when CHI method is used, V_{MCOR} is that when MCOR method is used.

When we analysis the weigh given by the both methods, we find that the average weight of the features are different. For example, when the reduction is 50%, the range of the CHI weight is (2.1, 6.81), while that of MCOR is (1.15, 1.76). Because CHI gives a much higher weight to all the features and its swing is wider, we should give a comparatively lower value to λ. If not, the value depends too much on CHI and the combined weigh method is meaningless. So we need a proper value of λ. In experience, we suppose that when the average weight of CHI and MCOR is the same, we can get the both advantage and the classification precision will be the highest. Therefore, we think that the best λ is as follows:

$$\frac{\lambda}{1-\lambda} = \frac{Mean(MCOR)}{Mean(CHI)}. \tag{7}$$

2.5 Competitive Learning Network

Competitive learning network usually consists of an input layer of N fan-out neurons and an output layer of K processing neurons. The neurons of output layer are full connected to that of input layer. The learning algorithm is shown as follows [4]:

Step 1: Initialize the small random weights.
Step 2: Each output neuron calculates the distance between the input vector and the weight vector connecting the input to it, if the i^{th} index of the closest competitive neuron is found and set its output activation to 1, the activation of all other competitive neurons to 0.
Step 3: The weight of the only winning neuron is updated so as to make the weights of the winning neuron closer to the current input pattern. Thus, the neuron whose weight vector was closest to the input vector is updated to be even closer.
Step 4: Check if the stopping criterion is satisfied, if no more changes of the weight vectors or the maximum epochs is reached, then learning stop, else go to Step 2.

The winning node is more likely to win the competition again when a similar input is presented, and less likely to win when a very different input is presented. Thus, the competitive network clusters the input data, the output vectors of resulting network represent to class indices.

2.6 RBF Network Classifier

In the RBF network, the activation function of the hidden nodes is Gaussian function [9], the input of the hidden node i is the product of threshold b_i and the Euclidean distance between weight vector W and input vector X:

$$\mathrm{k}_{\mathrm{i}}^{\mathrm{q}} = \sqrt{\sum_{j}(X_j^q - W_{ji})^2} \times b_i \cdot \tag{8}$$

Where, X^q_j is the j^{th} component of the q^{th} input vector, W_{ji} is the weight between the jth node in the input layer and the ith node in the hidden layer, b_i is a threshold to control the accuracy of the Gaussian function. The output of the same node is shown as follows:

$$\mathrm{r}_{\mathrm{i}}^{\mathrm{q}} = \exp(-(\mathrm{k}_{\mathrm{i}}^{\mathrm{q}})^2) = \exp(-\sqrt{\sum_{\mathrm{j}}(x_j^q - w_{ji})^2} \times \mathrm{b}_{\mathrm{i}}) \cdot \tag{9}$$

Instead of adjusting b_i, we can use the parameter of *spread* in Neural Network Toolbox of Matlab 7.0 to control the performance of the network. The larger *spread* is, the smoother the function approximation will be. The activation function of the network output is linear. The RBF network has a strong capability of approximation to the kernel vector in a limited part of the whole net, its training is divided into two processes. The first is unsupervised learning, which adjusts the weight vector between the input and hidden layers. The other is supervised learning, which adjusts the weight vector between the hidden and output layers.

3 Experiments

3.1 Experimental Data

The experimental corpus is based on 1204 documents from People's Daily from 1996 to 1998, all of which are first classified manually. These documents are separated into two parts: the training documents and the testing documents. The training corpus contains 1084 documents in six different categories, and the rest 120 documents are used as testing samples, with 20 documents in each category. Instead of using word frequency, we use the combined weighing method CHI-MCOR as features of each document as described in above, the dimension of the feature vector is 500.

3.2 Classification by the Competitive Network

In our experiment, we set the Kohonen learning rate of the competitive network to 0.01. To prevent the dead nodes problem from happening, we set the conscience mechanism to the competitive network. The network is unable to cluster the documents when conscience rate is larger than 0.001 or too close to zero, thus the conscience rate is set 0.000005. The clustering result of the competitive network is listed in Table 2.

Table 2. Experimental result of the competitive network

Positive accuracy	Negative accuracy	Average accuracy
0.5438	0.9159	0.7299

3.3 Classification by the RBF Network

The value of *spread* is 1.2. After 1084 iterations, the training completes. This process takes only 3 minutes or so. Then another 120 texts are used for testing. Comparing with the classification that has been made manually, we can get the precision, recall and *F*1 value of the classification made by RBF network. The experimental results are listed in the Table 3.

Table 3. Experimental result of the RBF network, *F*1=0.7722

	Economics	Politics	Computer	Sports	Education	Law	Average
P	0.615	0.435	1	1	0.947	.722	0.7866
R	0.8	0.5	0.8	0.9	0.9	0.65	0.7583

From the experimental results we can know that the classifier based on competitive network can classify document of different categories correctly, represented by a high negative accuracy rate. However, a low positive accuracy rate shows that many documents from different categories are not clustered together. This is partly because it is difficult to have information about the correct category in the unsupervised learning. In the experiment of the RBF network, the precision and recall are low in some categories, such as Politics category; the performances are fairly nice in Computer and Sports categories. The RBF network performs unsatisfactorily in the same category, the Politics, probably due to that some proper nouns have not been correctly recognized in certain categories and thus lost some important information about the text. Feature words in politics often have vague boundaries with other fields, such as economics, culture, military affaires, and so on. While sports and computer are highly characteristic fields, it is much easier choosing their feature words. Besides, the number of training documents are slightly too small compared to the dimension of the feature vector and the complexity of text classification.

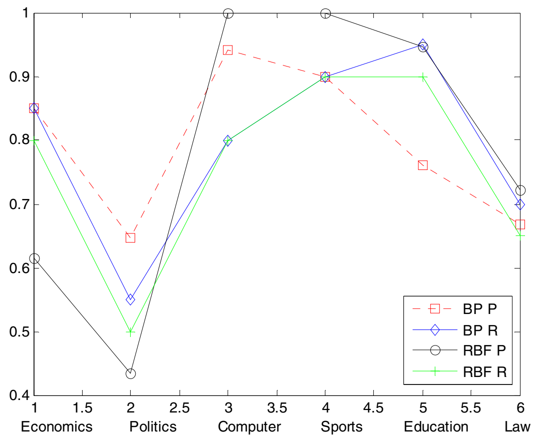


Fig. 2. Performance comparison between the BP and RBF network

At the same time, we do also experiment of the BP network, the result shows that the RBF network takes less than one-tenth of the training time BP takes, the RBF network performs well in the categories in which the results are satisfactory. In the sports category, it even outperforms the BP network.

4 Conclusions

When semantic sememes are used as the features set of text classification, we can efficiently reduce the feature dimension and reflect the original feature space to a more stable and smaller one. By setting a proper threshold, we can reserve the word whose concept definition is weak in expression. Meanwhile, as every sememe has its own expression power, we give them different values according to their expression power and relation to the word. Experimental results show that this combined feature set is much better than the word one or the semantic one. Because there are some words which are not highly occurred but useful in text classification, while the words with high occurrence frequency is usually useful, except the words in the stop word dictionary which are frequently used in the text but useless in classification, our CHI-MCOR method to take balance in the high occurring ones and the middle occurring ones. This method not only selects the highly occurring words, but also selects the words whose occurrence frequencies are middle or low but only occur in one or two categories. It is much better than CHI or MCOR method alone.

The competitive network sometime cannot cluster documents of the same category together, because it lacks category information as a method of unsupervised learning. As of the supervised learning methods, the RBF network shows its quickness in training and it can even outperform the BP's capability, especially being modified in some means. As a classifier, the RBF network can be a good substitute for the BP network, when the selected features are clear enough for the RBF network itself to produce satisfactory results.

Acknowledgement

This work was supported by National Natural Science Key Foundation of China (No. 60331010) and National Natural Science Foundation of China; Tsinghua University 985 Research Foundation; the Excellent Young Teacher Program, Ministry of Education, China; and State Key Lab of Pattern Recognition Open Foundation, Chinese Academy of Sciences.

References

1. Liao, S., Jiang, M.: An Improved Method of Feature Selection Based on Concept Attributes in Text Classification. In: Wang, L., Chen, K., and Ong, Y. S. (eds.): ICNC 2005, Lecture Notes in Computer Science, 3610 (2005) 1140 - 1149
2. Mlademnic, D., Gtobelnik, M.: Feature Selection for Unbalanced Class Distribution and Naïve Bayees. Proceedings of the Sixteenth International Conference on Machine learning, (1999) 258-267

3. Yang, Y., Pedersen, J. O.: A Comparative Study on Feature Selection in Text Categorization. In: Proceedings of the 14th International Conference on Machine Learning, (1997) 412-420
4. Wang, L., Jiang, M., Lu, Y., Noe, F., and Smith, J. C.: Clustering Analysis of Competitive Learning Network for Molecular Data. In: Wang, J. et al (eds.): ISNN06, Lecture Notes in Computer Science, 3971 (2006) 1244-1249
5. Wang, L., Jiang, M., Lu, Y., Noe, F., and Smith, J. C.: Self-Organizing Map Clustering Analysis for Molecular Data. In: Wang, J. et al (eds.): ISNN06, Lecture Notes in Computer Science, 3971 (2006) 1250-1255
6. Dong, Z., Dong, Q.: The Download of Hownet [EB/OL], http://www.keenage.com
7. Wang, L., Jiang, M., and Liao, S., et al.: A Feature Selection Method Based on Concept Extraction and SOM Text Clustering Analysis. International Journal of Computer Science and Network Security, 6 (1A), (2006) 20-28
8. Liao, S., Jiang, M.: A Combined Weight Method Based on Concept Extraction in Automatic Classification of Chinese Text. Proceedings of 2005 IEEE International Conference on Neural Networks and Brain, (2005) 625-630
9. Martin, T., Hagan, H., Demuth, B., and Beale, M.: Neural Network Design. PWS Publishing Company (1996)

Ontology Learning from Text: A Soft Computing Paradigm

Rowena Chau[1], Kate Smith-Miles[2], and Chung-Hsing Yeh[1]

[1] Clayton School of Information Technology,
Faculty of Information Technology,
Monash University, Clayton, Victoria 3800, Australia
{rowena.chau, chunghsing.yeh}@infotech.monash.edu.au
[2] School of Engineering and Information Technology
221 Burwood Highway, Burwood, Victoria 3125, Australia
katesm@deakin.edu.au

Abstract. Text-based information accounts for more than 80% of today's Web content. They consist of Web pages written in different natural languages. As the semantic Web aims at turning the current Web into a machine-understandable knowledge repository, availability of multilingual ontology thus becomes an issue at the core of a multilingual semantic Web. However, multilingual ontology is too complex and resource intensive to be constructed manually. In this paper, we propose a three-layer model built on top of a soft computing framework to automatically acquire a multilingual ontology from domain specific parallel texts. The objective is to enable semantic smart information access regardless of language over the Semantic Web.

1 Introduction

The Semantic Web [2] will augment the current WWW with ontological knowledge in order to overcome information overload by allowing semantically precise information search and retrieval. Ontology is the carrier of meaning that makes the Semantic Web content explicit. Despite the benefits that the Semantic Web promises, ontology availability and multilinguality pose two major challenges towards realization of the Semantic Web vision [1]. At the ontology level, the issue of multilinguality concerns the development of multilingual ontology, by which semantic annotation and reasoning of the multilingual Web content can be based on. However, manual development of multilingual ontology can be very time consuming and require extensive expertise. The need for automation is evident. To automate ontology development, ontology learning from text is a viable approach [3]. Currently, studies in automatic learning of ontology from text are mainly limited to the monolingual environment. To achieve this in a multilingual context, this paper proposes a soft computing framework to automate the acquisition of multilingual ontology from domain specific parallel texts. The objective is to enable semantic smart information access over the Web regardless of language. This paper is organized as follow: In Section two, an overview of the soft computing framework for automatic ontology development is presented. Following this, technical details of the framework, including the construction of a fuzzy multilingual association network, the self-organizing acquisition of the language-neutral concepts, and the formation of the ontology's growing self-organizing concept hierarchy, will be discussed in Section three, four and five, respectively. Finally, a conclusive remark is given in Section six.

2 The Soft Computing Ontology Learning Framework

A multilingual ontology can be conceived as a language-neutral concept hierarchy populated with domain-specific lexical entities (i.e. terms) in multiple languages. Semantic robustness towards lexical diversity among languages is thus crucial for developing a multilingual ontology. To cope with the linguistic diversity within a multilingual domain, a three-layer model, built on a soft computing framework, to multilingual ontology development, as depicted in Figure 1, is proposed. This model consists of three layers, namely, the lexical layer, the semantic layer and the structural layer. In the lexical layer, we establish the semantic association among semantically-similar multilingual lexical entries, which are content-bearing terms relevant to the domain of ontology. In the semantic layer, we develop a semantic unification scheme by considering each group of semantically-related multilingual terms as an individual language-neural concept. As such, linguistic diversity among multiple languages is unified. Finally, in the structural layer, we develop a language-neural concept hierarchy at the ontology structure level based on the notion of concept subsumption.

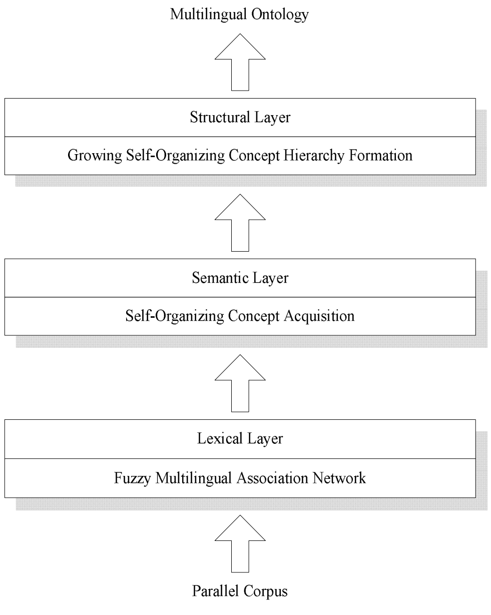

Multilingual Ontology
Structural Layer
Growing Self-Organizing Concept Hierarchy Formation
Semantic Layer
Self-Organizing Concept Acquisition
Lexical Layer
Fuzzy Multilingual Association Network
Parallel Corpus

Fig. 1. The three-layer soft computing ontology development framework

This three-layer model is built on top of a hybrid soft computing framework incorporating intelligent techniques from neural networks and fuzzy logic. First, for the lexical layer, a fuzzy multilingual association network that models the semantic relatedness among multilingual lexical entities is developed. This fuzzy multilingual association network will associate every term with all its cross-lingual related ones and indicate their semantic relatedness as a fuzzy association measure defined based on fuzzy set theory [8]. Second, for the semantic layer, we acquire language-neutral concept using neural network techniques based on the self-organizing map (SOM) [5]. Each lexical entry's semantic relatedness vector will be used as input to the self-organizing map to discover concepts by grouping semantically-similar lexical entries into clusters. Third, for the structural layer, we form a concept hierarchy using the language-neutral concepts as input. To do so, we discover the hierarchical relationship among concept within the ontology by applying the growing hierarchical self-organizing map (GHSOM) [4].

3 Fuzzy Multilingual Association Network

A fuzzy multilingual association network can be defined as an information structure consisting of sets of lexical entities in multiple languages and a specification of their cross-lingual semantic association. For automatically constructing a fuzzy multilingual association network, this approach involves the processing a parallel corpus, consists of identical texts in multiple languages, to exploit the cross-lingual semantic association among multilingual lexical entities. To determine the meaning of each lexical entity based on its corresponding degree of relevance to the documents, corpus statistics of the lexical entities' occurrences are analyzed. With each lexical entity's meaning as an integration of its document relevance, the lexical meaning of each lexical entity is then represented as a fuzzy subset of documents with the relevance degrees between the lexical entity and documents as the membership values. Based on the similarity of meanings, a degree of cross-lingual semantic association is computed. To get an association network that will allow partial association, a fuzzy relation representing the semantic relation of cross-lingual semantic association is established. Thereby, a fuzzy multilingual association network relating lexical entities across languages with their degrees of semantic association, ranging from 0 to 1, is constructed.

Given a parallel corpus D in two languages, L_A and L_B, we have:

$$D = \{d_k\} \tag{1}$$

where $d_{k\in\{1,2,\ldots z\}}$ is a parallel document containing identical text in both L_A and L_B versions.

Two sets of lexical entities, A and B, are extracted from the parallel corpus D using language-specific noun-phrase parsing techniques adopted from natural language processing research.

$$A = \{a_i\} \qquad \text{where } a_{i\in\{1,2,\ldots x\}} \text{ is a term of } L_A. \tag{2}$$

$$B = \{b_j\} \qquad \text{where } b_{j\in\{1,2,\ldots y\}} \text{ is a term of } L_B. \tag{3}$$

For the establishment of semantic association to be encoded in the fuzzy multilingual association network, meaning of lexical entities has to be determined. In our approach, each document of the parallel corpus is viewed as a specific semantic unit contributing to the meaning of the lexical entities it contains. As such, each document containing a lexical entity is considered contributing to the totality of the semantic content represented by the lexical entity as a whole. Accordingly, the degree of relevance between a lexical entity and a document is revealed by the lexical entity's relative frequency within a document. By counting the relative frequency of each lexical entity within every document of the parallel corpus, lexical meaning of every lexical entity is then represented as a fuzzy subset of its contributing documents with the degrees of relevance between lexical entity and documents as membership values.

For $a_i \in A$, its lexical meaning is represented by:

$$a_i = \sum_{d_k \in D} \mu_{a_i}(d_k)/d_k \tag{4}$$

where

$$\mu_{a_i}(d_k) = \frac{Frequency\ of\ a_i\ in\ L_A\ version\ of\ d_k}{Length\ of\ L_A\ version\ of\ d_k} \tag{5}$$

Similarly, for $b_j \in B$, its lexical meaning is represented by:

$$b_j = \sum_{d_k \in D} \mu_{b_j}(d_k)/d_k \tag{6}$$

where

$$\mu_{b_j}(d_k) = \frac{Frequency\ of\ b_j\ in\ L_B\ version\ of\ d_k}{Length\ of\ L_B\ version\ of\ d_k} \tag{7}$$

A fuzzy multilingual association network FN_{AB}, involving two languages, L_A and L_B, modeling the semantic relation of cross-lingual semantic association is expressed as a fuzzy relation [6, 7] $FN(A,B)$ as follows:

$$FN(A,B) = \begin{bmatrix} \mu_{FN}(a_1,a_1) & \cdots & \mu_{FN}(a_1,a_x) & \mu_{FN}(a_1,b_1) & \cdots & \mu_{FN}(a_1,b_y) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mu_{FN}(a_x,a_1) & \cdots & \mu_{FN}(a_x,a_x) & \mu_{FN}(a_x,b_1) & \cdots & \mu_{FN}(a_x,b_y) \\ \mu_{FN}(b_1,a_1) & \cdots & \mu_{FN}(b_1,a_x) & \mu_{FN}(b_1,b_1) & \cdots & \mu_{FN}(b_1,b_y) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mu_{FN}(b_y,a_1) & \cdots & \mu_{FN}(b_y,a_x) & \mu_{FN}(b_y,b_1) & \cdots & \mu_{FN}(b_y,b_y) \end{bmatrix} \tag{8}$$

where

$$\mu_{FN}(a_i,b_j) = \frac{|\mu(a_i) \cap \mu(b_j)|}{|\mu(a_i) \cap \mu(a_i)|} = \frac{\sum_{d_k \in D} min(\mu_{a_i}(d_k) \bullet \mu_{b_j}(d_k))}{\sum_{d_k \in D} max(\mu_{a_i}(d_k) \bullet \mu_{b_j}(d_k))} \tag{9}$$

is defined as the degree of cross-lingual semantic association between two terms, a_i and b_j, based on the similarity of their meanings. In other words, in terms of meanings, if b_j is similar to a_i as a_i is similar to a_i itself, then $\mu_{FN}(a_i, b_j) = 1$ and b_j is the translation equivalent of a_i.

4 Self-Organizing Concept Acquisition

To acquire language-neural concepts, which can effectively characterize the domain knowledge of a multilingual ontology, the fuzzy multilingual association network is used. Contextual contents of every multilingual term represented as their fuzzy semantic association vector in the fuzzy multilingual association network are used as the input for the self-organizing map algorithm to find concepts.

Let $\mathbf{x}_i \in R^M$ ($1 \le i \le M$) be the fuzzy semantic association vector of the i^{th} multilingual lexical entity, where M is the total number of multilingual lexical entities. The self-organizing map algorithm is applied to discover the language-neural concepts by clustering the multilingual lexical entities, using these lexical entities' fuzzy semantic association vectors as the input to the self-organizing map. The map consists of a regular grid of nodes. Each node is associated with an M-dimensional model vector. Let $\mathbf{m}_j = \lfloor m_{jm} | 1 \le m \le M \rfloor$ ($1 \le j \le G$) be the model vector of the j^{th} node on the map. The self-organizing concept acquisition algorithm is given below.

Step 1: Select a fuzzy semantic association vector $\mathbf{x}_i$ of a lexical entity at random.

Step 2: Find the winning node s on the map with the vector $\mathbf{m}_s$ which is closest to $\mathbf{x}_\mathrm{i}$ such that

$$\|\mathbf{x}_i - \mathbf{m}_s\| = \min_j \|\mathbf{x}_i - \mathbf{m}_j\| \tag{10}$$

Step 3: Update the weight of every node in the neighborhood of node s by

$$\mathbf{m}_t^{new} = \mathbf{m}_t^{old} + \alpha(t)(\mathbf{x}_i - \mathbf{m}_t^{old}) \tag{11}$$

where $\alpha(t)$ is the gain term at time t ($0 \le \alpha(t) \le 1$) that decreases in time and converges to 0.

Step 4: Increase the time stamp t and repeat the training process until it converges.

After the self-organizing process is completed, each multilingual lexical entity is mapped to a grid node closest to it on the self-organizing map. A partition of multilingual lexical entity space, represented by a map of language-neutral concepts, is thus formed. This process corresponds to a projection of the multi-dimensional fuzzy semantic association vectors onto an orderly two-dimensional concept space where the proximity of the multilingual lexical entities is preserved as faithfully as possible. Consequently, conceptual similarities among the multilingual lexical entities are explicitly revealed by their locations and neighborhood relationships on the map.

Multilingual lexical entities that are synonymous are associated to the same node. In this way, conceptual related multilingual lexical entities are organized into clusters, representing all domain-specific concepts, within a common semantic space. The problem of linguistic diversity in a multilingual environment is thus overcome.

5 Growing Self-Organizing Concept Hierarchy Formation

The multilingual ontology modeled as a concept hierarchy is generated with the application of the growing hierarchical self-organizing map algorithm using the concept prototype vectors obtained from the self-organizing map in the previous stage as inputs. The algorithm for the formation of the concept hierarchy is given below.

Step 1: Start the growing hierarchical self-organizing map (GHSOM) with layer *0* consisting of only one node whose weight vector m_0 is initialized as the average of all concept prototype vectors y. Then calculate the *mean quantization error* (*mqe*) of this single node by the Euclidean distance between the node and all concept prototype vectors mapped to it such that

$$mqe_0 = \frac{1}{d}\left\| m_0 - y \right\| \tag{12}$$

where d is the total number of concept prototype vectors y.

Step 2: Initialize layer *1* of the GHSOM as a small self-organizing map (SOM) of 2x2 nodes, which is trained with all concept prototypes according to the standard self-organizing map algorithm.

Step 3: Evaluate the mapping quality by calculating the *mean quantization error* of each node's *mqe* in the current layer to determine the *error node* according to the largest deviation between its weight vector and the input vectors mapped to it. Then, either a new row or column of nodes is inserted between the error node and its most dissimilar neighbor. The weight vectors of these new nodes are initialized as the average of their neighbors. After the insertion, training of the newly grown map continues according to the standard SOM algorithm. After a fixed number of iteration, calculate the *mean quantization error* (MQE_m) of the current map m with i nodes such that:

$$MQE_m = \frac{1}{u} \cdot \sum_i mqe_i \tag{13}$$

where mqe_i is computed as the average Euclidean distance between node vector m_i and the concept prototype vectors mapped to node i. The map m grows until its MQE_m is reduced to a predefined fraction (the growing-stopping criterion τ_1) of the mean quantization error mqe_p of the parent node p in the preceding layer of the hierarchy such that:

$$MQE_m < \tau_1 \cdot mqe_p \tag{14}$$

Step 4: Examine the nodes of map m for hierarchical expansion. Those nodes having a large mean quantization error will be expanded by adding a new SOM to the subsequent layer of the GHSOM. The parameter τ_2 is used to specify the level of granularity desired for the final hierarchy. Each node *i* fulfilling the criterion given in Equation (15) is subject to hierarchical expansion.

$$mqe_i < \tau_2 \cdot mqe_0 \tag{15}$$

The training process and node insertion procedure continues on the newly established SOM using only concept prototype vectors mapped to the SOM's corresponding parent node. The training and expansion process of the GHSOM will terminate when no more nodes requires further expansion.

The concept hierarchy resulted from the above process is thus a multilingual ontology representing the conceptual knowledge relevant to the domain of the training parallel corpus. This multilingual ontology, providing multilingual ontological knowledge, will thus act as the linguistic knowledge base for various multilingual Semantic Web applications, such as multilingual Semantic Web search engines and multilingual knowledge portal to facilitate linguistically smart and semantically precise multilingual information access.

6 Conclusion

In this paper, an automatic three-layer multilingual ontology development model built on top of a soft computing framework is proposed. By automating the laborious multilingual ontology development process using intelligent soft computing techniques, including fuzzy set theory and neural networks, the knowledge acquisition bottleneck problem (i.e. the difficulty of effectively modeling the knowledge of a particular domain) in ontological engineering is overcome. By making available a multilingual ontology as a linguistic knowledge base of metadata for multilingual Web content annotation, the Semantic Web vision of enabling linguistically smart and semantically precise global information access can be realized.

References

1. Benjamins, V. R., Contreras, J., Corcho, O., Gómez-Pérez, A. Six Challenges for the Semantic Web. *International Semantic Web Conference (ISWC2002)*, Sardinia, Italia, (2002). http://iswc2002.semanticweb.org/posters.html.
2. Berners-Lee, T., Hendler J., Lassila, O. The Semantic Web. *Scientific American*, 284(5), 34-43 (2001)
3. Buitelaar, P., Cimiano, P., Magnini, B. (Eds.) *Ontology learning from text: Methods, evaluations and applications*. IOS Press. (2005)
4. Dittenbach, M., Rauber, A., Merkl, D. Uncovering hierarchical structure in data using the growing hierarchical self-organizing map. *Neurocomputing*, 48 (2002) 199-216.
5. Kohonen, T. *Self-Organising Maps.* Springer-Verlag, Berlin, (1995)
6. Miyamoto, S. *Fuzzy sets in information retrieval and cluster analysis.* Kluwer Academic Publishers, (1990)
7. Zadeh, L. A. Similarity relations and fuzzy orderings. Information Sciences, vol. 3 (1971) 177-206
8. Zadeh, L. A. Fuzzy sets. *Information and Control,* vol. 8 (1965) 338-353

Text Categorization Based on Artificial Neural Networks

Cheng Hua Li and Soon Choel Park

Division of Electronics and Information Engineering, Chonbuk National University
Jeonju, Jeonbuk, 561-756, Korea
lchjk@msn.com, scpark@chonbuk.ac.kr

Abstract. This paper described two kinds of neural networks for text categorization, multi-output perceptron learning (MOPL) and back-propagation neural network (BPNN), and then we proposed a novel algorithm using improved back-propagation neural network. This algorithm can overcome some shortcomings in traditional back-propagation neural network such as slow training speed and easy to enter into local minimum. We compared the training time and the performance, and tested the three methods on the standard Reuter-21578. The results show that the proposed algorithm is able to achieve high categorization effectiveness as measured by the precision, recall and F-measure.

1 Introduction

Text categorization is a process of classifying documents with regard to a group of one or more existent categories according to themes or concepts present in the contents. The most common application of text categorization is in information retrieval and news classification.

Many different classification methods have been attempted, including the K-Nearest Neighbor[1], Rocchio[2], Decision Tree[3], Neural Networks[4,5] and Support Vector Machines [6].

There are many kinds of neural networks, based on the network topology; we can broadly classify the various neural networks into two classes, feed-forward networks and recurrent networks. Depending on whether an external teacher is present, we can have two different learning paradigms: supervised learning and unsupervised learning. Our described classification methods MOPL and BPNN have been known as linear classifier and non-linear classifier. Perceptron learning has it advantage of fast convergence, and the limitation is only can solve problems that are linearly separable. BPNN has the advantage of yield a good performance but the shortcomings are slow training speed and sometimes easy to enter into local minima. Based on the experience we learn that the limitations of BPNN are related to the morbidity neurons. In this paper, we proposed a novel algorithm based on improved back propagation neural network called MRBP (Morbidity neuron Rectified BPNN) which can detects and rectifies the morbidity neurons, this reformative BPNN divides the whole learning process into many learning phases. It will evaluate the learning mode used in the phase evaluation after every learning phase. This can improve the ability of the neural network, making it more adaptive and robust, so that the network can more easily escape from a local minimum, and be able to train itself more effectively.

The rest of this paper is organized as follows. Section 2 describes the theory of MOPL and BPNN including the basic BPNN theory and improved BPNN method; Experiments are discussed in section 3. The evaluation results are given in section 4. Finally, the conclusion and discussion of future work are given in section 5.

2 Theory of MOPL and BPNN Algorithms

2.1 Basic Theory of MOPL Algorithm

Perceptron had perhaps the most far-reaching impact of any of the early neural networks. The perceptron learning rule is a more powerful learning rule than the Hebb rule. Single neuron can be used to classify two categories, when the categories are more then two, the MOPL can be used. The architecture of the MOPL is shows in Fig. 1. In the network, there is an input layer, an output layer. The weights from the input layer to output layer were adjusted by the perceptron learning rule. For each training input, the net would calculate the response of the output neuron. Then the net would determine whether an error occurred for this pattern (by comparing the calculated output with the target value). If an error occurred for a particular training input pattern, that is $t_j \neq y_j$, then the weights and the biases would be changed. If the error did not occur, the weights and biases would not be changed. Training would continue until no error occurred.

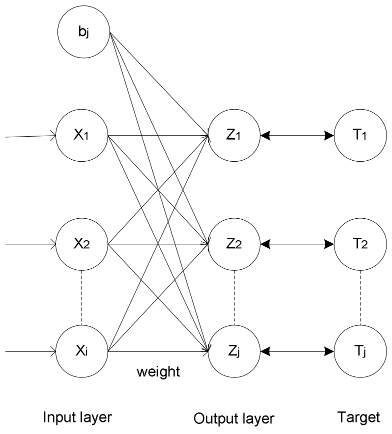


Fig. 1. Architecture of the MOPL

2.2 Basic Theory of the BPNN Algorithm

The back-propagation neural network is a generalization of the delta rule used for training multi-layer feed-forward neural networks with non-linear units. It is simply a gradient descent method designed to minimize the total error (or mean error) of the output computed by the network. Fig. 2 shows such a network.

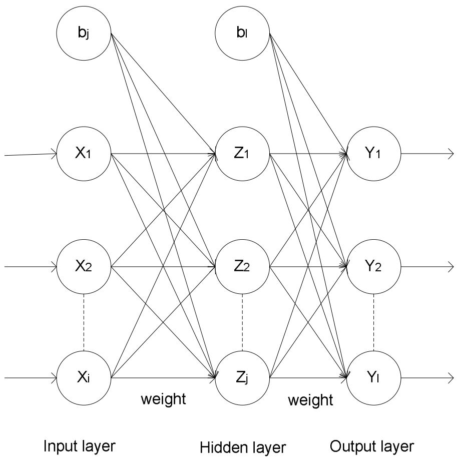


Fig. 2. Typical three layers BP network

In the network, there is an input layer, an output layer, with one or more hidden layers in between them.

During training, an input pattern is given to the input layer of the network. Based on the given input pattern, the network will compute the output in the output layer. This network output is then compared with the desired output pattern. The aim of the back-propagation learning rule is to define a method of adjusting the weights of the networks. Eventually, the network will give the output that matches the desired output pattern given any input pattern in the training set.

The training of a network by back-propagation involves three stages: the feed-forward of the input training pattern, the calculation and back-propagation of the associated error, and the adjustment of the weight and the biases.

2.3 BPNN Defect Analysis and Commonly Used Improved Methods

The three main defects of the BPNN and some commonly used improved methods are as follows:

Slow training speed. In the beginning, the learning process proceeds very quickly, in each epoch, and can make rapid progress, however it slows down in the later stages [7]. There are two commonly used methods of improving the speed of training for BPNNs. *a) Introduce momentum into the network.* Convergence is sometimes faster if a momentum term is added to the weight update formulas. The weight update formula for a BPNN with momentum is

$$W_{ij}(k+1)=W_{ij}(k)+\eta\delta_i x_j+u\left(W_{ij}(k)-W_{ij}(k-1)\right) \tag{1}$$

where momentum parameter u is constrained to be in the range from 0 to 1. The new weights for the training step t+1 are based on the weights at training steps t and t-1. *b) Using the adaptive learning rate to adjust the learning rate.* The role of the adaptive learning rate is to allow each weight to have its own learning rate, and to let the learning rates vary with time as training progresses. The formulas for a BPNN with an adaptive learning rate is

$$\eta^{(n+1)} = \eta^{(n)} \times \frac{E^{n-1}}{E^{n}} \tag{2}$$

where n is the epoch during the training process, and E is the absolute error in each epoch. When E decreases, the learning effect will increase (the weight may change to a greater extent). Otherwise, the learning effect will decrease.

These two kinds of methods accelerate the convergence of the BPNN, but they can not solve other problems associated with the BPNN, especially when the size of the network is large.

Local minimum. When training a BPNN, it is easy to enter into a local minimum, and usually the GA and simulated annealing algorithms have been used to solve this problem. These algorithms can prevent the problem of entering into a local minimum, but they cannot ensure that the network will not enter into a global minimum, and they are even slower than the traditional BPNN.

Network paralyses. During training, the value of the weights may be very large and, consequently, the input of the network will be very large. Thus, the output value of the activation functions, O_j (or O_l), tends to 1, according to the formula of error back propagation, and the back propagation error will tend to 0. This phenomenon is referred to as saturation. The speed of training becomes very slow when saturation occurs. Finally it will cause the weight not to change any more, and this will lead to network paralysis. P.D. Wasserman [8] provided the suggested formula to limit the weight between (-a, a), but it is only used for weight initialization. It cannot prevent the value of the weight increasing during training, and it also has the possibility of leading to network paralysis.

2.4 MRBP Algorithms

The defects mentioned above are all related to saturation, the convergence will become slower and the system will change to a higher learning rate. Also, the weight becomes larger due to the larger learning rate, and this will cause the output value of the activation function to tend to 1. Under this situation, the network can easily enter into a local minimum and ultimately become entrapped by network paralysis. Based on our experience with such problems, we also found that there is another phenomenon which can cause such defects. For some of the neurons, the range of input values is restricted to a small range during each epoch, and this causes the values of the output to be extremely close to each other at each epoch, while the error during each epoch changes slowly. In other words, the speed of convergence is slow. Finally, this situation causes a local minimum or even network paralysis. In this paper, we refer to these two kinds of phenomena as neuron overcharge and neuron tiredness respectively. We call these neurons morbidity neurons. In general, if some morbidity neurons occur within it, then the network cannot function effectively.

The MRBP improved method: During the learning process, neurons face two kinds of morbidity: overcharge and tiredness. If we avoid the appearance of morbidity neurons during the learning phase or rectify the problem in time, then the networks can train and evolve effectively.

[Definition 1]: Neuron overcharged. If the input value of the neuron is very big or very small, it will cause the output value to tend to -1 or 1, and cause the back-propagation error to tend to 0. We refer to such a neuron as being overcharged. That is, for the activation function,

$$f(net_j+\theta_j)=\frac{2}{(1+e^{-\lambda(net_j+\theta_j)})}-1\cdot \tag{3}$$

If $f(net_j+\theta_j)\to 1$ or $f(net_j+\theta_j)\to -1$, then $\delta_j\to 0$. When this happens, we refer to neuron j as being overcharged.

[Definition 2]: Neuron tiredness. If a certain neuron always receives the similar stimulation, then its response to this stimulation will be very similar, so that it is difficult to distinguish different stimulations by its response. We refer to such a neuron as being tired. That is, when neuron j during one learning phase (defined as follows) obeys,

$$\left(\underset{k}{MAX}\, f\left(net_j^k+\theta_j^k\right)-\underset{k}{MIN}\, f\left(net_j^k+\theta_j^k\right)\right)\to 0\cdot \tag{4}$$

When this happens, we refer to the neuron j as being tired.

[Definition 3]: Learning phase. Choosing N iterations (or leanings) as a period, during this period we record some important data, and calculate the effect of the learning process, as the direction for the next period. We called this period the learning phase and, based on our experience, we use 50 epochs as the learning phase.

According to the definition of an overcharged neuron and a tired neuron, we know that they are directly related to the activation function. In the conventional activation function $f(x)=\frac{2}{(1+e^{-\lambda x})}-1$, λ using 1 or other constants, whereas in our model, λ is an adjustable variable. V.P. Plagianakos [9] tried to use an adjustable value of λ in his paper. Actually, different combination of λ corresponds to different learning models.

The determinant rule of the morbidity neuron is: If $f(net_j+\theta_j)>=0.9$ or $f(net_j+\theta_j)<=-0.9$, then neuron j is overcharged. And if $\underset{k}{MAX}\, f\left(net_j^k+\theta_j^k\right)-\underset{k}{MIN}\, f\left(net_j^k+\theta_j^k\right)<=0.2$, then the neuron j is tired.

The formulae used to rectify the morbidity neuron are

$$\theta_j = \theta_j - \frac{\underset{k}{MAX} f\left(net_j^k + \theta_j^k\right) + \underset{k}{MIN} f\left(net_j^k + \theta_j^k\right)}{2} \tag{5}$$

and

$$\lambda_j = -\frac{Ln\left(\frac{2}{1.9} - 1\right)}{\underset{k}{MAX} f\left(net_j^k + \theta_j^k\right) - \underset{k}{MIN} f\left(net_j^k + \theta_j^k\right)} . \tag{6}$$

Formula (5) is used to normalize the maximum and minimum input values in the previous phase in order to make them symmetric with respect to the origin. Formula (6) is used to limit the maximum and minimum output values to the normal range. In our experiments, the range is (-0.9, 0.9). In our study, the morbidity neurons were rectified in each phase after their evaluation.

3 Experiments

3.1 Text Representation

In order to use an automated learning approach, we first need to transform a text into a feature vector representation. It involves several steps: word extraction, stop words removal, word stemming, and term weighting. In our experiment, we employ Porter's stemming algorithms [10] for word stemming, and Okapi rule as term weights,

$$W_{ij} = \frac{tf_{ij}}{tf_{ij} + 0.5 + 1.5 \times \frac{dl}{avgdl}} \times idf_j \tag{7}$$

where $idf_j = \log \frac{N}{n}$, N is the number of documents in the document sets, and n is the number of documents in which the i^{th} term appears; tf_{ij} is the i^{th} indexing term in document j ; dl is the length of document. Okapi rule normalizes the length of documents to replace simple $tf \times idf$.

3.2 Reuters Test Corpus

In order to measure its performance of our system, we tested the system on a standard test collection designed for text categorization used in the literature. This collection is known as Reuters-21578. We chose 1600 documents belonging to the Reuters data set with ten frequent categories. 600 documents were used for training and 1000 documents for testing.

After word stemming, we merged the sets of stems from each of the 600 training documents and removed the duplicates. This resulted in a set of 6122 indexing terms in the vocabulary.

In order to create the set of initial feature vectors to represent the 600 training documents, we measured the term weight for each of the 6122 indexing terms. The feature vectors were then formed term weights, and each of the feature vectors was of the form,

$$D_j = \langle W_{1j}, W_{2j}, ...W_{6122j} \rangle . \tag{9}$$

where W_{ij} is the term weight of the i^{th} indexing term in document j . For each training and testing documents, we created the feature vectors corresponding to the 600 training documents, where each feature vector had a dimensionality of 6122.

3.3 Dimensional Reductions

The main difficulty in the application of a neural network to text categorization is the high dimensionality of the input feature space which is typical for textual data. This is because each unique term in the vocabulary represents one dimension in the feature space, so that the size of the input of the neural network depends upon the number of stemmed-words. In [11], the authors introduce four kinds of methods designed to reduce the dimensional of the feature space. Based on our experience, we reduced this size by choosing the highest term weights. We chose 1000 terms as the neural network's input since it offers a reasonable reduction neither too specific nor too general.

3.4 Experimental Results

The number of output nodes is equal to the number of pre-defined categories. For MOPL, the input layer and output layer have 1000 and 10 nodes. And for BPNN, we select 15 as hidden nodes. Hence, our BPNN has three layers consisting of 1000, 15 and 10 nodes respectively.

Since the MOPL converged very fast, in our experiment we just compare the mean absolute error using traditional BPNN, the commonly used improved method (add the momentum and using adaptive learning rate, we also called Modified BPNN in our paper) and our proposed MRBP network.

In the case of the traditional BPNN, we can see in Fig. 3 that, at the beginning of the training phase, the error is reduced very rapidly. But this reduction slows down after a certain number of epochs, and then levels off. The Modified BPNN is 2-3 times faster than the first method at the beginning of the training phase. It slows down when the error reduction reaches a certain value and then fluctuates. However, in the case of our method, no morbidity neurons are produced in the first learning phase, with the result that the next learning phase is very similar to that of the Modified BPNN. However, from the third learning phase, our method progresses more rapidly than the Modified BPNN. It also has a good tendency in the later part of the training.

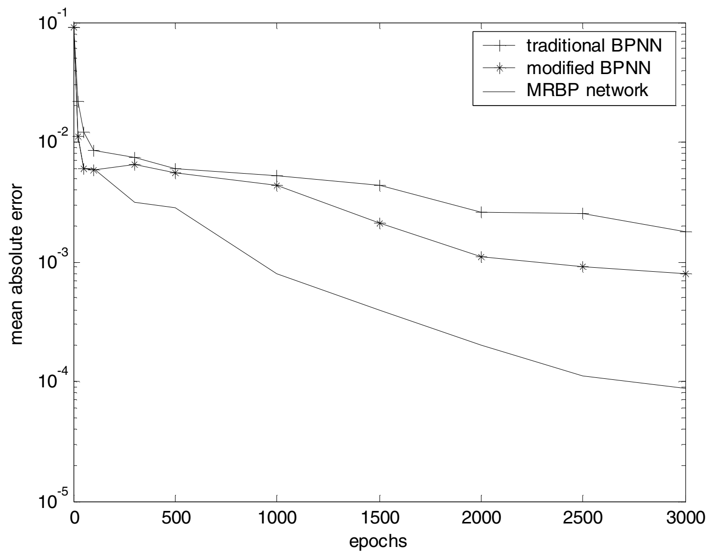


Fig. 3. Mean absolute error reduce during training with three methods

4 Evaluations

The performance of text categorization systems can be evaluated based on their categoryzation effectiveness. We used precision, recall and F-measure to measure our system and compare the performance of the MOPL, Modified BPNN and MRBP network.

We used the macro-average method to obtain the average value of the precision and recall. The F-measure is based on the micro-average value. The performance results are given in table1.

Table 1. Comparison of the performances of the three kinds of networks

Category	MOPL		Modified BPNN		MRBP Network	
	Precision	Recall	Precision	Recall	Precision	Recall
Money-supply	0.848	0.909	0.913	0.916	0.938	0.946
coffee	0.824	0.887	0.882	0.900	0.929	0.933
gold	0.913	0.901	0.944	1.000	0.955	1.000
sugar	0.889	0.853	0.954	0.883	0.927	0.914
trade	0.693	0.767	0.766	0.836	0.824	0.895
crude	0.946	0.903	0.945	0.916	1.000	0.932
grain	0.935	0.929	0.928	0.934	0.948	0.924
Money-fx	0.900	0.834	0.908	0.877	0.918	0.912
Acq	0.927	0.845	0.943	0.893	0.943	0.903
earn	0.932	0.918	0.957	0.934	0.967	0.952
micro-average	0.881	0.875	0.914	0.908	0.935	0.931
F-measure	0.878		0.911		0.933	

Table 2. The network size and parameters

Neural Networks	#Input Nodes	#Hidden Nodes	#Output Nodes	Learning Rate	Momentum	Threshold
MOPL	1000		10	0.01		2
BPNN	1000	15	10	0.01	0.8	

Table 3. The computation time of the networks

Neural Networks	Time	# Iterations	Mean error
MOPL	14.21 s	23	
Modified BPNN	776.55 s	3000	0.000854
MRBP Network	785.32 s	3000	0.000092

The size of the networks and some parameters used in our experiments are given in table 2 and the training time and mean error are given in table 3.

From the table 1, we can see that the MRBP Network outperform the MOPL and Modified BPNN, and from the table 3, the speed of MOPL is much faster than the Modified BPNN and MRBP Network. In fact, Modified BPNN also can converged earlier, for example, we can let the network stop training when the mean error is 0.01, it also take few time to converge, but the performance of F-measure is around 86%, even not as good as MOPL. Hence the BPNN is hard to reach trade-off between the speed and the performance. However, the MRBP Network, if we let it converged at the third learning phase (after 100 epochs), the performance is better than MOPL.

5 Conclusion and Future Works

This paper proposes an algorithm for text categorization using improved Back-propagation neural network. MRBP detects and rectifies the morbidity neuron in each learning phase. This method overcomes the network paralysis problem and has good ability to escape from the local minima. The speed of the training is also been increased and the results of our experiments show that the MRBP network can achieve higher categorization effectiveness than both MOPL and the Modified BPNN. The superiority of MRBP is obvious especially when the size of the networks is large.

Even though the MRBP does not solve the problems associated with the structure of the BPNN, it provides a rule for adjusting the neurons and training the network effectively. In a future study, we intend to analyze and generalize the previous training as a direction for the next trainings.

References

1. Yang, Y. and Liu, X. A Re-examination of Text Categorization Methods. In Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval, (1999) 42-49
2. Rocchio, Jr. J. JRelevance Feedback in Information Retrieval. The SMART Retrieval System: Experiments in Automatic Document Processing, editor: Gerard Salton, Prentice-Hall, Inc., Englewood Cliffs, News Jersey, (1971)

3. Cohen, W. W. and Singer, Y. Context–Sensitive Learning Methods for Text Categorization. ACM Trans. Inform. Syst. 17, 2, (1999) 141-173
4. J. Farkas, "Neural Networks and Document Classification," in: Proceedings of the 1993Canadian Conference on Electrical and Computer Engineering, Vol. I, Vancouver, B. C.,(1993) 14-17.
5. Ruiz, M. E. and Srinivasan, P. Hierarchical Neural Networks for Text Categorization. In Proceedings of SIGIR-99, 22nd ACM International Information Retrieval, (1999) 281-282
6. Joachions Thorsten. Text categorization with support vector machines: Learning woth many relevant features. In European Conference on Machine Learning (ECML) (1998)
7. Wei Wu, Guorui Feng, Zhengxue Li, and Yuesheng Xu Deterministic Convergence of an Online Gradient Method for BP Neural Networks IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 16, NO. 3, (2005)
8. P.D.Wasserman.Neural Computing: Theory and Practice [M].New York: Van Nostrand Reinhold. (1989)
9. V.P. Plagianakos, M.N. Vrahatis, Training Neural Networks with Threshold Activation Functions and Constrained Integer Weights, *ijcnn*, p.5161, IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)-Volume 5, (2000)
10. M. F. Porter. An algorithm for suffix stripping. Program, Vol.14 no. 3 (1980) 130-137
11. Savio L.Y. Lam, Dik Lun Lee, Feature Reduction for Neural Network Based Text Categorization, 6th International Conference on Database Systems for Advanced Applications (DASFAA '99), (1999) 195

Knowledge as Basis Broker — The Research of Matching Customers Problems and Professionals Métiers

Ruey-Ming Chao[1] and Chi-Shun Wang[2]

Dept. of Information Management / Digital Content & Industry Service Lab., Day-Yeh University, Taiwan, R.O.C.
rueyming@mail.dyu.edu.tw

Abstract. With the popularization of the concept knowledge economic management, it not only propels the whole development of knowledge economy but also directs the industry of "Basic Agent Service" of becoming the mainstream in the present markets. This research institute constructed a system called, "*Knowledge-based Broker Service Center*" (KBSC). It allows the customers to submit economic or business field questions online in the form of their natural language. By using Chinese phrase-cutting, key words weighted value calculations, and professional categorizations, it can automatically analyze the nature of the customer's problem and search for the relevant information in the HR database to list the most suitable names of specialists as the assigned coordinator for the clients. When each matching procedure was finished, the questionnaire was given to examine the correctness of the data search following adjustments of the system.

Keywords: Knowledge-based agent/broker, natural language, natural language arrangement, Chinese phrase-cutting, Fuzzy Sets theory.

1 Introduction

With the coming epoch of information, economy, labor, land, and capital being replaced by "Knowledge", it has become the most important tool in businesses' the survival of the fittest. "Knowledge" will not be viewed as a mere resource of the advantaged individuals or corporate organizations, but for everyone [1]. When the whole organization tries to integrate the internal and external knowledge resources by using "knowledge management" to enhance their competitive force, they usually adopt the external knowledge to offset the balance due to reasons such as the deficiency of internal knowledge resources, technology bottleneck, prime cost consideration etc [2], [13].

This research is based on the knowledge system of matching the customer's problems vs. professional specialties. By using fuzzy sets theories, automatically file categorized theories and techniques, it has built up a service center system [3]. This system enables the customers to submit questions online in the form of "natural language" and categorize and analyze automatically in response to the questions and problems. When each matching procedure was completed, the questionnaire was given to examine the correctness of the search as the data for the following adjustment of the system. In the long run, through sequence of adjustments, it is possible to cater to customer's needs and demands.

The knowledge based theory in this research was designed and constructed under the substances in real life. KBSC system invited the management research development center from central Taiwan to evaluate and research on the section of industrial service. This center is aimed for the goals and plans of "Modernization of Commerce" and "Business Management — Automatized Technology Education" each separately under the instruction of Department of Economy and Board of Education.

2 Relevant Research and Techniques

2.1 Knowledge Market

The sharing of knowledge shares a similarity towards the trading goods on the market [4], [5], [12]. When people try to seek solutions for complicated and uncertain problems, they usually turn out for help. This is precisely the background for the rising of the "knowledge buyer". The "knowledge seller", however, is the person who owns the profound knowledge, experience and holds keys to the solution during the process. The two parties decided on a satisfying price after a series of communications and negotiations. In addition, the "knowledge broker" combines the two sides, the demands and the supplies to become the bridging medium.

2.2 Knowledge Broker

Sharon [7] brought up that the broker only provides the connection to the recessive knowledge sources to the demanders. Once the function of the bridge is established, the interaction is solely between the two parties and the agent doesn't usually get involved in the actual transformation of the knowledge content.

2.3 Chinese Phrase-Cutting

In a natural language, the most basic unit is usually a "phrase". Here, the phrase is defined by linguistics as "the smallest language element that can be independently used and include a complete meaning." Many researches concerning Natural Chinese Application such as document index, Chinese input, and machinery translations could only be applied and dealt with using "phrase" as a unit. In the Chinese language, there is no space separating each word, therefore, separating the correct phrase becomes the most basic work in dealing with natural language.

There are three most commonly seen methods of Chinese phrase-cutting: phrase database cutting, lexicon cutting, statistics cutting. In addition to the above mentioned methods of phrase cutting, the utilization of the auto-Chinese phrase cutting 1.0 system invented by the Chinese Knowledge and Information Processing (CKIP), the group of Academia Sinica Chinese Electronic Dictionic Dictionary, can be use in the process. This system not only provides auto-Chinese phrase cutting functions but could also automatically label the syntactic functions and allow the users to choose different lexicons as references based on their needs [7].

2.4 Fuzzy Sets Theory

In the ordinary set theory, the relationship between an element x and a set, A can only be $x \in A$ and $x \notin A$. However in reality, there are a large amount of 'ambiguous' and 'paradoxical' situations. In order to indicate this concept, Zadeh [3] vivified the absolute membership function in the ordinary set theory and allow the character level of the elements to present from the value 0~1. It doesn't only confine within the binary opposition theory of common mathematical set (either 1 or 0) but it can use the membership function to show the reflective relationship between the elements and the character level.

2.5 Automatic Classification

The main function of the Automatic Classification is using computer calculations to find the characters of the documents and categorized them [8]. The Automatic Classification will first proceed words arrangement for the sample information and transform the information into phrase set data and find the characteristic key phrases set to represent the document.

When testing or categorizing new documents, it is also based on the same procedure which is to find of characteristic key phrase sets, weight and calculate them with the ones found in the database. The result of the similarity will be labeled and categorized as the classification of the new document based on the highest value, then the entire automatic classification of the document is done. There are three classical modes for automatic classification: ***Boolean Model***, ***Vector Space Model*** and ***Probabilistic Model***.

3 System Structure

The system structure of the knowledge mediating service is shown in Fig 1. Its main models include: "phrase database collecting model", "questions sorting model", "specialist matching model" and "accuracy adjustment model". Each function as described below:

(1) Phrase Database Collecting Model: This model's main function is to collect all the characteristic key phrase sets and the weighted values from each professional category in selected learning sample to be used for the question sorting model and the accuracy adjustment model.

(2) Question Sorting Model: This model could calculate by ways of proper auto-phrase cutting and calculation of similarity from customer's natural language questions to specify the relevant professional category in respond to the questions.

(3) Specialists Matching Model: The results of the question sorting model will be searched and compared in the specialist database to be listed as the knowledge brokers.

(4) Accuracy Adjustment Model: The accurate examination of the specialist group to the question sorting is in accordance with further adjustments.

The database is separated into two sections, "the key phrase database" and "specialist resource database". The key phrase database mainly stores information and sample documents which are built up by the professional sorting characteristic key phrase sets and customer request questions key phrase sets.

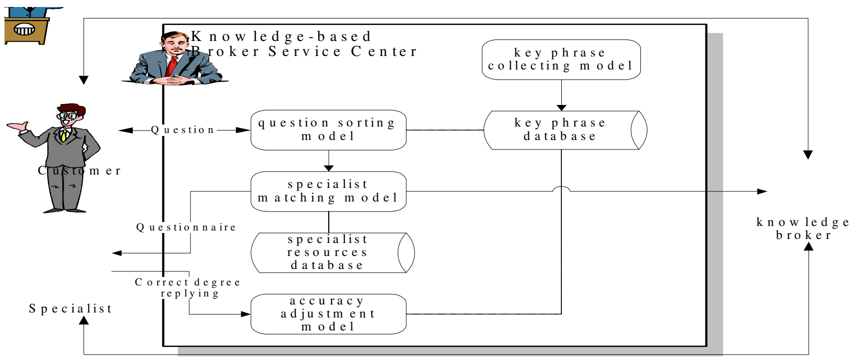


Fig. 1. System Structure Diagram

3.1 Key Phrase Collecting Model

(1) Establishing Professional Category Titles: First, establish the “professional category titles” data list according to the classification from knowledge based mediating center and then transformed these titles to the key phrases of the collected sample.

(2) Collecting Sample Documents: The collecting process was based on the following principles:

1. Collect the beforehand professional category title as the key phrase for the thesis researching.
2. Assign the three sections, “the Chinese title of the thesis”, “key phrase”, “abstract section (Chinese)” as the data retrieving section.
3. Assign “or” as the primary principle rule to the relationship between all the retrieving conditions and Boolean logic.
4. Only collect sample documents from “the Chinese title of the thesis”, “key phrase” and “abstract section (Chinese)”

(3) Phrase Cutting Arrangement of Sample Document: After collecting the sample documents, phrase cutting was conducted to get key phrases and establish the key phrase database for the system. The cutting tool used was the Chinese auto-cutting system from CKIP. After the initial cutting, the sentence was transformed into a phrase range to be further used in the next stage.

(4) Sifting Key Phrase: Each phrase contains the appellation and syntax after the cutting and then to the first stage key phrase sifting.

1. ex. 0、1、2…9)；Remove the half morphic and whole morphic digital units (ex. 0, 1, 2, ……9).
2. Remove the half morphic and whole morphic punctuation marks (ex., 、；。：！？「」『』….. etc.).
3. Remove the unimportant syntax terms such as adjectives, conjunctions, adverbs, interjections, expletives, prepositions and verbs.
4. The key phrase was mainly regarded as nouns but such non key phrase nouns such as Neu, Nes, Nep, Neqa, Nf, Ng, Nh were also removed.

(5) Calculating the Key Phrase Weighted Value: In order to specify the importance of the each key phrase to the professional category, it was calculated the number of times each phrase appeared in each category and weighted its value.

In general, there are two elements need to take into consideration: one is the relative frequency of the phrase to the document. However, the number of phrases in the documents vary from one to another, so it needs to be *normalized* and the result is the TF (*Term Frequency*) [9] ranging from 0~1. TF is used for testing the relative importance of a phrase to a document. The higher the value is, the more important the document is likely to be. Another important element is DF (*Document Frequency*). It represents the appearance frequency of the single phrase to the entire document set. The less the phrase appears in a different document, the more proper it is likely to be used as a specifying standard. So, calculating the importance of a single phrase is to multiply TF and IDF. This method is called TFIDF. The key phrase weighted value will require this method to get the values. The formula is as below:

$$W_{i,j} = TF_{i,j} \times IDF_{j} = \frac{tf_{i,j}}{MAX\left(tf_{i,j}\right)} \times \log\left(\frac{N}{n_j}\right) \tag{1}$$

$W_{i,j}$: Means the importance of one key phrase ***j*** in ***i*** professional category document—the weighted value.

$TF_{i,j}$: The *term frequency* means the relative frequency of the key phrase j being normalized, its value is between 0~1.

$tf_{i,j}$: Means the absolute frequency of the key phrase ***j*** appears in ***i*** professional category document.

$MAX(tf_{i,j})$: The most frequently shown key phrase in ***i*** professional category document.

IDF_j : Means the IDF of the key phrase ***j***. The inverse frequency of ***j*** in professional category document set.

N : Means the total number of the professional category document set.

n_j : Means the appearance frequency of the key phrase ***j*** in the total professional category document set ***N***.

3.2 Question Sorting Model

Through the predefined combination of the professional key phrase, it was used to depict different requests and professional categories. Each request and professional category may be related to one or more key phrases and these key phrases occupy different weighted values. This model used the *Keyword Fuzzy Sets* [10] to calculate the similarities within the professional categories in order to find the most adequate groups.

(1) Question Phrase Cutting Process

First we process the customer's questions were processed through *Phrase Cutting* [11] (using the phrase database cutting as the basis) and then using the longest matching method to cut the key phrase needed. The complete steps are below:

Step 1: Separating the questions into several sentences Q_1、Q_2、$Q_3 \ldots Q_m$ according to the paragraphs and syntax. The end of the sentence of based on punctuation marks, , ；。！？.....etc.

Step 2: Assign the first sentence m=1, $Q_1=(C_1C_2C_3C_4 \ldots C_n)$

Step 3: i=1, Inquire the first word C_1 of the sentence Q_m before cutting phrases.

Step 4: Check the database to see if there is phrases which begin with C_1 if not then the value i plus 1 then go on dealing with the next word C_2 then repeat Step 3. If so, then proceed the next step.

Step 5: Include all the phrases begin with C_1 and input the Term range.

Step 6: Terms={$<C_1>$、$<C_1C_2>$、$<C_1C_2C_3>$、$<C_1C_4>$、$<C_1C_5>$}

Step 7: Descending the *Term* array based on the phrase's length.

Step 8: Terms = {$<C_1C_2C_3>$、$<C_1C_2>$、$<C_1C_4>$、$<C_1C_5>$、$<C_1>$}

Step 9: Compare phrases from the *Terms* and the phrase string begin with C_1 to see if it's perfectly compatible.

Step 10: If not, then the value i plus 1 and dealing the next word C_2 and then repeat step 3. If not, proceed to the next step.

Step 11: Key words $<C_1C_2C_3>$ totally match with the word string begin with C_1 and get the result of $C_1C_2C_3$.

Step 12: The value ***i*** plus the length of the key phrase and deal with C_4, repeat step 3 until all the phrases are being processed.

Step 13: If this is the last sentence, then end the cutting phrase. If not, ***m*** value plus 1 and repeat step 3 and deal with next sentence.

(2) Depiction of Questions in Fuzzy Set

With regard to the customer question Q, the question key phrase set and the fuzzy set theory was used to depict the question Q in Keyword Fuzzy Sets [10];

$Q = \{(K_1, W_1), (K_2, W_2), (K_3, W_3) \ldots (K_n, W_n)\}$ or

$Q = \{(K_j, W_j) \mid K_j \in \mathrm{K}\}$, $j = 1,\ldots,n$

K_j : The ***j*** key phrase in customer question $\boldsymbol{Q}$

W_j : The weighted value of the ***j*** key phrase in customer question $\boldsymbol{Q}$

n : The key phrase unit of the customer question $\boldsymbol{Q}$ as to the characters description of each professional category, was used the same way to generate the key phrase fuzzy set $\boldsymbol{C_i}$:

$C_i = \{(K_1, W_{i1}), (K_2, W_{i2}), (K_3, W_{i3}) \ldots (K_n, W_{in})\}$ *Or.*

$C_i = \{(K_j, W_{ij}) \mid K_j \in \mathrm{K}\}$, $j = 1,\ldots,n$

K_j : The ***j*** key phrase in professional category ***i***.

W_{ij} : The weighted value of ***j*** key phrase in professional category ***i***. The weighted value has already been calculated and pre-stored by phrase collecting model in the key phrase database.

n : The key phrase unit $\boldsymbol{C_i}$ of in the professional category ***i***.

(3) Professional Category Gauging

The last step of this model was to gauge the most relative professional groups in respond to customer's question. This was convenient for the system to search and recommend from the specialist catalogue. In addition, it also provided the knowledge broker the guide and direction in aiming the service item. First, fuzzy similarity was use to calculate the key phrase set $\boldsymbol{Q}$ and the professional key phrase fuzzy set $\boldsymbol{C_i}$ and the result of the similarity between the two will be $\boldsymbol{R_i}$. the calculation formula is as below:

$$R_i = \frac{Q \cdot C_i}{MAX(Q \cdot Q, C_i \cdot C_i)} \tag{2}$$

Q : the key phrase fuzzy set of the customer's question.
$Q = \{(K_1 , W_1), (K_2 , W_2) , (K_3 , W_3) \ldots (K_n , W_n)\}$。
C_i : the key phrase fuzzy set of the professional category $\boldsymbol{i}$.
$C_i = \{(K_1 , W_{i1}), (K_2 , W_{i2}) , (K_3 , W_{i3}) \ldots (K_n , W_{in})\}$。

The gauging of the professional category can be based on the value of $\boldsymbol{R_i}$, the higher the value the higher the similarity and can be input in the most relevant result in the set D. On the contrary, the low value means the similarity is also low and can be ignored. In order to include many possibilities to the question, the descending permutation of the biggest differentiated value of the relative value $\boldsymbol{R_i}$ will be set as the standard. Any values that are bigger than the standards will be the results needed. The complete gauging steps are as follow:

Step 1. Calculate the similarity degree of the question $\boldsymbol{Q}$ and the professional category $\boldsymbol{R_i}$
Step 2. Descending the value $\boldsymbol{R_i}$ as the new permutation $\boldsymbol{R_k}$, and record the correspondent relationship between $\boldsymbol{i}$ and $\boldsymbol{k}$ in array $\boldsymbol{F}$.
Step 3. Search for the maximum value $\boldsymbol{(R_k\text{-}R_k+1)}$ in as oppose to item $\boldsymbol{k = y}$.
Step 4. Gauge whether or not $\boldsymbol{y}$ is bigger than $\boldsymbol{\alpha}$ value, if so the skip to step 5 if not, the skip to step 6. the $\boldsymbol{\alpha}$ is the system parameter, it can represent the maximum range of the service and provide the knowledge broker for adjustment. The restriction range is $1 \leqq \boldsymbol{\alpha} \leqq$ n, n is the total number of the professional category.
Step 5. Using range $\boldsymbol{F}$ to seek the correspondent item $\boldsymbol{i}$ in item $\boldsymbol{k}$ that matches with 1~$\boldsymbol{y}$ and record the titles in the category result set D.
Step 6. Ask the customer to adjust the content of the question and re-execute the problem sorting model.

3.3 Specialists Matching Model

The system search within the database for the items of specialties belonged to set D according to the results of the question sorting model set D. The prior rule was to make a list of specialists' names in descending permutation and this matching list can be recommended to the knowledge broker as his/her references before the counseling.

3.4 Accuracy Adjustment Model

In order to effectively promote the gauge of the system to customer's question, the system will be re-examining with the specialist group after each gauge and matching. The results will be regarded as the accordance for the further adjustment on sorting arrangement. After several adjustments, the sorting system is very likely to fit all the client's demands.

The accuracy adjustment model is only correspondent to the recommended group specialists, and the items of the examination are only aiming to see whether or not the result set of the professional categorizations will match with the specialists'. When the specialist respond to the model, we can choose the value ranging from 1~5 to show the degree of the matching between the professional category and the customer's question. The highest accuracy value is 5, the lowest 1. In addition, the system also set a TLV (Threshold Limit Value) for the accuracy. When the value respond of the specialist is higher than the TLV, the system will tune up the weighted value of the key phrase in the specific category; if lower, then tune down the weighted value. The adjustment formula 3 is as below:

$$W_{i,j} = W_{i,j} + (\gamma - \theta) * \frac{W_{i,j}}{\sum_{j=1}^{k} W_{i,j}} \tag{3}$$

If Wij >1 then Wij=1 ; If Wij <0 then Wij=0

$W_{i,j}$: The weighted value of the no.***j*** key phrase in the professional category ***i***.
γ : The accuracy value judged by the specialist.
θ : The accuracy TLV of the system.

$\sum_{j=1}^{k} W_{i,j}$: The total weighted value of the entire key phrase $\boldsymbol{C_i}$ under category ***i***.

k : The existing unit of the key phrase C_i under category ***i***.

After the adjustment, the key phrase weighted value matches with the limitation of the professional category fuzzy set, degree of subordination 0~1. After calculation, if the weighted value is over 1, it's regarded as the weighted value 1. If it's smaller than 0, then it's regarded as value 0.

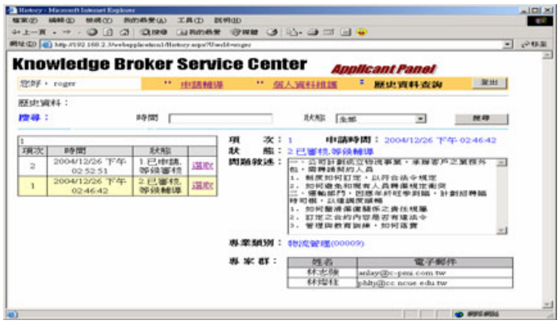
4 Experiment Procedure and Results Analysis

The testing objects of this experiment were a central management research development center and the group specialists: customers 48 people, specialists 16 people. The beginning date was Dec. 1st, 2004 and the termination date was Dec. 31st, 2005, within a year. The list of professional category includes 19 management fields. The system calculated the similarities between customer's question and the professional category and come up with the most possible result. The result showed that there are a total numbers of 41 application data, and all of the professional categories determine that they are all within one or two categories. It showed that the system was gathering focus instead of dispersing.

In the part of the questionnaire, the system automatically generated 214 questionnaires according to the recommended specialist list. The questionnaires were given to 16 specialists on the accuracy of the professional categorization. The valid ones are 214. The tested specialists can judge the accuracy by the syntactic variable or the scores. When the system is doing the evaluation, the corresponding score sheet (as Table 1 shows) will transform the variable into accuracy scores. The higher the score is, the more degree of affirmation there is. According to the result, 49.07% showed high degree of satisfaction. As we can see the specialists' view toward the gauge system is very positive, with a total accuracy score of 4.35.The following figures (as Fig 2) show the matching results:

Table 1. Corresponding Scores of Syntactic Variable and Accuracy Sheet

syntactic variable	**very inaccurate**	**inaccurate**	**normal**	**accurate**	**very accurate**
correspondent score	**1**	**2**	**3**	**4**	**5**
percentage	**4.21**	**16.36**	**10.75**	**19.63**	**49.07**



Knowledge Broker Service Center

Applicant Panel

您好，roger · 申請輔導 · 個人資料維護 · 歷史資料查詢 · 登出

歷史資料：

搜尋： 時間 [ ] 狀態 [全部] 搜尋

項次	時間	狀態	
2	2004/12/26 下午 02:52:51	1.已申請,等候審核	選取
1	2004/12/26 下午 02:46:42	2.已審核,等候輔導	選取

項 次：1 申請時間：2004/12/26 下午 02:46:42

狀 態：2.已審核,等候輔導

問題敘述：[illegible]

專業類別：物流管理(00009)

專家群：

姓名	電子郵件
[illegible]	anlay@c-pmi.com.tw
[illegible]	philtj@cc.ncue.edu.tw

Fig. 2. Matching Results

5 Conclusions

By ways of the discussion and summation of the knowledge mediating theory and methods, this research constructed a broker service center system based on knowledge. This system is capable of dealing with application question in the natural language form and auto-gauging its professional category and recommend lists of group specialists. The key phrase database samples are the Doctor and Master Thesis samples taken from The National Library. By using the key phrase fuzzy set calculation on the similarity between the question and professional category, it could compare and match with the specialties and come up with a list of group specialists that can be recommended to the broker center as the references for the case counseling.

However, there are still some parts that need to be further improved.

(1) The professional key phrase database mostly relies on the manpower to maintain its function and it has not yet occupied with the self studying function. In the future, it could be combining with analogic-nerve internet in order to further elevate its goal of self-learning function to semi-construction or automatic-construction.

(2) If the personal writings, other relevant counseling document, specialty fuzzy set of the specialists could be added in as part of the established learning samples, it

could cross-comparing the three fuzzy sets: customer questions, professional categories, professional specialties. It will help to improve the matching standard of the system.

(3) As to deal with a natural language, the proper inference ability and Chinese syntactic structure analysis were not mentioned in the present system. Under many circumstances, the combination of the phrases may extend to other meanings and the system can not cope with it. Right now these conditions are still lacking of solutions and need to be continuously researched on.

References

1. Wiig, K. M.: Knwoledge Management Methods : Practical Approaches to Managing Knowledge. Arlington: Schema Press (1994) 3-5, 20-26.
2. Myers, P. S.: Introduction to Series--why Knowledge, Why Now. In Paul. S. Myers. (Eds). Knowledge Management and Organizational Design. Boston: Butterworth-Heinmann (1997).
3. Zadeh, L.A.. : Fuzzy Sets. Information and Control (1965) Vol.8, 338-353.
4. Mahoney, T. A.: Journal Publishing and Organization Science: An Analysis of Intelligence Application Nstitrte. Organization Science (1996) 7(4), 443-455.
5. Davenport, T. H. and L. Prusak : Working Knowledge. Boston: Harvard Business School Press., Project Management Institute (1998) .
6. Sharon, J., L. Sasson, A. Parker, J. Horvath, and E. Mosbrooker: Identifying the Key People in Your KM Effort: The Role of Human Knowledge Intermediaries. Knowledge Management Review (2000) Vol. 3, No. 5, Nov. -Dec. 2000, 26-29.
7. Academia Sinica: “Academia Sinica Chinese Electronic Dictionic Dictionary Group-Academia Sinica Language Base Contents and Descriptions”□Chinese Phrase Knowledge Base, Technical Report of Academia Sinica #95-02(1995) .
8. Richardo, B. Y. and R. N. Berthier : Modern Information Retrieval. Addision Wesley Longman Limited (1999).
9. Salton, G. : Automatic Text Processing. Addison-Wesley (1989).
10. Hoch, R.: Using IR Techniques for Text Classification in Document Analysis. Proceedings of The Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval (1994) 31-40.
11. Li, B. Y., Lien, C. F., and Sun, M. S. : A Maximal Matching Automatic Chinese Word Segmentation Algorithm Using Corpus Tagging for Ambiguity Resolution. R.O.C. Computational Linguistics Conference, Taiwan (1991) 135-146.
12. Davenport, T. O.: Human Capital: What Is It and Why People Invest It, SanFrancisco: Jossey-Bass (1999).
13. Nonaka, I. and H. Takeuchi : The Knowledge Creating Company: How Japanese Companies Create the Dynamics of Innovation. New York: Oxford University Press. Harvard Business Review (1995) 69, Nov.-Dec. 96-104.

A Numerical Simulation Study of Structural Damage Based on RBF Neural Network

Xu-dong Yuan[1], Hou-bin Fan[2], Cao Gao[1], and Shao-xia Gao[1]

[1] Civil Engineering Institute, Dalian Fisheries University, Postfach 116021, Dalian, China
yxd0101@163.com
[2] ZheJiang Provincial Transportation Engineering Construction Group, Postfach 310030, Hangzhou, China

Abstract. It's natural and direct to identify the structural stiffness based on the measurement of static displacement; In addition, considering that the lower frequencies of structures can be tested with high precision and can reflect the global dynamic properties of structures, static displacements at partial nodes and several low frequencies were used to constitute the input parameter vectors for neural networks. A damage numerical verification on an arch bridge model was carried out using a radical basis function (RBF) network. Identification results indicate that the neural network has an excellence capability to identify the location and extent of structural damage with the limited noises and incomplete measured data.

1 Introduction

The health diagnosis and damage detection of civil engineering structures is a very active topic at present. The research in this area has extensive and profound engineering background. Although there are many effective methods, which developed for different problems, but how to use incomplete and inexact measured data to acquire acceptable and ideal damage identification results is hot and difficult problem [1].

The main load for existing engineering structures is static load. The static test has become a general method used to structural monitoring and damage identification. In addition, the static equilibrium equation of engineering structures is only connected with structural stiffness and unrelated with damp and inertia, the main aim of structural damage detection is the identification of structural stiffness, therefore the more natural and direct method for structural damage identification is based on static displacements [2-7]. In this paper, static displacements at partial nodes and several low frequencies were used to constitute the input parameter vectors for neural networks. Thus, overcoming the disadvantages of damage identification, which brought by using single date, and validate each other. And then numerical verification for the location and extent of structural damage were carried out using the radical basis function (RBF) networks with the limited measured data, the aim of the study was laid on the influences of different noise levels and the selected number of frequencies on the results of structural damage identification.

2 Radical Basis Function Neural Networks

When BP neural networks were used in function approximation, negative gradient descent method were adopted to adjust weights, this kind of weight method had its localization, such as comparative slow rate of convergence and local minimum value, whereas, no matter the capabilities of function approximation, classify and study speed of RBF are exceeded that of BP [8].

RBF neural network was composed with three layers shown in fig.1, the nodes of input layer just only transfer input signals to hidden layer, the nodes of hidden layer was composed with radiating action function, whereas, the nodes of output layers are simple liner function.

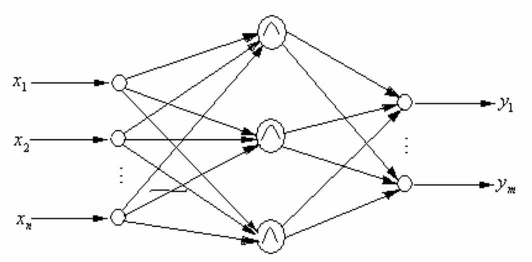


Fig. 1. Structure of RBF neural network

The kernel function of hidden layer nodes will bring some local effects to input signals, that is to say, when input signals close up the center of kernel function, relative great output will be produced in hidden layer nodes. So we can see that this kind of neural network has the local approach capability. In this paper, Gaussian kernel function is used as action function of hidden layer nodes, which shown as follow equation

$$u_j = \exp\left[-\frac{(X - C_j)^T (X - C_j)}{2\sigma_j^2}\right] \tag{1}$$

Where $(j = 1, 2, \cdots, N_h)$, u_j is the output of the jth hidden layer node, $X = (x_1, x_2, \cdots, x_n)^T$ is the input sample, C_j is the center value of Gaussian kernel function, σ_j is the standard constant, N_h is number of hidden layer nodes. We can know that the output of nodes is from 0 to 1, the input sample is more nearing the center of the nodes, and the output value is bigger.

The output of RBF neural network is the linear combination of hidden layer nodes.

$$y_i = \sum_{j=1}^{N_h} w_{ij} u_j - \theta = W_i^T U \tag{2}$$

$$W_i = \left(w_{i1}, w_{i2}, \cdots, w_{iN_h}, -\theta\right)^T \tag{3}$$

$$U = \left(u_1, u_2, \cdots, u_{N_h}, 1\right)^T \tag{4}$$

The process of network learning is divided two stages. First stage, C_j and σ_j are decided by making use of all input samples. Second stage, after the parameter of hidden layer determined, the weight value of output layer W_i is decided by using of the samples and least square algorithm. When second stage of learning finished, in order to improve the accuracy of the networks, the parameters of input and output layers are modified by the samples.

3 Damage Theory Analysis

3.1 Damage Location Detection Indicator

For healthy structure, the static response equation and the eigenvalue equation can be expressed as

$$Ku = p \tag{5}$$

$$(K + \omega_j^2 M)\varphi_j = 0 \tag{6}$$

Where K and M represent the global stiffness matrix and the global mass matrix for health structure; u and p represent the displacement vector and force vector; ω_i and φ_i are the *i*th natural frequency and the corresponding mode shape. Let the change of stiffness matrix caused by structural damage be defined as ΔK, thus, Eq.(5) and (6) can be rewritten as

$$(K + \Delta K)u^* = p \tag{7}$$

$$(K + \Delta K + (\omega_j^2 + \Delta\omega_j^2)M)(\varphi_j + \Delta\varphi_j) = 0 \tag{8}$$

u^* is damaged displacement vector and defined as

$$u^* = (K + \Delta K)^{-1} p \approx (K^{-1} - K^{-1}\Delta K K^{-1})p \tag{9}$$

The change of displacement vector caused by damage is defined as

$$\Delta u = u - u^* \approx K^{-1}\Delta K K^{-1} p \tag{10}$$

The change of natural frequency is expressed as

$$\Delta\omega_j^2 \approx \frac{\varphi_j^T \Delta K \varphi_j}{\varphi_j^T M \varphi_j} \tag{11}$$

When FEM model is used, the change of global stiffness matrix can be expressed by the change of element stiffness matrix

$$\Delta K = \sum_{i=1}^{ne} \alpha_i k_i \tag{12}$$

α_i is the damage parameter of element stiffness, $-1 \le \alpha_i \le 0$, k_i is element stiffness（$i = 1,2,\cdots,ne$）, ne is the number of structural elements.

DS of damage location detection presented in literature [9] was used as input vector for the neural network when individual element damage or several element damages with the same extent occurred. It can be defined as

$$DS = \frac{\Delta u}{\Delta\omega_j^2} = \frac{K^{-1}\Delta K K^{-1} p}{\dfrac{\varphi_j^T \Delta K \varphi_j}{\varphi_j^T M \varphi_j}} = \frac{K^{-1}\sum_{i=1}^{ne} k_i K^{-1} p}{\dfrac{\varphi_j^T \sum_{i=1}^{ne} k_i \varphi_j}{\varphi_j^T M \varphi_j}} \tag{13}$$

From Eq. (13), we can see that α_j is eliminated. Thus, DS is solely related to damage location and independent of damage extent

$$input - vector = \{DS_1, DS_2, \cdots, DS_m\} \tag{14}$$

m is the serial number of corresponding nodes

3.2 Damage Extent Detection Indicator

For structural damage extent detection, the terms of damage extent must be added to the input vector for damage location detection, namely

$$RNF = \frac{\Delta\omega_j^2}{\omega_j^2} \approx \frac{\varphi_j^T \Delta K \varphi_j}{\varphi_j^T K \varphi_j} = \frac{\varphi_j^T \sum_{i=1}^{ne} \alpha_i k_i \varphi_j}{\varphi_j^T K \varphi_j} \tag{15}$$

Eq. (13) is related to damage extent, thus, *DS* and *RNF* together constitute the input vector for the neural network,

$$input - vector = \{DS_1, DS_2, \cdots, DS_m, RNF\} \tag{16}$$

4 Numerical Verification of Arch Bridge Model

Numerical model is a plane arch bridge truss combined with 20 beam elements shown in Fig.2, elastic modulus $E = 210GPa$, mass density $\rho = 2600kg/m^3$, cross sectional area $A = 3.2m^2$, loads $P_1 = P_2 = P_3 = P_4 = P_5 = P_6 = 5000kg$.

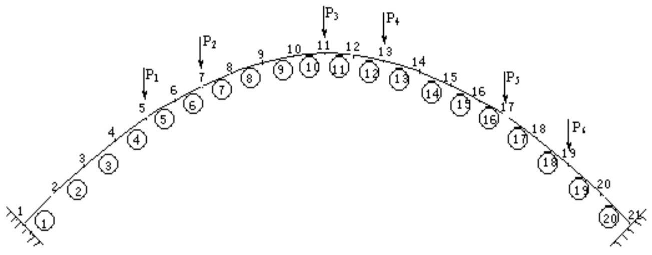


Fig. 2. Calculation model of arch bridge structure

4.1 Localization with Measured Noisy Data

When generating training samples of the network, the cases with one damaged element and two damaged elements of the arch bridge with three combined loads listed in Tab.1 are considered. In order to test the capability of identification for damage location with incomplete measured DOFs, the displacements of increment of X, Y directions at ten nodes with 20 DOFs were selected and composed the parameters of training samples of the network in term of Eq. (13). In order to enhance the robustness of the network and enlarge the number of training samples, 1%~4% random noises were added to the training samples calculated without data errors, and the training samples were enlarged 4 times.

Table 1. The cases of loads & damage of training for localization

Combined Load	damage element	damage extent(%)
1 (P_1,P_2,P_3,P_4)	2	50
	4	50
	8	50
	7/15	50
2 (P_3,P_4,P_5,P_6)	13	50
	18	50
	9/12	50
3 (P_1,P_2,P_5,P_6)	3	50
	6	50
	16	50
	8/17	50

Table 2. The cases of loads & damage of testing the network for localization

Combined loads	damage element	damage extent(%)
1 (P_1,P_2,P_3,P_4)	3	10
		60
	12	10
		60
	6/13	10
		60
3 (P_1,P_2,P_5,P_6)	4	10
		60
	16	10
		60
	8/15	10
		60

When generating testing samples of the network, the calculation data of the cases with one damaged element and two damage elements with two different combined loads listed in Tab.2 were used. Identification of damage locations was carried out with the data errors.

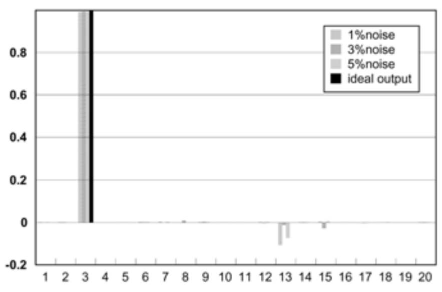


Fig. 3. Localization results with different noise levels in the 3th element

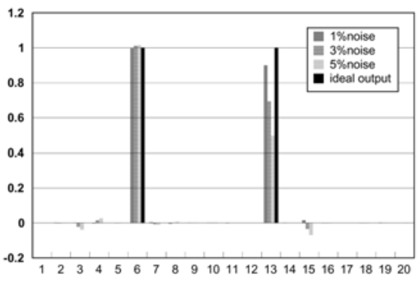


Fig. 4. Localization results with different noise levels in the 3th and 9th element

In the cases of one damaged element or two damaged elements, we can see that the identification errors have some increased with the increase of noise level from Fig.3~4. The network can accurately identify the damage locations with 1%~5% noises, the results indicate that the network has a favorable generalization capability. The identification capability of the network decreased and some error identifications occur with 10% noises.

4.2 Qualification with Change of the Number of Natural Frequencies

The parameters for consisting of training and testing samples are combined with two segments in term of Eq. (16), one is the parameters for identification of damage location, another is the ratio of the increment of frequency square to frequency square for damage extent identification. When training the network, the cases of one damaged element and two damaged elements with three combined loads listed in Tab. 3 were considered. In order to test damage extent identification capability of the network with incomplete measured DOFs, the increments of displacements of ten nodes with 20 DOFs were still used.

When generating training samples of the network, the cases of one damaged element and two damaged elements listed in Tab.4 were considered, and the damage extent identification were carried out with measured data errors.

During training and testing the network process. The samples were combined with 2, 3 and 5natural frequencies, the capability of damage extent identification of the network were tested with different natural frequencies.

The damage locations shown in Fig.5 and 6 were identified. We can see that the damage extent of damaged elements are more serious than that of undamaged elements and the damage extent identified was lighter than actual damage extent, this result had reflected that the training samples were not sufficient, and validated that the identification capability of the network is relied on the composition of the training samples.

The results of identification with the stiffness of two elements decrease 20% shown in Fig.7~9. With the increase of the number of natural frequencies, the identification accuracy of damage extent was improved, the cases of mistake identification in undamaged elements were not appeared and the results of identification were ideally. Hence, we can conclude that the network capability of damage extent identification depends on whether the training samples are sufficient.

Generally, the identification capability of damage extent is not as good as that of damage location. In spite of that, the damage location could be identified and damage

Table 3. The cases of loads & damage for training network

Combined Loads	damage element	damage extent(%)
1 (P_1,P_2,P_3,P_4)	2	20
	3	30
	4	40
	8	60
	12	40
	7/15	30/60
	6/13	30/20
2 (P_3,P_4,P_5,P_6)	13	20
	18	40
	9/12	30/60
3 (P_1,P_2,P_5,P_6)	3	20
	4	30
	6	40
	16	50
	8/17	40/20
	8/15	20

Table 4. The cases of load and damage of testing for quantification

Combined Loads	damage element	damage extent(%)
1 (P_1,P_2,P_3,P_4)	3	20
		40
	12	20
		40
	6/13	20
		40
3 (P_1,P_2,P_5,P_6)	4	20
		40
	16	20
		40
	8/15	20
		40
	15	40

extent could be judged basically. When detecting of structural damage, if we can combine damage location identification with damage extent identification and selected the number of natural frequencies properly. Thus, the satisfied results could be acquired.

4.3 Quantification with Measured Noisy Data

1%, 3% and 5% noises were added into the testing samples for verifying the capability of RBF networks to identify structural damage extent. The identification results of two elements stiffness decreased 20% were shown in Fig.10~12. We can see that all the results are not remarkable difference and are better ideally within 1%~5% noises. Meanwhile the results are shown that the identification capability of damage extent was not affected in definite error range.

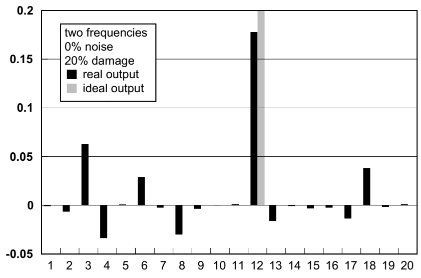


Fig. 5. Quantification results of 20% damage in the 12th element with 2 frequencies

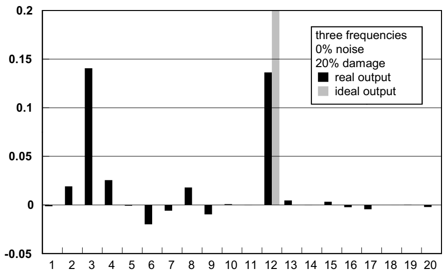


Fig. 6. Quantification results of 20% damage in the 12th element with 3 frequencies

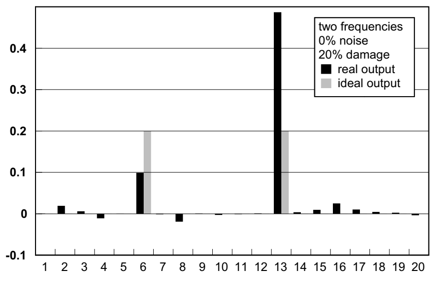


Fig. 7. Quantification results of 20% damage in the 6th and 13th element with 2 frequencies

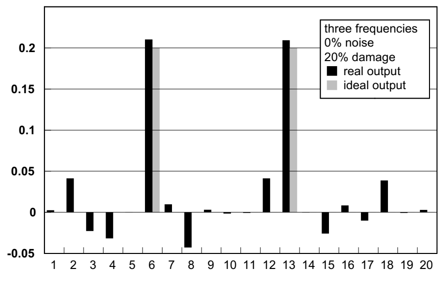


Fig. 8. Quantification results of 20% damage in the 6th and 13th element with 3 frequencies

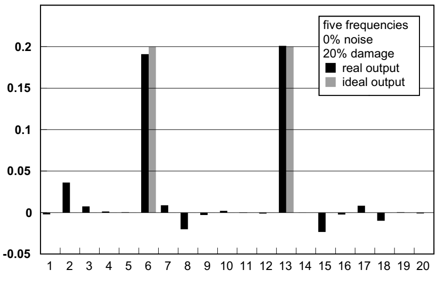


Fig. 9. Quantification results of 20% damage in the 6th and 13th element with 5 frequencies

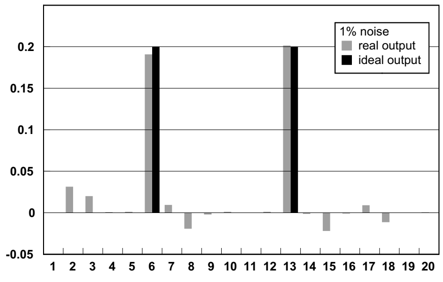


Fig. 10. Quantification results of 20% damage in the 6th and 13th element with 1% noise

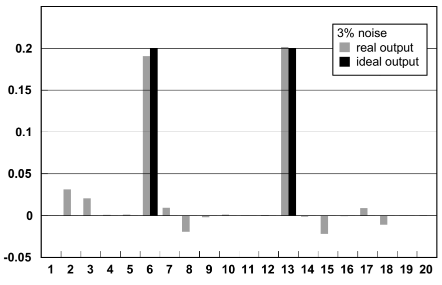


Fig. 11. Quantification results of 20% damage in the 6th and 13th element with 3% noise

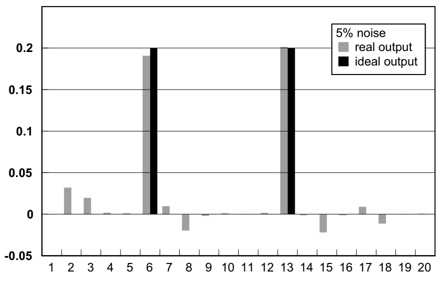


Fig. 12. Quantification results of 20% damage in the 6th and 13th element with 3% noise

5 Conclusion

In this paper, using of static displacements and several low natural frequencies forms the combined input parameter vectors of artificial neural network used to identify damage location and extent. The numerical example analyses were carried out on an arch bridge model by using RBF network. The results were summarized as follows:

(1) The accuracy of damage location identification is better than that of damage extent identification.
(2) The ideal results of identification depend on the number of the training samples whether is sufficient.
(3) The changes in the number of natural frequencies can improve the accuracy of damage extent identification, but it's limited
(4) The network has good robustness and the probability of RBF networks were embodied very well
(5) The identification of damage location and extent should be carried out simultaneously, thus, we can fix the damage location accurately and judge the damage extent broadly

References

1. Wang B S, Ding H J. Influence of modeling errors on structural damage identification using artificial neural networks [J]. China Civil Engineering Journal, 2000,.33(1): 50-55
2. Sheena Z, Unger A, and Zalmanovich A. theoretical stiffness matrix correction by using static test results. Isarel Journal of Technology, 1982(20): 245-253
3. Sanayei M, Scampoli SF. Structural element stiffness identification from static test data. Journal of Engineering Mechanics, 1991, 117(5): 1021-1036
4. Sanayei M, Onipede O. Assessment of structures using static test data. AIAA Journal, 1991,29(7): 1156-1179
5. Hejala P, Soeiro FJ. Recent development in damage detection based on system identification methods. Structural Optimization, 1990(2):1-10
6. Banan MR, and Hjelmstad KD. Parameter estimation of structures from static response, I: computational aspects. Journal of Structural Engineering, 1994,120(11): 3243-3258
7. Banan MR, and Hjelmstad KD. Parameter estimation of structures from static response II: numerical simulation studies. Journal of Structural Engineering, 1994,120(11): 3259-3283
8. Wen X, Zhou L. Neural networks application design based on MATLAB [M]. Beijing: Science Publishing Company, 2000. 207-32
9. X.Wang, N. Hu, Hisao Fukunaga, Z.H. Yao. Structure damage identification using static test data and changes in frequencies [J]. Engineering Structures, 2001, (23): 610-21

Word Frequency Effect and Word Similarity Effect in Korean Lexical Decision Task and Their Computational Model[⋆]

YouAn Kwon[1], KiNam Park[2], HeuiSeok Lim[3], KiChun Nam[1,⋆⋆], and Soonyoung Jung[2]

[1] Department of Psychology, Korea University, Korea
kichun@korea.ac.kr
http://coglab.korea.ac.kr
[2] Department of Computer Education, Korea University, Korea
spknn@korea.ac.kr
http://comedu.korea.ac.kr
[3] Division of Computer, Information, and Software, Hanshin University, Korea
limhs@hs.ac.kr
http://nlp.hs.ac.kr

Abstract. In this paper, we investigate whether the word frequency effect and the word similarity effect could be applied to Korean lexical decision task (henceforth, LDT). Also we propose a computational model of Korean LDT and present comparison results between human and computational model on Korean LDT. We found that the word frequency effect and the similarity effect in Korean LDT were language general phenomena in both the behavioral experiment and the proposed computational simulation.

1 Introduction

The lexical decision task gives us cues to understand the internal mechanism. There are several basic findings that are consistently obtained in behavioral LDT experiments(Taft, 1993). Among them, the major findings are frequency effect, lexical status effect, word similarity effect, semantic priming effect, and visual degradation effect. These effects should be explained by any models claiming to be an account of lexical access. Next descriptions are simple explanations of each effect on LDT.

Word frequency effect. It takes less time to make a lexical decision on high-frequency words than to low frequency words.

Lexical status effect. It takes more time to make a lexical decision on non-words than to real words.

⋆ This study has been supported by the Korea Research Foundation(2004-074-HM0004).

⋆⋆ Corresponding author.

Word similarity effect. A letter string is hard to make a lexical decision on, when sufficiently similar to real words.

Semantic priming effect. word is easy to make a decision on, when preceded by a semantically related word.

Visual degradation effect. When a letter string is visually degraded, it becomes more difficult to be classified as a word or non-word.

Recently many researchers have tried to investigate visual word recognition process using computational experiments. Computational experiments and simulations are very helpful to elaborate or to make the model complete by handling experimental environment. In addition, computational model may suggest fruitful areas for experimentalists to investigate. Also, partial lesions of neurological areas and pathways can be modeled in a straightforward way if connectionist modeling is used. The intensional lesion study using the connectionist computational model is never possible with human subjects.

Due to these advantages, researches on developing computational mental lexicon model (Hinton & Shallice, 1991; Plaut & Shallice, 1993) have been conducted. These researches commonly reported word frequency effect, word similarity effect, word status effect, semantic priming effect, and visual degradation effect which were shown in behavioral experiments. while most of previous studies have been focused on the model of English fewer studies have been done on Korean mental lexicon except for the symbolic computational model of Lim, Nam & Hwang (2005). Furthermore, there have never been behavioral experimental results reported on Korean LDT.

We have two aims in this paper. First, finding out whether major language phenomenon, especially word frequency effect and word similarity effect, can be observed on Korean LDT. Second, building computational model simulating Korean LDT and performing comparison studies between behavioral and computational results. The organization of this paper is as follows: the behavioral experiment and results, the description of computational model architecture training, the result of computational model, the comparison between behavioral results and computational results, and finally, we will conclude findings and suggest the future researches.

2 The Behavioral Experiment

2.1 The Word Frequency Effect and the Word Similarity Effect

Method

Participants. Thirty-five students of Korea University participated. They were all native speakers of Korean with corrected or normal vision.

Stimuli and Design. We collected two hundred words with frequency from 1 to 100 and invented two hundred pseudo-words which are pronounceable but not meaningful. Additionally we made another two hundred non-words which are not pronounceable nor meaningful. One hundred words (filler) were randomly selected from the corpus for the purpose of eliminating the behavioral bias. All words were selected from the Sejong Corpus. We randomly assigned materials to two lists in order to decrease artifacts of items. Each list contains 100 words, 100 pseudo-words, 100 non-words and 100 fillers. The experimenter presented randomly one of two lists to participants. We used an analysis of variance on word frequency (high vs. low) and word similarity (word vs. pseudo-word, word vs. non-word).

Apparatus and Procedure. Two PC586 were used to present stimuli and collect responses. Responses were received using the PST serial response box. Stimuli were presented on 17" CRT monitor.

When a participant sit in front of a computer, the experimenter presented the practice section. The practice section consisted of 5 words, 5 pseudo-words, 5 non-words, and 5 fillers. Any items used in practice section were not presented in the experiment section, All items were presented randomly. Each trial worked in the way that that a fixation ("***") appeared in the center of the screen for 500ms, and then a target was presented after the fixation. Each target was displayed long enough for participants to respond. Participants were asked to decide as quickly and accurately as possible whether the target was a legal word (word) or an illegal word(pseudo-word or non-word) and to press the appropriate response key. The time from stimulus onset to response was measured in ms.

3 The Behavioral Results

3.1 The Word Frequency Effect

The analysis was performed on only correct responses. We divided words into two groups(high frequency group vs. low frequency group). We defined a 'high frequency' group as a group of words with frequency with over fifty and a low frequency group as a group of words with frequency less than fifty. The mean response time of the high frequency group was 518.51ms and the low frequency group, 540.95ms. The difference between two groups was significant ($F(1, 31) = 7.39, p < .05$). Similar results were reported repeatedly in other studies (Forster, & Chambers, 1973; Taft, 1979). Actually, word frequency was considered to be one of basic variables in studies involving visual and auditory word process (Andrews, 1992).

3.2 The Word Similarity Effect

Three of thirty-five participants were excluded from analysis for being beyond 2.5 SD of the grand mean on error rates. We used analysis of variance (ANOVA) on word, pseudo-word, and non-word. Mean lexical decision times and error rates were presented on Table 1.

Conditions	Response time(ms)	Error rate(%)
Word	559.90	95.31
Pseudo-word	563.99	89.03
Non-word	519.64	98.34

In the error rate analysis, the difference between conditions was significant ($F(1, 62) = 10.93, p < .001$). In the response time analysis, the difference between 'pseudo-word' and 'non-word' was significant for items and participants analysis ($F_1(1, 62) = 7.70, p < .01; F_2(1, 398) = 25.79, p < .01$). Also, the difference between 'word' and 'non-word' was significant ($F_1(1, 62) = 6.34, p < .05; F_2(1, 398) = 17.80, p < .01$). According to these results, participants confused the lexicality of pseudo-words. However, participants immediately decided that non-words do not have a lexicality.

Because pseudo-words were pronounceable, it seems like that they cause interference with a decision-making on the lexicality of a target. The findings that pseudo-words take more time to make a lexical decision on than a non-words are consistent with other researches (Chambers, 1979; O'Connor, & Forster, 1981).

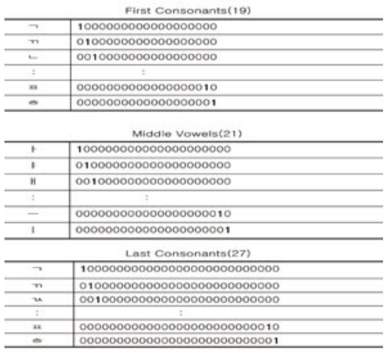
4 Input/Output Design of the Computational Model

Input to the proposed network is Korean 2 syllable string and the output of the model is a semantic representation. The network is regarded that it makes a lexical decision when it makes an output on an input.

Some representation methods of English orthography have been proposed such as in McClelland & Rumelhart (1981), Hinton & Shallice (1991), and Plaut & Shallice (1993). But, since Korean is very different from English, the same representation methods can not be used in representation of Korean Orthography. We propose a method for representing Korean orthography using connectionist model. A typical form of a Korean syllable is CVC(initial Consonant-medial Vowel-final Consonant) or CV(first Consonant-middle Vowel). The number of initial consonants and final consonants are 19 and 27 respectively. The number of vowels is 21. We may represent a Korean syllable as 15 bits: 5 bits are used to represent each consonant and vowel. But, with this scheme, two different syllables can be represented in the way of the very similar bit stream. This may be problematic in visual LDT experiment. Thus, we propose a orthogonal representation in which different consonants or vowels are represented as different bit streams.

The proposed orthography, a Korean syllable is represented as a 67-bit vector; 19 bits are used to represent initial consonants, 21 bits for medial vowels, and 21 bits for final consonants. Figure.1 shows an example of vector representation of Korean alphabets.

We represent the semantics of the input words in terms of 100 pseudo semantic features. The assignment of semantic features to words has the property that words from the same category tend to be more similar with each other than with words from different categories.



First Consonants(19)

ㄱ	1000000000000000000
ㄲ	0100000000000000000
ㄴ	0010000000000000000
⋮	⋮
ㅍ	0000000000000000010
ㅎ	0000000000000000001

Middle Vowels(21)

ㅏ	100000000000000000000
ㅑ	010000000000000000000
ㅐ	001000000000000000000
⋮	⋮
ㅡ	000000000000000000010
ㅣ	000000000000000000001

Last Consonants(27)

ㄱ	100000000000000000000000000
ㄲ	010000000000000000000000000
ㄳ	001000000000000000000000000
⋮	⋮
ㅍ	000000000000000000000000010
ㅎ	000000000000000000000000001

Fig. 1. The Representation of orthography

5 The Network Architecture

Figure. 2 depicts the network architecture of the proposed model. The network is a recurrent network which contains connections from the hidden units to a set of context units. The network consists of three layers: input layer, hidden layer, and output layer. The input layer includes 134 input units(grapheme units), and 250 context units. The 134 grapheme units and contexts units are fully connected to a group of 250 hidden units. The hidden units are also fully connected to 100 output units and context units.

Each unit except context unit has a real-valued activity level ranging between 0 and 1, computed by a smooth nonlinear function of the summed input received from other units. The activities of the context units are 0s or direct copies of the hidden unit activities on the previous time step whether the experimental task priming task or not. If it is not a priming task, all of context units are set to 0s. If it is a priming task such as a semantic priming task, the units are set by copying from hidden units.

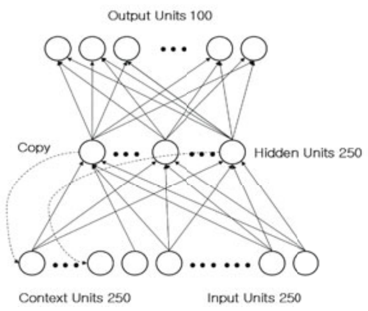


Fig. 2. The network architecture

6 Training and Evaluation of the Model

The network was trained by backpropagation algorithm which was modified slightly to copy value of hidden layer into the context layer. The error of output unit was calculated by sum of cross entropy of each output unit. Response time on lexical decision was measured in terms of semantic stress measure which was used in Plaut & Shallice (1993). More formally, the stress S_j of $unit_j$ is a measure of the information content (entropy) of its activation a_j, corresponding to the degree to which it differs from the "neutral" output of 0.5. The higher value of semantic stresses means that the network makes more correct semantic output pattern. If the network makes higher average value of semantic stresses of all output nodes in a fixed time, the network is taken to response more quickly on the input data.

7 The Computational Result

The current model was trained for 10,000 epochs according to the compressed frequency. After training, the total number of error rate was 0.34 and the error rate for one input data was 0.001 in average. However, as shown in the over training might occur so that the best trained point was determined to have the highest correlation between the semantic stress and the compressed frequency values. The best trained point was at the 500th trained state, and at this point the Pearson correlation value correlation between the semantic stress and the compressed frequency values was 0.738 ($p < 0.01$). The total error value of the 500th trained point was 536.85 and for one input data it was 0.630 in average.

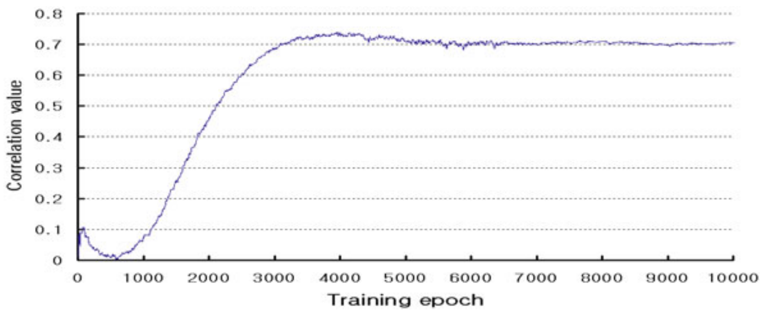


Fig. 3. A correlation value change according to training epoch

7.1 Frequency Effect

To simulate frequency effect, the correlation between the frequency and semantic stress has to be statistically significant. As the semantic stress value stands for reaction time and the frequency is a frequency value, the positive correlation means that when the frequency gets higher, the corresponding semantic stress value gets higher, and vice versa. The results show that the correlation coefficient between the frequency and semantic stress values is 0.738 ($p < 0.05$) (Fig. 3).

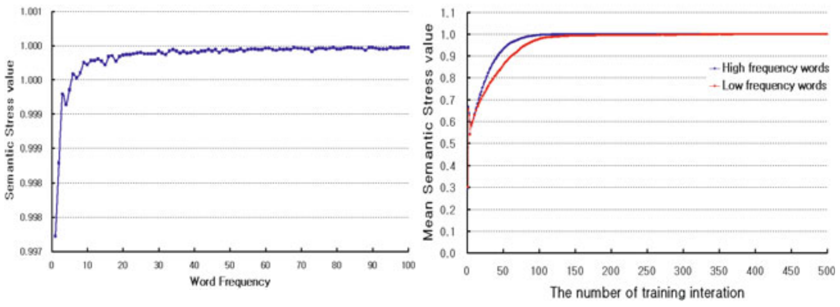


Fig. 4. Frequency effect

Also the results showed that there was a significant difference between two conditions($t(998.819) = 2.178, p < 0.05$), (0.973 for the high frequency word condition and 0.961 for the low frequency condition in average)(Fig. 4) So, the results indicate that this model simulates frequency effect.

7.2 Word Similarity Effect

The results showed that while non-words, which are unacceptable in the Korean orthography, had the semantic stress value of 0.924 in average, pseudo-words, which are acceptable, but do not exist in Korean language, had the semantic stress value of 0.913 in average regular word had the semantic stress vale of 0.978 in average. This difference was statistically significant($F(2, 167) = 2.0, p < 0.05$).(Fig. 5) Thus, we can conclude that that the proposed model is successfully simulating word similarity effect.

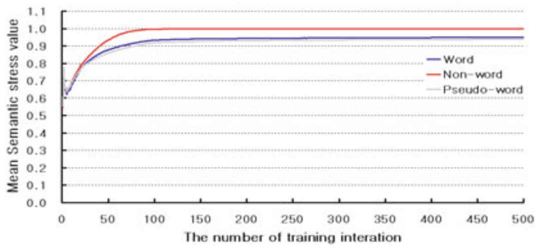


Fig. 5. Word similarity effect

8 Comparison with Human Performance

The right figure (Fig. 6) means that the higher the word's frequency is, the closer semantic value approaches 1 than less frequent words. In other words, words with high frequency can access the semantic system faster than words with low frequency. The left figure in (Fig. 6) represents human responses. The y-axis represents response time(ms) for participants to decide the lexicality. People also more quickly identified words with high frequency than words with low frequency.

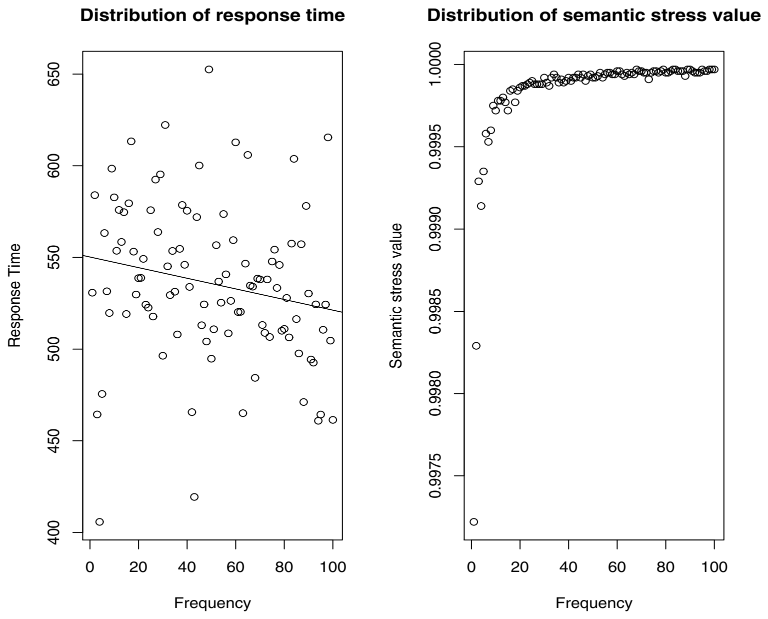


Fig. 6. Word Frequency Effect

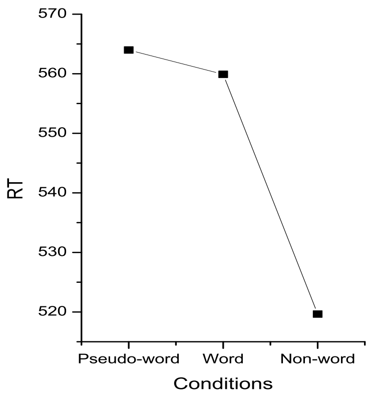


Fig. 7. Word similarity effect in behavioral experiment

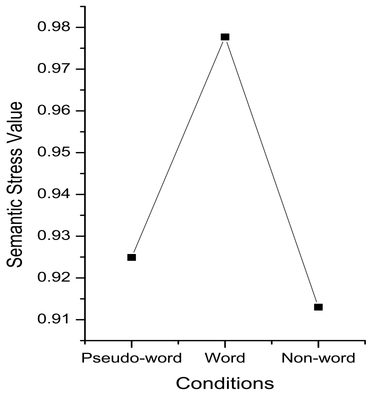


Fig. 8. Word similarity effect in computational experiment

(Fig. 7) represents the result of word similarity effect in behavioral experiment. According to (Fig. 7), participants had difficulties identifying pseudo-words rather than words or non-words. On the contrary as seen in, (Fig. 8), computational experiment had somewhat different results from behavioral experiment. In contrast to behavioral result, pseudo-word's semantic stress value is smaller than non-word's semantic value. The reason for this disagreement between behavioral and computational results can be explained by the lexical structure. It's probably because while human lexicon may be composed of phonological information for words (Grainger, Muneaux, Farioli, & Ziegeler, 2005), our computational lexicon is constructed based only on orthographic information.

9 Conclusion

In this paper, we propose a computational model of Korean LDT and make a comparison between behavioral and computational results on frequency effect and word similarity effect. High-frequency words were responded faster than the low-frequency words in both behavioral and computational experiment. Also, It took more time to respond to pseudo-words than to non-words. The present results have several implications for general models of the visual word recognition regardless of language. First, they provide additional evidence that the word frequency may be a basic information to access the mental lexicon. Second, the results support the interpretation that the phonological information may play a important role in lexical selection (Grainger, Muneaux, Farioli, & Ziegeler, 2005). However, this model is incomplete in that it still has room to be improved to simulate many other phenomena of language such as word status effect, semantic priming, word neighborhood density and frequency effect. In the future, we will try to find other effects related to visual word recognition process.

References

1. Andrews, S. (1992). Frequency and Neighborhood Effects on Lexical Access: Lexical Similarity or Orthographic Redundancy? *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 18, 234-254.
2. Becker, C.A. (1979). Semantic context and word frequency effects in visual word recognition. *Journal of Experimental Psychology: Human Preception and Performance*, 5, 252–259.
3. Chambers, S. M. (1979). Letter and order information in lexical access. *Journal of Verbal Learning and Verbal Behavior*, 18, 225-241.
4. Forster, K.I. (1976). Accessing the mental lexicon, In E.C.J. Walker & R.J. Wales (Eds.), *New approaches to language mechni, Amsterdam*: North-Holland.
5. Grainger, J., Muneaux, M., Farioli, F., & Ziegler, J. C.(2005). Effects of phonological and orthographic neighborhood density interaction in visual word recognition. *The Quarterly Journal of Experimental Psychology, 58A(6)*, 779-793.
6. Hinton, G.E., & Shallice, T. (1991). Lesioning an attractor network: Investigatins of acquired dyslexia. *Psychological Review*, 98(1), 74–95.
7. Lim, H.S., & Nam, K., & Hwang, Y. (2005). A Computational Model of Korean Mental Lexicon. *Lecture Notes in Computer Science, 3480*, 1129–1134.
8. McClelland, J.L. & Rumelhart, D.E. (1981). An interactive activation model of context effects in letter perception: Part 1. An account of basic findings. *Psychological Review*, 88, 375–407.
9. Morton, J. (1969). Interaction of information in word recognition. *Psychological Review*, 76, 165–178.
10. O'Connor, R. E., & Forster, K. I. (1981). Criterion bias and search sequence bias in word recognition. *Memory and Cognition*, 9, 78-92.
11. Plaut, D.C., & Shallice, T. (1993). Deep dyslexia: A case study of connectionist neuropsychology. *Cognitive Neuropsychology*, 10(5), 377–500.
12. Taft, M., (1993). *Reading and The Mental Lexicon.* Hove [etc.] : Lawrence Erlbaum Associates.

Content-Based 3D Graphic Information Retrieval

Soochan Hwang and Yonghwan Kim

Department of Computer Engineering, Hankuk Aviation University, 412-791, Korea
{schwang, yhkim93}@hau.ac.kr

Abstract. Supporting the content-based retrieval of 3D graphic data has been little addressed in the most current 3D graphic systems. The systems focus on visualizing 3D images. This paper presents a 3D graphic data model which models 3D scenes using domain objects and their spatial relations. The model also supports the content-based query. The user can pose a visual query involving various 3D graphic features such as an inclusion of a given object, object's shape, descriptive information, and spatial relations on the web interface. We discuss the 3D data modeling technique and content-based retrieval in detail.

1 Introduction

The explosive growth in the number of web-based applications has made the support of multimedia data in web-based information systems a hot research topic. Graphic data is probably one of the most frequently used data types in today's web applications. The significance of 3-Dimensional (3D) graphic information has been demonstrated in many areas such as e-commerce, virtual reality, geographic information system, and entertainment [1, 2, 3].

As more graphic data are used in applications, the methods that support content-based retrievals of 3D graphic information are desired. Most current 3D graphic systems focus on visualizing 3D images. One problem with this approach is that it is difficult to retrieve or manipulate a particular domain object separately from others.

We have developed a 3D graphic data model that supports a content-based retrieval for 3D scenes. This paper presents the data model and a retrieval system based on it. The 3D graphic data model, called 3DGML (3-Dimensional Graphical Markup language), allows the semantics of 3D objects to be incorporated into a 3D scene. The semantics implied in a 3D image include roles of component objects, semantic or spatial relationships among objects and composition hierarchies of objects. This support of semantic information allows for a content-based retrieval of scenes. Scenes are modeled as compositions of 3D graphic objects. A set of primitive 3D objects is used as building blocks for modeling 3D scenes instead of lines and polygons. Larger 3D objects are defined through a composition of other objects.

In our system, the user can search scenes using various 3D graphic features such as object's shape, descriptive information and spatial relations of 3D objects they contain. The user can also look for scenes or objects that contain a given object as a component. A query on the shape of an object enables a scene to be retrieved based on a particular shape it contains. The shape is determined by the contour information of an object. A query on a spatial relation allows scenes to be searched based on relative placements of objects in the space. Finally, a query can use textual descriptions of objects and scenes.

The 3D graphic retrieval system presented in this paper is implemented using XML. 3DGML is an XML vocabulary defined using an XML DTD (Document Type Definition). The choice of XML as the description mechanism of our data model makes the database system suited to web-based information systems. The user can pose content-based queries using a web browser. A query result is presented to the user through the web browser.
The remainder of this paper is organized as follows. The next section discusses previous researches related to our work. Section 3 describes 3DGML and modeling 3D images with it. Section 4 presents the information retrieval system for 3D graphic information that we developed. Section 5 concludes the paper.

2 Related Works

Most current works on graphic database systems are centered on the processing of 2D graphic data such as images and maps [4, 5, 6]. Research on 3D graphics has mainly concerned about the visualization of data to provide the user with a 3D feel [2, 3, 7]. Existing graphic systems traditionally treat a 3D object as a collection of lines and polygons rather than a unit of manipulation. It prevents them from supporting content-based retrievals or manipulations of 3D objects.

MPEG standard committees have been worked on the issues of modeling 3D images [8, 9]. MPEG-4 aims to develop techniques for representing synthetic images with natural objects efficiently. 3D objects are represented by 3D meshes, norm vectors, and their features such as color, texture, etc. However, modeling 3D objects as semantic units is not addressed by MPEG.

XML has been widely used for describing complex data types. The openness and the extensibility make XML an excellent vehicle for defining new languages with a relatively small effort [10]. Several domain specific languages have already been designed with XML before. 3D graphic data modeling can also benefit from XML. X3D (eXtensible 3D) is an open standards XML-enabled 3D file format and related access services for describing interactive 3D objects and worlds [11]. It may be used in a variety of application areas such as engineering and scientific visualization, CAD, training and simulation, multimedia, entertainment, educational, and more. However, X3D also does not consider the representation of semantics in 3D graphic modeling, which is used to retrieve 3D images from the database.

There have been some efforts to model the spatial relations of objects for 3D scenes. Xiong and Wang described a technique supporting similarity search for a chemical application [12]. They represent a 3D object using points in the Euclidean Space. An object is a 3D graph consisting of one or more substructures of connected subgraphs. Similarity of two objects is determined by comparing their substructures and edges. Gudivada and Jung proposed an algorithm for retrieving images of relevance based on similarity to user queries [13]. In their image representation scheme, an image is converted to an iconic image with human assistance. This method also determines the spatial relation of objects using the connectivity of graphs.

While the works discussed above considered geometrical similarity, they did not discuss modeling the semantics of 3D objects and querying based on the contents of 3D images. Their views still remain graph-oriented. In order to address the problems discussed above a new data model is needed that treats 3D objects as first class objects in modeling. The new data model should represent 3D data using semantic units rather than primitive geometrical objects.

3 3D Graphic Modeling in 3DGML

3.1 Data Model and 3DGML

We first present the graphic data model used in our system. 3DGML models 3D scenes using domain objects and their spatial semantics. A 3D scene is an image consisting of one or more 3D objects which are meaningful in a domain. 3DGML is defined by an XML DTD which is shown in Fig. 1. We use the tags of X3D as much as possible to convert 3DGML documents to X3D easily.

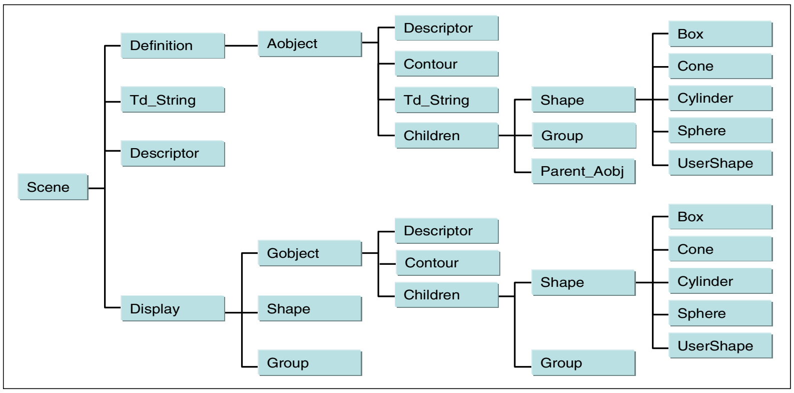


Fig. 1. 3DGML DTD

A 3D scene is modeled using three types of components: 3D objects contained in the scene, spatial relations on the objects, and descriptors. A simple 3D object is modeled using basic objects called shapes. A *shape* object is a system-defined 3D graphic object like VRML. Examples of shapes are box, cone, cylinder and sphere. An object of an arbitrary shape that is difficult to model with basic objects only, such as pyramid, triangular prism, etc, is defined using one or more polygons. Such an object is called a *user-defined shape* (*UserShape*). Each polygon of a user-defined shape may have associated properties as with basic objects. A complex 3D object is modeled as a composition of shapes, user-defined shapes, and other complex objects.

Every 3D object within a scene exists in the form of a *Gobject* (*Graphic object*). A Gobject is defined by extending an abstract object. An *abstract object* is a skeletal object that is used as a prototype of other objects. We called it as *Aobject*. It is a template that does not physically exist in a scene. It specifies the shape of a 3D object and partially describes its appearance. For example, an abstract object may define structure, color, and texture for its child objects. Hence, the modeling of a scene typically involves defining abstract objects first. In many cases, abstract objects represent semantic units such as desk, chair, etc that are germane to an application domain.

The definition of an abstract object consists of descriptive meta-information called descriptor, contour, compound objects (*Children*), and 3D string [14] that defines spatial relation on the objects. A compound object may be comprised of one or more of shapes, user-defined shapes, or other compound objects. The contour information is represented as a feature value of an object. The contour object of a 3D object is defined as a set of wrapped surfaces of the object, which has a cube-like shape [15]. The contour value is a sum of norm vectors of surfaces composing a contour object.

Every 3D object embedded in a scene is a Gobject. To create a Gobject, an abstract object needs to be specialized by adding further information required to render its clone in a specific scene context. The information on the translation, rotation, and scaling in the context of a particular scene needs to be specified in addition to what the abstract object has already defined. Separating the structure information of an object from its physical rendering for a scene and the use of prototype based object instantiation simplify the process of creating new 3D objects. Objects of a kind can easily be modeled without creating many superfluous classes.

A 3D string can express the concepts such as A is located to the left of B (left-right relation), A is above B (top-bottom relation), and A is closer than B (front-back relation). The representation of a 3D string is derived from the 1D string technique [16]. A 1D string is an encoding of the order of the positions of objects in the linear space. The ordering symbols used for 1D strings are "<" and "=" which means closer and equal, respectively. A 3D string is a 3-tuple (u, v, w), where u, v, and w represent the 1D strings obtained when objects are projected to the X, Y, and Z-axis, respectively.

3.2 A 3D Modeling Example

We now show an example 3DGML document that models a 3D scene. Fig. 2 shows a scene of an office that will be used in our discussion below. It is a 3D scene of furniture in an office. We label Gobject's id and related Aobject's id in the parenthesis for each object. Fig. 3 is a stripped version of a 3DGML document for the scene.

The model in Fig. 3 defines the scene "OFFICE_02." It contains two major blocks: the *Definition* block that defines abstract objects used in the scene and the *Display* block that defines the actual contents of the scene. The *Td_string* element defines the spatial relation for the objects contained in this scene. It specifies the spatial relation of Gobjects G001 through G009. For example, the u value of the 3D string specifies the ordering of the nine objects with respect to the X-axis. "G001<G002" means that G001 is located to the left of G002. Since G005, G006 and G007 are on the same locations with respect to the X-axis and located to the left of G008, the 3D string is represented as "G005=G006=G007<G008." We discuss the *Display* block first.

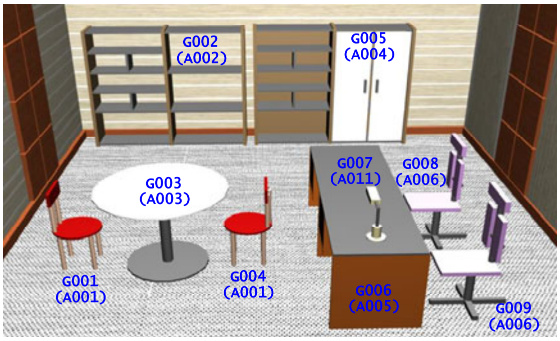
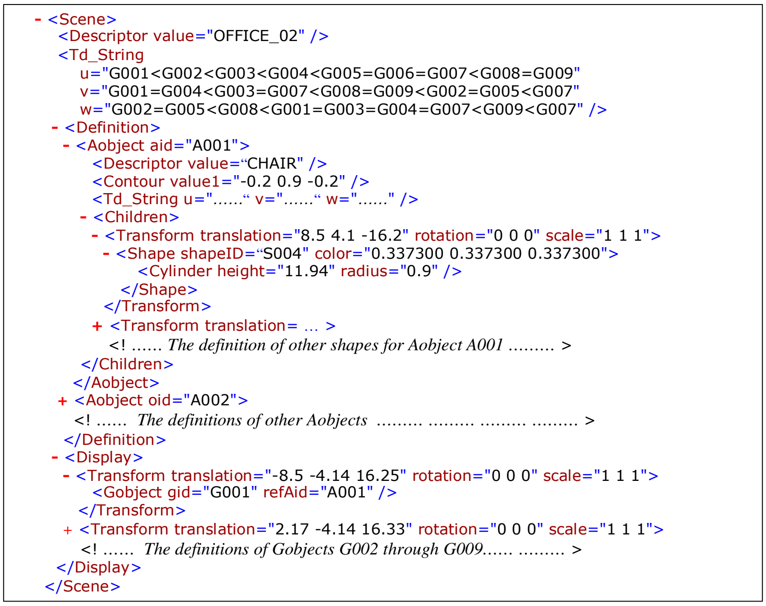


Fig. 2. The 3D scene of an office



```
- <Scene>
    <Descriptor value="OFFICE_02" />
    <Td_String
      u="G001<G002<G003<G004<G005=G006=G007<G008=G009"
      v="G001=G004<G003=G007<G008=G009<G002=G005<G007"
      w="G002=G005<G008<G001=G003=G004=G007<G009<G007" />
  - <Definition>
    - <Aobject aid="A001">
        <Descriptor value="CHAIR" />
        <Contour value1="-0.2 0.9 -0.2" />
        <Td_String u="......" v="......" w="......" />
      - <Children>
        - <Transform translation="8.5 4.1 -16.2" rotation="0 0 0" scale="1 1 1">
          - <Shape shapeID="S004" color="0.337300 0.337300 0.337300">
                <Cylinder height="11.94" radius="0.9" />
              </Shape>
          </Transform>
        + <Transform translation= ... >
            <! ...... The definition of other shapes for Aobject A001 ......... >
        </Children>
      </Aobject>
    + <Aobject oid="A002">
      <! ......  The definitions of other Aobjects  ......... ......... ......... ......... >
    </Definition>
  - <Display>
    - <Transform translation="-8.5 -4.14 16.25" rotation="0 0 0" scale="1 1 1">
        <Gobject gid="G001" refAid="A001" />
      </Transform>
    + <Transform translation="2.17 -4.14 16.33" rotation="0 0 0" scale="1 1 1">
       <! ......  The definitions of Gobjects G002 through G009...... ......... >
    </Display>
  </Scene>
```

Fig. 3. The definition of the scene in Fig. 2

The *Display* block contains the definitions of Gobjects G001 through G009 which model the objects labeled in Fig. 2, correspondingly. A Gobject may be defined by using an abstract object as its prototype, in which case its features should be modified as needed. Object G001 is an example of a Gobject defined using A001, an abstract object, as its prototype. It is a concrete extension of A001 with a translation.

We now narrow our discussion to the modeling of chair G001 in the Fig. 2. The 3DGML model shown in Fig. 2 defines chair G001 in a two-step process. It first defines an abstract object, A001, which resembles the shape of the target object, G001. It then defines G001 as a Gobject by declaring A001 to be its prototype and specifying additional information needed to create a concrete graphic object to be inserted in the scene.

According to the definition of the abstract object A001, it consists of many basic shapes including a cylinder whose height has 11.94 with the radius of size 0.9. The information defined by the shape object in the abstract definition still is not specific enough to be used in a scene as it barely defines the information needed by A001. This example defines the relative location and scaling factors within A001 and the default color for the objects contained in it. These values may need to be modified or complemented with further information to fit it in a specific scene. The definition of A001 contains one descriptor, which describes it as a chair. The *Contour* element of A001 contains a vector value of the contour object for A001. The *Td_string* element defines the spatial relation for the objects contained in it. It specifies the spatial relation among component objects of A001.

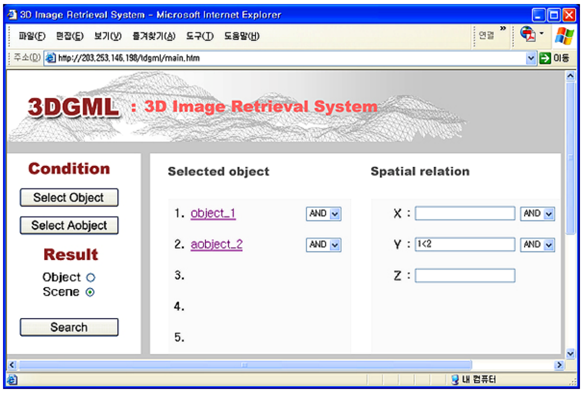
4 Content-Based Graphic Information Retrieval

In 3DGML, a 3D scene is modeled hierarchically such that a scene is composed of objects and the objects are composed of component objects. This relationship composes a "*scene-object-component*" hierarchy and can be used as selection criteria in a query. In the 3D graphic retrieval system, queries can be classified into the following three categories.

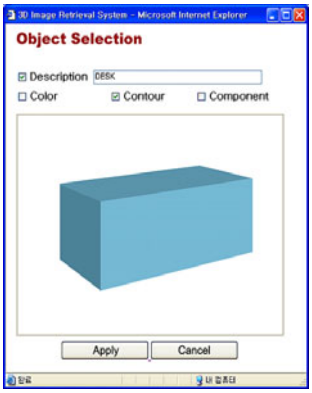
- The descriptions on scenes and objects can be used to querying database. An example query is "find a scene containing an object whose descriptor is desk."
- 3D objects have features such as color, contour and spatial relations. These 3D graphic features can be used as query conditions. A query like "find a scene including a lamp on a desk" is an example.
- Finally, user can retrieve scenes based on a hierarchical relationship among components of a 3D scene. It is possible to find an Aobject that contains a given shape or find a scene that contains a given Aobject or Gobject. Finding Gobjects that is defined by using a given Aobject is also possible.

A sample query on the 3D database system is now described. It retrieves the scenes that contain a lamp on a desk. Fig. 4 shows the screens displayed by our retrieval system. The query screen shown in Fig. 4(a) is the user interface with which the user enters queries.

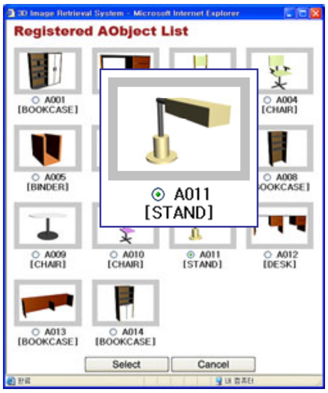
The user specifies object selection conditions by pressing "Select Object" or "Select Aobject" button. When "Select Object" button is pressed, the object selection window of Fig. 4(b) is popped up for the expression of conditions. The conditions of Fig. 4(b) mean an object whose descriptor contains a word "DESK" and contour object is the same shape as the given box object. Since the color checkbox is not selected, the color feature of the given object is not considered. If the component checkbox is selected, objects containing the given shape as a component are selected. If "Select Aobject" button is pressed, the Aobject list window of Fig. 4(c) is shown. The user selects an Aobject to be used as a query condition from the list. A lamp object is selected in our example. For each time the user specifies a search condition for an object, a conditioned object is added to "Selected object" part of the user interface.



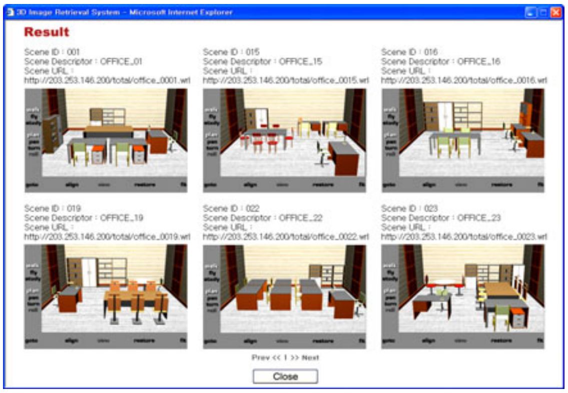
(a) The query screen



(b) Object selection



(c) Aobject list


Result
Scene ID : 001 Scene Descriptor : OFFICE_01 Scene URL : http://203.253.146.200/total/office_0001.wrl
Scene ID : 015 Scene Descriptor : OFFICE_15 Scene URL : http://203.253.146.200/total/office_0015.wrl
Scene ID : 016 Scene Descriptor : OFFICE_16 Scene URL : http://203.253.146.200/total/office_0016.wrl
Scene ID : 019 Scene Descriptor : OFFICE_19 Scene URL : http://203.253.146.200/total/office_0019.wrl
Scene ID : 022 Scene Descriptor : OFFICE_22 Scene URL : http://203.253.146.200/total/office_0022.wrl
Scene ID : 023 Scene Descriptor : OFFICE_23 Scene URL : http://203.253.146.200/total/office_0023.wrl
Prev << 1 >> Next
Close

(d) The result screen

Fig. 4. A sample query and the result

After expressing all conditions on objects, the user states 3D string conditions in "Spatial relation" part using the id's of conditioned objects. In our example, spatial relation "1<2" is expressed, which means the "aobject_1", the lamp object selected in Fig. 4(c), is above the "object_1" that has the condition mentioned above.

The query conditions for the desk object are processed using the description and the contour information. All descriptions of both Gobjects and Aobjects are searched for the given text. The contour values of all objects are also compared with the given shape's contour value. The query processing for the lamp object is based on a simple comparison of oid's of abstract objects. The scenes shown in Fig. 4(d) are returned as the result of the query.

The implementation of the 3DGML system that we have discussed so far is based on the XML technology. The system consists of the semantic editor, the 3D object manager, the query coordinator and the database wrapper.

The user creates a 3D scene using a 3D graphic editor which generates a VRML file like 3D Max. The semantic editor converts the VRML document to a 3DGML document. In the process, descriptor information is added and feature values such as contour and 3D string are calculated. Then, the database wrapper parses a 3DGML document and maps it to a relational database. A relational schema is defined to reflect the part of 3DGML structure that is used querying 3D images.

The query coordinator controls overall phases of query generation. It provides a graphical user interface so that users can enter a Query-By-Example style query. In the query processing step, the database wrapper generates an SQL query from the given content-based query, which returns 3DGML documents. The retrieved documents are converted to VRML documents to display its 3D image on the browser.

The current system has been implemented and runs with IIS(Internet Information Server) of Microsoft on the Windows NT platform. The XML parser was implemented in ASP using DOM API [17]. Parsed XML documents are stored in Oracle 9i.

5 Conclusion

While X3D has become a popular tool in many 3D application domains, few graphics applications are known to support content-based retrievals of 3D graphic data. We presented a 3D graphic system that offers a content-based retrieval of 3D scenes. One of the significant features that our system introduced is its support of semantic modeling for 3D scenes. 3D objects are modeled using semantic units rather than the primitive geometrical objects such as lines and polygons. An XML-based data modeling language called 3DGML was described in detail. It separates the implementation details of a 3D object from its semantic usage and supports modeling scenes in an object-oriented way. A content-based retrieval of 3D objects on our system was described using an example. Searching may be based on 3D shapes and spatial relations.

For future work, we are planning on providing more elaborate algorithms to represent contour information of objects and the support of similarity query based on shapes of objects and scenes. We are also considering the indexing methods that help fast searching of 3D graphic data using their features.

Acknowledgements

This work was supported by the Internet information Retrieval Research Center (IRC) in Hankuk Aviation University. IRC is a Regional Research Center of Gyeong-gi Province, designated by ITEP and Ministry of Commerce, Industry and Energy, Korea.

References

1. Menendez, R. G., Bernard, J. E.: Flight Simulation in Synthetic Environments. IEEE Proc. of the Digital Avionics Systems Conferences, Vol. 1 (2000)
2. Jie, S.: Visualizing 3D Geographical Data with VRML. IEEE Proc. of the Int. on Computer Graphics (1998) 108–110

3. Weinhaus, F. M., Devarajan, V.: Texture Mapping 3D Models of Real-World Scenes. ACM Computing Survey, Vol. 29, No. 4 (1997) 325–368
4. Bimbo, A. D.: Visual Information Retrieval. Morgan Kaufmann (1999)
5. Brunelli, R., Mich, O.: Image Retrieval by Example. IEEE Transactions on Multimedia, Vol. 2, No. 3 (2000) 164–171
6. Hung, K. W., Yong, M. A.: A Content-based Image Retrieval System Integration Color, Shape and Spatial Analysis. IEEE Proc. of the Int. Conference on Systems, Man, and Cybernetics, Vol. 2 (2000) 1484–1488
7. Hwang, S., Cho, S., Wang, T., Sheu, P. C. Y.: A Fast 3D Visualization Methodology Using Characteristic Views of Objects. Int. J. of Software Eng. and Knowledge Eng., Vol. 8, No. 1 (1998)
8. Huang, Q., Puri, A., Liu, Z.: Multimedia Search and Retrieval: New Concepts, System Implementation, and Application. IEEE Transactions on Circuits and System for Video Technology, Vol. 10, No. 5 (2000) 679–692
9. Hunter, J.: MPEG-7 Behind the Scenes. D-Lib Magazine, Vol. 5, No. 9 (1999)
10. Bray T., Paoli, J., Sperberg-McQueen, C. M.: Extensible Markup Language (XML) 1.0. http://www.w3.org/TR/1998/REC-xml-19980210 (1998)
11. Web 3D Consortium. X3D and Related Specifications. http://www.web3d.org/x3d/specification.html (2005)
12. Xiong, W., Wang, J. T. L.: Fast Similarity Search in Database of 3D Objects. IEEE Int. Conference on Tools with Artificial Intelligence (1998) 16–23
13. Gudivada, V. N., Jung, G. S.: Spatial Knowledge Representation and Retrieval in 3D Image Database. IEEE Pro. of the Int. Conference on Multimedia Computing and Systems (1995) 90–97
14. Hwang, J., Roh, J., Lee, K., Park, J., Hwang, S.: An XML-based 3-Dimensional Graphic Database System. The Human Society and the Internet. LNCS, Vol. 2105 (2001) 454–467
15. Hwang, J., Hwang, S.: An XML-based 3D Graphic Database System Supporting Content-based Retrievals: Proc. of SPIE: Internet Multimedia Management Systems II (2001) 73–83
16. Chang, S. K., Shi, Q. Y., Yan, C. W.: Iconic Indexing by 2-D Strings. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 9, No. 3 (1987)
17. W3C Consortium. Document Object Model (DOM). http://www.w3.org/DOM/ (1998)

Performance Improvement in Collaborative Recommendation Using Multi-Layer Perceptron

Myung Won Kim and Eun Ju Kim

School of Computing, Soongsil University, 511, Sangdo-Dong, Dongjak-Gu, Seoul, Korea
mkim@comp.ssu.ac.kr, blue7786@ssu.ac.kr

Abstract. Recommendation is to offer information which fits user's interests and tastes to provide better services and to reduce information overload. It recently draws attention upon Internet users and information providers. Collaborative filtering is one of the widely used methods for recommendation. It recommends an item to a user based on the reference users' preferences for the target item or the target user's preferences for the reference items. In this paper, we propose a neural network based collaborative filtering method. Our method builds a model by learning correlation between users or items using a multi-layer perceptron. We also investigate integration of diverse information to solve the sparsity problem and selecting the reference users or items based on similarity to improve performance. We finally demonstrate that our method outperforms the existing methods through experiments using the EachMovie data.

Keywords: recommendation system, collaborative filtering, personalization, multi-layer perceptron.

1 Introduction

Recommendation is to select and offer information or items which fits user's interests and tastes by filtering the given set of information or items to provide better service and to reduce information overload. For recommendation there are generally three different methods depending on the kinds of information used to filtering: content-based filtering, demographic filtering, and collaborative filtering. Content-based filtering estimates a user's preference for an item based on its similarity in some properties with the items with known preferences. Demographic filtering performs such estimation based on the user's demographic information such as gender, age, hobbies, and occupation. Collaborative filtering predicts a user's preference for an item based on other correlated users' preferences for the item. Different filtering methods can be best applied to different application problems and they have their strengths and weaknesses. For example, content-based filtering is good for recommending books and documents for which contents can be clearly specified. However, it may not be useful for recommending items for which contents cannot be clearly defined. It is also difficult to recommend items not similar in contents but closely related to some given items. Collaborative filtering can be best applied to problems for which items are recommended based on their correlated preference behaviors without the need of knowing the contents of items or demographic information of the users. Collaborative filtering can be considered more general than the other two filtering methods. It has been reported that content-based filtering and demographic filtering lack flexibility in recommendation and their performances are generally lower compared with collaborative filtering [1], [2], [3].

There have been two typical methods for collaborative filtering: the *k*-NN method and the association rule method [4], [5]. In spite of some advantages, their performances are generally low. The low performances of the methods are mainly due to the limitation of their underlying models. The *k*-NN method assumes that the attributes of data are independent, however, it is usually not the case. Association rules are limited in representing complex quantitative relationships among data [6], [7].

In this paper, we propose collaborative filtering based on neural network. In general, it has several advantages over the conventional models. Some important features of a neural network include: (1) it can learn a complex relationship between input and output values; (2) it can easily integrate diverse information; (3) it can handle various data types; (4) it can handle incomplete information efficiently. These distinguishing features can well fit to problems such as collaborative filtering. In our method, a multi-layer perceptron is adopted as the basic neural network architecture. A multi-layer perceptron is trained to learn a correlation among preferences of the target user and the reference users. The resulting model is called a user model. The same principle can be applied to build an item model.

Our neural network model has several advantages over the existing methods. First of all, it is expected that its performance is improved because it can learn a complex relationship of preferences among users or items. In addition, the hidden nodes of the model represent latent concepts for recommendation and it can improve performance. Next, it is easy to handle diverse data types including continuous numeric data, binary or logical data, and categorical data. Finally, it is easy to integrate diverse kinds of information such as contents and demographic information for efficient filtering. We investigate integration of contents information into the user model to examine the possibility of solving the sparsity problem. In training a neural network it is important to use good features as input data. For this we also investigate selection of the reference users or items based on the similarity between the target user and the reference users or the target item and the reference items.

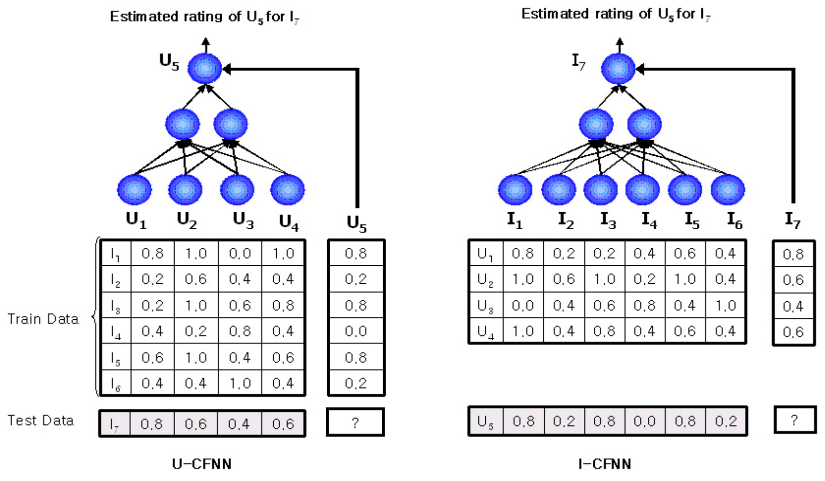
This paper is organized as following. In Section 2 we describe our neural network based collaborative filtering and some methods to improve the performance. In Section 3 we briefly review the existing collaborative filtering methods. In Section 4 we describe our experiments with our method and compare the performance of our method with those of the existing methods. Finally, in Section 5 we make a conclusion and describe future research.

2 Collaborative Filtering Based on Neural Network

Among many different types of neural networks we adopt as the basic neural network model the multi-layer perceptron(MLP) [8], which is commonly used in various applications. In this section we describe our method of building collaborative recommendation models based on the MLP. We also describe integration of additional information and selection of the reference data to improve the performance of our model.

2.1 Neural Network Models

There are two neural network models for collaborative filtering (CFNN): a user model called U-CFNN and an item model called I-CFNN. In the U-CFNN model the input nodes correspond to the reference users' preferences and the output node corresponds to the target user's preference for the target item. In the I-CFNN model the input nodes correspond to the target user's preferences for the reference items and the output node corresponds to his preference for the target item. CFNN can be designed to include more than one output node for multiple users or items. In this case it needs to train the whole network for incremental learning for even a small number of users or items. Furthermore, it is difficult to utilize user or item specific information such as the selected reference users or items. Thus we do not consider such models in this paper.


Estimated rating of U_5 for I_7
U_5
U_1 U_2 U_3 U_4
U_5
Train Data
Test Data
U-CFNN
Estimated rating of U_5 for I_7
I_7
I_1 I_2 I_3 I_4 I_5 I_6
I_7
I-CFNN

U-CFNN

	U_1	U_2	U_3	U_4	U_5
I_1	0.8	1.0	0.0	1.0	0.8
I_2	0.2	0.6	0.4	0.4	0.2
I_3	0.2	1.0	0.6	0.8	0.8
I_4	0.4	0.2	0.8	0.4	0.0
I_5	0.6	1.0	0.4	0.6	0.8
I_6	0.4	0.4	1.0	0.4	0.2
I_7	0.8	0.6	0.4	0.6	?

I-CFNN

	I_1	I_2	I_3	I_4	I_5	I_6	I_7
U_1	0.8	0.2	0.2	0.4	0.6	0.4	0.8
U_2	1.0	0.6	1.0	0.2	1.0	0.4	0.6
U_3	0.0	0.4	0.6	0.8	0.4	1.0	0.4
U_4	1.0	0.4	0.8	0.4	0.6	0.4	0.6
U_5	0.8	0.2	0.8	0.0	0.8	0.2	?

Fig. 1. Example of Training CFNN

The U-CFNN model is produced by learning the target user's (U_t) correlation with the reference users (U_1, U_2, ... $U_{n-1,}$ U_n). When an item is given for recommendation the model outputs an estimation of the target user's preference for the item based on other users' preferences for the item. For example, suppose we are given the user-item preference ratings and we need to predict user U_5's rating for item I_7 by referencing other users' ratings. We associate the input nodes with users U_1, U_2, U_3 and U_4 as the reference users and the output node with the target user U_5. We train a U-CFNN model using the ratings of users for each item. The resulting model represents a general behavior of the target user's preference as associated with the reference users' ratings. Now, for a new item, for example $I_{7,}$ the U-CFNN model can estimate the target user's (U_5) rating for the item based on the reference users' ratings for the item.

An I-CFNN model can be built in a similar way to that described in the above and it is basically of the same structure as that of the U-CFNN model. It mainly differs from the U-CFNN model in that it represents preference association among items.

Given an item for recommendation its rating is estimated based on ratings for other items rated by the target user.

2.2 Information Integration

Collaborative filtering often suffers the sparsity problem in which the performance is low when the rating information is not sufficient in the data. For a new item, only a limited number of users rate their preferences for the item. In this case collaborative filtering has difficulty to estimate the preference for the item correctly. To solve the sparsity problem and to improve the performance, [10] used the singular value decomposition (SVD) to use latent structure in the data. [1] and [11] proposed integration of collaborative filtering and content-based filtering. In [11] the content-based predictor estimates ratings of users for the target item and that rating information is fed into the collaborative filtering system to estimate the rating for the target item.

In this paper we investigate integration of additional information into CFNN as shown in Fig. 2, to solve the sparsity problem. We consider integrating content information of items into the U-CFNN model. For example, we can integrate content information such as the genre of movies into the U-CFNN model. Also we can integrate into the I-CFNN model user's demographic information such as age, gender, job, and hobbies.

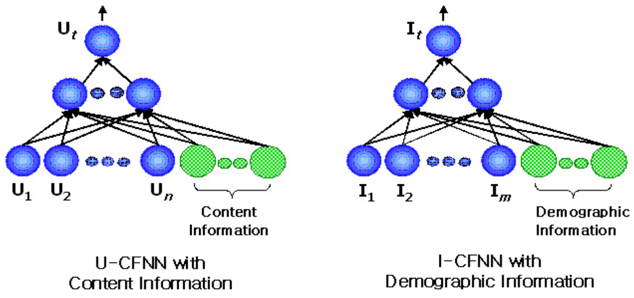


Fig. 2. CFNN with Information Integration

An MLP for integrating additional information can be built simply by adding new input nodes corresponding to the additional information and connecting them to hidden nodes as shown in Fig. 2. Our method takes advantage that it is very easy to integrate different kinds of information using neural networks. However, we describe later an experiment with integrating the genre information into the U-CFNN model. We do not consider integrating the demographic information into the I-CFNN model because of lacking useful demographic information in the data.

2.3 Selection of Reference Data

We investigate using similarity in selecting the reference users or items. In the CFNN model there is no restriction on selecting the reference users or items. However, For the U-CFNN model if we select those users who are similar to the target user in preference, the model can learn stronger correlation among users than random selection of the reference users and consequently, the performance can be improved. Fig. 3 illustrates a U-CFNN model with the selected reference users as input users. In this paper we adopt Pearson's correlation coefficient as similarity measure. We select k users with the highest Pearson's correlation coefficients.

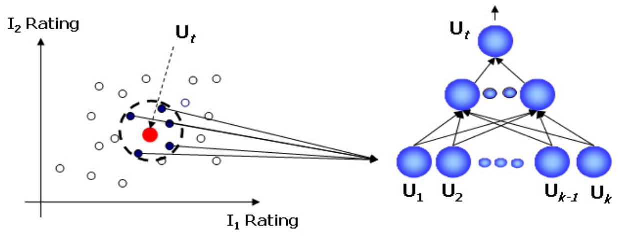
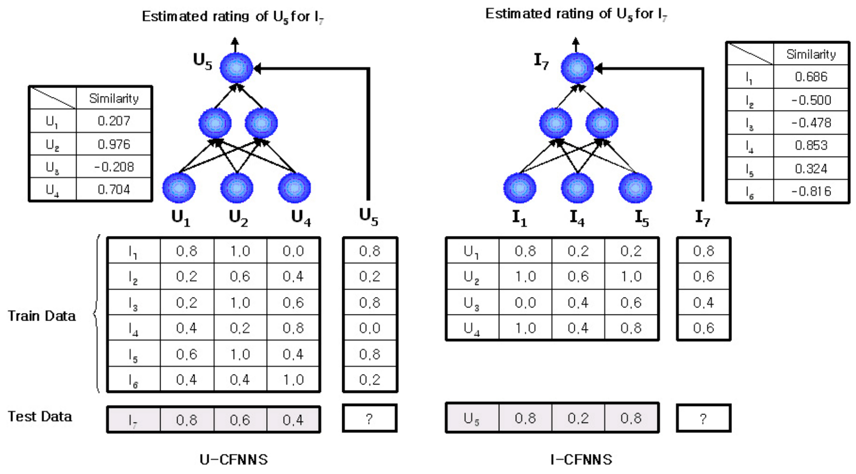


Fig. 3. U-CFNNS

Fig. 4 illustrates building a U-CFNNS (U-CFNN with the selected reference users). It computes the similarities of preferences between the target user (U_5) and other users (U_1, U_2, U_3, and U_4). If *k* is three, it produces a model by selecting three users (U_1, U_2, and U_4) as the reference users, who have the highest similarities with the target user. The selected reference users correspond to the input nodes. Similarly, for the I-CFNNS model three items (I_1, I_4, and I_5) are selected as similar to the target item (I_7).



	Similarity
U_1	0.207
U_2	0.976
U_3	-0.208
U_4	0.704

	U_1	U_2	U_4	U_5
I_1	0.8	1.0	0.0	0.8
I_2	0.2	0.6	0.4	0.2
I_3	0.2	1.0	0.6	0.8
I_4	0.4	0.2	0.8	0.0
I_5	0.6	1.0	0.4	0.8
I_6	0.4	0.4	1.0	0.2
I_7	0.8	0.6	0.4	?

	Similarity
I_1	0.686
I_2	-0.500
I_3	-0.478
I_4	0.853
I_5	0.324
I_6	-0.816

	I_1	I_4	I_5	I_7
U_1	0.8	0.2	0.2	0.8
U_2	1.0	0.6	1.0	0.6
U_3	0.0	0.4	0.6	0.4
U_4	1.0	0.4	0.8	0.6
U_5	0.8	0.2	0.8	?

Fig. 4. Example of CFNNS

3 Related Works

The *k*-NN method, which was used in GroupLens for the first time [9], is a memory-based collaborative filtering. In the *k*-NN method, a subset of users are chosen based on their similarities to the target user in preference and a weighted combination of their ratings is used to produce a prediction for the target user. The *k*-NN method is simple and easy to use, however, its performance is generally low. The reason for this is that the *k*-NN algorithm assumes that the attributes are independent. Consequently, it may not be efficient if independence among attributes is not guaranteed. It also has the scalability problem that computation time increases as the size of data increases [4], [6].

Association rules represent the relations between properties of data in the form of 'IF *A* Then *B*.' In association rules we can consider that the correlations (weights) between users or items are represented in terms of support and confidence of rules. However, more detail correlation may not be represented by support or confidence. In addition the performance is low because they rely on a simple statistics of co-occurrences of data patterns without considering intrinsic semantic structure in the data. It is also difficult to represent complex (not logical) relationships among data only in rules of attribute-value pairs.

[10] proposed a method, which is based on dimensionality reduction through the SVD of an initial matrix of user ratings. It exploits latent structure to essentially eliminate the need for users to rate common items and it solves the sparsity problem to some degree. [11] investigated combining the user model and the item model for performance improvement.

4 Experiments

We experimented with our method over the domain of movie recommendation. We used the EachMovie dataset [12]. It consists of 72,916 users, 1,628 movies, and 2,811,983 numeric ratings. Movies are rated by six different levels between 0.0 and 1.0 for the lowest preference (or "dislike"), the highest preference (or "like"), respectively. In our research we chose the first 1000 users who rated more than 100 movies.

Table 1. Rating Encoding

User Rating	0.0	0.2	0.4	0.6	0.8	1.0	missing
CFNN Encoding	-1.0	-0.6	-0.2	0.2	0.6	1.0	0.0

For our method, the ratings were transformed for efficient training as shown in Table 2. We also represented the missing ratings by 0.0. Generally, handling missing values is one of the difficult problems in data processing. Here we simply considered missing ratings the same as the in-between rating to be ignored in MLP processing and in our experiments we noticed that it did not affect the performance significantly. In training the MLP we only considered the target ratings greater than 0.7 or less than 0.3 and other ratings were ignored. We also transformed the ratings greater than 0.7 into 1.0 and the ratings less than 0.3 into 0.0 as the target output for the MLP.

For its generalization power we also fixed the number of hidden nodes to be small, such as five. In our experiments we used the learning rate 0.05 and the momentum constant 0.0.

We use four different measures for performance evaluation: accuracy, precision, recall, and F-measure as defined in equations (1) – (4). In the equations *T* (Total Items) represents the number of items rated and *TL* (Total Like Items) represents the number of items that users like. *R* is the number of items which are recommended to users by the model, *RL* (Recommended Like Items) is the number of items that are recommended and the target users like, and *NRD* (Not-Recommended Dislike Items) is the number of items that are not recommended and the target users dislike. For performance evaluation using those measures we quantized the output ratings of our model greater than or equal to 0.5 to "like" and less than 0.5 to "dislike", respectively.

$$accuracy = \frac{RL + NRD}{T} \tag{1}$$

$$precision = \frac{RL}{R} \tag{2}$$

$$recall = \frac{RL}{TL} \tag{3}$$

$$F - measure = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4}$$

In this section, we analyzed the performance of our method for recommendation and compared it with the existing methods in various aspects. First we analyzed and compared CFNN and CFNNS, and then evaluated the performance for integrating the genre information into the U-CFNNS model. Finally, we compared CFNN with the *k*-NN and the association rule methods.

4.1 Comparison of CFNN and CFNNS

We built 20 models for 10 users and 10 movies. We particularly selected 10 users whose preferences are unbiased in that they rated almost equal number of items as "like" and "dislike". In this case it is generally difficult to predict user's preferences correctly. Each model is evaluated by 4-fold cross validation. In our research for efficiency we limit the number of the reference users or movies to 100. In CFNN the reference users or movies are selected randomly while in CFNNS users whose preferences are similar to that of the target user are selected. Table 2 compares CFNN and CFNNS in performance. As shown in the table CFNNS clearly outperforms CFNN. The performance improvement can be accounted for by the difference of average similarities between random selection and similarity based selection of the reference users or items. We notice that the performance improvement for the item model is greater than for the user model. In this case the difference is 0.32 for the item model while it is 0.27 for the user model.

Table 2. Performance Comparison of CFNN and CFNNS (%)

	User Model		Item Model	
	U-CFNN	U-CFNNS	I-CFNN	I-CFNNS
Accuracy	82.1	83.6	77.7	81.5
Precision	81.7	82.6	77.9	82.0
Recall	84.1	85.6	75.8	79.4
F-Measure	82.9	84.1	76.2	80.7

4.2 Information Integration

It is often the case that integrating diverse relevant information or combining multiple methods yields better performance. This is also true with recommendation. [1] and [11] demonstrated performance improvement by integrating additional information and combining different filtering methods. Especially, such integration can solve the sparsity problem in collaborative filtering. In this paper we describe integration of diverse information for performance improvement and solving the sparsity problem using a neural network. Demographic information in the EachMovie data includes gender, age, residential area, and job. However, we decided to ignore it. It is because such demographic information is of little use: residential area and job are little correlated with movie preference while gender and age appear to be significantly biased in the data.

Instead, we integrated the genre of movies into the U-CFNN model as well correlated to user's preference. The EachMovie dataset has 10 different genres of movies and each movie can have more than one genre specified. Although each genre is represented in binary, when we trained the model we transformed it by multiplying the target user's rating for the movie and used the result as input to the neural network model as shown in Fig. 2. We experimented with the 10 users under the same conditions as described previously.

Table 3. Performance Comparison of U-CFNNS and U-CFNNS with Genre (%)

	No. of Reference Users (U-CFNNS)					
	10		50		100	
	User	User/Genre	User	User/Genre	User	User/Genre
Accuracy	77.3	82.8	80.5	84.2	83.6	84.1
Precision	75.2	81.4	80.3	83.4	82.6	84.0
Recall	85.9	87.8	82.8	87.0	85.6	85.2
F-measure	79.6	83.4	81.4	84.4	84.1	83.7

Table 3 compares the performances of U-CFNNS with and without the genre information. In the experiment we examined the effect of the genre information by varying the number of the reference users. As we can see in the table when the number of the reference users is smaller, the performance improvement is larger. Integration of diverse relevant information will help recommending items correctly to a new user who only has a limited number of reference users. This result demonstrates that integration of relevant information can solve the sparsity problem in collaborative filtering and our neural network model proves to be easy to integrate such information.

4.3 Performance Comparison of CFNN and the Existing Methods

In this section, we compare our collaborative filtering method with the existing methods. For comparison we used the first 1000 users in the EachMovie dataset, who rated more than 100 movies as the training data, and the first 100 users whose user IDs are greater than 70,000 and who rated more than 100 movies as the test data. Using the data we built 30 user and item models. Especially we selected 30 target users randomly among those who have user ID over 70,000. Two thirds of 30 models are trained using data of unbiased preferences and the rest are trained using data of a little biased preferences.

Tables 4 and 5 compare in performance our method with the existing methods such as *k*-NN and association rule methods. As shown in the tables the CFNN and CFNNS models show a significant improvement of performance compared with the existing methods.

Table 4. Perfomance Comparison of the User model and the Existing Methods (%)

	k-NN	Assoc. Rule	U-CFNN		U-CFNNS	
			User	User/Genr e	User	User/ Genre
Accuracy	67.8	72.0	81.6	81.4	87.2	88.1
Precision	60.3	75.1	77.4	78.0	86.6	88.5
Recall	55.7	58.4	69.6	65.7	82.8	88.3
F-measure	57.9	65.7	73.3	71.3	83.3	85.7

Table 5. Performance Comparison of the Movie Model and the Existing Mehtods (%)

	k-NN	Assoc. Rule	I-CFNN	I-CFNNS
Accuracy	64.7	61.1	76.2	79.1
Precision	67.8	75.4	76.1	74.4
Recall	59.8	22.6	72.1	76.6
F-measure	62.4	34.	72.7	74.6

5 Conclusions

In this paper, we propose a collaborative filtering method based on neural network called CFNN. We also propose some methods to improve the performance including integration of additional information and selection of the reference users or items based on similarity. Our model utilizes the advantages of a neural network over other methods. It is powerful to learn a complex relationship among data and it is easy to integrate diverse information. The experiment results prove that our method shows a significant improvement of performance compared with the existing methods. One of the weaknesses of our method is that the neural network model is not comprehensible, however, in recommendation it is usually the case that comprehensibility is not be important.

Acknowledgments. This works was supported by Seoul R&BD Program (Project No.:10581cooperateOrg93111).

References

1. Pazzani, M.J.: A Framework for Collaborative, Content-Based and Demographic Filtering. Artificial Intelligence Review. 13(5-6). (1999) 393-408
2. Ziegler, C, McNee, S. M., Konstan, J.A., and Lausen, G.: Improving Recommendation Lists Through Topic Diversification. Proceedings of the International World Wide Web Conferences 2005, ACM. (2005) 22-32
3. Cheung, K.W., Kwok, J.T., Law, M.H., Tsui, K.C.: Mining Customer Product Ratings for Personalized Marketing. Decision Support Systems. Volume 35. Issue 2. (2003) 231-243
4. Sarwar, B.M., Karypis, G., Konstan, J.A., and Ried, J.: Analysis of Recommendation Algorithms for E-Commerce. In Proceedings of the ACM EC'00 Conference. Minneapolis. MN. (2000) 158-167
5. Lin, W., Ruiz, C., and Alverez, S.A.: Collaborative Recommendation via Adaptive Association Rule Mining. International Workshop on Web Mining for E-Commerce(WEBKDD2000) (2000)
6. Herlocker, J.L., Konstan, J.A., Borchers, A. and Riedl, J.: An Algorithmic Framework for Performing Collaborative Filtering. In Proceedings on the 22nd annual international ACM SIGIR conference on research and development in information retrieval. Berkeley. CA. (1999) 230-237
7. Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., and Sartin, M.: Combining Content-Based and Collavorative Filters in an Online Newspaper. In Proceedings of ACM SIGIR'99 Workshop in Recommender Systems: Algorithms and Evaluation. Univ. of California. Berkely. (1999)
8. Haykin, S.: Neural Network : A Comprehensive Foundation, 2nd edition. Prentice-Hall. Inc. (1999)
9. Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L. And Riedl, J.: GroupLens: Applying Collaborative Filtering to Usenet News. Communications of the ACM, 40(3). (1997) 77-87
10. Billsus, D. and Pazzani, M.J.: Learning Collaborative Information Filters. In Proceedings of the Fifteenth International Conference on Machine Learing. (1998) pp. 46-53
11. Patrick Paulson and Aimilia Tzanavari: Combining Collaborative and Content-Based Filtering Using Conceptual Graphs. Modelling with Words, LNAI 2873, pp. 368-185 (2003)
12. McJones, P.: Eachmovie Collaborative Filtering Data Set. http://www.rearchdigital.com/SRC/eachmovie. DEC Systems Research Center. (1997)
13. Do, Y.A., Kim, J.S., Ryu, J.W. and Kim, M.W.: Efficient Method for Combining User and Article Models for Collaborative Recommendation. Journal of Kiss : Software and Application. Vol. 30. No. 56. (2003)

NN-OPT: Neural Network for Option Pricing Using Multinomial Tree

Hung-Ching (Justin) Chen and Malik Magdon-Ismail

Rensselaer Polytechnic Institute, Dept. of Computer Science, Troy, NY 12180, USA
{chenh3, magdon}@cs.rpi.edu

Abstract. We provide a framework for learning to price complex options by learning risk-neutral measures (Martingale measures). In a simple geometric Brownian motion model, the price volatility, fixed interest rate and a no-arbitrage condition suffice to determine a unique risk-neutral measure. On the other hand, in our framework, we relax some of these assumptions to obtain a *class* of allowable risk-neutral measures. We then propose a framework for learning the appropriate risk-neural measure. In particular, we provide an efficient algorithm for backpropagating gradients through multinomial pricing trees. Since the risk-neutral measure prices all options simultaneously, we can use all the option contracts on a particular stock for learning. We demonstrate the performance of these models on historical data. Finally, we illustrate the power of such a framework by developing a real time trading system based upon these pricing methods.

1 Introduction

In 1973, Black and Scholes published their pioneering paper [1] which introduced the first option pricing formula and also developed a general framework for derivative pricing. Since then, derivative pricing has become a popular research topic. A modern, popular approach to pricing has been though the Martingale measure (see, for example, [2]). The origin of the fundamental theorems on the Martingale measure can be traced to Cox and Ross' paper [3] describing the method of *risk neutral valuation.* The Martingale measure was developed into a more mature pricing technique, such as [4]. Other related topics can be found in [2,5].

Option trading by directly predicting prices and then building trading systems based on the predictions have been considered in the neural network literature [6,7]. An alternative to predicting prices and then trading is to use direct reinforcement to trade directly (see for example [8]). Learning to trade directly has the advantage of avoiding an additional price-prediction step. When multiple instruments are available, for example multiple options on a single underlying stock, then the state space of possible trading actions grows exponentially and direct reinforcement for learning to trade becomes infeasible. In addition, price prediction of each individual option leads to an excessive number parameters, and it now makes sense to develop a unified price prediction mechanism for all the options simultaneously. Once prices are predicted for all the options, trading can be performed independently on each of these options based on their respective prices. This is the motivation for this work, namely to present a unified framework for learning to price *all* the derivatives on a particular underlying stock.

The tool we use for accomplishing this task is the Martingale measure, which relates to the *stock dynamics*. If we can predict the stock dynamics in the risk neutral world, then we can price all derivatives on a particular stock. We summarize the advantages of predicting the risk neutral stock dynamics:

(i) Simultaneously prices all derivatives on a stock.
(ii) All derivative data can be used in learning.
(iii) No-arbitrage constraints exist for the risk neutral dynamics.

In contrast, learning to directly price each option suffers from two problems. The first is that more parameters must be learned, one set for each option. The second is that the data on a single option can be used only to learn to predict that particular option's price. On the other hand, only one set of prediction parameters need be learned for predicting the risk neutral dynamics, and all the option prices can be used to learn this single set of parameters – in effect, more data to learn fewer parameters. Also mentioned above are no-arbitrage constraints, which limit the possible risk neutral measures. The no-arbitrage requirement thus provides an economic constraint to regularize the learning in the right direction, further improving the generalization performance of the system.

The underlying theory for the pricing based on the risk neutral dynamics is that the prices can be computed as expected values of cashflows over the risk neutral stock price dynamics. Often the Martingale measure is not unique, and this is where learning comes in. We develop a framework for *learning* the Martingale measure. We assume that the stock dynamics can be represented on a multinomial tree. Binomial trees have often been used to price options [9,2,10]. In this work, we present the framework for general multinomial trees, and illustrate with trinomial trees [11], which is more complicated, more flexible and better illustrate the general principles – in the binomial model, there is no learning because the Martingale measure is unique. For background on option pricing and other financial topics, we suggest [10,11].

The outline of this paper is as follows: first, we give some basics of multi-period, multinomial trees and option pricing, before presenting the NN-OPT algorithm. We then give some experimental results (training and test) on high-frequency paper trading of IBM stock options based on the learned price prediction. Our results indicate that learning Martingale measures together with no-arbitrage regularization constraints performs best.

2 Multi-period Economy with Multinomial Tree

Before introducing NN-OPT, an algorithm to price options, we need set up the notation to describe the economy. Describe the price of an instrument by C and consider, for example, a 2-period economy (see Fig. 1(a)). Consider time steps m and $m+1$ (corresponding to times mT and $(m+1)T$). At time step

m, the instrument could be in one of many states, indexed by α, with price C_α^m. From state α at time step m, assume that the instrument can transition to one of L states, with prices $\{C_{\alpha 1}^{m+1}, \ldots, C_{\alpha L}^{m+1},\}$. Thus we use $C_{\alpha\beta}^{m+1}$ to denote the possible prices which the instrument can transition into at time step $m+1$ from state α at time step m. When $L = 2$, we have a binomial model, $L = 3$ is a trinomial model and for $L > 3$ a multinomial model. Let P_j denote the probability to transition to state j, $j = 1, \ldots, L$, and $\sum P_j = 1$. When P_j is independent of m and α, we have a standard multinomial tree dynamics for the instrument price. We can represent $C_{\alpha\beta}^{m+1}$ and P_j in vector notation,

$$\boldsymbol{C}_\alpha^{m+1} = \begin{bmatrix} C_{\alpha 1}^{m+1} \\ C_{\alpha 2}^{m+1} \\ \vdots \\ C_{\alpha L}^{m+1} \end{bmatrix} \quad \text{and} \quad \boldsymbol{P} = \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_L \end{bmatrix} .$$

When $m = 0$ (time 0), there is only one state C_0^0, and after each time period T, each state can transition to L possible states, which creates a multinomial tree in a multi-period economy (shown in Fig. 1.(b)).

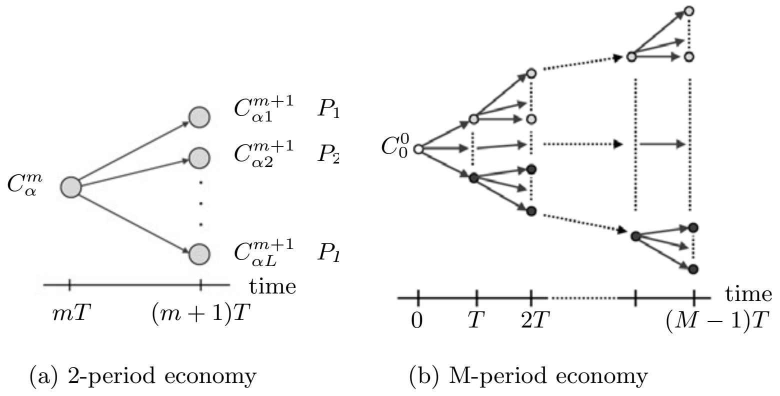


Fig. 1. The dynamics of economy

3 Option Pricing

NN-OPT is based on Martingale methods for options pricing and we briefly discuss some background on Martingale pricing. The basic theorem is that the discounted price is a Martingale with respect to some measure $\boldsymbol{P}$.

$$\begin{aligned} C_\alpha^m &= D(T) \times E_{\boldsymbol{P}}\left[C_{\alpha\beta}^{m+1}\right] \\ &= D(T) \times \textstyle\sum_{j=1}^{L} P_j C_{\alpha j}^{m+1} \\ &= D(T) \times \boldsymbol{P}^T \boldsymbol{C}_\alpha^{m+1} \quad , \end{aligned} \tag{1}$$

where $D(T)$ is the *risk free discount factor*, which depends on the interest rate and T. Intuitively, this formula means that the current prices are the present value of the expected future prices, where the expectation is with respect to the so called risk neutral probabilities $\boldsymbol{P}$. In this paper, we consider C to be the price of an American option, whose value can be realized by either exercising now or holding and optimally exercising later. Let $G(S^m, K)$ be the value of exercising at time m with strike K and stock price S^m. Then

$$C_\alpha^m = \max\left\{G(S^m, K), D(T) \times \boldsymbol{P}^T \boldsymbol{C}_\alpha^{m+1}\right\} \ . \tag{2}$$

Thus, we can use backward propagation to compute the current prices of options. To initiate the backward propagation, note that at last time step $(M-1)$, the options don't have any future value, and the option prices become

$$C_\alpha^{M-1} = \max\left\{G\left(S^{M-1}, K\right), 0\right\} \ . \tag{3}$$

Therefore, if we know the appropriate values[1] for S^{M-1} (i.e. the stock dynamics), we are able to determine C_α^{M-1} for all states at the last time step, and then we use the recursive algorithm to compute the current price C_0^0 of the option. The details of pricing options using multinomial trees can be found in numerous techniques of option pricing, for example [2,11].

4 The NN-OPT Learning Algorithm

The NN-OPT learning algorithm includes two parts, a standard neural network[2] probability predictor and a multinomial pricing tree. Figure 2 shows the structure of neural networks, where $w_{\eta\delta}^\theta$ denote the weights from node η to node δ in layer θ, and the set of weights are denoted by vector $\boldsymbol{w}$. The neural networks will predict and learn the probabilities $\boldsymbol{P}$ for pricing, used in the multinomial pricing tree. The input of the neural networks can be anything, such like short term interest rates, long term interest rates, technical indexes or previous historical data. The output of the neural networks is a set of amplitudes, $\{g_1, g_2, \ldots, g_{L-1}\}$, where $g_i \in [0,1]$. There is also a set of transfer functions $\{H_1, H_2, \ldots, H_L\}$, denoted by a vector $\boldsymbol{H}$, to transfer $\{g_1, g_2, \ldots, g_{L-1}\}$ into probabilities $\boldsymbol{P}$, where,

$$P_i = H_i(g_1, g_2, \ldots, g_{L-1}) \quad i = 1, \ldots, L \ . \tag{4}$$

The outputs g_i, $i = 1, \ldots, L-1$, are independent of each other. This property makes the process of backward propagation easier (in Sect. 4.3). In a trinomial model, for example, one choice for the transfer functions is

$$P_1 = g_1 \tag{5}$$
$$P_2 = (1-g_1) \times g_2 \tag{6}$$
$$P_3 = 1 - g_1 - (1-g_1) \times g_2 \ , \tag{7}$$

which satisfy $\sum P_i = 1$ and $0 \leq P_i \leq 1$, for $i = 1, \ldots, 3$.

[1] There are many techniques to determine appropriate values for S^{M-1}, such as historical volatility and GARCH volatility predictors, [9,12].

[2] The details of the structure and the procedures of a standard neural network, please refer to [13,14].

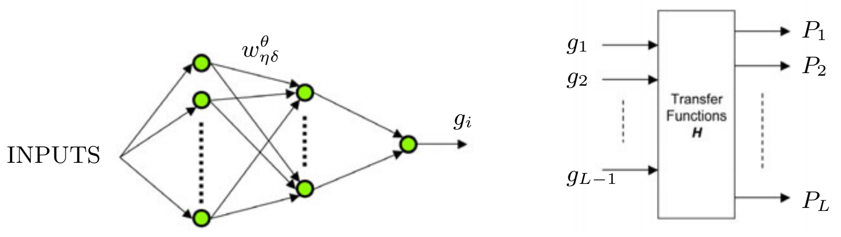


Fig. 2. The structure of the neural network

The second part of the NN-OPT learning algorithm is the multinomial pricing tree which uses the probabilities in $\boldsymbol{P}$ to price all options and compute the feedback to the neural networks for learning (updating of the weights).

4.1 Forward Propagation: Computation of Current Price C_0^0

Figure 3 shows the framework for forward propagation. Assume that we are given a data set, $D = \{\boldsymbol{x}_i, y_i\}_{i=1}^N$, where $\boldsymbol{x}_i$ is the input vector, and y_i is the expected output. The standard neural networks just apply the regular forward propagation to obtain $\{g_1, g_2, \ldots, g_{L-1}\}$, see for examples [13,14]. Then using the transfer functions $\boldsymbol{H}$, we calculate pricing probabilities $\boldsymbol{P}$. The pricing algorithm for the multinomial tree is as follows:

1. Use (3) to compute C_α^{M-1} for all states at last time step $(M-1)$.
2. Loop from $i = (M-2)$ to 0,
   – Use (2) and $\boldsymbol{C}_\alpha^{i+1}$ to compute C_α^i for all states at time step i .
3. Output C_0^0.

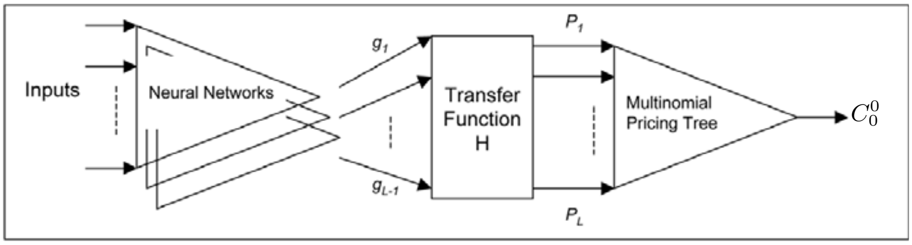


Fig. 3. The framework for forward propagation

4.2 Computation of Error Function $\boldsymbol{Error(w)}$

We define the error function as

$$Error(\boldsymbol{w}) = \frac{1}{N}\sum_{i=1}^{N}\left(C_0^0 - y_i\right)^2 \quad . \tag{8}$$

The goal is to find a set of weights $\boldsymbol{w}$ that minimizes $Error(\boldsymbol{w})$. To do this, using a gradient descent algorithm, we will need to backpropagate gradients through a multinomial tree and then through the neural network. Backpropagation of gradients through neural networks is standard (see for example [13,14]). We now develop an efficient algorithm for backpropagating gradients through a multinomial tree pricing framework.

4.3 Backward Propagation: Computation of the Gradients

The framework for backward propagation is shown in Fig. 4. In order to implement the gradient descent algorithm, we need

$$\frac{\partial Error(\boldsymbol{w})}{\partial w^{\theta}_{\eta\delta}} = \frac{2}{N}\sum_{i=1}^{N}\left(C^0_0 - y_i\right)\frac{\partial C^0_0}{\partial w^{\theta}_{\eta\delta}} \quad . \tag{9}$$

There are $L-1$ neural networks, and we need to compute (9) for each one. We will focus on a particular neural network $j \in \{1, 2, \ldots, L-1\}$. In (9), C^0_0 and y_i are known after forward propagation, and we have

$$\frac{\partial C^0_0}{\partial w^{\theta}_{\eta\delta}} = \frac{\partial C^0_0}{\partial g_j} \times \frac{\partial g_j}{\partial w^{\theta}_{\eta\delta}} \quad , \tag{10}$$

thus, we need $\partial C^0_0/\partial g_j$ and $\partial g_j/\partial w^{\theta}_{\eta\delta}$ to compute derivatives, (9). For the second term, $\partial g_j/\partial w^{\theta}_{\eta\delta}$, we just apply the regular backward propagation on a standard neural network, so we won't discuss it further. In the following, we will focus on the process of backward propagation in the multinomial pricing tree to determine $\partial C^0_0/\partial g_j$.

From (2) and (3), we know

$$C^m_\alpha = \begin{cases} \max\left\{G\left(S^m, K\right), 0\right\} & \text{when } m = M-1 \\ \max\left\{G\left(S^m, K\right), D(T)\boldsymbol{P}^T\boldsymbol{C}^{m+1}_\alpha\right\} & \text{when } m = 0, \ldots, M-2 \end{cases} , \tag{11}$$

then

$$\frac{\partial C^m_\alpha}{\partial g_j} = \begin{cases} \partial \max\left\{G\left(S^m, K\right), 0\right\}/\partial g_j & \text{when } m = M-1 \\ \frac{\partial \max\left\{G\left(S^m, K\right), D(T)\boldsymbol{P}^T\boldsymbol{C}^{m+1}_\alpha\right\}}{\partial g_j} & \text{when } m = 0, \ldots, M-2 \end{cases} . \tag{12}$$

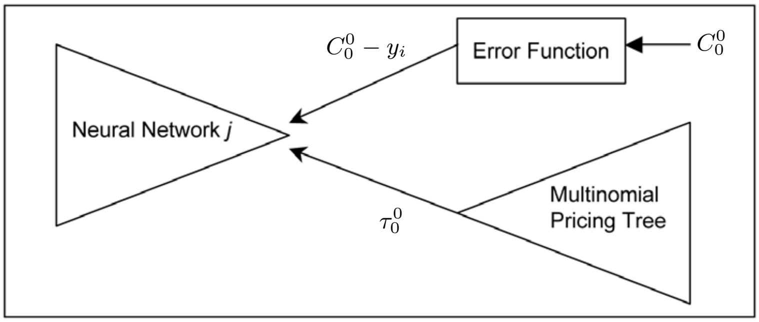


Fig. 4. The framework for backward propagation

Let $\tau_\alpha^m = \partial C_\alpha^m / \partial g_j$,

- Case 1: $C_\alpha^m = G(S^m, K)$. We know S^m and K are not related to g_j, then

$$\tau_\alpha^m = \frac{\partial G(S^m, K)}{\partial g_j} = 0 \ . \tag{13}$$

- Case 2: $C_\alpha^m = D(T)\boldsymbol{P}^T \boldsymbol{C}_\alpha^{m+1}$. Then,

$$\begin{aligned} \tau_\alpha^m &= \frac{\partial\left(D(T)\boldsymbol{P}^T\boldsymbol{C}_\alpha^{m+1}\right)}{\partial g_j} \\ &= D(T)\sum_{i=1}^{L}\frac{\partial P_i C_{\alpha i}^{m+1}}{\partial g_j} \\ &= D(T)\sum_{i=1}^{L}\left(\frac{\partial P_i}{\partial g_j}C_{\alpha i}^{m+1} + P_i\frac{\partial C_{\alpha i}^{m+1}}{\partial g_j}\right) \\ &= D(T)\sum_{i=1}^{L}\left(\frac{\partial P_i}{\partial g_j}C_{\alpha i}^{m+1} + P_i\tau_{\alpha i}^{m+1}\right) \ . \end{aligned} \tag{14}$$

We have discussed the value of $D(T)$ (in Sect. 3), and from (4), we know P_i is the output of $H_i(g_1, g_2, \ldots, g_{L-1})$. Then, in (14), we need to determine the rest of three terms, $C_{\alpha i}^{m+1}$, $\partial P_i/\partial g_j$, and $\tau_{\alpha i}^{m+1}$.

Computation of $\boldsymbol{C_{\alpha i}^{m+1}}$. In the forward propagation, Sect. 4.1, when calculating C_0^0, we also compute $C_{\alpha i}^t$ for all states at all time steps, where $t = 0, \ldots, (M-1)$, and we just save those values for backward propagation.

Computation of $\boldsymbol{\partial P_i/\partial g_j}$. From (4), we obtain

$$\frac{\partial P_i}{\partial g_j} = \frac{\partial H_i(g_1, g_2, \ldots, g_{L-1})}{\partial g_j} \ . \tag{15}$$

In the beginning of Sect. 4, we discuss $\boldsymbol{H}$ which are known because we construct the neural networks and decide those transfer functions, and we also know g_j is independent of $\{g_1, \ldots, g_{j-1}, g_{j+1}, \ldots, g_{L-1}\}$. Then, (15) is solvable. From the same trinomial example, see (5)(6), and (7), and then (15) can be computed easily; for instance,

$$\frac{\partial P_3}{\partial g_1} = \frac{1-g_1-(1-g_1)\times g_2}{\partial g_1} \ .$$

Computation of $\boldsymbol{\tau_{\alpha i}^{m+1}}$. From (14), we know τ_α^m at time step m can be computed from $\tau_{\alpha j}^{m+1}$, $j = 1, \ldots, L$, at next time step $m+1$, and from (12) and (13), we know τ_α^{M-1} for any state at last time step are 0. Then, we can use backward propagation for the multinomial tree to compute τ_α^m for any state at any time step. The algorithm is as follows:

1. Set all τ_{α}^{M-1} to 0.
2. Loop from $i = (M-2)$ to 0,
   – Use (14) to compute τ_{α}^{i} for all states at time step i from the value of $\tau_{\alpha_j}^{i+1}$, $j = 1, \ldots, L$.
3. Output τ_0^0.

Now, we have determine the derivative of error function $Error(\boldsymbol{w})$, (9), and then apply the regular process to update the set of weights $\boldsymbol{w}$ in standard neural networks. Then repeat the all process for each iteration of learning.

5 Results

We developed a simple trading system to evaluate the NN-OPT, which we tested using intraday real market data (5 minutes time period) for IBM (stock and option data) and interest rate data, from July 20, 2004 to April 29, 2005. We used the first 80 days, from July 20, 2004 to November 9, 2004, as the training data set, and used the remaining 118 days as test data set (see Fig.5). We compared the trading performance between different algorithms and different trading costs. Some algorithms, using NN-OPT with no-arbitrage constraint to learn *risk-neutral probabilities*, [2,15]. Intuitively, arbitrage is the possibility to make money out of nothing.

1. **Enforcing No-arbitrage, with learning:** This is based on a no-arbitrage condition and also uses NN-OPT learning algorithm to predict the *risk-neutral probabilities* for pricing.
2. **Not Enforcing No-arbitrage, with learning:** This approach is only based on NN-OPT learning algorithm without no-arbitrage constraints.

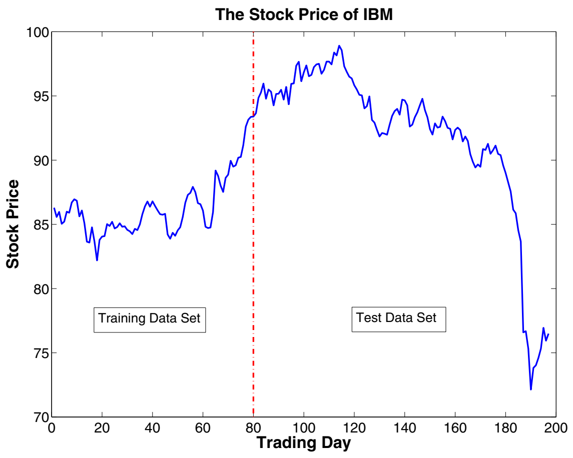


Fig. 5. The intraday market price of IBM

3. **Enforcing No-arbitrage, no learning:** This approach is to demonstrate that a no-arbitrage constraint alone, without learning the Martingale measures is worse than our framework.
4. **Not Enforcing No-arbitrage, no learning (random strategy):** This approach is to develop a benchmark performance using a random strategy.

The results of trading using these approaches are shown in Fig. 6, and the figure on right hand side has higher trading cost. Using both no-arbitrage constraint and NN-OPT clearly has the best performance. Note that the system still makes money even when the market crashes. As we move further from the training window, the performance degrades, though it remains positive. The results of the other algorithms are also reasonable because any random trading strategy will systematically lose the transaction cost on each trade which means that the total profit will drop linearly; the results also show that it is useful to use a no-arbitrage condition because it narrows the range of Martingale measures to obtain a set of plausible prices, rather than pure random.

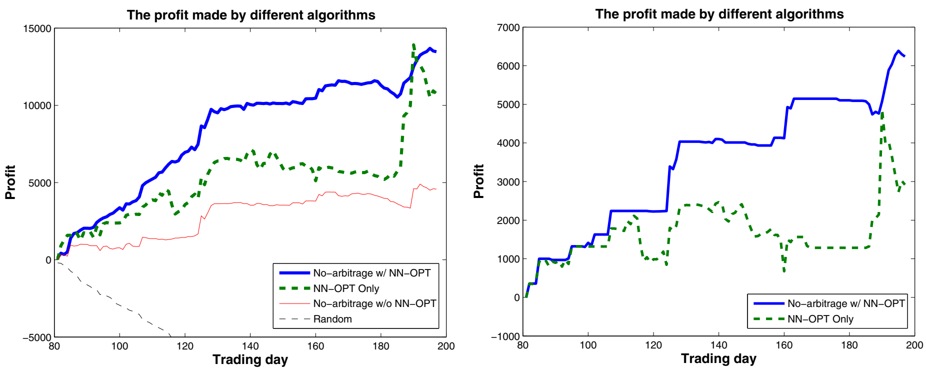


Fig. 6. Comparison of the trading results of stock IBM using different algorithms

6 Conclusions

Our results show that the right constraint (no-arbitrage) for option pricing can enable one to potentially learn the Martingale measure. By using a no-arbitrage condition, we can narrow the range of possible Martingale measures for the learning - the no-arbitrage constraint regularizes the learning in the right direction to yield a better learning outcome. Another benefit of our framework is that one can learn the Martingale measure from *all* data on all derivatives of the same underlying instrument simultaneously. The derivatives of the same underlying instrument should have correlations which our framework can utilize to yield better performance. Our future work includes using a moving training window to increase the performance of predicting the option prices, as we observe a degradation further from the training window. We are also expanding the framework to include derivatives from various different financial markets such as Futures, Commodities and many others.

References

1. Black, F., Scholes, M.S.: The pricing of options and corporate liabilities. Journal of Political Economy **3** (1973) 637–654
2. Magdon-Ismail, M.: The Equivalent Martingale Measure: An Introduction to Pricing Using Expectations. IEEE Transactions on Neural Netork **12**(4) (2001) 684–693
3. Cox, J.C., Ross, S.A.: The valuation of options for alternative stochastic processes. Journal of Financial Economics (1976) 145–166
4. Harrison, J.M., Pliska, S.R.: A stochastic calculus model of continuous trading: Complete markets. Stochastic Processes and their Applications (1983) 313–316
5. Musiela, M., Rutkowski, M.: Martingale Methods in Financial Modeling (Applications of Mathematics, 36). New York: Springer-Verlag (1977)
6. Paul R. Lajbcygier, J.T.C.: Improve option pricing using artificial neural networks and bootstrap methods. International Journal of Neural System **8**(4) (1997) 457–471
7. Amari SI, Xu L, C.L.: Option pricing with neural networks. In Progress in Neural Information Processing **2** (1996) 760–765
8. Moody, J. Saffell, M.: Learning to trade via direct reinforcement. IEEE Transactions on Neural Networks **12**(4) (2001) 875–889
9. Cox, J.C., Ross, S.A., Rubinstein, M.: Option pricing: A simplified approach. Journal of Financial Economics (1979) 229–263
10. Ross, S.M.: An Elementary Introduction to Mathematical Finance. Second edn. Cambridge University Press (2003)
11. Baxter, M., Prnnie, A.: Financial Calculus: An Introduction to Derivative Pricing. Cambridge University Press (1996)
12. Bollerslev, T.: Generalized autoregressive conditional heteroskedasticity. Journal of Econometrics **31** (1986) 307–327
13. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
14. Haykin, S.: Neural Networks: A Comprehensive Foundation. 2nd edn. Prentice Hall (1998)
15. Harrison, J.M., Pliska, S.R.: Martingales and stochastic integrals in the theory of continuous trading. Stochastic Processes and their Applications **11** (1981) 215–260

A Brain-Inspired Cerebellar Associative Memory Approach to Option Pricing and Arbitrage Trading

S.D. Teddy[1], E.M.-K. Lai[2], and C. Quek[3]

Centre for Computational Intelligence, School of Computer Engineering,
Nanyang Technological University, Nanyang Avenue, Singapore 639798.
[1]sdt@pmail.ntu.edu.sg, {[2]asmklai,[3]ashcquek}@ntu.edu.sg

Abstract. *Option pricing* is a process to obtain the theoretical fair value of an option based on the factors affecting its price. Currently, the nonparametric and computational methods of option valuation are able to construct a model of the pricing formula from historical data. However, these models are generally based on a global learning paradigm, which may not be able to efficiently and accurately capture the dynamics and time-varying characteristics of the option data. This paper proposes a novel brain-inspired cerebellar associative memory model for pricing American-style option on currency futures. The proposed model, called PSECMAC, constitute a local learning model that is inspired by the neurophysiological aspects of the human cerebellum. The PSECMAC-based option pricing model is subsequently applied in a mis-priced option arbitrage trading system. Simulation results show a return on investment as high as 23.1% for a relatively risk-free investment.

1 Introduction

Options, as a derivative security, provide a means to manage financial risks. They are playing an increasingly important role in modern financial markets [1]. The buyer of an option enters into a contract with the right, but not the obligation, to purchase or sell an underlying asset at a later date at a price agreed upon today. The price of an option is determined by a set of pricing factors such as time to expiry and the intrinsic value of the option. A vital aspect of option trading is to arrive at the theoretical fair value of an option. This process is called *option pricing*. The conventional approach to option pricing is to construct parametric models that are based on the assumptions of continuous-time finance theory [2]. The pioneering models are the *Black-Scholes formula* [3] and the *Binomial Pricing Model* [4]. However, these models presumed complex and rigid statistical formulations from which the prices are deduced [5].

Nonparametric methods of option pricing based on neural networks [6,7,8,9], genetic algorithms [10] and kernel regression [11], on the other hand, are model-free approaches. The pricing model, which is usually represented as a nonlinear functional mapping between the input factors and the theoretical option price, is derived from vast quantities of historical data. However, these methods involve heuristics and therefore suffer from poor interpretability. More recently, neuro-fuzzy approaches [12] are introduced to overcome this problem. With these techniques, a set of comprehensible semantic rules can be extracted from historical trading data for rational pricing of the options.

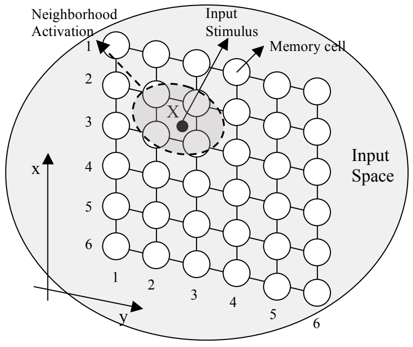
Currently, nonparametric option pricing methods are generally based on a global learning paradigm, in which the system attempts to use a single formulated model to generalize or fit the behaviors/characteristics of the entire set of historical pricing data. Some researchers have argued that it is difficult, if not impossible, to obtain a general and accurate global learning model [13]. Moreover, a financial market generally has a dynamic nature and thus is characterized by time-varying trading/pricing patterns. Historical option pricing data may contain contradicting time-varying characteristics that make it hard for a global learning model to accurately approximate the underlying pricing function. In contrast, a local learning paradigm focuses on capturing only useful local information from the observed data [14]. Instead of having a single formulated model, a local learning system can be considered as a collection of locally-active models, where each sub-model is learning from different subset of the training data.

In option trading, the prices of the options are determined by a set of pricing factors, such as time to expiry and the intrinsic values of the options. The complex relationship between the valuation of an option and its influencing factors may be modeled as combinatorial associations to be extracted from the historical pricing data. This motivates the use of a local associative model as a nonparametric computational method to option pricing. In this paper, a novel brain-inspired cerebellar associative memory approach to the pricing of American-style option on currency futures of British Pound versus US dollar is investigated. The cerebellar associative memory model, referred to as the Pseudo Self Evolving Cerebellar Model Arithmetic Computer (PSECMAC), constitutes a local learning model to approximate the associative characteristics between the option price and its influencing factors. The structure of the PSECMAC network is inspired by the neurophysiological properties of the human cerebellum [15], and emulates the information processing and knowledge acquisition of the cerebellar memory.

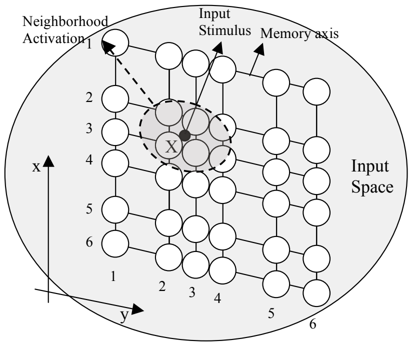
This paper is organized as follows. Section 2 briefly describe the architecture of the PSECMAC network and highlights the cerebellar-inspired memory formation and knowledge acquisition process of the network. Section 3 presents an overview of the proposed cerebellar associative memory based option pricing model and defines the selected input factors considered to have an impact on the pricing of American-style currency futures options. In Section 4, the autonomous option trading system that employs the proposed option pricing model is introduced and evaluated using real-life British Pound versus US dollar futures options trading data. Section 5 concludes this paper.

2 The PSECMAC Network

The cerebellum constitutes a part of the human brain that is important for motor control and a number of cognitive functions [16], including motor learning and memory. The human cerebellum is postulated to function as a movement calibrator [17], which is involved in the detection of movement error and the subsequent coordination of the appropriate skeletal responses to reduce the error. The human cerebellum functions by performing *associative mappings* between the input sensory information and the cerebellar output required for the production of temporal-dependent precise behaviors [15].



(a) CMAC



(b) PSECMAC

Fig. 1. Comparison of CMAC and PSECMAC memory quantization for 2D input problem

The human cerebellum has been classically modelled by the Cerebellar Model Articulation Controller (CMAC) [18]. As a computational model of the human cerebellum, CMAC manifests as an associative memory network [19], where the memory cells are uniformly quantized to cover the entire input space. The CMAC network operation is characterized by the table lookup access of its memory cells. This allows for localized generalization and rapid algorithmic computation, and subsequently motivates the prevalent use of CMAC for control applications [20,21].

This paper proposes the use of a brain-inspired cerebellar-based learning memory model named Pseudo Self-Evolving Cerebellar Model Arithmetic Computer (PSECMAC) as a generic functional model of the human cerebellum for solving approximation, modeling, control and classification problems. This architecture differs from the CMAC network in *two* aspects. Firstly, the PSECMAC network employs *one* layer of network cells, but maintained the computational principles of the layered-based CMAC network by adopting a neighborhood activation of its computing cells to facilitate: (1) smoothing of the computed output; (2) distributed learning paradigm; and (3) activation of highly correlated computing cells in the input space. Secondly, instead of uniform partitioning of the memory cells, the PSECMAC network employs the PSEC clustering technique [22] to form an experience-driven adaptive memory quantization mechanism of its network cells. Figure 1 illustrates this fundamental architectural distinction.

The adaptive quantization process of the PSECMAC network is performed in per dimension basis. The non-uniform quantization of the PSECMAC memory structure is inspired by the neurophysiological properties of the brain development, where the precise wiring in the adult brain is a result of experience-dependent refinement of initial architecture through repeated exposures to external stimuli. This experience-dependent plasticity is also observed in the human cerebellum [23], and is incorporated to the PSECMAC network through the PSEC clustering algorithm. Each training data point is a learning episode to the network. In each input dimension, the PSEC clustering algorithm is used to compute clusters of data density, and the memory axes in each dimension are allocated based on the observed density profile of the training data. Thus, more memory cells are allocated to the densely populated regions of the input space. The details on the adaptive quantization algorithm is reported in [24].

The PSECMAC network employs a *Weighted Gaussian Neighborhood Output* or WGNO computational process, where a set of neighborhood-bounded computing cells is activated to derive an output response to the input stimulus. For each input stimulus $\mathbf{X}$, the computed output is derived as follows:

Step 1: Determine the region of activation
Each input stimulus $\mathbf{X}$ activates a neighborhood of PSECMAC computing cells. The neighborhood size is governed by the neighborhood constant parameter N, and the activated neighborhood is centered at the input stimulus (see Fig 1(b)).

Step 2: Compute the Gaussian weighting factors
Each activated cell has a varied degree of activation that is inversely proportional to its distance from the input stimulus. These degrees of activation functioned as weighting factors to the memory contents of the active cells.

Step 3: Retrieve the PSECMAC output
The output is the weighted sum of the memory contents of the active cells.

Following this, the PSECMAC network adopts a modified *Widrow-Hoff learning rule* [25] to implement a *Weighted Gaussian Neighborhood Update* (WGNU) learning process. The network update process is briefly described as follows:

Step 1: Computation of the network output
The output of the network corresponding to the input stimulus $\mathbf{X}$ is computed based on the WGNO process.

Step 2: Computation of learning error
The learning error is defined as the difference between the expected output and the current output of the network.

Step 3: Update of active cells
The learning error is subsequently distributed to all of the activated cells based on their respective weighting factors.

3 A PSECMAC Based Option Pricing Model

The PSECMAC network is used to construct a pricing model to predict the correct valuations for American call options on the British pound (GBP) and US dollar (USD) exchange rate futures contract. In this study, the option pricing formula is represented as a function of the following inputs: S_0, X, T, and σ_{30}; where S_0 is the current GBP vs. USD exchange rate futures value; X is the strike price of the option on the GBP vs. USD exchange rate futures; T is time to maturity of the option in years; and σ_{30} is the historical price volatility for the last 30 trading days. We introduce the notion of *moneyness* (or intrinsic value) of the futures option, which is computed as the difference between the current futures value S_0 and the options strike price X (i.e. $S_0 - X$). Thus, the pricing function f to be approximated by the PSECMAC network is:

$$C_0 = f(S_0 - X, T, \sigma_{30}) \tag{1}$$

where C_0 is current option price; and $(S_0 - X)$ reflects the moneyness of the options.

Table 1. Simulation set-ups based on permutations of the three sub-groups A, B and C to define the training and testing sets of the proposed PSECMAC option pricing model

Configuration	Simulation	Training set	Testing set
$1/3$ training and $2/3$ testing	I	Sub-group A	Sub-groups B and C
	II	Sub-group B	Sub-groups A and C
	III	Sub-group C	Sub-groups A and B
$2/3$ training and $1/3$ testing	IV	Sub-groups A and B	Sub-group C
	V	Sub-groups B and C	Sub-group A
	VI	Sub-groups A and C	Sub-group B

Table 2. Performances of the proposed PSECMAC option pricing model

		Recall		Generalization	
Configuration	Simulation	RMSE	Correlation	RMSE	Correlation
$1/3$ training and $2/3$ testing	I	0.1299	0.9956	0.2386	0.9858
	II	0.1376	0.9954	0.2727	0.9816
	III	0.1178	0.9964	0.2638	0.9847
$2/3$ training and $1/3$ testing	IV	0.1382	0.9952	0.2103	0.9889
	V	0.1404	0.9949	0.2210	0.9885
	VI	0.1353	0.9954	0.2007	0.9902
	Average	0.1332	0.9955	0.2345	0.9866

The data used in this study consists of the daily closing quotes of the GBP versus USD currency futures and the daily closing bid and ask prices of American style options on such futures in the Chicago Mercantile Exchange (CME) [26] during the period of Sept 2002 to Aug 2003. In total, 792 data samples are available in the selected futures option data set, which contains the historic pricing data for five different strike prices: \$158, \$160, \$162, \$166 and \$168, with 159, 158, 173, 137 and 165 data samples respectively. The presentation order of the 792 data samples is randomized and subsequently partitioned into three evenly distributed sub-groups denoted as A, B and C, each containing 264 data tuples. A total of six different cross-validation sets are constructed based on the permutations of the sub-groups, as outlined in Table 1.

A PSECMAC network with a memory size of 12 cells per dimension is constructed. A neighborhood size of 0.2 and Gaussian weighting factor of 0.5 is employed. Table 2 tabulates the *recall* (training) and *generalization* (testing) performances of the PSECMAC option pricing model under the various cross-validation sets. *RMSE* denotes the root-mean-square-error between the predicted and desired option prices, and *Correlation* is the Pearson correlation coefficient, a statistical measure reflecting the goodness-of-fit between the predicted and desired pricing functions. The performances of the PSECMAC option pricing model are generally good, with an average RMSE of around 0.13 and 0.23 for the recall and generalization process respectively. An average correlation of 0.98 is achieved in the generalization process, indicating a very low performance degradation as the evaluation emphasis is shifted from the recall to the generalization capability of the system. From Table 2, one can also observe that a larger training data set improves the generalization performance of the option pricing model.

Table 3. Benchmarking results for various global and local option pricing model

		Recall		Generalization	
System	**Type**	**RMSE**	**Correlation**	**RMSE**	**Correlation**
MLP(3-8-1)	global	0.0384	0.9997	0.0982	0.9963
PSECMAC	local	0.1332	0.9955	0.2345	0.9866
CMAC	local	0.0605	0.9990	0.2738	0.9813
GenSoFNN-TVR	global	0.1808	0.9946	0.2576	0.9873

As benchmarks, the set of option pricing simulations is repeated using two global nonparametric approximators: the multi-layered perceptron (MLP) and the GenSoFNN-TVR [12] networks; as well as the CMAC network, which is a well-established local learning model. Table 3 summarizes the benchmarking results. The network structure of the MLP, which consists of three input, eight hidden and one output nodes respectively, has been empirically determined, while the GenSoFNN-TVR network is a self evolving structure. Also, for a fair comparison, the size of the CMAC network is set as 12 cells per dimension. From Table 3, one can observe that the MLP network possesses the most accurate pricing decisions as compared to the other benchmarked systems. However, it is a black-box model as its complex synaptic weight structure is hardly human interpretable. There is no mechanism to explain the logical steps that the MLP network employs for its pricing decisions. Moreover, the empirical determination of the network structure often renders the MLP network hard to use. In contrast, the global learning-based GenSoFNN-TVR network offers interpretable semantic rules, at the expense of lower pricing accuracy. The performances of both the CMAC and PSECMAC local models, on the other hand, exceed those of the global GenSoFNN-TVR model. Furthermore, the pricing decisions of the proposed PSECMAC option pricing model outperform those of the CMAC model. The associative structure of the PSECMAC model also enables discrete pricing rules to be extracted. For example, ”IF the *time-to-maturity* is between 0 - 0.04 years and the *volatility* is between 5.08 - 5.28 and the *moneyness* is between \$5.03 - \$7.98 THEN the Option-Price (on average) is \$9.4” is a representative discrete rule extracted from the PSECMAC model that expresses the knowledge induced from the training data.

4 Cerebellar Associative Memory Approach to Arbitrage Trading

This section introduces a mis-priced option *arbitrage* trading system, where the PSECMAC option pricing model is employed to detect any misalignments between the market spot value and the theoretical valuation of an option. When such mis-pricing occur, potential arbitrage trading opportunities on that option are created and investors can exploit these opportunities to derive risk-free profits.

An arbitrage opportunity arises when the *Law of One Price* [1] is violated, making it possible for an investor to make a *risk-less* profit. In this paper, an arbitrage trading strategy known as the *Delta Hedge Trading Strategy* (DHTS) [1] is employed in the proposed PSECMAC-based trading system. In the DHTS, a delta hedge ratio h is computed to determine the quantity of the underlying asset (e.g. stock) required to cover the risk of taking a naked position on the call option. Hence, the selling of one call option is hedged by the buying of h quantity of the underlying asset and vice versa. The hedge ratio h is computed as:

$$h_t = \frac{\varDelta C}{\varDelta S} = \frac{(\hat{C}_{u,t+1} - \hat{C}_{d,t+1})}{(S_{u,t+1} - S_{d,t+1})} \in [0,1] \tag{2}$$

where h_t is the hedge ratio at current time t (i.e. this trading opportunity) employed to build up a risk-free portfolio with proper ratio of call option and the underlying asset; $S_{u,t+1}$ is the price of the underlying asset at time $t+1$ (i.e. the next trading opportunity) if the price goes up; $S_{d,t+1}$ is the price of the underlying asset at time $t+1$ (i.e. the next trading opportunity) if the price goes down; $\varDelta S$ is the change in value of the underlying asset due to the projected change in price S_t at time $t+1$; $\varDelta C$ is the change in value of the call option due to the projected change in price of the underlying asset at time $t+1$; $\hat{C}_{u,t+1}$ is the predicted price of the call option if the value of the underlying asset is $S_{u,t+1}$ at time $t+1$; and $\hat{C}_{d,t+1}$ is the predicted price of the call option if the value of the underlying asset is $S_{d,t+1}$ at time $t+1$.

However in this study, for simplicity, the price of the underlying asset is assumed to either go up by 0.5 unit price or go down by 0.5 unit price (i.e. $S_{u,t+1} = S_t + 0.5$ and $S_{d,t+1} = S_t - 0.5$) such that the variable $\varDelta S$ in equation (2) evaluates to unity. That is, there is only a unit change in the price of the underlying asset from time t to time $t+1$. Hence, equation (2) can be reduced to:

$$h_t = \frac{(\hat{C}_{u,t+1} - \hat{C}_{d,t+1})}{(S_{u,t+1} - S_{d,t+1})} = \frac{(\hat{C}_{u,t+1} - \hat{C}_{d,t+1})}{(S_t + 0.5 - (S_t - 0.5))} = (\hat{C}_{u,t+1} - \hat{C}_{d,t+1}) \tag{3}$$

Thus, the hedge ratio of the portfolio at current time t is computed as the difference in the predicted prices of the call option at time $t+1$.

4.1 Trading Strategy

Based on the DHTS discussed in the last section, the PSECMAC-based option trading system is implemented. The general framework of the trading system proposed in this paper is a modified version of the generic trading decision model found in [27], and is illustrated in Figure 2.

The format of the training set is as described in Section 3. Historic data of options with strike prices of \$158, \$160, \$162, \$166 and \$168 respectively from Sept 2002 to Feb 2003 is used to train the PSECMAC-based option pricing model. The test set contains out-of-sample data consisting of the intra-day bid and ask prices of options with strike prices of \$158, \$159, \$160, \$164 and \$170 respectively from Mar 2003 to Aug 2003. The trading algorithm is summarized as follows:

1. The proposed trading system takes in the theoretical option value C_t computed by the PSECMAC-based option pricing model and subsequently compares it to the spot bid-ask prices of the option.
2. If the predicted theoretical option value C_t falls out of the bid-ask spread range, the trading system assumes a mis-priced arbitrage opportunity as being detected.

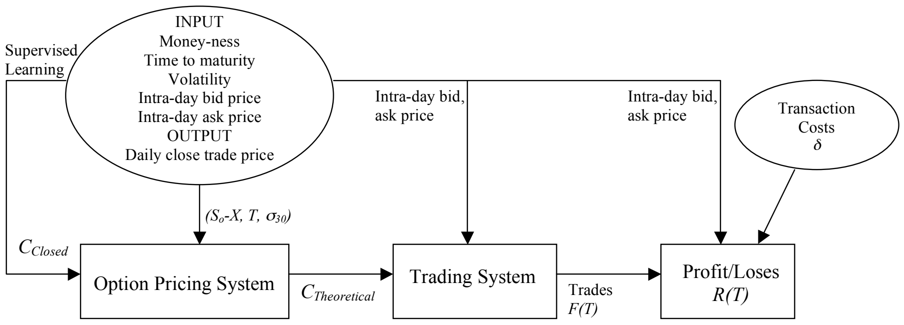


Fig. 2. General framework of the proposed mis-priced option arbitrage system

3. The trading system would take up trading positions according to the following trading strategy:
   (a) Evaluate if the call option is overpriced or under-priced using equation (4).

$$\text{Call option} = \begin{cases} \text{Overpriced,} & \text{if } C_t < \text{Option bid-price at time } t \\ \text{Underpriced,} & \text{if } C_t > \text{Option ask-price at time } t \end{cases} \quad (4)$$

   (b) If the call option is overpriced, short sell the call option and hedge the risk by buying in h_t quantity of the underlying asset, i.e. the GBP vs. USD currency futures. The hedge ratio h_t is computed using equation (3). Else, if the call option is under-priced, buy in the call option and short sell h_t quantity of the GBP vs. USD futures to hedge the risk.
4. If the trading system already possessed a portfolio (i.e. has either a long or short open position on the call option with the appropriate ratio of hedged futures), it would continuously check whether the mis-priced option has pulled back into the option bid-ask spread range. If it is the case, the trading system closes all the outstanding position immediately; else, it continues to hedge the portfolio by computing a new hedge ratio h_{t+1} and adjusting the portfolio composition.

4.2 Results and Analysis

The proposed PSECMAC-based trading system is evaluated by observing its arbitrage performances using real-life GBP vs. USD currency futures options with various strike prices. To simplify the simulation setup, transaction costs are omitted here. The results are tabulated in Table 4. The "total capital outlay" column denotes the overall amount of investment made on the sales and purchases of the respective options and futures in the hedging exercises, while "return on investment" (ROI) denotes the profit earned from the trading endeavors. As shown in Table 4, the PSECMAC-based trading system has demonstrated fairly high returns for investment, with ROI of 23.1% for the option with a strike price of $\$164$ and an average ROI of around 12.8% across all the five options. Such an average rate of return is considered encouraging given the risk-free nature of the investment portfolio constructed and when compared against other risk-free investments during the same period. For example, according to the Federal Reserve

Table 4. Arbitrage performances of the proposed PSECMAC-based option trading system. (Note: UO is option under-priced arbitrage opportunity; OO is option over-priced arbitrage opportunity; and ROI denotes the return on investment).

Option Strike Price X ($)	**Sim Period** (days)	**Num of UO transaction**	**Num of OO transaction**	**Total Capital Outlay** ($)	**Absolute ROI** ($)	**Percentage ROI** (%)
158	156	26	19	143300	7964.80	5.56
159	61	7	15	50940	4228.60	8.3
160	65	0	17	30820	1809.30	5.87
164	97	17	10	20560	4759.60	23.15
170	94	10	12	5560	1175.20	21.14
					Average ROI (%)	12.80

Board, the 3-months compounding interest rate of US Treasury Bill is 0.93% on 30th September 2003, while the 3-month fixed deposit interest rate in Singapore is only 0.25% on 3rd October 2003 according to data provided by the Development Bank of Singapore (DBS).

5 Conclusions

This paper proposes the use of a brain-inspired cerebellar associative learning memory structure named PSECMAC to perform nonparametric option pricing of American style call options on the British pound (GBP) versus US dollar (USD) currency futures. The PSECMAC-based option pricing system constitutes a local learning approach to the approximation of the associative characteristics between the option price and its influencing factors. Evaluation results have demonstrated that the modeling capabilities of the proposed pricing system exceed those of the global learning-based GenSoFNN-TVR model, as well as the well established local learning-based CMAC network. The associative structure of the PSECMAC model also enables discrete pricing rules to be extracted from the pricing system. Subsequently, the PSECMAC-based option pricing model is employed in a mis-priced option arbitrage trading system. Simulation results on various options with different strike prices demonstrated that such a mis-priced arbitrage trading system is able to construct risk-free investment portfolios with a satisfactory rate of return on investment. Future studies will attempt to incorporate other external factors such as transaction costs, as well as to extend the PSECMAC-based option pricing system to a fuzzy associative system to enable the extraction of fuzzy semantic rules.

References

1. Chance, D.M.: An Introduction to Derivatives & Risk Management. 6 edn. Thomson (2004)
2. Nielsen, L.T.: Pricing and Hedging of Derivative Securities – Textbook in continuous-time finance theory. Oxford University Press (1999)
3. Black, F., Scholes, N.: The pricing of options and corporate liabilities. Journal of Political Economy **81** (1973) 637–659
4. Jr., R.J.R., Bartter, B.J.: Two-state option pricing. Journal of Finance **34** (1979) 1093–1110

5. Radzikowski, P.: Non-parametric methods of option pricing. Proc. of Informs-Korms (Seoul 2000 conference) (2000) 474–480
6. Amilon, H.: A neural network versus black-scholes: A comparison of pricing and hedging performances. Scandinavian Working Papers in Economics, Lund University series, Department of economics, Lund, Sweden (2001)
7. Anders, U., Korn, O., Schmitt, C.: Improving the pricing of options - a neural network approach. Journal of Forecasting **17**(5–6) (1998) 369–388
8. Qi, M., Maddala, G.S.: Option-pricing using artificial neural networks: the case of s&p500 index call options. Neural Networks in Financial Engineering (1995) 78–92
9. Hutchinson, J., Lo, A., Poggio, T.: A nonparametric approach to pricing and hedging derivative securities via learning networks. Journal of Finance **49** (1994) 851–889
10. Keber, C.: Option pricing with the genetic programming approach. Journal of Computational Intelligence in Finance **7**(6) (1999) 26–36
11. Ait-Sahalia, Y., Lo, A.W.: Nonparametric estimation of state-price densities implicit in financial asset price. LFE-1024-95, MIT-Sloan School of Management (1995)
12. Tung, W.L., Quek, C.: GenSo-OPATS: A brain-inspired dynamically evolving option pricing model and arbitrage trading system. Proc. IEEE CEC 2005, Edinburgh, Scotland **3** (2005) 2429–2436
13. Huang, K., Yang, H., King, I., Lyu, M.: Local learning vs. global learning: An introduction to maxi-min margin machine. In: Support Vector Machines: Theory and Applications, Wang L. (Eds.). Volume 177., Springer (2005) 113–132
14. Bottou, L., Vapnik, V.: Local learning algorithms. Neural Computation (4) (1992) 888–900
15. Kandel, E.R., Schwartz, J.H., Jessell, T.M.: Principles of Neural Science. 4 edn. McGraw-Hill (2000)
16. Middleton, F.A., Strick, P.L.: The cerebellum: An overview. Trends in Cognitive Sciences **27**(9) (1998) 305–306
17. Albus, J.S.: Marr and Albus theories of the cerebellum two early models of associative memory. Proc. IEEE Compcon (1989)
18. Albus, J.S.: A new approach to manipulator control: The Cerebellar Model Articulation Controller (CMAC). J. Dyn. Syst. Meas. Control, Trans. ASME (1975) 220–227
19. Albus, J.S.: Data storage in Cerebellar Model Articullation Controller (CMAC). J. Dyn. Syst. Meas. Control, Trans. ASME (1975) 228–233
20. Yamamoto, T., Kaneda, M.: Intelligent controller using CMACs with self-organized structure and its application for a process system. IEICE Trans. Fundamentals **82**(5) (1999) 856–860
21. Commuri, S., Jagannathan, S., Lewis, F.L.: CMAC neural network control of robot manipulators. J. Robot Syst. **14**(6) (1997) 465–482
22. Ang, K., Quek, C.: Stock trading using PSEC and RSPOP: A novel evolving rough set-based neuro-fuzzy approach. IEEE Congress on Evolutionary Computation (2005)
23. Federmeier, K.D., Kleim, J.A., Greenough, W.T.: Learning-induces multiple synapse formation in rat cerebellar cortex. Neuroscience Letters **332** (2002) 180–184
24. Teddy, S.D., Quek, C., Lai, E.M.K.: Psecmac: A brain-inspired multi resolution cerebellar learning memory model. Neural Computation, *under review* (2006)
25. Widrow, B., Stearns, S.D.: Adaptive Signal Processing. Prentice-Hall (1985)
26. Chicago Mercantile Exchange, U. ([Online] http://www.cme.com)
27. Gencay, R.: The predictability of security returns with simple trading rules. Journal of Empirical Finance **5**(4) (1998) 347–359

A Reliability-Based RBF Network Ensemble Model for Foreign Exchange Rates Predication

Lean Yu[1,2], Wei Huang[3], Kin Keung Lai[2,4], and Shouyang Wang[1,4]

[1] Institute of Systems Science, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100080, China
{yulean, sywang, whuang}@amss.ac.cn
[2] Department of Management Sciences, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong
{msyulean, mskklai}@cityu.edu.hk
[3] School of Management, Huazhong University of Science and Technology, 1037 Luoyu Road, Wuhan 430074, China
[4] College of Business Administration, Hunan University, Changsha 410082, China

Abstract. In this study, a reliability-based RBF neural network ensemble forecasting model is proposed to overcome the shortcomings of the existing neural ensemble methods and ameliorate forecasting performance. In this model, the ensemble weights are determined by the reliability measure of RBF network output. For testing purposes, we compare the new ensemble model's performance with some existing network ensemble approaches in terms of three exchange rates series. Experimental results reveal that the prediction using the proposed approach is consistently better than those obtained using the other methods presented in this study in terms of the same measurements.

1 Introduction

Combining the outputs of several neural networks into an aggregate output often gives improved accuracy over any individual output [1]. Usually, the output of an ensemble network is a weighted average of the outputs of each network, with the ensemble weights determined as a function of the relative error of each network determined in training [1]. Accordingly, the resulting ensemble network often outperforms the individual networks. Therefore there is a growing research stream [1-5] into ensemble methods. For example, performance improvement can result from training the individual networks to be decorrelated with each other [2] with respect to their errors. Usually, the weights of the existing ensemble methods are determined by error or error variance [1, 9, 11] without considering the reliability of neural network output. However, error-dependent neural network ensemble methods do not often obtain consistent good performances in forecasting. In some experiments, such as [1, 6], error-dependent ensemble network sometimes performs worse than the individual neural networks.

Under such backgrounds, we propose a novel neural network ensemble forecasting approach that differs in that the ensemble weights are determined the reliability of neural network output. In this study, the reliability is used as a confidence measure of network output. That is, the ensemble weights are proportional to the reliability measure of the respective outputs. In addition, the neural network type used in this study is radial basis function (RBF) neural network because it can generate a confidence measure due to its specificity.

The motivation of this study is to formulate a reliability-based RBF network ensemble forecasting model for exchange rates prediction and compare its performance with other existing network ensemble forecasting approaches. The rest of the study is organized as follows. The next section presents some previous work done in the ensemble methods in terms of forecasting. Section 3 describes the reliability-based RBF neural network ensemble method in detail. To verify the effectiveness of the proposed method, several experiments are performed in Section 4. Section 5 concludes.

2 Previous Studies

This section presents some earlier work done in ensemble methods in terms of prediction. Suppose there are n individual neural networks trained on a set $\boldsymbol{D} = \{x_i, y_i\}$ ($i = 1, 2, \dots, n$).

2.1 The Brief Description of Neural Ensemble Predictor

According to the previous assumption, there are n individual neural network outputs, i.e., $\hat{f}_1(x), \hat{f}_2(x), \cdots, \hat{f}_n(x)$. The main question of neural ensemble predictor is how to combine (ensemble) these different outputs into an aggregate output $\hat{f}(x)$, which is assumed to be a more accurate output. The general form of the model for such an ensemble predictor can be defined as

$$\hat{f}(x) = \sum_{i=1}^{n} w_i \hat{f}_i(x) \tag{1}$$

where w_i denotes the assigned weight of $\hat{f}_i(x)$, and in general the sum of the weight is equal to one, i.e., $\sum_i w_i = 1$. In neural network ensemble forecasting, how to determine ensemble weights is a focus issue. As earlier mentioned, there are a variety of methods for determining ensemble weights in the past work, which is presented in the following. Typically, there are four ensemble methods, which are described below.

2.2 The Simple Averaging Ensemble Method (SAE)

A simple method for combining network outputs is to simply average individual network predictors together, so this approach is called as the "simple averaging ensemble (SAE) method". The SAE is defined as

$$\hat{f}_{SAE}(x) = \sum_{i=1}^{n} w_i \hat{f}_i(x) = \frac{1}{n}\sum_{i=1}^{n} \hat{f}_i(x) \tag{2}$$

where the weight of each individual network output $w_i = 1/n$.

Simple averaging ensemble method is one of the most frequently used combination approaches is easy to understand and implement [4]. Some experiments [8-9] have shown that this approach by itself can lead to improved performance [1] and it is an effective approach to improve neural network performance. Specially, it is more useful when the local minima of ensemble members are different, i.e., when the local minima of ensemble networks are different. Different local minima mean that ensemble members are diverse. Thus averaging can reduce the ensemble variance.

However, this approach treats each member equally, i.e., it does not stress ensemble members that can make more contribution to the final generalization. That is, it does not take into account the fact that some networks may be more accuracy than others. If the variances of ensemble networks are very different, we do not expect to obtain a better result using simple averaging [10]. In addition, since the weights in the combination are so unstable, a simple average may not the best choice in practice [6].

2.3 The Simple MSE Ensemble Method (SMSE)

The simple MSE method estimates the linear weight parameter wi in Equation (1) by minimizing the mean squared error (MSE) [11], that is, for *i* = 1, 2, …, *n*,

$$w_{opt,i} = \arg\min_{w_i}\left\{\sum_{j=1}^{m}\left(w_i^T \hat{f}_i(x_j) - d_{ji}(x_j)\right)^2\right\} = \left(\sum_{j=1}^{m}\hat{f}_i(x_j) f_i^T(x_j)\right)^{-1}\sum_{j=1}^{m} d_{ji}(x)\hat{f}_i(x_j) \tag{3}$$

where $d(x)$ is the expected value.

The simple MSE solution seems to be reasonable, but, as Breiman [9] has pointed out, this approach has two serious problems in practice: a) the data are used both in the training of each predictor and in the estimation of w_i, and b) individual predictors are often strongly correlated since they try to predict the same task. Due to these problems, this approach's generalization ability will be poor.

2.4 The Stacked Regression Ensemble Method (SRE)

The stacked regression method was proposed by Breiman [9] in order to solve the problems associated with the previous MSE method. Thus, the stacked regression method is also called the modified MSE method. This approach utilizes cross-validation data to modify the simple MSE solution, i.e.,

$$w_{opt,i} = \arg\min_{w_i}\left\{\sum_{j=1}^{m}\left(w_i^T g_i(x_j) - d_{ji}(x_j)\right)^2\right\},\quad i = 1,2,\ldots,n \tag{4}$$

where $g_i(x_j) = \left(\hat{f}_i^{(1)}(x_j; D_{cv}), \cdots, \hat{f}_i^{(m)}(x_j; D_{cv})\right)^T \in \Re^M$ is a cross-validated version $\hat{f}_i(x_j) \in \Re^M$ and D_{cv} is the cross-validated data.

Although this approach overcomes the limitations of the simple MSE method, the solution is based on the assumption that the error distribution of each validated set is normal [10]. In practice, however, this normal assumption does not always hold and thus this approach does not lead to the optimal solution in the Bayes sense [10].

2.5 The Variance-Based Weighting Ensemble Method (VWE)

The variance-based weighting ensemble approach estimates the weight parameter w_i by minimizing error variance σ_i^2 [1]; all predictors are error-independent networks, i.e.,

$$w_{opt,i} = \arg\min_{w_i} \left\{ \sum_{i=1}^{n} \left(w_i \sigma_i^2 \right) \right)^2 \right\}, \quad (i = 1, 2, \cdots, n) \tag{5}$$

under the constraints $\sum_{i=1}^{n} w_i = 1$ and $w_i \geq 0$. Using the Lagrange multiplier, the optimal weights are:

$$w_{opt,i} = \frac{\left(\sigma_i^2\right)^{-1}}{\sum_{j=1}^{n} \left(\sigma_j^2\right)^{-1}}, \quad (i = 1, 2, \cdots, n) \tag{6}$$

The variance-based weighting method is based on the assumption of error independence. Moreover, as earlier mentioned, individual predictors are often strongly correlated for the same task. This indicates that this approach has serious drawbacks for minimizing error-variance when neural predictors with strong correlation are included within the combinatorial members.

According to the previous descriptions and literature review, the above four neural network ensemble methods have widely been used, but we can also find that the weights of these ensemble methods are determined by the error or error variance. In practice, error-dependent network ensemble model may not often obtain good forecasting performance when error is correlated each other [2]. Furthermore, the accuracy of each individual network predictor is different in different conditions because neural network learning by itself is “the state of the art”. Thus, it is necessary to measure the confidence of the forecasting results of each output before they are combined. In such situations, a novel reliability-based neural ensemble method is proposed to overcome the above problems, which is presented in the following. It is worth noting that the reliability is used to measure the confidence of neural network output and the RBF network is used to constitute a neural network ensemble.

3 The Reliability-Based RBF Network Ensemble Model

In this section, we first introduce the radial basis function (RBF) neural network briefly. Then the confidence measure — reliability is induced from the RBF neural network. Finally, based on the reliability measure, a reliability-based RBF neural network ensemble method is formulated.

3.1 Overview of RBF Neural Networks

The RBF neural network [12, 13] is generally composed of three layers: input layer, hidden layer and output layer. The input layer feeds the input data to each of the nodes of the hidden layer. The hidden layer of nodes differs greatly from other neural networks in that each node represents a data cluster which is centered at a particular point with a given radius. Each node in the hidden layer calculates the distance from the input vector to its own center. The calculated distance is transformed via some basis function and the result is output from a node. The output from the node is multiplied by a constant or weighting value and fed into the output layer. The output layer consists of only one node which acts to sum the outputs of the previous layer and to yield a final output value. A generic architecture of an RBF network with *k* input and *m* hidden nodes is illustrated in Fig. 1.

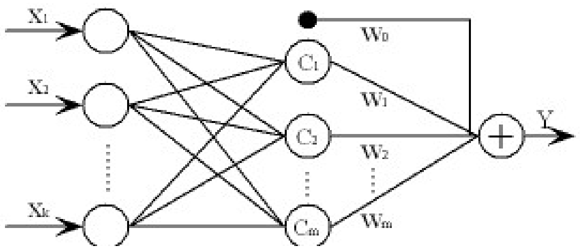


Fig. 1. The generic architecture of the RBF neural network

The computation process of the RBF neural network follows the following procedures. When the network receives a k dimensional input vector $\boldsymbol{X}$, the network computes a scalar value using the following formula:

$$Y = f(X) = w_0 + \sum_{i=1}^{m} w_i \varphi(D_i) \tag{7}$$

where w_0 is the bias, w_i is the weight parameter, m is the number of nodes in the hidden layers of the RBF neural network, and $\varphi(D_i)$ is the radial basis function. In this study, the Gaussian function is used as radial basis function, as shown below.

$$\varphi(D_i) = \exp(-D_i^2 / \sigma^2) \tag{8}$$

where σ is the radius of the cluster represented by the center node, the D_i representes the distance between the input vector $\boldsymbol{X}$ and all the data centers. It is clear that $\varphi(D_i)$ will return values between 0 and 1. Usually, the Euclidean norm is used to calculate distance, but other metrics can also be used. The Euclidean norm is calculated by

$$D_i = \sqrt{\sum_{j=1}^{k} (x_j - c_{ji})^2} \tag{9}$$

where c is a cluster center for any of the given nodes in the hidden layer.

Complex nonlinear systems such as time series data are generally difficult to model using standard linear regression [13]. Dissimilar to the regression, neural networks are nonlinear and their parameters are determined by some learning techniques and search algorithms such as error back propagation and steep gradient algorithm. The main drawback of the standard neural networks is that their parameters learning algorithm is time-consuming and have a tendency to get stuck at local minima [14]. But RBF neural networks overcome the above problems to obtain good performance since their parameters that need to be trained are the ones in the hidden layer of the network. Finding their values is the solution of a linear problem and can be obtained through interpolation [12]. Therefore, their parameters are found much faster than in other neural networks. Furthermore, the RBF network can usually reach near perfect accuracy on the training data set without trapping into local minima [13].

3.2 Confidence Measure and Reliability

Once the parameters are determined by training the data, the RBF network can be applied to perform prediction for any unknown input vectors. But the confidence of the output results is an important issue for any prediction problems. In this study, other than the basic output, the RBF network is also used to generate confidence measure to verify the reliability of network output.

Actually, using RBF neural network to generate a confidence measure is not a new idea. Lee [15] used this principle in a handwritten digit recognition program and achieved good results. Lee used as a measure of confidence the numeric difference between the two output nodes with the highest values. Leonard et al. [16] explored two methods of measuring the reliability of prediction: the maximum activation value function and Parzen estimator and concluded the Parzen method superior as it took into account the distribution of the training data. Wedding and Cios [13] also used the output of the Gaussian basis function to calculate a certainty factor value. The certainty factor was calculated from the input vector's proximity to the hidden layer's node centers rather than the data distribution of the training vectors. The reliability proposed in this study is a variation of [13].

In this study, a new confidence measure called reliability for RBF neural networks is proposed. The reliability can be generated by RBF through using the $\varphi(D_i)$ value from the basis functions in the hidden layer, as shown in Equation (8). When the output $\varphi(D_i)$ of a hidden layer node is high (near one), then this indicates that a data point lies near the center of a cluster and the data is familiar to that particular node and therefore the network is confident in the output. Conversely, if the $\varphi(D_i)$ is very small (near zero), this implies that the data point lies outside of a cluster and thus the network is not reliable in the result. Because the reliability is used to measure the overall network, then all the values from different centers should be combined into a single one. According to the positive correlation between the reliability and $\varphi(D_i)$, the reliability measure of overall network can be computed by

$$R_j = \frac{1}{m}\sum_{i=1}^{m}\varphi(D_i) \tag{10}$$

where R_j is the reliability of the individual network j and $\varphi(D_i)$ is the corresponding output value at hidden neuron i. As Equation (10) revealed, the larger the R, the larger the reliability. Then the weight of this individual network should be larger than that of others in the ensemble. That is, the large reliability should correspond to large weight in the ensemble network so as to improve the prediction accuracy. In terms of reliability measure, a reliability-based RBF network ensemble forecasting model is formulated in the following.

3.3 Reliability-Based Weighted Ensemble (RWE) Method

Instead of choosing error-dependent weights, we allow the weights to adjust to be proportional to the reliability measure of the respective network outputs. We might achieve better performance. Define the reliability-based weighted ensemble (RWE) network as:

$$\hat{f}_{RWE}(x) = \sum_{i=1}^{n} w_i \hat{f}_i(x) \tag{11}$$

where the w_i s are as:

$$w_i = \frac{R_i}{\sum_{j=1}^{n} R_j} \tag{12}$$

The w_i s sum to one, so $\hat{f}_{RWE}$ is a weighted average of the individual network outputs. The difference is that the weight vector is determined by confidence measure — reliability to try to give the best prediction under consideration, instead of choosing error-dependent weights with respect to a cross validation set. Each RBF network's contribution to the sum is proportionate to its confidence measure, i.e., reliability. For testing, several experimental examples are presented below.

4 Simulations

4.1 Data

In this study three foreign exchange series are selected for comparison purposes. The foreign exchange data used in this paper are monthly and are obtained from Pacific Exchange Rates Services (http://fx.sauder.ubc.ca/), provided by Professor Werner Antweiler, University of British Columbia, Vancouver, Canada. They consist of the US dollar against each of the three currencies — German marks (DEM), British pound (GBP) and Japanese yen (JPY) studied in this paper. We take monthly data from January 1971 to December 2000 as in-sample (training periods) data sets (360 observations including 60 samples for validations). We also take the data from January 2001 to December 2004 as out-of-sample (testing periods) data sets (48 observations), which is used to evaluate the good or bad performance of prediction based on some evaluation measurement. In order to save space, the original data are not listed here, and detailed data can be obtained from the website or from the authors. For comparison, two typical indicators, mean squared error (*MSE*) and directional statistics (D_{stat}) were used in this study. Usually, *MSE* is an ordinary level predication measurement and while D_{stat} is a directional prediction measurement, as defined in [7].

4.2 Empirical Results

In this study each of the three ensemble methods was implemented and tried on several data sets for comparison. Ten feed-forward neural networks with sigmoidal activation functions and four hidden nodes were trained for each training set, then tested as an ensemble for each method for the testing set. Each network was trained with standard back-propagation for 100 iterations with a learning rate of 0.2 using the Matlab software package, which is producted by Mathworks Laboratory Corporation. In addition, the best individual network using cross-validation (CV) (NCV for short) [3] method (i.e., select the individual network by minimizing the mean squared error on CV) is chosen as benchmark model for comparison. Accordingly, the results obtained are reported in Tables 1 and 2.

Table 1. A comparison of MSE between different methods for the three exchange rates

	DEM		GBP		JPY	
	MSE	Rank	*MSE*	Rank	*MSE*	Rank
NCV	0.0035	5	0.0019	5	0.0048	6
SAE	0.0038	6	0.0018	4	0.0043	5
SMSE	0.0031	4	0.0021	6	0.0040	4
SRE	0.0029	3	0.0015	2	0.0037	3
VWE	0.0025	2	0.0016	3	0.0031	2
RWE	0.0022	1	0.0009	1	0.0028	1

Table 2. A comparison of D_{stat} between different methods for the three exchange rates

	DEM		GBP		JPY	
	D_{stat} (%)	Rank	D_{stat} (%)	Rank	D_{stat} (%)	Rank
NCV	64.58	6	70.83	5	62.50	6
SAE	72.91	4	68.75	6	66.67	5
SMSE	66.67	5	70.83	4	75.00	4
SRE	79.17	3	83.33	3	77.08	3
VWE	81.25	2	85.41	2	81.25	2
RWE	87.50	1	89.58	1	83.33	1

Tables 1 and 2 give clear comparisons of various methods for the three currencies via *MSE* and D_{stat}. Generally speaking, these tables provide comparisons of level and direction among these different methods. Experimental results reveal that the prediction performance of the dynamically weighted ensemble forecasting model is better than those of other ensembles models.

Focusing on the *MSE* indicator, our proposed ensemble method performs the best in all the cases, followed by the VWE, SRE. Interestingly, the *MSE* of the SAE are not better than those of the best individual network model for the DEM testing case, and the *MSE* of the NCV are worse than those of the SMSE for the GBP case, implying that the SAE and SMSE does not consider the fact that some networks may be more accurate than the others.

However, the low *MSE* does not necessarily mean that there is a high hit ratio for foreign exchange movement direction prediction. Thus the D_{stat} comparison is necessary for practitioners. Focusing on D_{stat} of Table 2, we are not hard to find that the proposed dynamically weighted ensemble forecasting model outperforms the other ensemble models and the benchmark model according to the rank; furthermore, from the business practitioners' point of view, D_{stat} is more important than MSE because the former is an important decision criterion in foreign exchange trading decision. With reference to Table 2, the differences between the different models are very significant. For instance, for the JPY testing case, the D_{stat} for the best individual ANN model via NCV is only 62.50%, for the SAE method it is 66.67%, for the SMSE, it is 75.00%, and the D_{stat} for SRE is 77.08%, and for the VWE, D_{stat} is 81.25%; while for the RWE method, D_{stat} reaches 83.33%. Furthermore, like *MSE* indicator, the proposed ensemble method performs the best in all the cases, followed by VWE, SRE, and the poorest is the individual network model via NCV. The main reason is that our proposed approach can adjust its weights dynamically, giving it an advantage over other ensemble methods whose weights are fixed as a part of training.

5 Conclusions

This study proposes a reliability-based RBF neural network ensemble forecasting model to obtain accurate prediction results and improve prediction quality further. In terms of the empirical results, we can find that across different ensemble models for the test cases of three main currencies — German marks (DEM), British pound (GBP) and Japanese yen (JPY) — on the basis of different criteria, our proposed ensemble method performs the best. In the proposed reliability-based RBF network ensemble model testing cases, the *MSE* is the lowest and the D_{stat} is the highest, indicating that the proposed reliability-based RBF network ensemble model can be used as a viable alternative ensemble solution to exchange rates prediction.

Acknowledgements

This work is partially supported by National Natural Science Foundation of China, Chinese Academy of Sciences, Key Research Institute of Humanities and Social Sciences in Hubei Province-Research Center of Modern Information Management and Strategic Research Grant of City University of Hong Kong (SRG No. 7001677).

References

1. Perrone, M.P., Cooper, L.N.: When Networks Disagree: Ensemble Methods for Hybrid Neural Networks. In Mammone, R.J. (ed.): Neural Networks for Speech and Image Processing, Chapman-Hall (1993) 126-142
2. Rosen, B.E.: Ensemble Learning Using Decorrelated Neural Networks. Connection Science 8 (1996) 373-384
3. Krogh, A., Vedelsby, J.: Neural Network Ensembles, Cross Validation, and Active Learning. In Tesauro, G., Touretzky, D., Leen, T. (eds.): Advances in Neural Information Processing Systems, Cambridge, MA, MIT Press 7 (1995) 231-238
4. Bishop, C.M.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
5. Tumer, K., Ghosh, J.: Error Correlation and Error Reduction in Ensemble Classifiers. Connection Science 8 (1996) 385-404
6. Kang, B.H.: Unstable Weights in the Combination of Forecasts. Management Science 32 (1986) 683-695
7. Yu, L., Wang, S.Y., Lai, K.K.: A Novel Nonlinear Ensemble Forecasting Model Incorporating GLAR and ANN for Foreign Exchange Rates. Computers and Operations Research 32 (2005) 2523-2541
8. Hansen L.K., Salamon, P.: Neural Network Ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence 12 (1990) 993-1001
9. Breiman, L.: Bias, Variance, and Arcing Classifiers. Technical Report TR460, Department of Statistics, University of California, 1994

10. Ueda, N.: Optimal Linear Combination of Neural Networks for Improving Classification Performance. IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000) 207-215
11. Benediktsson, J.A., Sveinsson, J.R., Ersoy, O.K., Swain, P.H.: Parallel Consensual Neural Networks. IEEE Transactions on Neural Networks 8 (1997) 54-64
12. Bishop, C.M.: Improving the Generalization Properties of Radial Basis Function Neural Networks. Neural Computation 3 (1991) 579-588
13. Wedding II, D.K., Cios, K.J.: Time Series Forecasting by Combining RBF Networks, Certainty Factors and the Box-Jenkins Model. Neurocomputing 10 (1996) 149-168
14. Chen, S., Billings, S.A., Cowan, C.F.N., Grant, P.M.: Nonlinear Systems Identification Using Radial Basis Functions. International Journal of Systems Science 21 (1990) 2513-2539
15. Lee, Y.C.: Handwritten Digit Recognition Using K Nearest Neighbor, Radial Basis Function and Backpropagation Neural Networks. Neural Computation 3 (1991) 440-449
16. Leonard, J.A., Kramer, M.A., Unger, L.H.: A Neural Network Architecture that Computes its Own Reliability. Computers and Chemical Engineering 16 (1992) 819-835

Combining Time-Scale Feature Extractions with SVMs for Stock Index Forecasting

Shian-Chang Huang[1] and Hsing-Wen Wang[2]

[1] Department of Business Administration, National Changhua University of Education, Changhua, Taiwan
shhuang@cc.ncue.edu.tw

[2] Department of Business Administration, National Changhua University of Education, Changhua, Taiwan
shinwen@cc.ncue.edu.tw

Abstract. Support vector machine (SVM) has appeared as a powerful tool for time series forecasting and demonstrated better performance over other methods. This paper proposes a novel hybrid model which combines time-scale feature extractions with SVM models for stock index forecasting. The time series of explanatory variables are decomposed by the wavelet basis, and the extracted time-scale features then serve as inputs of a SVM model which performs the nonparametric forecasting. Compared with pure SVM models or traditional GARCH models, the performance of the new method is the best. The root-mean-squared forecasting errors are significantly reduced. The results of this study can help investors for controlling and reducing their risks in international investments.

1 Introduction

With the expansion of international financial links and the continued liberalization of cross-border cash flows, international financial markets are highly correlated, and some markets are information leader of the other markets. With these time series features, investors can develop an accurate forecasting strategy on the stock index evolution. The objective of this study is to help investors for implementing a forecasting model based on these important characteristics of financial time series.

Stock index predictions are one of the challenging applications of modern time series forecasting and very important for the success of many businesses and financial institutions. The increasingly tight correlations among financial markets can help investors make a good forecast on the co-movements of stock indices. However, international investors are a diverse group. They operate on very different time scales. As a result, the correlations between international market indices are not fixed over every time scale. The new forecasting model should extract these important time scale features among financial markets to make a good prediction.

For the feature extraction, wavelet analysis is a powerful tool to extract time series features among various time scales. Reviewing the literature on wavelet analysis, it is widely used in engineering, important applications including the signal processing and image compressions. However, it is relatively new in economics and finance. For forecasting strategies, Box and Jenkins' Auto-Regressive Integrated Moving Average (ARIMA) technique has been widely used for time series forecasting. However, ARIMA is a general univariate model and it is developed based on the assumption that the time series being forecasted are linear and stationary. In recent years, neural networks (NN) has found useful applications in financial time-series analysis and forecasting (Yao and Tan [1], Zhang and Hu [2]). Many researches have shown that NNs perform better than ARIMA models, specifically, for more irregular series and for multiple-period-ahead forecasting.

Recently, support vector machine (Cristianini and Shawe-Taylor [3], Schoelkopf, Burges, and Smola [4], Vapnik [5]) has appeared as a powerful tool for forecasting and demonstrated better performance over neural networks or ARIMA-based models (Cao and Tay [6], Gestel et al. [7]). In very simple terms an SVM corresponds to a linear method in a very high dimensional feature space that is nonlinearly related to the input space. Even though we think of it as a linear algorithm in the high dimensional feature space, in practice, it does not involve any computations in that high dimensional space. By the use of kernels, all necessary computations are performed directly in input space. Support vector machines for regression, as described by Vapnik [5], also exploit the idea of mapping input data into a high dimensional reproducing kernel Hilbert space (RKHS) where a linear regression is performed. The advantages of support vector regression are: the presence of a global minimum solution resulting from the minimization of a convex programming problem; relatively fast training speed; and sparseness in the solution representation.

The major innovation of this paper lies in combing the above two powerful tools and developing a hybrid SVM model for forecasting stock indices. In the empirical analysis, owing to that most high technology investments over the world have moved closer and closer, especially in U.S. Silicon Valley and Eastern Asia, the NASDAQ index becomes strongly correlated with major major Asian stock indices, and it's always a leading indicator with great prediction power on these indices. Therefore, the NASDAQ return serves as an important explanatory variables for three major Asian indices. For forecasting an Asian index, wavelet analysis is used to decompose and extract important features from the NASDAQ and lagged returns of itself, and then these features are fed into an SVM model to perform the nonparametric forecasting. Compared with neural network systems, SVM minimize the structural risk of the prediction model as opposed to empirical risk minimization as employed by NN systems. Thus the new forecasting model is more robust and gets rid of the overfitting problem of traditional nonparametric regression models.

The remainder of the paper is organized as follows. Section 2 describes wavelet analysis and SVM models. Section 3 introduces the GARCH prediction model. Section 4 describes the data used in the study, and discusses the empirical findings. Conclusions are given in Section 5.

2 Wavelet-Based Feature Extraction

The basic tool of wavelet analysis is the multiresolution decomposition (MRD). For a thorough review of wavelet analysis I refer to Daubechies [8], and Percival and Walden [9]. Practical applications of wavelet analysis is given in Lee [10] and Gençay et al. [11]. Technical details of wavelet analysis are discussed in Bruce and Gao [12].

2.1 Multi-resolution Decomposition

Any function $f(t)$ in $L^2(R)$ can be decomposed by a sequence of projections onto the wavelet basis. The wavelet representation of the signal or function $f(t)$ in $L^2(R)$ can be written as

$$\begin{aligned} f(t) = & \sum_k s_{J,k}\phi_{J,k}(t) + \sum_k d_{J,k}\psi_{J,k}(t) \\ & + \sum_k d_{J-1,k}\psi_{J-1,k}(t) + ... + \sum_k d_{1,k}\psi_{1,k}(t), \end{aligned}$$

where ϕ is the father wavelet and ψ the mother wavelet. $\phi_{j,k}$ and $\psi_{j,k}$ are scaling and translation of ϕ and ψ, defined as

$$\phi_{j,k}(t) = 2^{-j/2}\phi(2^{-j}t - k) = 2^{-j/2}\phi\left(\frac{t - 2^j k}{2^j}\right) \tag{1}$$

$$\psi_{j,k}(t) = 2^{-j/2}\psi(2^{-j}t - k) = 2^{-j/2}\psi\left(\frac{t - 2^j k}{2^j}\right). \tag{2}$$

In the representation J is the number of multiresolution components, and $s_{J,k}$ are called the smooth coefficients, and $d_{j,k}$ are called the detailed coefficients. If we define

$$S_J(t) = \sum_k s_{J,k}\phi_{J,k}(t) \tag{3}$$

$$D_j(t) = \sum_k d_{j,k}\psi_{j,k}(t) \quad \text{for } j = 1, 2, ..., J. \tag{4}$$

The functions (3) and (4) are called the smooth signal and the detail signals, respectively, which constitute a decomposition of a signal into orthogonal components at different scales. A signal $f(t)$ can thus be expressed in terms of these signals:

$$f(t) = S_J(t) + D_J(t) + D_{J-1}(t) + ... + D_1(t). \tag{5}$$

2.2 Support Vector Machines

The support vector machines (SVMs) were proposed by Vapnik [5]. Based on the structured risk minimization (SRM) principle, SVMs seek to minimize an upper bound of the generalization error instead of the empirical error as in other neural networks. Additionally, the SVMs models generate the regress function by applying a set of high dimensional linear functions. The SVM regression function is formulated as follows:

$$y = w\phi(x) + b, \tag{6}$$

where $\phi(x)$ is called the feature, which is nonlinear mapped from the input space x to the future space. The coefficients w and b are estimated by minimizing

$$R(C) = C\frac{1}{N}\sum_{i=1}^{N} L_\varepsilon(d_i, y_i) + \frac{1}{2}||w||^2, \tag{7}$$

where

$$L_\varepsilon(d, y) = \begin{cases} |d-y| - \varepsilon & |d-y| \geq \varepsilon, \\ 0 & \text{others,} \end{cases}, \tag{8}$$

where both C and ε are prescribed parameters. The first term $L_\varepsilon(d, y)$ is called the ε-intensive loss function. The d_i is the actual option price in the ith period. This function indicates that errors below ε are not penalized. The term $\frac{C}{N}\sum_{i=1}^{N} L_\varepsilon(d_i, y_i)$ is the empirical error. The second term, $\frac{1}{2}||w||^2$, measures the smoothness of the function. C evaluates the trade-off between the empirical risk and the smoothness of the model. Introducing the positive slack variables ξ and ξ^*, which represent the distance from the actual values to the corresponding boundary values of ε-tube. Equation (7) is transformed to the following constrained formation:

$$\min_{w,b,\xi,\xi^*} R(w, \xi, \xi^*) = \frac{1}{2}w^T w + C\left(\sum_{i=1}^{N}(\xi_i + \xi_i^*).\right) \tag{9}$$

Subject to

$$w\phi(x_i) + b_i - d_i \leq \varepsilon + \xi_i^*, \tag{10}$$

$$d_i - w\phi(x_i) - b_i \leq \varepsilon + \xi_i, \tag{11}$$

$$\xi_i, \xi_i^* \geq 0. \tag{12}$$

After taking the Lagrangian and conditions for optimality, One can get the dual representation of the model,

$$y = f(x, \alpha, \alpha^*) = \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)K(x, x_i) + b, \tag{13}$$

where α_i and α_i^* are Lagrangian multipliers, which are the solution to the dual problem, and $K(x, x_i)$ is the kernel function. b follows from the complementarity Karush-Kuhn-Tucker (KKT) conditions.

The value of the kernel is equal to the inner product of two vectors x_i and x_j in the feature space, such that $K(x_i, x_j) = \phi(x_i)\phi(x_j)$. Any function that satisfying Mercer's condition (Vapnik [5]) can be used as the Kernel function. The Gaussian kernel function

$$K(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right) \tag{14}$$

is specified in this study, because Gaussian kernels tend to give good performance under general smoothness assumptions.

3 GARCH Prediction Models

For comparison, a generalized autoregressive conditional heteroscedasticity models [13] are also used for predictions. The following model is employed to predict daily returns of the major Asia indices. In the conditional mean part,

$$r_t = \alpha + \beta r_{t-1} + \gamma x_{t-1} + \varepsilon_t, \tag{15}$$

this equation describes the causal relationship between current returns and lagged returns among market indices. r_t is the daily return at time t for each index, x_{t-1} is the lagged NASDAQ return which serves as a explanatory variable, and $\varepsilon_t \sim N(0, \sigma_t)$, the innovation or shock at time t. We model σ_t to follow a univariate GJR-GARCH(1,1) process,

$$\sigma_t = a_0 + a_1\varepsilon_{t-1}^2 + c_1 S_{t-1}\varepsilon_{t-1}^2 + b_1\sigma_{t-1}^2, \tag{16}$$

where

$$S_{t-1} = \begin{cases} 1 & \text{if} \quad \varepsilon_{t-1} < 0 \\ 0 & \text{if} \quad \varepsilon_{t-1} \geq 0. \end{cases} \tag{17}$$

That is, depending on whether ε_{t-1} is above or below the threshold value of zero, ε_{t-1}^2 has different effects on the conditional variance σ_t^2. The asymmetric impacts of ε_{t-1} on σ_t^2 has known as the leverage effects.

4 Empirical Results and Analysis

The data employed in the study are composed of the following daily indices: NASDAQ(US), NK225(Japan), TWSI(Taiwan) and KOSPI(South Korea). All the index data encompass the period from January 2003 to December 2004. There are 435 observations. These stock market indices are then transformed into daily returns (by 100 times their log differences). Selected descriptive statistics of daily log returns of these indices are presented in Table 1. Sample means, standard deviations, maxima, minima, skewness, kurtosis are reported. The time-series plots of these four indices are shown in Figure 1 and 2.

In this study, only one-step-ahead forecasting is considered. One-step-ahead forecasting can prevent problems associated with cumulative errors from the previous period for out-of-sample forecasting. The SVM model is trained in a batch manner, namely, 300 data points before the day of prediction are treated

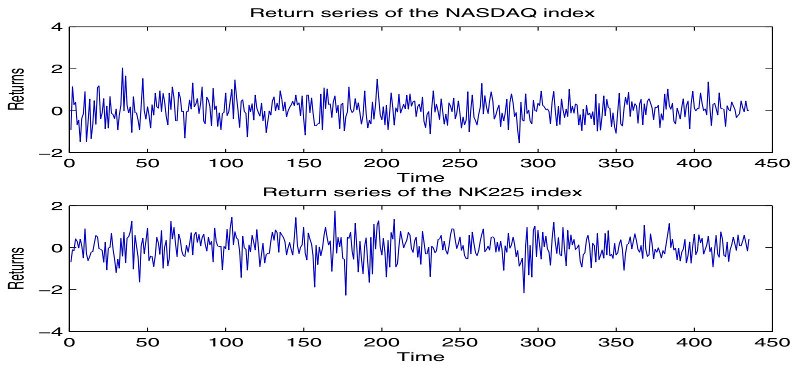


Fig. 1. Return Series of NASDAQ and NK225 Indices

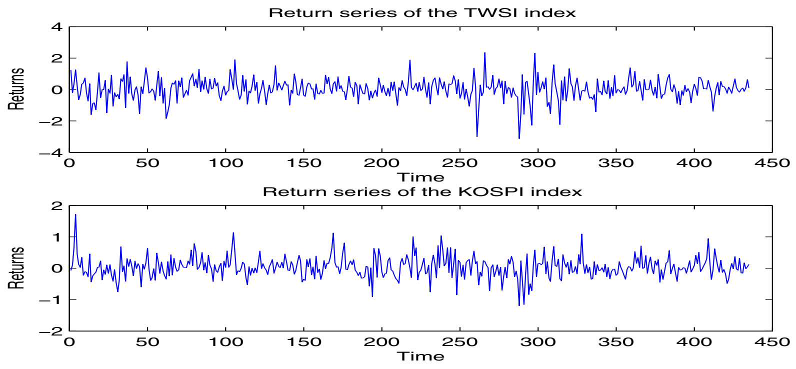


Fig. 2. Return Series of TWSI and KOSPI Indices

Table 1. Descriptive Statistics of Daily Returns of Four Indices

Series	Mean	Standard Deviation	Skewness	Kurtosis
NASDAQ	0.0419	0.5676	0.0504	3.1555
NK225	0.0283	0.5854	-0.4265	3.8608
TWSI	0.0260	0.6545	-0.4805	6.0437
KOSPI	0.0373	0.3330	0.3824	5.4790

as the training data set, and the window of the training data set slides with the current prediction. The daily returns in the last 135 days of the data series are used as the test set to evaluate the performances of the pure SVM, the hybrid SVM and the GARCH prediction models.

We apply the Daubechies least asymmetric filters with length 8 to decompose the explanatory variables, the NASDAQ returns and lagged returns of itself (r_{t-1}). These returns are decomposed into four mutually orthogonal different periodicity series, ranging from the shortest-periodicity series to the longest-periodicity series. As shown in Table 2, the correlations between each Asian Index and the NASDAQ Index are not fixed on every time scale. Their correlation coefficients range from -0.0464 to 0.6237. Thus, the new method can capture important time scale features which can't be revealed in aggregated explanatory variables.

Table 2. Correlations on Every Time Scale Component

correlation coefficient	NK225	TWSI	KOSPI
$corr(D_1, D_1^N)$	-0.0464	0.1019	-0.0330
$corr(D_2, D_2^N)$	0.3709	0.3280	0.2213
$corr(D_3, D_3^N)$	0.5589	0.5730	0.3594
$corr(S_3, S_3^N)$	0.6237	0.5140	0.3537

To model nonlinear relationship between these features and output data, a SVM model is applied to forecast the future evolution of the target stock index; that is, the all of the decomposed series or extracted features served as the inputs of the SVM model. The parameters used in the SVM model are set as follows: $\varepsilon = 0.01$, $C = 100$ and $\sigma = 0.8$ for the Gaussian Kernel. Traditional performance indices such as MSE (mean square error), RMSE (root mean squared error), MAE (mean absolute error), and MAPE (mean absolute percent error), can be used as measures of the forecasting accuracy. In this studies, we adopted the RMSE as the performance index. The RMSE index is defined as follows:

$$RMSE = \left(\frac{1}{N} \sum_{t=1}^{N} (r_t - \widehat{r}_t)^2 \right)^{1/2}, \tag{18}$$

where N is the number of forecasting periods, r_i is the actual return at period t, and $\widehat{r}_t$ is the forecasting return at period t.

Table 3 shows the forecasting performance of the GARCH model, the pure SVM model, and the hybrid SVM model on three major Asian stock indices. The actual returns, predicted values and model residuals on various stock indices are displayed in Figures 1-3.

Compared the pure SVM model with the traditional GARCH model, the performance of the pure SVM is slightly poor. However, when hybrid with the

Table 3. Relative Forecasting Performance of Three Models on Major Asian Stock Indices

	NK225	TWSI	KOSPI
GARCH Predictions	0.2013	0.2183	0.1514
Pure SVM	0.3038	0.3774	0.1725
Hybrid SVM	0.1187	0.1779	0.0379

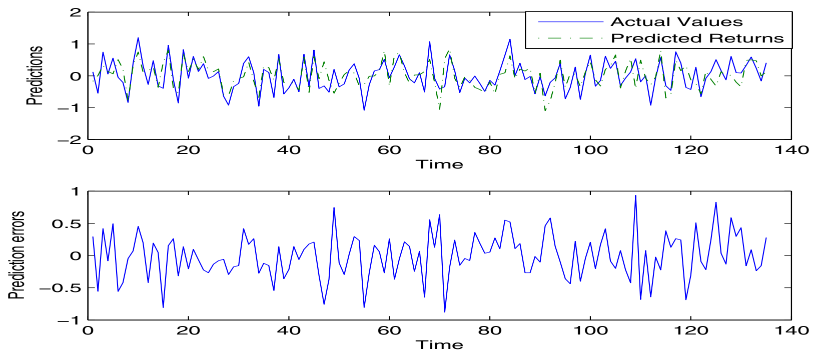


Fig. 3. Hybrid SVM Forecasts on the NK225 Index Returns

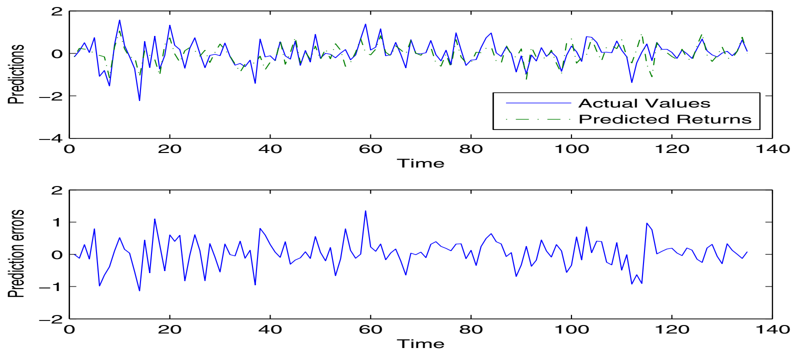


Fig. 4. Hybrid SVM Forecasts on the TWSI Index Returns

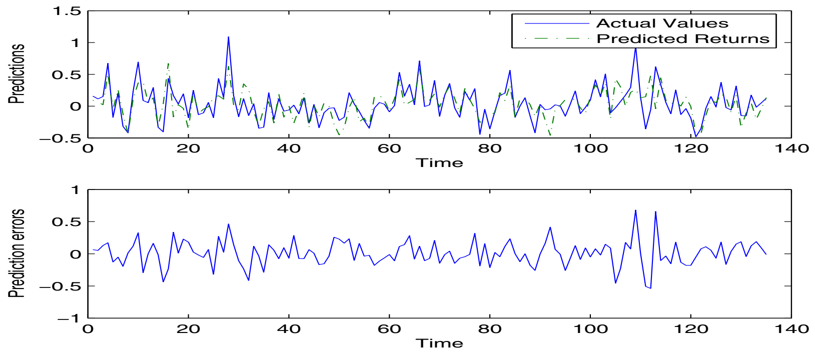


Fig. 5. Hybrid SVM Forecasts on the KOSPI Index Returns

wavelet-based feature extraction, the hybrid SVM methods shows superior performance, and significantly reduces the root-mean-squared forecasting errors. Among the three methods, the performance of the hybrid model is the best. The success of this new forecasting model can be attributed to the following two reasons: first, the relationship between the NASDAQ and these Asian indices is not fixed over every time scale. The proposed wavelet-based SVM model is capable to capture all these important features. Conventional GARCH predictions only incorporate information about the "average" correlations among these indices, and thus its performance is not as good as the new model.

Second reason for the excellent performance of the proposed model is that SVMs map input data into a high dimensional reproducing kernel Hilbert space (RKHS) where a linear regression is performed. The RKHS has richer algebraic and topological structures to capture nonlinear relationship between input and output data. Consequently, SVMs provide a valuable and flexible framework for the representation of relations in the data.

5 Conclusions

Combining wavelet analysis and support vector machines (SVM), this study develops a new hybrid SVM prediction model, which operates on multiple resolutions and uses a flexible nonparametric regression to predict the future evolution of the return series.

International investors are a diverse group. They operate on very different time scales. Intraday traders, market makers and hedging funds strategists trade on very short time scales ranging from seconds to hours. International portfolio managers are traders on the intermediate time scales. Their trades typically occur on a weekly to monthly basis. Central banks are the main traders on the longest time scales, they consider long-term economic fundamentals with the longest investment horizon.

Since the aggregated predictive power of lagged NASDAQ returns is unequally divided into the four time scales, the correlations among the NASDAQ and these Asian indices change over every time scale. The pure SVM or GARCH model only incorporating the "average" of the above information have very poor performance. By using wavelet analysis and by mapping data to the high dimensional reproducing kernel Hilbert space, the proposed hybrid SVM model is capable to capture all the important features, and makes an excellent forecasts.

To summarize, the powerful framework established in this study can also applied to other problems involving financial forecasts. The results of this paper can be used to perform a good hedge on international investments. A challenging future task is to combine Bayesian methods for important feature selections.

Acknowledgment

This work was supported in part by the National Science Foundation of Taiwan.

References

1. Yao, J. and Tan, C. L.: A case study on using neural networks to perform technical forecasting of forex, Neurocomputing. 34 (2000) 79-98
2. Zhang, G. and Hu, M. Y.: Neural network forecasting of the British Pound/US Dollar exchange rate, OMEGA: Int. Journal of Management Science. 26 (1998) 495-506
3. Cristianini, N. and Shawe-Taylor, J.: An Introduction to Support Vector Machines, Cambridge University Press. (2000)
4. Schoelkopf, B., Burges, C. J. C., and Smola, A. J.: Advances in kernel methods - support vector learning, MIT Press, Cambridge, MA. (1999)
5. Vapnik, V. N.: The Nature of Statistical Learning Theory. New York: Springer-Verlag. (1995)
6. Cao, L. and Tay, F.: Financial forecasting using support vector machines, Neural Computing & Applications. 10(2001) 184-192
7. Gestel, T. et. al.: Financial time series prediction using least squares support vector machines within the evidence framework, IEEE trans. Neural Network, 12 (2001), 809-821
8. Daubechies, I.: Ten Lectures on Wavelets, SIAM, Philadelphia, PA. (1992)
9. Percival, D. and Walden, A.: Wavelet Methods for Time Series Analysis, Cambridge University Press, Cambridge. (2000)
10. Lee, G. H.: Wavelets and wavelet estimation: a review, Journal of Economic Theory and Econometrics. 4 (1998) 123-158
11. Gençay, R., Selçuk, F., and Whitcher, B.: An introduction to wavelets and other filtering methods in finance and economics. Academic Press, London. (2002)
12. Bruce, A. and Gao, H. Y.: Applied Wavelet Analysis with SPLUS, Springer-Verlag, New York. (1996)
13. Bollerslev, T.: Generalized autoregressive conditional heteroskedasticity, Journal of Econometrics. 31 (1986) 307-327

Extensions of ICA for Causality Discovery in the Hong Kong Stock Market

Kun Zhang and Lai-Wan Chan*

Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, Hong Kong
{kzhang, lwchan}@cse.cuhk.edu.hk

Abstract. Recently independent component analysis (ICA) has been proposed for discovery of linear, non-Gaussian, and acyclic causal models (LiNGAM). As in practice the LiNGAM assumption usually does not exactly hold, in this paper we propose some methods to perform causality discovery even when LiNGAM is violated. The first method is ICA with a sparse separation matrix. By incorporating a suitable penalty term, the separation matrix produced by this method tends to satisfy the LiNGAM assumption. The other two methods are proposed to tackle nonlinearity in the data generation procedure, which violates the LiNGAM assumption. In the second method, the post-nonlinear mixing ICA model is exploited to do causality discovery when the nonlinearity is componentwise. The third method is proposed for the case where the nonlinear distortion in data generation is of arbitrary form, but smooth and weak. The separation system for such data is a linear transformation coupled with a nonlinear one, and the nonlinear one is as weak as possible such that it can be neglected when performing causality discovery. The linear causal relations in the data are then revealed. The proposed methods are applied to discover the causal relations in the Hong Kong stock market, and the last method works very well. The resulting causal diagram shows some interesting information in the stock market.

1 Introduction

It is well known that financial assets are not independent of each other, and that there may be some relations among them. Such relations can be described in different ways. In risk management, correlations are used to describe them and help to construct portfolios. The business group, which is a collection of firms bound together in some formal and/or informal ways, focuses on ties between financial assets and has attracted a lot of interests [4,8]. But these descriptions do not tell us the causal relations among the financial assets.

* This work was partially supported by a grant from the Research grants Council of the Hong Kong Special Administration Region, China. We are grateful to the LiNGAM group and Prof. L.B. Almeida for providing the MATLAB codes for LiNGAM analysis and MISEP, respectively. We would also thank Dr. Heng Peng for helpful discussions.

The return of a particular stock may be influenced by those of other stocks, for many reasons, such as the ownership relations and financial interlinkages [8]. According to the efficient market hypothesis, such influence should be reflected in the stock returns immediately. In this paper we aim to discover the causal relations among selected stocks by analyzing their daily returns.

Traditionally, causality discovery algorithms for continuous variables usually assume that the dependencies of variables are of a linear form and that the variables are Gaussian distributed [9]. Under the Gaussianity assumption, only the correlation structure of variables is considered and all higher-order information is neglected. As a consequence, one would obtain some possible causal diagrams which are equivalent in their correlation structure, and could not find the true causal directions. Recently, it has been shown that the non-Gaussianity distribution of the variables allows us to distinguish the explanatory variable from the response variable, and consequently, to identify the full causal model [2,11].

In particular, in [11] an elegant and efficient method was proposed for identifying the *linear, non-Gaussian, acyclic causal model* (abbreviated LiNGAM) by exploiting the independent component analysis (ICA) technique. If the data are generated according to the LiNGAM model, theoretically, the ICA separation matrix $\mathbf{W}$ of the data can be permuted to lower triangularity. However, in practice, this may not hold, due to the finite sample effect, the existence of unobserved confounder variables [9], or the fact that mild nonlinearity and noise are often encountered in the data generation procedure.

In this paper, we propose a learning approach to continuously tune $\mathbf{W}$ such that it can be permuted to lower triangularity, when LiNGAM is not violated much. Our another contribution is to propose two methods to tackle the nonlinearity in data generation when performing causality discovery. First, the post-nonlinear (PNL) mixing ICA model [12,14] is used to account for a particular type of nonlinearity. Second, a linear transformation coupled with a nonlinear one modeled by a multi-layer perceptron (MLP) is proposed to represent the nonlinear ICA transformation. With certain penalties, this structure helps to discover the linear causal relations in the data, when nonlinear distortion is weak. Finally, we apply the proposed methods for causality discovery in the Hong Kong stock market.

2 Causality Discovery by ICA: Basic Idea

The LiNGAM model assumes that the generation procedure of the observed data follows the following properties [11]. *1.* The generation procedure is recursive. This means that the observed variables x_i, $i = 1, ..., n$, can be arranged in a causal order, such that no later variable causes any earlier variable. This causal order is denoted by $k(i)$. *2.* The value of x_i is a linear function of the values assigned to the earlier variables, plus a disturbance term e_i and an optional constant c_i: $x_i = \sum_{k(j)<k(i)} b_{ij} x_j + e_i + c_i$. *3.* e_i are continuous-valued variables with non-Gaussian distributions of non-zero variances (or at most one is Gaussian), and are independent of each other.

After centering of the variables, the causal relations among x_i can be written in the matrix form: $\mathbf{x} = \mathbf{B}\mathbf{x} + \mathbf{e}$, where $\mathbf{x} = (x_1, ..., x_n)^T$, $\mathbf{e} = (e_1, ..., e_n)^T$, and the matrix $\mathbf{B}$ can be permuted (by simultaneous equal row and column permutations) to strict lower triangularity if one know the causal order $k(i)$ of x_i. We then have $\mathbf{e} = \mathbf{W}\mathbf{x}$, where $\mathbf{W} = \mathbf{I} - \mathbf{B}$. This is exactly the ICA separation procedure [6].

Therefore, the LiNGAM model can be estimated by ICA. Let $\mathbf{W}$ be the ICA separation matrix of $\mathbf{x}$. We can permute the rows of $\mathbf{W}$ such that it produces a matrix $\widetilde{\mathbf{W}}$ without any zero on its diagonal (or in practice, $\sum_i |\widetilde{\mathbf{W}}_{ii}|$ is maximized). Dividing each row of $\widetilde{\mathbf{W}}$ by the corresponding diagonal entry, a new matrix $\widetilde{\mathbf{W}}'$ with all entries on its diagonal being 1 is obtained. Finally, by applying equal row and column permutations on $\mathbf{B} = \mathbf{I} - \widetilde{\mathbf{W}}'$, we can find the matrix $\widetilde{\mathbf{B}}$ which is as close as possible to strictly lower triangularity. $\widetilde{\mathbf{B}}$ contains the causal relations of x_i. For details, see [11].

In practice, $\mathbf{W}$ may not be able to be permuted to lower triangularity, due to the estimation error, or violation of the LiNGAM assumption. In [11] some statistical tests are exploited for pruning the entries of $\mathbf{W}$, so that $\mathbf{W}$ is permuted to lower triangularity. This method is a discrete process, meaning that after statistical tests, $\mathbf{W}_{ij}$ are either retained or set to 0. With this scheme, small changes in the data may result in a very different model. In addition, if an entry is thought of insignificant and pruned, the ICA outputs will change, the independence between them may not hold, and the significance of other entries will also be affected.

3 Methodology

3.1 ICA with Sparse Transformation Matrix

Why ICA with Sparse Transformation Matrix. When performing ICA, sometimes it is desirable not only to achieve the independence between outputs, but also to make the transformation matrix ($\mathbf{W}$ or $\mathbf{A}$) as sparse as possible, i.e. to make its zero entries as many as possible, for the following reasons. First, consider the case where the data dimension is high and the true transformation matrix is sparse. If the zero entries can be automatically detected and are set to 0 during the learning process, the model complexity will be reduced and parameter estimation will be more reliable. Second, a sparse mixing matrix means that the observations x_i are affected by only a smaller subset of the independent sources s_i, and this makes the interpretation easier. The third reason is application-oriented. When applying ICA for estimating the LiNGAM model, $\mathbf{W}$ is expected to be permuted to lower triangularity. In order to achieve this, $\mathbf{W}$ with as many as possible zero entries is preferred. The sparsity of the transformation matrix can be achieved by introducing some penalty term.

Penalties Producing Sparse Coefficients in Linear Regression. In the statistics literature, the behavior of different penalties in the linear regression problem has been intensively studied. By incorporating the L_γ penalty on β_i, the penalized least square error (LSE) estimate for the coefficients $\boldsymbol{\beta} = (\beta_1, ..., \beta_p)^T$ is obtained by minimizing the mean square error plus the penalty term $\lambda \sum_{j=1}^{p} |\beta_j|^\gamma$, where $\lambda \geq 0$ is a parameter controlling the extent to which the penalty influences the solution. $\gamma = 2$ results in ridge regression, which tends to make coefficients smaller, but cannot set any coefficient to 0. The L_2 penalty also results in the weight decay regularizer in neural networks learning. The *least absolute shrinkage and selection operator* (LASSO) emerges when $\gamma = 1$, i.e. the L_1 penalty is adopted [13]. With a suitable λ, it automatically sets insignificant coefficients to 0 . The L_1 penalty corresponds to a Laplacian prior on β_i. From now on we drop the subscript in β_i for simplicity.

In [3] it was claimed that a good penalty function should result in an estimator with the following three properties: *1.* unbiasedness for the resulting estimator of significant parameters, *2.* sparsity, which means that insignificant coefficients are automatically set to 0, to reduce the model complexity, and *3.* continuity of the resulting estimator with respect to changes in data to avoid unnecessary variation in model prediction. The *smoothly clipped absolute deviation* (SCAD) penalty, which has the above properties, was proposed. The derivative of the SCAD penalty (including the coefficient λ) is given by: $p'_\lambda(\beta) = \lambda \Big\{ I(\beta \leq \lambda) + \frac{(a\lambda - \beta)_+}{(a-1)\lambda} I(\beta > \lambda) \Big\}$, for some $a > 2$ and $\beta > 0$, where $I(\cdot)$ is the indicator function, and the typical value for a is 3.7. The SCAD penalty corresponds to an improper prior on β. For details of the SCAD penalty, see [3].

Now we propose a generalized version of the SCAD penalty and consider some computational problems. In fact, in the SCAD penalty, the parameter controlling the strength of the penalty and that controlling the range the penalty applies to are not necessarily equal. We then propose the following *generalized SCAD* (GSCAD) (for $\beta \geq 0$):

$$p'_\lambda(\beta) = \lambda \Big\{ I(\beta \leq \lambda_1) + \frac{(a\lambda_1 - \beta)_+}{(a-1)\lambda_1} I(\beta > \lambda_1) \Big\} \tag{1}$$

GSCAD plays a trade-off between SCAD and the L_1 penalty. When λ_1 is very large, it tends to be the L_1 penalty. When the data are very noisy, we can choose the parameter $\lambda_1 > \lambda$ so that the penalty operates on a wider range of the parameter value. The penalty for weight elimination in neural networks learning can be considered as a heuristic approximate to GSCAD. But it just shrinks small parameters and can not set any parameter to 0.

Any penalty which can set small parameters to 0 and results in a continuous estimator, including the L_1 penalty and (G)SCAD, must be singular (not differentiable) at the origin [3]. Consequently, we can not use the gradient-based learning rule to optimize the objective function when β is around 0. We use some approximation to tackle this optimization problem. The function $\tanh(m\beta)$ (m is a large number, say 200) is used to approximate the derivative of $|\beta|$. The GSCAD (Eq. 1) is then approximated by $p'_\lambda(\beta) = \lambda \Big\{ \tanh(m\beta) \cdot I(\beta \leq \lambda_1) + \tanh(m\lambda_1) \cdot \frac{(a\lambda_1 - \beta)_+}{(a-1)\lambda_1} I(\beta > \lambda_1) \Big\}$. As the price, β will not exactly shrink to

0 even if its true value is 0; it will converge to a very small number (about 10^{-2}) instead. In practice, after the convergence of the algorithm, we need to set the parameters whose values are small enough, say, smaller than 0.02, to 0.

ICA with Sparse Separation Matrix. Under some weak regularity conditions, the ordinary maximum likelihood estimates are asymptotically normal, and the above penalties can be applied to likelihood-based models [3]. As ICA algorithms can be derived from a maximum likelihood point of view [10], *ICA with a sparse separation matrix* can be derived with the data (log-)likelihood together with the above penalties as the objective function. Consider the ICA separation procedure $\mathbf{y} = \mathbf{W}\mathbf{x}$, where $\mathbf{x} = (x_1, ..., x_n)^T$ are the observed variables assumed to be generated from independent variables $\mathbf{s} = (s_1, ..., s_n)^T$ by $\mathbf{x} = \mathbf{A}\mathbf{s}$. The penalized log-likelihood is $\frac{1}{N}\ell(\mathbf{x}; \mathbf{W}) - \sum_{i,j=1}^{n} p_\lambda(w_{ij})$, where $\ell(\mathbf{x}; \mathbf{W})$ denotes the log-likelihood function of the observations $\mathbf{x}$ given the ICA model and N denotes the number of samples. The corresponding natural gradient learning rule for $\mathbf{W}$ is $\triangle\mathbf{W} \propto \big(\mathbf{I} - E\{\boldsymbol{\psi}(\mathbf{y})\mathbf{y}^T\} - [p'_\lambda(w_{ij})]\mathbf{W}^T\big)\mathbf{W}$, where $[p'_\lambda(w_{ij})]$ denotes the matrix whose (i, j)-th entry is $p'_\lambda(w_{ij})$, and $\boldsymbol{\psi}(\mathbf{y})$ denotes the score function of $\mathbf{y}$. In practice, some modifications can be adopted to ensure that y_i are of unit variance at convergence; for example, see [12].

ICA with Sparse Separation Matrix for Estimation of LiNGAM. The parameters involved in the penalty $p_\lambda(w_{ij})$, denoted by $\boldsymbol{\theta}$, can be estimated by cross-validation or generalized cross-validation [3]. But when using ICA with a sparse $\mathbf{W}$ for LiNGAM analysis, we need to choose a suitable $\boldsymbol{\theta}$ such that $\mathbf{W}$ can be permuted to lower triangularity. A greedy method can be adopted for determining the parameter λ. (When adopting the GSCAD penalty, we set $\lambda_1 = 1.2\lambda$ and $a = 3.2$ empirically.) Starting from a small value, each time λ is increased by a fixed increment. After the algorithm converges for the new value of λ, we check whether the LiNGAM property holds for $\mathbf{W}$. The check can be easily done with Algorithm B in [11]. Once the LiNGAM property holds, we terminate the above procedure and LiNGAM is estimated by analyzing $\mathbf{W}$; if λ reaches the upper bound set in advance, we conclude that the data do not follow the LiNGAM model. The computation involved in this method is not high, since the convergence of the algorithm for the new value of λ usually requires just several iterations.

3.2 To Incorporate Component-Wise Nonlinearity

ICA with a sparse separation matrix can estimate the LiNGAM model when it is slightly violated. It may fail to discover the causality when the data generation procedure is actually nonlinear. Now let us take into account such nonlinearity as it is often encountered in practice. Consider the following data generation procedure. Suppose the observed data x_i do not follow the LiNGAM model; however, z_i, as the actual effect of x_i, is a continuous and invertible function of x_i, and z_i follow the LiNGAM model. We call this model *component-wise nonlinear LiNGAM* (CWN-LiNGAM). Let $z_i = g_i(x_i)$. Without loss of generality, we assume that the nonlinear functions g_i to be strictly monotonically increasing.

Let $\mathbf{z} = (z_1, ..., z_n)^T$ and f_i be the inverse of g_i. We have $x_i = f_i(z_i)$ and $\mathbf{Wz} = \mathbf{e}$. Clearly the observed data $\mathbf{x}$ are post-nonlinear (PNL) mixtures of the independent sources e_i [12]. To identify the CWN-LiNGAM model, we first need to separate the PNL mixtures x_i and to estimate g_i and $\mathbf{W}$. Some algorithms can be found in [12,14]. Next, the causal relations among x_i can be discovered from $\mathbf{W}$, with the method mentioned in Section 2.

3.3 With Mild Nonlinearity Modeled by MLP

We now consider the general case of the nonlinear distortion often encountered in the data generation procedure, provided that the nonlinear distortion is smooth and mild. The structure in Fig. 1 is used to model the nonlinear transformation from the the observed variables x_i to the disturbance variables e_i. This structure is *a linear transformation coupled with a MLP*. The MLP accounts for the nonlinear distortion if necessary. We would like to address that this structure was adopted to perform nonlinear ICA in the experiments of [1]. It was called a MLP with direct connections between inputs and outputs there. In fact, this structure implicitly introduces the regularization condition that the nonlinear ICA mapping should be as close as possible to linear.[1]

According to Fig. 1, we have $\mathbf{e} = \mathbf{Wx} + \mathbf{h}(\mathbf{x})$, and consequently $\mathbf{x} = (\mathbf{I} - \mathbf{W})\mathbf{x} + \mathbf{h}(\mathbf{x}) + \mathbf{e}$, where $\mathbf{h}(\mathbf{x})$ denotes the output of the MLP. As it is difficult to analyze the relations among x_i implied by the nonlinear transformation $\mathbf{h}(\mathbf{x})$, we expect that $\mathbf{h}(\mathbf{x})$ is weak such that its effect can be neglected. The *linear* causal relations among x_i can then by discovered by analyzing $\mathbf{W}$.

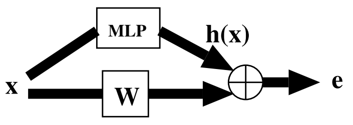


Fig. 1. Structure used to model the transformation from the observed data x_i to independent disturbances e_i

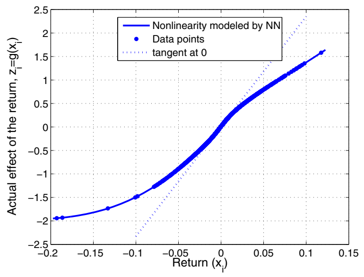


Fig. 2. Nonlinear function g_i transforming the return x_i to its actual effect z_i

In order to do causality discovery, the separation system in Fig. 1 is expected to exhibit the following properties. *1.* The outputs e_i are mutually independent, since

[1] We conjecture that with this regularization condition, nonlinear ICA leads to nonlinear blind source separation (BSS) when the nonlinear distortion is mild [15]. It was ever believed that smooth mappings provided by MLP's are sufficient to ensure that nonlinear ICA leads to nonlinear BSS [1], but a counterexample against this conjecture has been given in [7].

independence of e_i is a crucial assumption in LiNGAM. This can be achieved since nonlinear ICA always has solutions. *2.* The matrix $\mathbf{W}$ is sparse enough such that it can be permuted to lower triangularity. This can be enforced by incorporating the L_1 or (G)SCAD penalty on the entries of $\mathbf{W}$. *3.* The nonlinear mapping modeled by the MLP is weak enough such that we just care about the linear causal relations indicated by $\mathbf{W}$. We initialize the system with linear ICA results and use early stopping to ensure this property: $\mathbf{W}$ is initialized by the linear ICA separation matrix, and the initial values for weights in the MLP are very close to 0; early stopping means that we stop the training process once the LiNGAM property holds for $\mathbf{W}$. In addition, some techniques are used to make the mapping from $\mathbf{x}$ to $\mathbf{e}$ as close as possible to linear [15]. Thus, the system in Fig. 1 is learned by penalized maximum likelihood, or by minimizing the mutual information between e_i together with certain penalty terms. After the algorithm terminates, $\frac{\text{var}(h_i(\mathbf{x}))}{\text{var}(e_i)}$ can be used to measure the level of nonlinear distortion, if needed.

4 Data

We aim at discovering the causality network among 14 stocks[2] selected from the Hong Kong stock market. The Hong Kong stock market has some structural features different from the US and UK markets [5]. One typical feature is that the concentration of market activities and equity ownership in relatively small group of stocks, which probably makes causal relations in the Hong Kong stock market more obvious. However, we should be aware that it is probably very hard to discover the causal relations among the selected stocks, since the financial data are somewhat non-stationary, the data generation mechanism is not clear, and there may be many confounder variables [9].

The selected 14 stocks are constituents of Hang Seng Index (HSI).[3] They are almost the largest companies of the Hong Kong stock market. We use the daily dividend/split adjusted closing prices from Jan. 4, 2000 to Jun. 17, 2005, obtained from the Yahoo finance database. For the few days when the stock price is not available, we use the simple linear interpolation to estimate the price. Denoting the closing price of the ith stock on day t by P_{it}, the corresponding return is calculated by $x_{it} = \frac{P_{it} - P_{i,t-1}}{P_{i,t-1}}$. The observed data are $\mathbf{x}_t = (x_{1t}, ..., x_{14,t})^T$. Each return series contains 1331 samples.

5 Empirical Results

We first apply a standard ICA algorithm to perform ICA on the data $\mathbf{x}$. The natural gradient algorithm with the score function adaptively estimated from data is adopted. We use the LiNGAM software to permutate $\mathbf{W}$ and obtain the matrix $\mathbf{B} = \mathbf{I} - \widetilde{\mathbf{W}}'$. $\mathbf{B}$ seems unlikely to be lower-triangular; in fact, the

[2] For saving space, they are not listed here; please see the legend in Fig. 4.

[3] Except that Hang Lung Development Co. Ltd (0010.hk) was deleted from HSI on Dec. 2, 2002.

ratio of the sum of squares of its upper-triangular entries to that of all entries is 0.24, which is very large. We may conclude that the data **x** do not satisfy the LiNGAM model.

Second, we exploit ICA with a sparse separation matrix to do causality discovery. The GSCAD penalty is adopted, and the upper bound for λ is set to 0.25 empirically. We find that the learned separation matrix **W** does not follow LiNGAM for λ less than the upper bound. So again we conclude that the LiNGAM model does not hold for the data **x**.

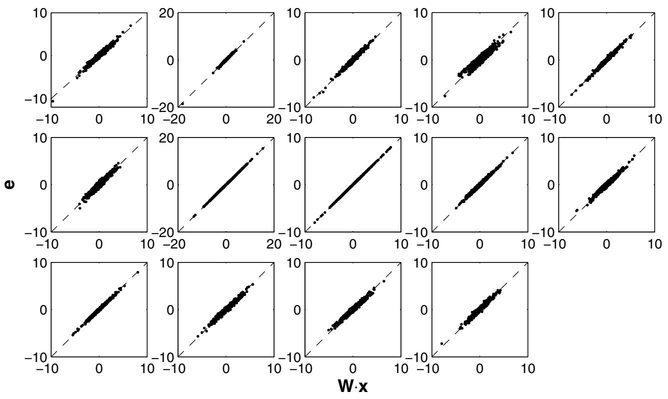


Fig. 3. Scatter plot of each output of the system in Fig. 1 and its linear part. The nonlinear distortion level $\frac{\mathrm{var}(h_i(\mathbf{x}))}{\mathrm{var}(e_i)}$ is 0.0485, 0.0145, 0.0287, 0.2075, 0.0180, 0.0753, 0, 0.0001, 0.0193, 0.0652, 0.0146, 0.0419, 0.0544, and 0.0492, respectively, for the 14 outputs e_i.

Third, we check if the data are generated according to the CWN-LiNGAM model discussed in Section 3.2. We conjecture that the post nonlinearities f_i should be almost the same for all stocks, which is confirmed by the PNL ICA result of the extended-Gaussianization method [14]. But the nonlinear functions g_i estimated by extended-Gaussianization are not smooth. We further adopt the algorithm in [12] to do PNL ICA, but to reduce the model complexity and to achieve a reliable result, all g_i are modeled by the same MLP. The MLP has two layers with eight hidden units. The activation function for hidden units is the 'tansig' function, and that in the output layer is linear. The learned nonlinear function g_i is shown in Fig. 2. It is very interesting that the shape of this function is somewhat similar to the value function in the prospect theory in behavior finance. The matrix **B** estimated from **W** by the LiNGAM software still seems not close to lower-triangular, since the ratio of the sum of squares of its upper-triangular entries to that of all entries is 0.16. However, it is smaller than in the first experiment.

Finally we adopt the method proposed in Section 3.3. The system in Fig. 1 is used to separate the data **x**. The GSCAD penalty is applied to entries of **W** with $\lambda = 0.04$. After 195 epoches, **W** satisfies the LiNGAM assumption and the training process is terminated. Fig. 3 shows the scatter plot of each output e_i and its linear part, from which we can see that the nonlinear distortion is very weak. Based on the learned $\mathbf{W}$, we find the linear causal relations among these stocks, as shown in Fig. 4. This figure was plotted using the LiNGAM software.

From Fig. 4 we have some interesting findings. *1.* The ownership relation tends to cause the causal relation. If A is a holding company of B, there tends to be a causal relation from B to A. There are two significant relations $x_8 \to x_5$ and $x_{10} \to x_1$. In fact, x_5 owns some 60% of x_8, and x_1 holds about 50% of x_{10}. *2.* Stocks belonging to the same subindex tend to be connected together. For example, x_2, x_3, and x_6, which are linked together, are the only three constituents of Hang Seng Utilities Index. x_1, x_9, and x_{11} are constituents of Hang Seng Property Index. *3.* Large bank companies are the cause of many stocks. Here x_5 and x_8 are the two largest banks in Hong Kong. 4. Returns of stocks in Hang Seng Property Index tend to depend on many other stocks, while they hardly influence other stocks. Note that Here x_1, x_9, and x_{11} are in Hang Seng Property Index. Further interpretations of this causal diagram is to be done.

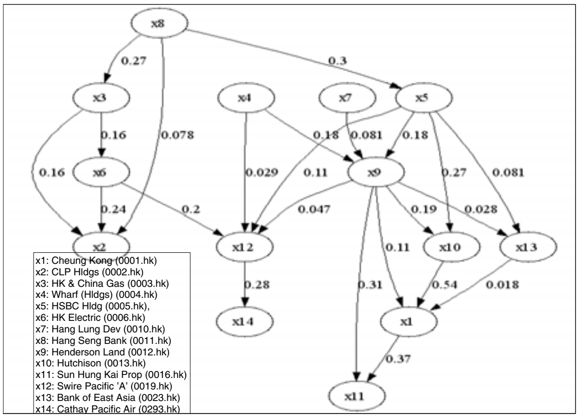

x8
x3
x4
x7
x5
x6
x9
x2
x12
x10
x13
x14
x1
x11
0.27
0.3
0.16
0.078
0.18
0.081
0.18
0.16
0.029
0.11
0.27
0.081
0.24
0.2
0.047
0.19
0.028
0.28
0.11
0.31
0.54
0.018
0.37
x1: Cheung Kong (0001.hk)
x2: CLP Hldgs (0002.hk)
x3: HK & China Gas (0003.hk)
x4: Wharf (Hldgs) (0004.hk)
x5: HSBC Hldg (0005.hk),
x6: HK Electric (0006.hk)
x7: Hang Lung Dev (0010.hk)
x8: Hang Seng Bank (0011.hk)
x9: Henderson Land (0012.hk)
x10: Hutchison (0013.hk)
x11: Sun Hung Kai Prop (0016.hk)
x12: Swire Pacific 'A' (0019.hk)
x13: Bank of East Asia (0023.hk)
x14: Cathay Pacific Air (0293.hk)

Fig. 4. Casual diagram of the 14 stocks

6 Conclusion

We extended the idea of causality discovery by ICA to a wider application area. By incorporating a certain penalty on the ICA separation matrix, ICA with a sparse separation matrix can identify the LiNGAM model when the LiNGAM assumption is not violated much. We further considered two nonlinear models to describe the data generation procedure. If the nonlinear transformation in data generation is component-wise, PNL mixing ICA can be exploited to do causality discovery. The data generation procedure with unknown but mild nonlinearity was then studied. A linear transformation coupled with a MLP was adopted to separate such data. Certain techniques were proposed to ensure that the nonlinearity modeled by the MLP is as weak as possible. Neglecting the nonlinear distortion, linear causality discovery can be easily done. We applied the proposed methods for causality discovery in the Hong Kong stock market, and found that the method with a MLP modeling weak nonlinear distortion behaves very well. The resulting casual diagram revealed some interesting information in the Hong Kong stock market.

References

1. L.B. Almeida. MISEP - linear and nonlinear ICA based on mutual information. *Journal of Machine Learning Research*, 4:1297–1318, 2003.
2. Y. Dodge and V. Rousson. On asymmetric properties of the correlation coefficient in the regression setting. *The American Statistician*, 55(1):51–54, 2001.
3. J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statist. Assoc.*, 96:1348–1360, 2001.
4. M. Granovetter. Business groups. In *Handbook of Economic Sociology*, chapter 18. Princeton University Press, Princeton, 1994.
5. R.Y. Ho, R. Strange, and J. Piesse. The structural and institutional features of the Hong Kong stock market: Implications for asset pricing. Research Paper 027, The Management Centre Research Papers, King's College London, 2004.
6. A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis.* John Wiley & Sons, Inc, 2001.
7. C. Jutten and J. Karhunen. Advances in nonlinear blind source separation. In *Proc. ICA2003*, pages 245–256, 2003. Invited paper in the special session on nonlinear ICA and BSS.
8. T. Khanna and J.W. Rivkin. Interorganizational ties and business group boundaries: Evidence from an emerging economy. Forthcoming in Organization Science, 2006.
9. J. pearl. *Causality: Models, Reasoning, and Inference.* Cambridge University Press, Cambridge, 2000.
10. D.T. Pham and P. Garat. Blind separation of mixture of independent sources through a quasi-maximum likelihood approach. *IEEE Trans. on Signal Processing*, 45(7):1712–1725, 1997.
11. S. Shimizu, P.O. Hoyer, A. Hyvärinen, and A.J. Kerminen. A linear non-Gaussian acyclic model for causal discovery. Submitted to Journal of Machine Learning Research, 2006.
12. A. Taleb and C. Jutten. Source separation in post-nonlinear mixtures. *IEEE Trans. on Signal Processing*, 47(10):2807–2820, 1999.
13. R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1):267–288, 1996.
14. K. Zhang and L.W. Chan. Extended Gaussianization method for blind separation of post-nonlinear mixtures. *Neural Computation*, 17(2):425–452, 2005.
15. K. Zhang and L.W. Chan. Nonlinear ICA with limited nonlinearity. Technical report, The Chinese Univerity of Hong Kong, 2006. To be available soon.

Pricing Options in Hong Kong Market Based on Neural Networks

Xun Liang[1,2], Haisheng Zhang [1], and Jian Yang [1,*]

[1] Institute of Computer Science and Technology, Peking University, Beijing 100871, China
[2] Department of Economics and Operations Research, Stanford University, CA 95035, USA
{liangxun, zhanghaisheng, yangjian, linan}@icst.pku.edu.cn

Abstract. Option pricing is one of the important issues in the financial industry and has been studied for decades. Many classical and successful pricing models have been presented to implement the pricing processing either by numerical computing or by simulation. In this paper, a new option pricing model based on a three-layer feedforward neural network is established to improve the pricing performance. The new model combines 4 traditional pricing models to obtain a better forecasting result based on learning and cutting down their forecasting errors. Numerical experiments are conducted on the data of Hong Kong option market from March 2005 to July 2005. The new model improves the pricing performance remarkably compared to the traditional option pricing models.

1 Introduction

Over the recent years, options have become one of the hottest securities in financial markets. Huge amounts of profits from trading options have attracted many investors. However, the irrational management in stock options sometimes severely jeopardizes the investors' properties. As a result, the skills and tactics of trading stock options remain a mysterious yet provocative topic.

The option price could be affected by many factors, such as the current asset price, the interest rate, the expiration date, the exercise price, and the market risk. Many efficient methods have been presented in the past. Among them, the most successful pricing models [3] include the binomial options (BI) model, the Black-Scholes (BS) model [2], the finite difference (FD) model, and the Monte Carlo (MC) model.

The BI, BS and FD model are classified as the numerical methods, while the MC model is grouped into the simulation method. Each one of them has its own advantages and disadvantages. Accordingly, it is a natural idea to combine the merits of these methods to obtain more accurate prediction results.

At the resurgence of neural networks, many scholars have contributed to the area of applying neural networks in the financial markets [1][6][9]. Due to the complexity of option calculation, the neural network has not been extensively experimented on the option markets until the commencement of this century [5]. The typical ways of making use of the neural network in improving the pricing include calculating the implied volatility and revising the BS model by using neural network. However, fundamentally they are still regarded as traditional methods.

[*] Corresponding author.

The neural network is a universal learner [4] and is capable of extracting rules from the complex systems. Efforts have been made to price the options by learning the trend and rules of the option market based on the current option market features (for example, the sensitivity to risks). The obtained features are "stored" in the neural network as weight values and biases, forming the so-called "knowledge" of current option market. Consequently, the knowledge could be used for predicting more accurate prices.

This paper aims to utilize the nonlinear learning capability of neural network to obtain a better pricing algorithm for option market based on the 4 successful models.

This paper is organized as follows. In section 2, the traditional methods as well as the improved method are illustrated. In section 3 the experiments on predicting the option prices in Hong Kong market are conducted. Section 4 concludes the paper.

2 Model and Methodology

2.1 Traditional Option Pricing Models

As described above, there are 4 widely-used traditional pricing models, the BI model, the BS model, the FD model, and the MC model. The BI model divides the whole time span from the current day to the expiration date into the designated time intervals. After each time interval Δt, the underlying asset price either increases by a certain ratio μ or decreases by a certain ratio d. Then, the current option price could be derived backward from the expiration date to the current date. The BS model supposes that the underlying asset price observes the logarithmic normal distribution. As a result the famous Black-Scholes differential equation could be solved by using the Ito's theorem. Also, the pricing formula could be obtained based on the BS differential equation. The FD model converts the differential equation supported by the underlying asset price to a series of finite equations, based on which the option's current price is obtained. In the MC method, a certain number (for example, 10000) of times of simulations is executed. First, in each time, a sampling path for the movement of the underlying asset's price is formed in a risk-neutral world by using the normal distributed random numbers. Second, the profit and loss of each sampling path are calculated. Third, the mean value of all the profit and loss are discounted back to the current date by the risk-free interest rate, and the current option price is solved. The detail of these 4 traditional models can be found in many books and papers in the field of financial engineering [2][3].

2.2 Methods Based on Traditional Models- Weighted Average Model (WAM)

The simplest way to combine the traditional methods described above is to calculate the average of the 4 methods' results. Suppose that the method with better pricing performance during the past few days would lead to a better pricing result of the current day in sequence. The contribution of each method should be considered by respectively setting a weight value w_i,

$$C = \sum_{i=0}^{3} w_i C_i \,, \tag{1}$$

where C_i is the calculation result of the ith pricing method. Each w_i might be determined by observing the performance of the ith method for pricing a certain option during the past L days, where L is the length of sliding window.

Let $C_r(k)$ be the real option price on the kth day, $C_i(k)$ the forecasting price on the kth day by the ith method, i = 0, ..., 3. Then the overall forecasting error by the ith method is written as

$$d_i = \sum_{k=-L}^{-1} f_d(k) \, | \, C_i(k) - C_r(k) \, | , \tag{2}$$

where $f_d(\cdot)$ is a decay function, such as $-\frac{1}{k}$ or e^k. Clearly, the most recent historical data should be given more attention or weight. The decay function is utilized to simulate the fact that the earlier error results in a smaller effect on the current weight value

Weight w_i could therefore be calculated as

$$w_j = c \frac{\sum_{i=0}^{3} d_i}{d_j} \,, \tag{3}$$

where the constant c normalizes w_j to ensure $\sum_{j=0}^{3} w_j = 1$.

2.3 Neural Network Approach- Multilayer Perceptron (MLP)

The method in subsection 2.2 holds a preliminary capability of "learning" in the meaning that it could revise the pricing result by considering each method's short-term historical performance. Apparently, the neural network could be used to learn their historical performance automatically and more powerfully.

The option market is a complex nonlinear system. If f is set to be the nonlinear multivariate function mapping from the pricing results of the traditional methods to the real option price, then the nonlinear model MLP could be used to fit f nonlinearly.

In the MLP, nonlinear activation functions and a hidden layer provide the nonlinear feature. In this paper a three-layer feedforward nonlinear neural network is employed to learn the pricing processing, as shown in Fig. 1.

Noticing that all the possible values of parameters S, X, k, r, and σ are already incorporated in the predicted prices given by the 4 traditional methods, the assignment of these parameters as the input of the neurons duplicates the functions of them.

Similarly, the five parameters S, X, k, r, and σ are not necessary to be set as the input values of the neurons. Furthermore, it is desired to decrease the prediction errors through learning the errors of 4 traditional models, and the framework of decreasing

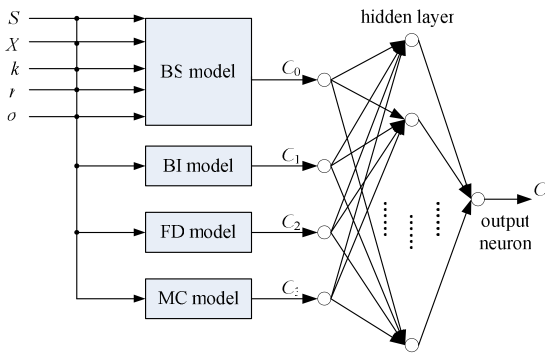


Fig. 1. MLP model

errors by the feedforward neural network is more straightforward and explainable compared with other types of neural networks. Consequently, the feedforward neural network is employed.

In Fig.1, C_i is the forecasted result of the 4 traditional methods, the hidden layer comprises H hidden neurons adopting nonlinear activation functions such as the sigmoidal function, the output neuron also has a nonlinear activation function, and C is the output of the overall network. The training process is implemented by the BP algorithm.

Specifically, the network has 4 input neurons which receive the pricing results of the 4 traditional methods. In addition, the sliding window with length L is selected to form the training patterns. Namely, in each pricing cycle, there are L training patterns corresponding to the past L days' data, and the trained network is used for forecasting the current day's option price.

The errors are used as training pattern as opposed to the forecasting prices. For example, let

$$X(k) = \begin{bmatrix} C_0(k) - A(k) \\ C_1(k) - A(k) \\ C_2(k) - A(k) \\ C_3(k) - A(k) \end{bmatrix}, k = -L, -L+1, \ldots, -1, \tag{4}$$

and

$$Y(k) = C_r(k) - A(k), k = -L, -L+1, \ldots, -1, \tag{5}$$

where $A(k) = \frac{\sum_{i=0}^{3} C_i(k)}{4}$ is the mean value of the pricing results on the kth day before the current day obtained by the 4 traditional methods, $k \in \{-L, -1\}$.

Let

$$X' = \begin{bmatrix} X(-L) & X(-L+1) & \cdots & X(-1) \end{bmatrix}, \tag{6}$$

and

$$Y' = \left[Y(-L) \quad Y(-L+1) \quad \cdots \quad Y(-1) \right]. \tag{7}$$

It follows that $< X', Y' >$ is input to the network.

After the training processing, the current day's experimental sample

$$X'(0) = \begin{bmatrix} C_0(0) - A(0) \\ C_1(0) - A(0) \\ C_2(0) - A(0) \\ C_3(0) - A(0) \end{bmatrix} \tag{8}$$

is input to the network and the output value $Y'(0)$ is obtained. At last, the final forecasted option price is found by adding back the mean value of the pricing results as $Y'(0) + A(0)$.

3 Numerical Experiments

3.1 Data Preparation

The options of Hong Kong securities market are selected as the data in experiments. Specifically, the historical underlying stock asset prices from March 2005 to July 2005 and the historical option prices during the same period of time are used as the experimental data. Totally, there are 62 days' data in all, among which in the offline training, the former 40 days' data are for training, and the latter 22 days' data are for testing. However, in the experiments, the data are utilized more sufficiently. In the stage of forecasting, an online training is also in progress on a rolling basis. Namely, a sliding window of *L* days in length is designed to across the 22 days. The data in the past *L* days continue to train the neural network so as to make the forecast of (*L*+1)th day more precisely.

The call options with code HWL, HSB, CKH and HKB in the Hong Kong option market is selected in experiments. The expiration date is 2005-09-16.The source of all the data can be found on the website of the Hong Kong Exchange and Clearing Limited with http://www.hkex.com.hk/tod/markinfo/setdata.asp. The respective parameters of each option are in Table 1.

Table 1. Parameters of 4 options

option code	underlying stock	exercise price (HK$)
HWL	HUTCHISON, HKSE:0013.HK	70.0
HSB	HANG SENG BANK, HKSE:0013.HK	105.0
CKH	CHEUNG KONG, HKSE:0001.HK	75.0
HKB	HSBC HOLDINGS, HKSE:0005.HK	130.0

3.2 Evaluation Criterion

The purpose of this paper is to improve the pricing result of the 4 traditional methods. The mean absolute error of a method during T days is selected to evaluate,

$$R = \frac{\sum_{t=1}^{T} | C(t) - C_r(t) |}{T} . \tag{9}$$

3.3 Experimental Results

3.3.1 Methods Based on Traditional Model

Firstly, the 4 traditional pricing methods including BI, BS, FD, and MC are implemented by the Java program. The respective parameters of each method are in Table 2.

Table 2. Parameters of 4 traditional methods

model	step length	grid size	risk free rate	max price	simulation times
BI	1/100 year		0.025		
BS			0.025		
FD		100×100	0.025	1.2 × exercise price	
MC	1/100 year		0.025		20000

Based on the pricing result of the 4 traditional methods, the adjusted pricing results for the latter 22 days by using the WAM model are calculated. The mean absolute errors are evaluated. Figs. 2 show the mean of absolute error values in pricing HKB.

In Figs. 2, BI, BS, FD and MC represent the respective absolute errors for option prices of the 4 traditional methods, WAM1, WAM3, and WAM7 are the absolute error for the WAM model with the lengths of sliding windows 1, 3, and 7, respectively. In Fig.3, the means of absolute errors of WAM1, WAM3 and WAM7 are 0.0456, 0.0457 and 0.0457, which are all a little smaller than some of the traditional methods (BI, FD and MC).

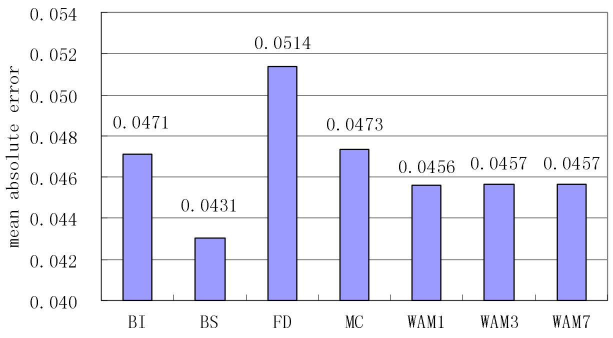


Fig. 2. Mean absolute errors in pricing HKB by the 4 traditional methods and the WAM

3.3.2 Neural Network Approach

The simulation processing is realized by using the Matlab neural network toolbox. The training processing adopts the regularization training function `trainbr`. The training cycle is set to be 100 and the training goal of error is set to be 0.1. The activation function in the MLP is `tansig`. Figs. 3 show the evaluation results the MLP model. In the figures, NN2, NN3, NN5, NN8 and NN12 represent the absolute errors of the MLP model and the respective sliding window lengths are 2, 3, 5, 8 and 12. NN5_3, NN5_5, NN5_8, NN5_12 and NN5_17 represent the results of the MLP model with a sliding window of 5 days' length and the numbers of hidden neurons in each network are 3, 5, 8, 12 and 17 respectively.

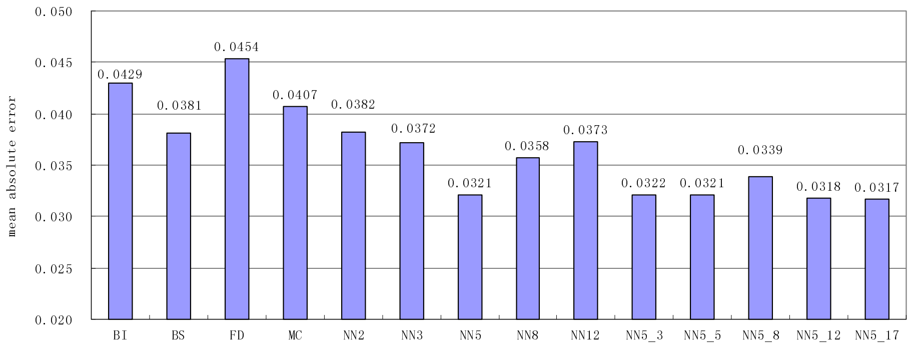


Fig. 3. Mean absolute errors in pricing HKB by neural network approach

From Figs. 3, a conclusion can be drawn that the neutral network methods significantly reduced the forecasting errors for pricing each options. Further, the MLP model with a sliding window of 3 to 5 days in length and 12 hidden neurons could improve the forecasting performance remarkably. As a result, much more profit can be made comparing with the 4 traditional methods.

The experimental results of pricing 4 options HWL, HSB, CKH and HKB are listed below in Tables 3, 4 and 5. From Table 3, it can be seen that compared to the 4 traditional methods, WAM and MLP all improve the precision of the option pricing, especially the MLP method. From Table 4, it is found that the length of the sliding window *L* is around 3 to 5 days. From Table 5, it is concluded that the optimal number of hidden neurons is some 12.

Table 3. Mean absolute errors of different methods

option	BI	BS	FD	MC	WAM	NN3	NN5
HKB	0.043	0.038	0.045	0.041	0.045	0.037	0.032
HWL	0.183	0.187	0.194	0.197	0.190	0.061	0.064
HSB	0.233	0.226	0.230	0.226	0.229	0.043	0.054
CKH	0.244	0.245	0.263	0.255	0.252	0.063	0.057

Table 4. Mean absolute errors of different lengths of sliding windows

option	NN2	NN3	NN5	NN8	NN12
HKB	0.038	0.037	0.032	0.036	0.037
HWL	0.067	0.061	0.064	0.064	0.063
HSB	0.046	0.043	0.054	0.056	0.062
CKH	0.066	0.063	0.057	0.055	0.060

Table 5. Mean absolute errors of different numbers of hidden neurons

option	number of hidden neurons				
	3	5	8	12	17
HKB	0.037	0.041	0.033	0.032	0.032
HWL	0.063	0.064	0.064	0.062	0.064
HSB	0.053	0.054	0.054	0.054	0.053
CKH	0.057	0.057	0.058	0.058	0.057

More experiments are conducted on another 6 options, CLP, HKG, WHL, CPA, HEH, and PCC. The experimental results are similar to the above.

3.4 Discussion

There are two important parameters in the MLP model, the length of sliding window *L* and the number of hidden neurons *H*.

The selection of *L* depends on the features of the specific option, for example, from the aforementioned experiments the best length of sliding window for HSB is 3 days and the best length of sliding window for HKB is 5 days. Furthermore, a short sliding window's length (for example, 2, 3, or 5 days) is selected in this paper for that the experiment with longer history data would not affect the current price remarkably in practice. Moreover, in this paper a "dynamic" training strategy is adopted to update the network continuously. For example, firstly, the 4th day's price is forecasted by training the network with the history data from the 1st to the 3rd days. When the 5th day comes, the network should be trained again with the market data from the 2nd to the 4th days, and the 1st day's data are abandoned.

The problem of determining the number of hidden neurons is difficult [7][8][10]. Too many hidden neurons will lead to a long training process and the possibility of overfitting. However, too few hidden neurons will lead to the loss of accuracy. In this paper, different numbers of hidden neurons are tested and the best ones are found. Although the upper and lower bounds for the number of hidden neurons could be estimated in specific problems, in practice the number of hidden neurons has to be obtained by the trail-and-error method and a developed automatic process could help us to find the best number of hidden neurons.

The experimental results described above show that the forecasting error for pricing options is reduced significantly by adding neutral network training processing to traditional methods. It should be note that the traditional methods implemented here are the traditional standard ones which do not take into consideration any improved methods proposed by researchers, so that the forecasting errors of these traditional methods are comparatively a little higher. However, even if such improved methods are employed instead of the standard ones, since the neutral network approach is established on those traditional methods, the pricing performance could also be improved and more profits could be gained.

4 Concluding Remarks and Future Work

This paper presents a new option pricing model which utilizes a three-layer feedforward neural network. By carefully selecting the model parameters, for example, the number of hidden neurons, and the length of sliding window, the forecasting accuracy could be improved remarkably.

Noticing that the new pricing model in this paper is based on 4 traditional and successful pricing models, known as the binomial tree model, the Black-Scholes model, the finite differential model and the Monte Carlo model, the result in this paper does not surprise us - the neural-network-based method takes advantages from the other 4 methods. However, the improvement does support the financial professionals to make more profits in the option markets.

In addition, the pricing model in this paper explores a new option pricing approach from the machine learning point of view. It highlights a new option pricing framework by automatically learning the complex rules from the performance of the traditional models as well as the option markets.

It should be acknowledged that the neural network pricing method in this paper is not a substitute for the other 4 methods. Instead, it is one of the better and improved methods over the traditional ones. Namely, in the meaning of statistics, the method in this paper predicts more closely to the market as long as the forecasting results are given first by the 4 traditional methods. Clearly, all methods are naturally complementary and a proper utilization of their main strengths and weaknesses should lead to synergetic effects beneficial to their common goals.

The work in this paper is still fairly preliminary. Future work includes conducting experiments on more data from the worldwide option markets, characterizing the method with more statistical features, studying the case in which improved methods are adopted instead of the standard ones, as well as incorporating the option pricing knowledge hinted by the 4 traditional methods more meticulously into the hidden layer of the neural network.

Acknowledgments

The Project sponsored by NSF of China under grant number 70571003 and SRF for ROCS, SEM under grant number 4131522.

References

1. Amilon, H.: Neural network versus Black-Scholes: a comparison of pricing and hedging performances. Journal of Forecast 22 (2003) 317-335
2. Black, F., Scholes, M.: The pricing of options and corporate liabilities. Journal of Political Economy 81 (1973) 637-654

3. Hull, C.J.: Options, Future and other Derivatives. Prentice Hall, 3rd Edition (2004)
4. Kolmogorov, A.N.: On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. Dokl. Akad. Nauk USSR 6 (1957) 953-956
5. Lajbcyg, P.: Improving option pricing with the product constrained hybrid neural network. IEEE Trans on Neural Networks 15 (2004) (2) 465-476
6. Lajbcygier, P.R., Jerome, T.C.: Improved option pricing using artificial neural networks and bootstrap methods. International Journal of Neural Systems 8 (1997) 457-471
7. Liang, X., Xia, S.: Methods of training and constructing multilayer perceptrons with arbitrary pattern sets. International Journal of Neural Systems 6 (1995) (3) 233-247
8. Liang, X., Wang, X.: Information crosswise propagation (CP) learning in multilayer perceptrons - a novel way to neuron pruning. In: Proc of World Congress on Neural Networks Washington D.C. 1 (1995) 642-645
9. Zapart, A.C.: Beyond Black Scholes: a neural networks-based approach to options pricing. International Journal of Theoretical and Applied Finance 6 (2003) 469-489
10. Zurada, J.M.: Artificial Neural Systems. West Publishing Company (1992)

Global Optimization of Support Vector Machines Using Genetic Algorithms for Bankruptcy Prediction

Hyunchul Ahn[1], Kichun Lee[2], and Kyoung-jae Kim[3]

[1] Graduate School of Management, Korea Advanced Institute of Science and Technology, 207-43 Cheongrangri-Dong, Dongdaemun-Gu, Seoul 130-722, Korea
hcahn@kaist.ac.kr
[2] R&D Center, Samsung Networks Inc., 8F, ASEM Tower, World Trade Center, 159-1, Samseong-Dong, Kangnam-Gu, Seoul 135-798, Korea
skylee1020@gmail.com
[3] Department of Management Information Systems, Dongguk University, 3-26, Pil-Dong, Chung-Gu, Seoul 100-715, Korea
kjkim@dongguk.edu

Abstract. One of the most important research issues in finance is building accurate corporate bankruptcy prediction models since they are essential for the risk management of financial institutions. Thus, researchers have applied various data-driven approaches to enhance prediction performance including statistical and artificial intelligence techniques. Recently, support vector machines (SVMs) are becoming popular because they use a risk function consisting of the empirical error and a regularized term which is derived from the structural risk minimization principle. In addition, they don't require huge training samples and have little possibility of overfitting. However, in order to use SVM, a user should determine several factors such as the parameters of a kernel function, appropriate feature subset, and proper instance subset by heuristics, which hinders accurate prediction results when using SVM. In this study, we propose a novel approach to enhance the prediction performance of SVM for the prediction of financial distress. Our suggestion is the simultaneous optimization of the feature selection and the instance selection as well as the parameters of a kernel function for SVM by using genetic algorithms (GAs). We apply our model to a real-world case. Experimental results show that the prediction accuracy of conventional SVM may be improved significantly by using our model.

1 Introduction

Prediction of corporate bankruptcies has long been an important topic and has been studied extensively in the finance and management literature because it is an essential basis for the risk management of financial institutions. Bankruptcy prediction models have used various statistical and artificial intelligence techniques. These techniques include discriminant analysis, logistic regression, the decision tree, *k*-nearest neighbor, and backpropagation (BP) neural network. Among them, the BP network (BPN) has become one of the most popular techniques for the prediction of corporate bankruptcy due to its high prediction accuracy. However, many financial companies still have difficulties in using BPN. The difficulty stems from inherent limitations of

BPN such as the requirement of large data samples, the possibility of overfitting, and poor explanatory power for the results.

Support vector machines (SVMs) may be an alternative to relieve these limitations of BPN [12]. General BPN models implement the empirical risk minimization principle for seeking to minimize the misclassification error or deviation from the correct solution of the training data. However, SVM implements the structural risk minimization principle for searching to minimize an upper bound of generalization error. In addition, the solution of SVM may be the global optimum, while BPN models may tend to fall into a local optimal solution. Therefore, overfitting of the results is unlikely to occur with SVM. Consequently, several recent studies for bankruptcy prediction used SVM as a classifier, and they showed that it might be an effective technique for predicting corporate financial distress [4,16,17,22].

However, SVM also has some factors that affect the prediction performance – these factors are usually set by heuristics. In particular, the selection of an appropriate kernel function and its parameters (e.g. C, d, δ^2) and the selection of proper feature subset in SVM have been popular research topics. Other than these factors, the selection of appropriate instance selection (in other words, prototype selection) may also improve the classification accuracy of SVM by eliminating irrelevant and distorting training samples. Nonetheless, there have been few studies that have applied instance selection to SVM, especially in the domain of bankruptcy prediction.

Thus, in this study, we propose a novel hybrid SVM classifier with simultaneous optimization of feature subsets, instance subsets, and kernel parameters. This study introduces genetic algorithms (GAs) to optimize the feature selection, instance selection, and kernel parameters simultaneously. Our study applies the proposed model to the real-world case for bankruptcy prediction, and presents experimental results from the application.

2 Prior Studies

In this study, we propose the combined model of two artificial intelligence techniques, SVM and GA for effective bankruptcy prediction. Thus, in this section, we first review the basic concepts of SVM and GA, which are the core algorithms of our model. After that, we introduce prior studies that attempt to optimize SVM using GA.

2.1 Support Vector Machine (SVM)

SVM uses a linear model to implement nonlinear class boundaries by nonlinear mapping of the input vectors x into the high-dimensional feature space. A linear model constructed in the new space can represent a nonlinear boundary in the original space. In the new space, an optimal separating hyperplane is constructed [25].

Thus, SVM is known as the algorithm that finds a special kind of linear model, the *maximum margin hyperplane*. The maximum margin hyperplane gives the maximum separation between the decision classes. The training examples that are closest to the maximum margin hyperplane are called *support vectors*. All other training examples are irrelevant for defining the binary class boundaries.

SVM constructs a linear model to implement nonlinear class boundaries through the transformation of the inputs into the high-dimensional feature space. The function, $K(\mathbf{x_i},\mathbf{x_j})$, which is called 'kernel function', does this work. There are some different kernels for generating the inner products to construct machines with different types of nonlinear decision surfaces in the input space. Choosing among different kernels the model that minimizes the estimate, one chooses the best model. Common examples of the kernel function are the polynomial kernel $K(\mathbf{x_i},\mathbf{x_j})= (1+ \mathbf{x_i}^T\mathbf{x_j})^d$ and the Gaussian radial basis function (RBF) $K(\mathbf{x_i},\mathbf{x_j})= \exp(-1/\delta^2(\mathbf{x_i} - \mathbf{x_j})^2)$ where d is the degree of the polynomial kernel and δ^2 is the bandwidth of the Gaussian RBF kernel [11].

As mentioned above, BPN has been widely used in the area of financial forecasting because of its broad applicability to many business problems and preeminent learning ability. On the other hand, there are no parameters to tune except the upper bound C for the non-separable cases in linear SVM [3]. Overfitting is also unlikely to occur with SVM. Overfitting may be caused by too much flexibility in the decision boundary, but the maximum hyperplane is relatively stable and gives little flexibility [26].

Although SVM has the above advantages, there are a few studies on the application of SVM in financial forecasting. Mukherjee et al. [18] showed the applicability of SVM to time-series forecasting. Tay and Cao [24] examined the predictability of financial time-series with SVMs. They showed that SVMs outperformed the BPNs on the criteria of normalized mean square error, mean absolute error, directional symmetry and weighted directional symmetry. Kim [11] applied SVM to predicting the future direction of the stock price index. In his study, SVM outperformed BPN and case-based reasoning for the prediction of the stock price index. Recently, several studies investigated the efficacy of applying SVM to bankruptcy prediction. Fan and Palaniswami [4] showed that SVM outperformed traditional classifiers for bankruptcy prediction such as DA, multi-layer perceptron, and learning vector quantization. Shin et al. [22] pointed out that the accuracy and generalization performance of SVM were better than those of BPN as the training set size got smaller. Min and Lee [16] showed that SVM outperformed LOGIT, DA, and BPN for bankruptcy prediction.

2.2 Genetic Algorithm (GA)

The genetic algorithm is a popular optimization method that attempts to incorporate ideas of natural evolution. Its procedure improves the search results by constantly trying various possible solutions with some kinds of genetic operations. In general, the process of GA proceeds as follows.

First of all, GA generates a set of solutions randomly that is called an initial population. Each solution is called a chromosome and it is usually in the form of a binary string. After the generation of the initial population, a new population is formed that consists of the fittest chromosomes as well as offspring of these chromosomes based on the notion of survival of the fittest. The value of the fitness for each chromosome is calculated from a user-defined function. Typically, classification accuracy (performance) is used as a fitness function for classification problems.

In general, offspring are generated by applying genetic operators. Among various genetic operators, selection, crossover and mutation are the most fundamental and popular operators. The selection operator determines which chromosome will survive.

In crossover, substrings from pairs of chromosomes are exchanged to form new pairs of chromosomes. In mutation, with a very small mutation rate, arbitrarily selected bits in a chromosome are inverted. These steps of evolution continue until the stopping conditions are satisfied [5,6].

2.3 Optimization of SVM Using GA

Until now, researchers have studied optimization of SVM using GA in three ways. First, some studies have tried to optimize 'the kernel function and its parameters'. For example, Pai and Hong [19] used GA to optimize the free parameters used in the kernel function of SVM. The SVM model of their study used Gaussian RBF as the kernel function, and they designed the proposed system to optimize C, δ^2, ε parameters using GA. Howley and Madden [8] extended the area of optimization. Their proposed model optimized the kernel parameters (e.g. C, δ^2, d, ε) as well as the kernel function itself. Consequently, their model could present a globally optimized kernel function and its optimized parameters.

The second approach of GA-optimization of SVM is 'feature subset selection'. Feature subset selection is a method that uses only a small subset of features that prove to be relevant to the target concept. In most classification problems, the selection of an appropriate feature subset is important because it enhances classification performance by characterizing each sample more accurately, and it also reduces computational requirements. Thus, many researchers have tried to optimize the input features of SVM by using GA. For example, Lee and Byun [14] and Sun et al. [23] used this technique for image identification, and Li et al. [15] used it for cancer detection. In addition, this technique is adopted in various application areas including gear fault detection [21], abnormal key stroke detection [27], direct marketing [28], and bankruptcy prediction [17].

The final approach is 'simultaneous optimization of kernel parameters and feature subset selection'. As mentioned above, both kernel parameters and feature selection affects the classification performance of SVM. Thus, it may be more effective to optimize these factors simultaneously. Nonetheless, this is still an undiscovered area, so there are few related studies. Jack and Nandi [9] applied this technique for machinery fault detection, and Kim et al. [10] used it for network intrusion detection. Zhao et al. [29] proposed this approach to enhance protein sequence classification.

Although there have been many prior studies that optimized the various factors of SVM using GA, there is another factor to be optimized – optimal instance selection. Instance selection is the technique that selects an appropriate reduced subset of the training samples and only uses the selected subset for training. This prevents the distorted training of SVM by reducing the possibility of selecting noisy training samples as the support vectors, so it may improve classification accuracy of SVM. Due to its advantages, it has been applied to various classification techniques including neural networks [20] and case-based reasoning [1]. However, there has been no study that has introduced instance selection using GA for SVM, as far as we know. Thus, in this study, we propose a global optimization model that optimizes the selection of features, instances, and kernel parameters simultaneously by using GA.

3 Simultaneous Optimization of SVM Using GA

This study proposes a novel SVM model whose feature selection, instance selection, and kernel parameter settings are globally optimized, in order to improve prediction accuracy of typical SVM. We employ GA to optimize these factors simultaneously. Hereafter, we call our model SOSVM - Simultaneous Optimization of SVM using GA. The detailed explanation for each step of SOSVM is presented as follows.

Phase 1. Initiation

In the first step, the system generates the initial population that would be used to find global optimum factors – feature and instance selection variables, and kernel parameters. The values of the chromosomes for the population are initiated into random values before the search process. To enable GA to find the optimal factors, we should design the structure of a chromosome as a form of binary strings. Each chromosome for SOSVM has all the information for feature selection, instance selection, and kernel parameter settings. The length of each chromosome is m+n+12 bits when m is the number of features and n is the number of instances. The values of the codes for feature selection and instance selection are set to '0' or '1'. '0' means the corresponding feature or instance is not selected and '1' means it is selected. The sign for feature and instance selection needs just 1 bit. As a result, m+n bits are just required to implement feature and instance selection by GA. The remaining 12 bits are used for selecting appropriate kernel parameters. Similar to the study by Pai and Hong [19], we use the Gaussian radial basis function (RBF) as the kernel function of SVM. Tay and Cao [24] showed that the upper bound C and the kernel parameter δ^2 play an important role in the performance of SVM using Gaussian RBF. Setting these two parameters improperly can cause overfitting or underfitting problems. Thus, SOSVM tries to optimize these parameters using GA, and it assigns 6 bits to represent each variable. Thus, 12 bits in total are used for setting C and δ^2.

Phase 2. Training

After generating the initial population, the system performs a typical SVM process using the assigned value of the factors in the chromosomes, and calculates the performance of each chromosome. The performance of each chromosome can be calculated through the fitness function for GA. In this study, the main goal is to find the optimal or near optimal parameters that produce the most accurate prediction solution. Thus, we set the fitness function for the test data set to the prediction accuracy of the test dataset [5,12,13].

Phase 3. Genetic operation

In the third step, a new generation of the population is produced by applying genetic operators such as selection, crossover, and mutation. According to the fitness values for each chromosome, the chromosomes whose values are high are selected and used for the basis of crossover. The mutation operator is also applied to the population with a very small mutation rate.

After the production of a new generation, phase 2 – the training process with calculation of the fitness values – is performed again. From this point, phase 2 and phase 3 are iterated again and again until the stopping conditions are satisfied. When the stopping conditions are satisfied, the genetic search finishes and the chromosome that shows the best performance in the last population is selected as the final result.

Phase 4. Checking generalizability

Occasionally, the optimized parameters determined by GA fit quite well with the test data, but they don't fit well with the unknown data. The phenomenon occurs when the parameters fit too well with the given test data set. Thus, in the last stage, the system applies the finally selected parameters – the optimal selections of features and instances, and the optimal kernel parameters – to the hold-out (unknown) data set in order to check the generalizability of the determined factors.

4 The Research Design and Experiments

4.1 Application Data

The application data used in this study consists of financial ratios and the status of bankruptcy or non-bankruptcy for corresponding corporate. The data was collected from one of the largest commercial banks in Korea. The sample of bankrupt companies was 774 companies in heavy industry that filed for bankruptcy between 1999 and 2002. There were also 774 non-bankrupt companies from the same industry and period. Thus, the total size of the sample was 1548 companies.

The financial status for each company is categorized as "0" or "1" and it is used as a dependent variable. "0" means that the corporation is bankrupt, and "1" means that the corporation is solvent. For independent variables, we first generate 162 financial ratios from the financial statement from each company. Finally, we get 41 financial ratios as independent variables through the two independent sample t-test, the forward selection procedure based on logistic regression, and the opinions of the experts who are responsible for approving and managing loans in the bank. We split the data into three groups: training, test, and hold-out datasets. The portion of these groups is 60% (928 companies), 20% (310 companies) and 20% (310 companies) each.

4.2 Comparative Models

To test the effectiveness of the proposed model, we compare the result of SOSVM to the results of four different models. The first model, labeled COSVM (COnventional SVM), uses the conventional approach of SVM. This model considers all initially available features as a feature subset. That is to say, there is no special process of feature subset selection. In addition, instance selection is not considered here, so all instances are used in this model. The kernel parameters in this model are determined by varying their values to select optimal values that produce the best prediction performance.

The second model determines the optimal kernel parameters by applying GA. We call this model KPSVM (Kernel Parameter optimization for SVM by GA). Similar to COSVM, KPSVM also does not contain any function of feature selection or instance selection. Pai and Hong [19] proposed a similar model.

The third model selects relevant features using GA. This model is called FSSVM (Feature Selection for SVM by GA). Here, we try to optimize feature selection and kernel parameters by GA, but we are still unconcerned with instance selection. The studies by Jack and Nandi [9], Kim et al. [10], and Zhao et al. [29] are the examples that used this model.

The final model uses GA to select a relevant instance subset. This model is called ISSVM (Instance Selection for SVM by GA). In this model, we try to optimize instance selection and kernel parameters by GA, but we are unconcerned with feature selection.

4.3 Research Design and System Development

For the controlling parameters of GA search for ISSVM and SOSVM, the population size was set at 200 organisms and the crossover and mutation rates were set at 70% and 10%. As the stopping condition, 100 generations were permitted. However, the genetic search space of KPSVM and FSSVM is much smaller than the space of ISSVM and SOSVM. Thus, we assigned 100 organisms for the population, and set the mutation rate at 15% in the case of KPSVM and FSSVM.

These experiments are done by our private experimental software that is designed to perform SVM training by using parameters optimized by GA. This software is developed on a Java platform, and the class for SVM training is programmed using LIBSVM, a public software for SVM [2].

5 Experimental Results

In this section, the prediction performances of SOSVM and other alternative models are compared. Table 1 describes the average prediction accuracy of each model. As shown in Table 1, SOSVM achieves the higher prediction accuracy than COSVM, KPSVM, FSSVM, and ISSVM by 5.16%, 3.23%, 2.26%, and 0.33% for the hold-out data. Comparing the performance of FSSVM and ISSVM, we can find that ISSVM outperforms FSSVM by 1.93%. It may be understood that appropriate instance selection is more important than feature selection for improving prediction accuracy of SVM.

Table 1. Average prediction accuracy of the models

Model	Train	Test	Hold-out	Kernel parameter	F#[a)]	I#[b)]
COSVM	82.65%	-	74.52%	C=100, δ^2=25	41	928
KPSVM	84.81%	77.42%	76.45%	C=55.88, δ^2=13.04	41	928
FSSVM	82.54%	77.74%	77.42%	C=93.29, δ^2=8.76	25	928
ISSVM	84.55%	80.00%	79.35%	C=36.82, δ^2=12.65	41	492
SOSVM	81.46%	81.94%	79.68%	C=80.94, δ^2=25.85	39	701

[a)] The number of selected features, [b)] The number of selected instances

We use the two-sample test for proportions to examine whether the differences of prediction accuracy between SOSVM and other comparative algorithms are statistically significant. By applying this test, it is possible to check whether there is a difference between two probabilities when the prediction accuracy of the left-vertical methods is compared with the right horizontal methods [7]. In this test, the null hypothesis is H_0: $p_i - p_j = 0$ where i=1,…,4 and j=2,…,5, while the alternative hypothesis is H_a: $p_i - p_j > 0$ where i=1,…,4 and j=2,…,5. p_k means the classification performance of the kth method. Table 2 shows *Z values* for the pairwise comparison of the performance of the models.

Table 2. Z values of the two sample test for proportions

	KPSVM	FSSVM	ISSVM	SOSVM
COSVM	0.560	0.846*	1.430**	1.529**
KPSVM		0.286	0.871*	0.971*
FSSVM			0.585	0.685
ISSVM				0.100

* significant at the 10% level, ** significant at the 5% level

As shown in Table 2, SOSVM is better than COSVM at the 5% and better than KPSVM at the 10% statistical significance level. But, SOSVM does not outperform FSSVM and ISSVM with statistical significance.

6 Concluding Remarks

We have proposed a new hybrid SVM model using GA called SOSVM. Our proposed model optimizes feature selection, instance selection, and kernel parameters simultaneously. Although GA-optimization models for feature selection and kernel parameter selection of SVM have been suggested in the literature, our proposed model is designed to include 'instance selection', which reduces distorted training samples that may lead erroneous prediction. Compared to other models such as COSVM, KPSVM and FSSVM, SOSVM as well as ISSVM showed higher prediction accuracy in the empirical test for real-world bankruptcy prediction. Thus, instance selection seems to be very important for improving classification accuracy of SVM in our experiment. It is quite difficult to theoretically show the reason that instance selection in SVM improves classification performance. However, we just anticipate that appropriate instance selection may help SVM to find more proper support vectors by eliminating misleading training samples near the classifying hyperplane.

However, this study has some limitations. First of all, our model requires a high level of computational resources. Similar to other GA-based optimization models, SOSVM iterates the SVM training process whenever genetic evolution occurs. In particular, the search space of our model is very large, so it takes more time to get enough training. Consequently, the efforts to make SOSVM more efficient should be followed in the future. Second, the generalizability of SOSVM should be tested further. Although we apply this model to bankruptcy prediction, SOSVM can be applied to any domain that requires accurate prediction. Moreover, SOSVM did not outperform most comparative models with statistical significance in our experiment because of the insufficient sample size. Thus, it is necessary to validate the general applicability of SOSVM by applying it to other problem domains in the future.

References

1. Babu, T.R., Murty, M.N.: Comparison of genetic algorithm based prototype selection schemes. Pattern Recognition, 34(2) (2001) 523-525
2. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm (2001)

3. Drucker, H., Wu, D., Vapnik, V.N.: Support vector machines for spam categorization. IEEE Transactions on Neural Networks, 10(5) (1999) 1048-1054
4. Fan, A., Palaniswami, M.: Selecting bankruptcy predictors using a support vector machine approach. Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (2000) 354-359
5. Fu, Y., Shen, R.: GA based CBR approach in Q&A system. Expert Systems with Applications, 26(2) (2004) 167-170
6. Han, J., Kamber, M.: Datamining: Concepts and Techniques. Morgan Kaufmann Publishers. San Francisco, CA (2001)
7. Harnett, D.L., Soni, A.K. Statistical methods for business and economics. Addison-Wesley. Massachusetts, MA (1991)
8. Howley, T., Madden, M.G.: The Genetic Kernel Support Vector Machine: Description and Evaluation. Artificial Intelligence Review, 24(3-4) (2005) 379 - 395
9. Jack, L.B., Nandi, A.K.: Fault detection using support vector machines and artificial neural networks, augmented by genetic algorithms. Mechanical Systems and Signal Processing, 16(2-3) (2002) 373-390
10. Kim, D.S., Nguyen, H.-N., Park, J.S.: Genetic algorithm to improve SVM based network intrusion detection system. Proceedings of the 19th International Conference on Advanced Information Networking and Applications (2005) 155-158
11. Kim, K.: Financial forecasting using support vector machines. Neurocomputing, 55(1-2) (2003) 307-319
12. Kim, K.: Toward global optimization of case-based reasoning systems for financial forecasting. Applied Intelligence, 21(3) (2004) 239-249
13. Kim, K.: Artificial neural networks with evolutionary instance selection for financial forecasting. Expert Systems with Applications, 30(3) (2006) 519-526
14. Lee, K., Byun, H.: A New Face Authentication System for Memory-Constrained Devices. IEEE Transactions on Consumer Electronics, 49(4) (2003) 1214-1222
15. Li, L., Tang, H., Wu, Z., Gong, J., Gruidl, M., Zou, J., Tockman, M., Clark, R.A.: Data mining techniques for cancer detection using serum proteomic profiling. Artificial Intelligence in Medicine, 32(2) (2004) 71-83
16. Min, J.H., Lee, Y.-C.: Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters. Expert Systems with Applications, 28(4) (2005) 603-614
17. Min, S.-H., Lee, J., Han, I.: Hybrid genetic algorithms and support vector machines for bankruptcy prediction. Expert Systems with Applications, (2006) Forthcoming
18. Mukherjee, S., Osuna, E., Girosi, F.: Nonlinear prediction of chaotic time series using support vector machines. Proceedings of the IEEE Workshop on Neural Networks for Signal Processing (1997) 511-520
19. Pai, P.-F., Hong, W.-C.: Forecasting regional electricity load based on recurrent support vector machines with genetic algorithms. Electric Power Systems Research, 74(3) (2005) 417-425
20. Reeves, C.R., Taylor, S.J.: Selection of training sets for neural networks by a genetic algorithm. In Eiden, A.E., Back, T., Schoenauer M., Schwefel, H.-P.: Parallel problem-solving from nature-PPSN V. Springer. Berlin (1998)
21. Samanta, B.: Gear fault detection using artificial neural networks and support vector machines with genetic algorithms. Mechanical Systems and Signal Processing, 18(3) (2004) 625-644
22. Shin, K.-S., Lee, T.S., Kim, H.-j.: An application of support vector machines in bankruptcy prediction model. Expert Systems with Applications, 28(1) (2005) 127-135

23. Sun, Z., Bebis, G., Miller, R.: Object detection using feature subset selection. Pattern Recognition, 37(11) (2004) 2165-2176
24. Tay, F.E.H., Cao, L.: Application of support vector machines in financial time series forecasting. OMEGA: The International Journal of Management Science, 29(4) (2001) 309-317
25. Vapnik, V.N.: Statistical Learning Theory. Wiley. New York (1998)
26. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann Publishers. San Francisco, CA (2000)
27. Yu, E., Cho, S.: Keystroke dynamics identity verification: its problems and practical solutions. Computers & Security, 23(5) (2004) 428-440
28. Yu, E., Cho, S.: Constructing response model using ensemble based on feature subset selection. Expert Systems with Applications, 30(2) (2006) 352-360
29. Zhao, X.-M., Cheung, Y.-M., Huang, D.-S.: A novel approach to extracting features from motif content and protein composition for protein sequence classification. Neural Networks, 18(8) (2005) 1019-1028

Neural Networks, Fuzzy Inference Systems and Adaptive-Neuro Fuzzy Inference Systems for Financial Decision Making

Pretesh B. Patel and Tshilidzi Marwala

School of Electrical and Information Engineering, University of the Witwatersrand, Private Bag 3, 2050, Johannesburg, South Africa
p.patel@ee.wits.ac.za, t.marwala@ee.wits.ac.za
http://dept.ee.wits.ac.za/~marwala/

Abstract. This paper employs pattern classification methods for assisting investors in making financial decisions. Specifically, the problem entails the categorization of investment recommendations. Based on the forecasted performance of certain indices, the Stock Quantity Selection Component is to recommend to the investor to purchase stocks, hold the current investment position or sell stocks in possession. Three designs of the component were implemented and compared in terms of their complexity as well as scalability. Designs that utilized 1, 4 and 16 classifiers, respectively, were developed. These designs were implemented using Artificial Neural Networks, Fuzzy Inference Systems as well as Adaptive Neuro-Fuzzy Inference Systems. The design that employed 4 classifiers achieved low complexity and high scalability. As a result, this design is most appropriate for the application of concern.

1 Introduction

Pattern recognition could be defined as the study of the ability of machines to observe the environment, learn to differentiate between patterns of interest from their backgrounds and formulate reliable as well as sensible decisions about the categories of the patterns [1]. This is a complex task that is an innate ability for humans. However, to develop a system to solve such problems poses formidable research challenges.

This research focuses on a pattern classification problem utilized within an application that could assist individual as well as institutional investors in making financial decisions. It is anticipated that this application would be used in conjunction with other financial analysis methodologies. As a result, such an application should be employed to confirm an investment decision.

Pattern classification is the process of assigning an input pattern to one of a predefined set of classes. It consists of developing a functional relationship between the input features and the target classes. Accurately estimating such a relationship is vital to the success of a classifier. Specifically, the quantity of stocks or shares to be purchased based on the forecasted performance of certain indices is the pattern classification problem. The Dow Jones Industrial Average, Johannesburg Stock Exchange or the JSE Securities Exchange (JSE) All Share, Nasdaq 100 and Nikkei 225 Stock Average indices are considered. However, the computational intelligent techniques as well as their implementation methodology utilized in this research could be adapted for decision making systems in other industry sectors.

The classification of data into various classes has been an important research area for many years. Artificial neural networks (ANNs) have been applied to pattern classification [2]. Research has also been conducted on fuzzy classification. This resulted in many algorithms, such as fuzzy K-nearest neighbour [3] and fuzzy c-means [4], being applied to decision making systems. Fuzzy systems constructed using genetic algorithms have been utilized [5][6]. Fuzzy neural networks have also been employed in pattern classification applications [7][8]. Support Vector Machines have been applied to multi-category classification problems [9]. These classification tasks have also been implemented by combining multiple simpler specialized classifiers [10][11].

In this research, artificial neural network (ANN) architectures, Fuzzy Inference Systems (FISs) as well as Adaptive Neuro-Fuzzy Inference Systems (ANFISs) have been considered. Specifically, the Multi-Layer Perceptron (MLP) and the Radial Basis Function (RBF) neural network architectures have been considered. FISs developed employed subtractive clustering to generate the required membership functions and set of fuzzy inference rules. Information on these computational intelligent techniques can be found in [12], [13] and [14], respectively.

The next section briefly examines the application of concern. Thereafter, the implementation methodology is described. The paper concludes with the comparison of the various models developed and the selection of the superior classifiers.

2 The Developed System

The developed system is to be used in assisting an investor in making financial decisions. As a result, the system should be based on a profitable trading strategy. There are numerous trading strategies available. This research focuses on the "Buy low, sell high" trading strategy. The strategy has been implemented as well as compared to the "Buy and hold" trading strategy in terms of profits generated.

The "Buy low, sell high" trading strategy entails purchasing certain stocks at a low price and selling these stocks when the price is high. The "Buy and hold" trading strategy, as the name suggests, involves an investor purchasing certain stocks and retaining them for a particular duration. The method used to implement the "Buy low, sell high" trading strategy involved classifying the change in index or delta into certain categorizes. Delta is defined as the difference between the closing index value for the next day and the closing index value for the previous day. This functionality has been implemented within the Forecasting Component (FC). Depending on the classification of this component, the strategy would recommend the investor to purchase stocks, hold the current investment position or sell stocks. This responsibility can be found in the Stock

Quantity Selection Component (SQSC). Table 1 illustrates the forecasted classes as well as the corresponding investment recommendation.

It has been determined that the "Buy low, sell high" trading strategy, with the percentage threshold combination of 0.8% and -0.20% of the closing value for the previous day, is most profitable. As a result, the system has been based on this trading strategy. Further information on the comparison of the 2 trading strategies considered can be found in [15].

Pattern classification problems can be grouped as either dichotomous or polychotomous problems. Dichotomous classification can be interpreted as 2-class classification problems, whereas polychotomous classification involves problems with more than 2 classes to be categorized. The SQSC module is the center of this research. Based on the forecasted performance of the closing price of the index, the component is to recommend the investor to purchase stocks, hold the current investment position or sell stocks. It is evident that this is a polychotomous classification problem as there are more than 2 classes. Further information on the FC module can be found in [15].

Various classifier designs of the SQSC module were considered. Each of these designs were developed using both ANNs as well as fuzzy logic techniques. The first design employed 1 classifier. This classifier consisted of 16 inputs and 16 outputs. The inputs to the model are the forecasted performance of the closing price of the indices considered. The outputs of the classifier are the investment recommendations for the indices. The second design involved 4 classifiers. Each classifier has 4 inputs and 4 outputs. Each classifier is used to generate an investment recommendation for an index considered. The input to a classifier is the forecasted performance of the closing price of an index. The output of a classifier is the investment recommendation for the index. The third and final design considered utilized 16 classifiers. Each classifier has 4 inputs and 1 output. Each classifier is employed to categorize whether or not to execute an investment recommendation. The input to a classifier is the same as design 2 above. The outputs of the classifiers are fed into an interpretation function that generates the final investment recommendations for the indices. This design has been implemented to investigate the method of utilizing simpler classifiers to generate a multi-category classifier.

3 Implementation Methodology

The data used to develop the SQSC module has been generated based on the 4 forecasted closing price performance classes illustrated in Table 1. The development process was divided into various stages.

The following procedure has been pursued in the creation of the various classifiers employed:

1. Selection and processing of data to be used by the classifiers during training, validation and testing.
2. Optimization of the classification threshold of the various classes to be categorized.

Table 1. The "Buy low, sell high" trading strategy categorizes

Class	Requirement	Investment recommendation
Large Rise (LR)	Delta > Positive threshold percentage of previous day closing price.	If LR is forecasted for the next day, sell stocks in possession.
Slight Rise (SR)	0 < Delta <= Positive threshold percentage of previous day closing price.	If SR is forecasted for the next day, hold current investment position.
Slight Drop (SD)	Negative threshold percentage of previous day closing price <= Delta <= 0.	If SD is forecasted for the next day, buy stocks to the value of 15 % of available trading capital.
Large Drop (LD)	Delta < Negative threshold percentage of previous day closing price.	If LD is forecasted for the next day, buy stocks to the value of 25 % of available trading capital.

3. Optimization of the classifier architectures.
4. Comparison of the various classifiers developed and the selection of the superior model.

The remainder of this section will elaborate on the various stages of implementation mentioned above.

3.1 Selection and Processing of Data

The data utilized in developing and testing the various classifiers has been created by analyzing all the possible combinations of the 4 forecasted closing price performance classes. As a result, the entire data set consisted of 256 unique data records.

In order to present the forecasted closing price performance classes to the classifiers, a binary notation is employed. These inputs are presented to the classifier using 4 inputs. This input representation format is used for all indices considered. A similar binary notation scheme is also utilized to present the investment recommendation outputs. Table 2 illustrates the manner in which the inputs and outputs of the component are to be interpreted. As previously mentioned, design 1 has 16 inputs and 16 outputs. The input representation format is the same as above. However, the first group of 4 inputs corresponds to the forecasted performance of the Dow Jones Industrial Average index. Similarly, the second, third and fourth group of 4 inputs characterizes the forecasted performance of the JSE All Share, Nasdaq 100 and Nikkei 225 Stock Average indices, respectively. The outputs are to be interpreted in a similar manner.

The data is divided into a training, validation and test set. During the implementation of all 3 designs considered, the training data set consisted of all data records where the inputs were classified into 2 of the 4 closing price performance classes. However, the models developed were validated and tested with the remaining possible closing price performance class combinations. The

Table 2. Classifier input and output representation

Classifier inputs					Classifier outputs				
Input	1	2	3	4	Output	1	2	3	4
LR	1	0	0	0	Sell stocks in possession	0	1	0	0
SR	0	1	0	0	Hold current position	1	0	0	0
SD	0	0	1	0	Buy stocks to the value of 15 % of available trading capital	0	0	1	0
LD	0	0	0	1	Buy stocks to the value of 25 % of available trading capital	0	0	0	1

training data set is used to train the ANN to find the general pattern between its inputs and outputs. The validation data set is used to assess the network and the test data is employed to confirm the classification quality of the developed model.

The training data set is used to create the cluster centers within the FISs. However, the validation and test data sets are utilized to assess the classification ability of the inference systems.

3.2 Optimization of the Classification Threshold

MLP and RBF neural network architectures were utilized in the classification of investment recommendations. The MLP and RBF neural network architectures are possibly the most extensively employed ANNs in pattern classification [2]. Due to the non-linear capabilities of these networks, they are said to be excellent universal approximators that provide highly accurate solutions. As a result, these networks produce very practical tools for classification and inversion problems [12].

It has been stated that a network with 1 hidden layer, provided with sufficient data, can be used to model any function [12]. As a result, the ANN architectures employed consisted of only 1 hidden layer. The MLP network hidden layer consists of non-linear activation functions. The choice of the activation function is largely dependent on the application of the model [12]. However, it has been found that the hyperbolic tangent activation function offers a practical advantage of faster convergence during training [2]. As a result, this function has been employed within the MLP network.

The MLP network output layer also contains activation functions. There are 3 major forms of the function that should be considered. These are the linear, logistic sigmoidal and softmax activation functions [2]. It has been stated that the appropriate selection of the output layer activation function for a classification problem is the logistic sigmoidal function [2]. As a result, this function has been employed within the output layer of the MLP network. The RBF networks that have been developed contained a Gaussian activation function within its hidden layer and a linear activation function within its output layer.

As previously mentioned, the FISs developed utilized subtractive clustering to create the required membership functions and set of fuzzy inference rules. During this stage of implementation, the number of hidden nodes within the ANNs and the cluster radius utilized by the cluster centers within the FISs were assumed to be arbitrary. This will be optimized at a later stage of development. During this stage of development, the number of hidden nodes within the ANNs as well as the cluster radius utilized by the FISs was 10 and 0.5, respectively. This stage of implementation involved the optimization of the interpretation of the classifiers. As a result, this involved the selection of an appropriate classification threshold value that would yield the most accurate results.

The classification threshold has been optimized by minimizing an error function that mapped the classification thresholds to the accuracy of the developed classifiers. The process has been performed on the validation data set.

Since this is a classification implementation, the accuracy of the models can no longer be calculated using the sum of square error of the difference between the target and investment recommendation classifier output. Instead a confusion matrix is utilized to identify the number of true and false classifications that are generated by the models developed. This is then used to calculate the true accuracy of the classifiers, using the following equation:

$$Accuracy = \sqrt{\frac{TP * TN}{(TP + FN) * (FP + TN)}} \tag{1}$$

where
TP is the true positive (1 classified as a 1),
TN is the true negative (0 classified as a 0),
FN is the false negative (1 classified as a 0),
FP is the false positive (0 classified as a 1).

The classification threshold was optimized by initially creating classifiers utilizing a threshold value of 0.5. This implies that if the classifier outputs a value less than 0.5, the output will be regarded as a 0. Similarly, if the output value is larger than or equal to 0.5, the output will be interpreted as a 1. This threshold value of 0.5 proved to be adequate for the MLP networks as well as the FISs implementations. The threshold value resulted in 100% accurate classifications. This has been demonstrated on the training as well as validation data sets. However, the RBF classifier employed in design 1 did not perform well utilizing this threshold value. As a result, the classification threshold of this model had been varied from 0.1 to 0.5 in iterations of 0.01. Table 3 illustrates the threshold values that resulted in the largest accuracy value for the validation data set. The threshold value of 0.5 proved to be satisfactory for design 2 and design 3 RBF classifiers.

3.3 Optimization of the Classifier Architectures

This stage of implementation involved the optimization of the ANN and Fuzzy Inference System (FIS) architectures. As a result, this step of development involved the selection of the correct number of hidden neurons that would yield the most accurate results. It also entailed selecting the correct cluster radius that would concede the largest investment recommendation classification accuracy.

Table 3. Results of varied classification threshold for design 1 RBF classifier. DJ, JSE, Nas and Nik corresponds to Dow Jones Industrial Average index, JSE All Share index, Nasdaq 100 index and Nikkei 225 Stock Average index, respectively.

	Classification thresholds			
Class.	DJ	JSE	Nas	Nik
Sell stocks in possession	0.19	0.23	0.20	0.11
Hold current position	0.17	0.10	0.12	0.17
Buy stocks to the value of 15 % of available trading capital	0.19	0.16	0.13	0.17
Buy stocks to the value of 25 % of available trading capital	0.24	0.17	0.17	0.19

The number of hidden neurons or nodes has been optimized by minimizing an error function that mapped the number of hidden nodes to the accuracy of the developed network. The process was performed on the validation and test data sets.

The hidden nodes were optimized by creating various MLP and RBF ANNs with hidden nodes of 1 to 75. As a result, 150 ANNs were developed. These developed networks employed the classification thresholds stated in the previous section. The networks also utilized the same activation functions mentioned in the previous section. Utilizing the training data set, these networks are trained. The validation and test data are then presented to the ANN. Thereafter, the accuracies for the training, validation and test data sets are calculated. When presented with the validation and test data, ANNs that resulted in the largest accuracy were analyzed.

Fig. 1 illustrates the sell stocks in possession at the next day closing price investment recommendation results of design 1. Similar results were achieved for the other design implementations as well as investment recommendations. Similar results were also obtained for the other indices considered.

The investigation revealed that a design 1 MLP and RBF network with number of hidden nodes larger than 12 and 52, respectively, yield 100% accurate models for categorizing the investment recommendations appropriately. The investigation also determined that design 2 MLP and RBF ANNs with number of hidden nodes greater than 2 and 5, respectively, achieved the same results. Similar results were obtained with design 3 MLP and RBF networks that contained more than 1 hidden neuron.

The cluster radius indicates the range of influence of a cluster. A small cluster radius results in small clusters in the data and, therefore, many fuzzy rules. Large cluster radii yield few large clusters in the data and, hence, fewer fuzzy rules [13]. The cluster radius has been optimized by minimizing an error function that mapped the radius to the accuracy of the developed inference systems. This process was performed on the validation and test data sets.

During this step of implementation, the optimization process entailed the construction of various inference systems with the cluster radius ranging from 0.01 to 1. The investigation determined that design 2 FISs with a cluster radius equal to or greater than 0.01 achieve 100% accuracy in categorizing the investment recommendations appropriately. However, the design 1 FIS did not achieve 100% investment recommendation classification accuracies. It has been determined that a cluster radius of 0.11 achieved the most accurate results. The lowest accuracy value attained was 83%. The largest accuracy value was 100%.

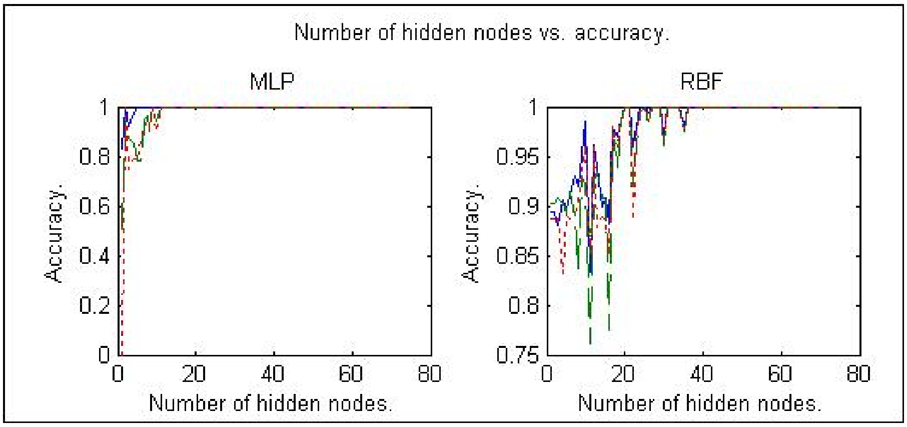


Fig. 1. This figure illustrates the results of sell stocks in possession at the next day closing price investment recommendation for design 1. The number of hidden nodes was varied and the corresponding accuracy values achieved were noted. The solid, dashed and dotted line represent the training, validation and test data sets, respectively.

3.4 Comparison of the Various Designs Implemented and the Selection of the Superior Model

This stage of implementation entailed the comparison of the various designs that were developed. It also involves the selection of the best design to classify the investment recommendations. Table 4 illustrates the various models that have been created. The above designs have been compared in terms of their complexity as well as scalability. Complexity, in this context, is defined as the number of classifiers employed by the design. Scalability is defined as the ability of the design to accommodate the classification of additional investment recommendations.

It is evident that design 1 has low complexity and low scalability. When additional investment recommendations are to be added to the component, the classifier employed is to be re-trained. However, design 2 has low complexity as there are only 4 classifiers utilized. The design also has high scalability. It is not required to re-create the existing classifiers, when additional recommendations are added. Table 4 indicates that design 3 has high complexity. The design contains 16 classifiers. In order to add investment recommendations to the component, the existing classifiers do not have to be re-created. As a result, the design has high scalability.

Table 4. This table illustrates the various models that were created. Accuarcies are presented as percentages.

Design	Classifier topology	Hidden nodes	Fuzzy rules	Membership functions	Accuracy (Training)	Accuracy (Validation)	Accuracy (Test)
1	MLP	12	-	-	100	100	100
1	RBF	52	-	-	100	100	100
1	FIS	-	85	1360	100	83	87
2	MLP	2	-	-	100	100	100
2	RBF	5	-	-	100	100	100
2	FIS	-	4	16	100	100	100
3	MLP	1	-	-	100	100	100
3	RBF	1	-	-	100	100	100
3	ANFIS	-	4	16	100	100	100

Due to the above analysis, design 2 is most appropriate for this application. It does not employ many classifiers and the design does not require re-work when additions are to be made. It is evident from Table 4 that both the ANN and FIS implementations of design 2 perform satisfactorily. As a result, either of the classifier architectures could be used.

4 Conclusion

This research involved the development of a component that could categorize investment recommendations, based on the forecasted performance of indices, appropriately. The Dow Jones Industrial Average, JSE All Share, Nasdaq 100 and Nikkei 225 Stock Average indices were considered.

Various designs of the component were considered. Designs that utilized 1, 4 and 16 classifiers were implemented. The development methodology employed in the creation of these designs, initially, involved the selection of appropriate classification thresholds. Thereafter, the number of hidden nodes within the ANNs as well as the cluster radius of the cluster centers within the FISs was varied. This resulted in creating acceptable classifier architectures. Acceptable investment recommendation classification accuracies were achieved.

The designs were compared in terms of complexity as well as scalability. Complexity is concerned with the number of classifiers that are used within the design. Scalability is the ability of the design to accommodate the classification of additional investment recommendations. Design 2 has low complexity and high scalability. This design consisted of 4 classifiers. Each classifier has 4 inputs and 4 outputs. This design is most appropriate for the application of concern.

References

1. Jain, A. K., Duin, R. P. W., Mao, J.: Statistical pattern recognition: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence. **22** (2000) 4–37
2. Nabney, I. T.: Netlab: Algorithms for Pattern Recognition. Springer (2002)

3. Keller, J. M., Gray, M., Givens, J.: A fuzzy k-nearest neighbor algorithm. IEEE Transaction on System, Man and Cybernetics. **15** (1985) 580–585
4. Bezdek, J. C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum (1981)
5. Ishibuchi, H., Nozaki, K., Yamamoto, N., Tanaka, H.: Construction of fuzzy classification systems with rectangular fuzzy rules using genetic algorithms. Fuzzy Sets and Systems. **65** (1994) 237–253
6. Russo, M.: FuGeNeSys A fuzzy genetic neural system for fuzzy modeling. IEEE Transaction on Fuzzy Systems. **6** (1998) 373–388
7. Nauck, D., Kruse, R.: A neuro-fuzzy method to learn fuzzy classification rules from data. Fuzzy Sets and Systems. **89** (1997) 277–288
8. Sun, C-T., Jang, J-S. R.: A neuro-fuzzy classifier and its applications. Proceedings of the IEEE International Conference on Fuzzy System. **1** (1993) 94–98
9. Hsu, C-W., Lin, C-J.: A comparison of methods for multi-class support vector machines. IEEE Transactions on Neural Neworks. **13** (2002) 415–425
10. Friedman, J. H.: Another approach to polychotomous classification. Department of Statistics, Stanford University. Technical Report. (1996)
11. Schurmann, J.: Pattern Classification. A Unified View of Statistical and Neural Principles. John Wiley & Sons Inc. (1996)
12. Bishop, C. M.: Neural Networks for Pattern Recognition. Oxford University Press. (1995)
13. Chiu, S.: Fuzzy Model Identification Based on Cluster Estimation. Journal of Intelligent and Fuzzy Systems. **2** (1994) 267–278
14. Jang, J-S. R.: ANFIS: Adaptive-Network-Based Fuzzy Inference System. IEEE Transaction on System, Man and Cybernetics. **23** (1993) 665-685
15. Patel, P. B., Marwala, T.: Forecasting closing price indices using neural networks. IEEE System, Man and Cybernetics Conference. (to appear)

Online Forecasting of Stock Market Movement Direction Using the Improved Incremental Algorithm

Dalton Lunga and Tshilidzi Marwala

University of the Witwatersrand
School of Electrical and Information Engineering
Private Bag 3 Wits 2050,
Johannesburg, South Africa
{d.lunga, t.marwala}@ee.wits.ac.za
http://www.ee.wits.ac.za/∼marwala

Abstract. In this paper we present a particular implementation of the Learn++ algorithm: we investigate the predictability of financial movement direction with Learn++ by forecasting the daily movement direction of the Dow Jones. The Learn++ algorithm is derived from the Adaboost algorithm, which is denominated by sub-sampling. The goal of concept learning, according to the probably approximately correct weak model, is to generate a description of another function, called the hypothesis, which is close to the concept, by using a set of examples. The hypothesis which is derived from weak learning is boosted to provide a better composite hypothesis in generalizing the establishment of the final classification boundary. The framework is implemented using multi-layer Perceptron (MLP) as a weak Learner. First, a weak learning algorithm, which tries to learn a class concept with a single input Perceptron, is established. The Learn++ algorithm is then applied to improve the weak MLP learning capacity and introduces the concept of online incremental learning. The proposed framework is able to adapt as new data are introduced and is able to classify.

1 Introduction

The financial market is a complex, evolutionary, and non-linear dynamical system. The field of financial forecasting is characterized by data intensity, noise, non-stationary, unstructured nature, high degree of uncertainty, and hidden relationships [1]. Many factors interact in finance including political events, general economic conditions, and traders' expectations. Therefore, predicting market price movements is quite difficult. Increasingly, according to academic investigations, movements in market prices are not random. Rather, they behave in a highly nonlinear and dynamical manner. The standard random walk assumption of future prices may merely be a veil of randomness that shrouds a noisy nonlinear process [2]. Incremental learning is the solution to such scenarios, which can be defined as the process of extracting new information without losing prior knowledge from an additional dataset that later becomes available. Various definitions and interpretations of incremental learning can be found in literature, including online learning [3], relearning of previously misclassified instances, and growing and pruning of classifier architectures [4]. An algorithm possesses incremental learning capabilities, if it meets the following criteria:

- Ability to acquire additional knowledge when new stock data are introduced
- Ability to retain previously learned information about the stock closing prices.
- Ability to learn new classes of stock data if introduced by new data.

Some applications of online classification problems have been reported recently [5]. In most cases, the degree of accuracy and the acceptability of certain classifications are measured by the error of misclassified instances. Although Learn++ has mostly been applied to classification problems, we show in this paper that the choice of Learn++ algorithm can boost a weak learning model to classify stock closing values with minimum error and reduced training time. For the practitioners in financial market, forecasting methods based on minimizing forecast error may not be adequate to meet their objectives. In other words, trading driven by a certain forecast with a small forecast error may not be as profitable as trading guided by an accurate prediction of the direction of movement. The main goal of this study is to explore the predictability of financial market movement direction using an ensemble of classifiers implemented using the Learn++ algorithm. This paper discusses the ensemble systems, introduces the basic theory on incremental learning and the Learn++ algorithm, and gives the experimental scheme as well as results obtained.

2 Ensemble of Classifiers

Ensemble systems have attracted a great deal of attention over the last decade due to their empirical success over single classifier systems on a variety of applications. Such systems combine an ensemble of generally weak classifiers to take advantage of the so-called instability of the weak classifier. This causes the classifiers to construct sufficiently different decision boundaries for minor modifications in their training parameters and as a result each classifier makes different errors on any given instance. A strategic combination of these classifiers, such as weighted majority voting [6], then eliminates the individual errors, generating a strong classifier. A rich collection of algorithms has been developed using multiple classifiers, such as AdaBoost [7], with the general goal of improving the generalization performance of the classification system. Using multiple classifiers for incremental learning, however, has been largely unexplored. Learn++, in part inspired by AdaBoost, was developed in response to recognizing the potential feasibility of ensemble of classifiers in solving the incremental learning problem. Learn++ was initially introduced in [8] as an incremental learning algorithm for the MLP type networks. A more versatile form of the algorithm was presented in [9] for all supervised classifiers. We have recently recognized that the inherent voting mechanism of the algorithm can also be used in effectively determining the confidence of the classification system in its own decision making. In this work, we describe the algorithm Learn++, along with representative results on incremental learning and confidence estimation obtained on the application of the algorithm to predict the direction of the movement for the Dow Jones Average Indicators.

3 Incremental Learning

An incremental learning algorithm is defined as an algorithm that learns new information from unseen data, without necessitating access to previously used data [10]. The algorithm must also be able to learn new information from new data and still retains knowledge from the original data. Lastly, the algorithm must be able to learn new classes that may be introduced by new data. This type of learning algorithm is sometimes referred to as a 'memoryless' online learning algorithm. Learning new information without requiring access to previously used data, however, raises 'stability-plasticity dilemma' [11]. This dilemma indicates that a completely stable classifier maintains the knowledge from previously seen data, but fails to adjust in order to learn new information, while a completely plastic classifier is capable of learning new data but lose prior knowledge. The problem with the MLP is that it is a stable classifier and is not able to learn new information after it has been trained. Different procedures have been implemented for incremental learning. One procedure of learning new information from additional data involves discarding the existing classifier and training a new classifier using accumulated data. Other methods such as pruning of networks or controlled modification of classifier weight or growing of classifier architectures are referred to as incremental learning algorithm. This involves modifying the weights of the classifier using the misclassified instances only. The above algorithms are capable of learning new information; however, they suffer from 'catastrophic forgetting' and require access to old data. One approach evaluates the current performance of the classifier architecture. If the present architecture does not sufficiently represent the decision boundaries being learned, new decision clusters are generated in response to new pattern. Furthermore, this approach does not require access to old data and can accommodate new classes. However, the main shortcomings of this approach are: cluster proliferation and extreme sensitivity to selection of algorithm parameters. In this paper, Learn++ is implemented for online prediction of stock movement direction using the Dow Jones average indicators. The Learn++ algorithm is summarized in the next section.

4 Learn++

Learn++ is an incremental learning algorithm that uses an ensemble of classifiers that are combined using weighted majority voting. Learn++ was developed from an inspiration by a boosting algorithm called adaptive boosting (AdaBoost).

Each classifier is trained using a training subset that is drawn according to a distribution D. The classifiers are trained using a weakLearn algorithm. The requirement for the weakLearn algorithm is that it must be able to give a classification rate of atleast 50% initially. For each database D_k that contains learning examples and their corresponding classes, Learn++ starts by initializing the weights, w, according to the distribution D_T, where T is the number of hypothesis. Initially the weights are initialized to be uniform, which gives equal probability for all instances to be selected to the first training subset and the distribution is given by

$$D = \frac{1}{m} \tag{1}$$

Where m represents the number of training examples in database S_k. The training data are then divided into training subset T_R and testing subset T_E to ensure weakLearn capability. The distribution is then used to select the training subset T_R and testing subset T_E from S_k. After the training and testing subset have been selected, the weakLearn algorithm is implemented. The weakLearner is trained using subset, T_R. A hypothesis, h_t obtained from weakLearner is tested using both the training and testing subsets to obtain an error,ϵ_t:

$$\epsilon_t = \sum_{t:h_t(x_i)\neq y_i} D_t(i) \tag{2}$$

The error is required to be less than $\frac{1}{2}$; a normalized error β_t is computed using:

$$\beta_t = \frac{\epsilon_t}{1-\epsilon_t} \tag{3}$$

If the error is greater than $\frac{1}{2}$, the hypothesis is discarded and new training and testing subsets are selected according to D_T and another hypothesis is computed. All classifiers generated so far, are combined using weighted majority voting to obtain composite hypothesis, H_t

$$H_t = \arg\max_{y\in Y} \sum_{t:h_t(x)=y} \log\frac{1}{\beta_t} \tag{4}$$

Weighted majority voting gives higher voting weights to a hypothesis that performs well on its training and testing subsets. The error of the composite hypothesis is computed as in Eq. 5 and is given by

$$E_t = \sum_{t:H_t(x_i)\neq y_i} D_t(i) \tag{5}$$

If the error is greater than $\frac{1}{2}$, the current composite hypothesis is discarded and the new training and testing data are selected according to the distribution D_T. Otherwise, if the error is less than $\frac{1}{2}$, the normalized error of the composite hypothesis is computed as:

$$B_t = \frac{E_t}{1-E_t} \tag{6}$$

The error is used in the distribution update rule, where the weights of the correctly classified instances are reduced, consequently increasing the weights of the misclassified instances. This ensures that instances that were misclassified by the current hypothesis have a higher probability of being selected for the subsequent training set. The distribution update rule is given by

$$w_{t+1} = w_t(i) \cdot B_t^{[|H_t(x_i) \neq y_i|]} \tag{7}$$

Once the T hypotheses are created for each database, the final hypothesis is computed by combining the composite hypothesis using weighted majority voting given by

$$H_t = \arg\max_{y \in Y} \sum_{k=1}^{K} \sum_{t: H_t(x)=y} \log \frac{1}{\beta_t} \tag{8}$$

5 Confidence Measurement

An intimately relevant issue is the confidence of the classifier in its decision, with particular interest on whether the confidence of the algorithm improves as new data become available. The voting mechanism inherent in Learn++ hints to a practical approach for estimating confidence: decisions made with a vast majority of votes have better confidence than those made by a slight majority [12]. We have implemented McIver and Friedl's weighted exponential voting based confidence metric [13] with Learn++ as

$$C_i(x) = P(y = i|x) = \frac{\exp^{F_i(x)}}{\sum_{k=1}^{N} \exp^{F_k(x)}}, 0 \leq C_i(x) \leq 1 \tag{9}$$

Where $C_i(x)$ is the confidence assigned to instance x when classified as class i, $F_i(x)$ is the total vote associated with the i^th class for the instance x and N is the number of classes. The total vote $F_i(x)$ class received for any given instances is computed as

$$F_i(x) = \sum_{t=1}^{N} \begin{pmatrix} \log \frac{1}{\beta_t}, \ if \ h_t(x) = i \\ 0, \ otherwise \end{pmatrix} \tag{10}$$

The confidence of winning class is then considered as the confidence of the algorithm in making the decision with respect to the winning class. Since $C_i(x)$ is between 0 and 1, the confidences can be translated into linguistic indicators as shown in Table 1. These indicators are adopted and used in interpreting our experimental results.

Equations (9) and (10) allow Learn++ to determine its own confidence in any classification it makes. The desired outcome of the confidence analysis is to observe a high confidence on correctly classified instances, and a low confidence on misclassified instances, so that the low confidence can be used to flag those instances that are being misclassified by the algorithm. A second desired outcome is to observe improved confidences on correctly classified instances and reduced confidence on misclassified instances, as new data become available, so that the incremental learning ability of the algorithm can be further confirmed.

Table 1. Confidence estimation representation

Confidence range (%)	Confidence level
$90 \leq C \leq 100$	Very High (VH)
$80 \leq C < 90$	High (H)
$70 \leq C < 80$	Medium (M)
$60 \leq C < 70$	Low (L)
$C < 60$	Very Low (VL)

6 Forecasting Framework

6.1 Experimental Design

In our empirical analysis, we set out to examine the daily changes of the Dow Jones Index. The Dow Jones averages are unique in that they are price weighted rather than market capitalization weighted. Their component weightings are therefore affected only by changes in the stock prices, in contrast with other indexes' weightings that are affected by both price changes and changes in the number of shares outstanding [14]. When the averages were initially created, their values were calculated by simply adding up the component stock prices and dividing by the number of components. Later, the practice of adjusting the divisor was initiated to smooth out the effects of stock splits and other corporate actions. The Dow Jones Industrial Average measures the composite price performance of over 30 highly capitalized stocks trading on the New York Stock Exchange (NYSE), representing a broad crosssection of US industries. Trading in the index has gained unprecedented popularity in major financial markets around the world. The increasing diversity of financial instruments related to the Dow Jones Index has broadened the dimension of global investment opportunity for both individual and institutional investors. There are two basic reasons for the success of these index trading vehicles. First, they provide an effective means for investors to hedge against potential market risks. Second, they create new profit making opportunities for market speculators and arbitrageurs. Therefore, it has profound implications and significance for researchers and practitioners alike to accurately forecast the movement direction of stock prices.

6.2 Model Input Selection

Most of the previous researchers have employed multivariate input. Several studies have examined the cross-sectional relationship between stock index and macroeconomic variables. The potential macroeconomic input variables which are used by the forecasting models include term structure of interest rates (TS), short-term interest rate (ST), long-term interest rate (LT), consumer price index (CPI), industrial production (IP), government consumption (GC), private consumption (PC), gross national product (GNP) and gross domestic product (GDP). Other macroeconomic variables data are not available for our study. Thus for our study only the closing values of the Index were selected as inputs.

A one step forward prediction of the Index was performed on a daily basis. The output of this prediction model was used as inputs to the learn++ algorithm for classification into the correct category that would give an indication of whether the predicted index value is 1 (indicating a positive increase in next day's predicted closing value compared to the previous day's closing value) or a predicted closing value of −1, indicating a decrease in next day's predicted closing value compared to the previous day's closing value. Figure 1 depicts the conceptual model of all processes required for this study. The first prediction model can be written as depicted by Eq. 11 below:

$$CV_t = F(cv_{t-1}, cv_{t-2}, cv_{t-3}, cv_{t-4}) \tag{11}$$

Where CV_t is the predicted close value at time t, cv_{t-1} indicates the close value at day i, where $i = 1, 2, 3, , t-1$.The second model takes the output of the first model as its input in predicting the direction of movement for the index. The classification prediction stage can be represented by Eq. 12:

$$Direction_t = F(CV_t) \tag{12}$$

Where CV_t is the first model prediction of the fifth day stock closing value when given the raw data at time $t-1$ to $t-4$ respectively. $Direction_t$ is a categorical variable to indicate the movement direction of Dow Jones Index at time t. If Dow Jones Index at time t is larger than that at time $t-1$, $Direction_t$ is 1. Otherwise, $Direction_t$ is −1.

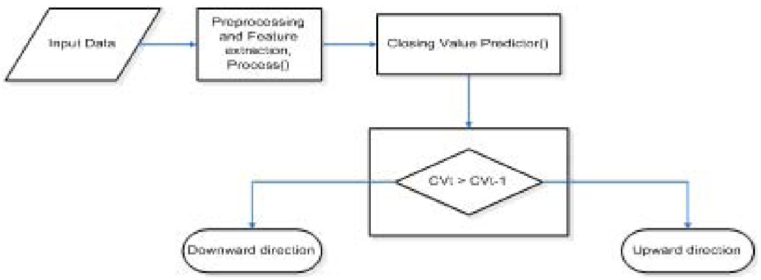

Input Data
Preprocessing and Feature extraction, Process()
Closing Value Predictor()
CVt > CVt-1
Downward direction
Upward direction

Fig. 1. Proposed model for online stock forecasting

6.3 Experimental Results

The forecasting model described in the sections above is estimated and validated by insample data. The model estimation selection process is then followed by an empirical evaluation which is based on the out-of-sample data. At this stage, the relative performance of the model is measured by the classification accuracy of the final hypothesis chosen for all given databases. The confidence of the algorithm on its own decision is used in establishing the accuracy of predicted closing value category. The first experiment implements a one step forward prediction of the next day's stock closing value. After predicting the next day's closing value this value is fed into a classification model to indicate the direction of movement for the stock prices. As discussed above the database consisted of 1476 instances of the Dow Jones average closing value during the period from January 2000 to November 2005; 1000 instances is used for training and all the remaining instances is used for validation. The two binary classes are 1, indicating an upward direction of returns in Dow Jones stock, and -1 to indicate a predicted fall/downward direction of movement for the Dow Jones stock. Four datasets S_1, S_2, S_3, S_4, where each dataset included exactly one quarter of the entire training data, were provided to Learn++ in four training sessions for incremental learning. For each training session k,($k = 1, 2, 3, 4$) three weak hypothesis were generated by Learn ++. Each hypothesis h_1, h_2 and h_3 of the k^th training session was generated using a training subset TR_t and a testing subset TE_t. The WeakLearner was a single hidden layer MLP with 15 hidden layer nodes and 1 output node with an MSE goal of 0.1. The test set of data, Validate consisted of 476 instances that were used for validation purposes. On average , the MLP hypothesis, weakLearner, performed little over 50%, which improved to over 80% when the hypothesis were combined by making use of weighted majority voting. This improvement demonstrates the performance improvement property of Learn++, as inherited from AdaBoost, on a given database. The data distribution and the percentage classification performance are given in Table 2. The performances listed are on the validation data, Validate following each training session. Table 3 provides an actual breakdown of correctly classified and misclassified instances falling into each confidence range after each training session. The trends of the confidence estimates after subsequent training sessions are given in Table 3. The desired outcome on the actual confidences is high to very high confidences on correctly classified instances, and low to very low confidences on misclassified instances. The desired outcome on confidence trends is increasing or steady confidences on correctly classified instances, and decreasing confidences on misclassified instances, as new data is introduced.

Table 2. Training and generalisation performance of Learn++

Database	Class(1)	Class(-1)	Test Performance (%)
S_1	132	68	72
S_2	125	75	82
S_3	163	37	85
S_4	104	96	86
Validate	143	57	–

The performance shown in Table 2 indicates that the algorithm is improving its generalization capacity as new data become available. The improvement is modest, however, as majority of the new information is already learned in the first training session. Table 4 indicates that the vast majority of correctly classified instances tend to have very high confidences, with continually improved confidences at consecutive training sessions. While a considerable portion of misclassified instances also had high confidence for this database, the general desired trends of increased confidence on correctly classified instances and decreasing confidence on misclassified ones were notable and dominant, as shown in Table 3.

Table 3. Confidence results

		VH	H	M	VL	L
Correctly classified	S_1	96	14	13	15	6
	S_2	104	7	22	17	14
	S_3	111	11	6	3	39
	S_4	101	13	42	12	4
Incorrectly classified	S_1	23	7	13	3	8
	S_2	27	0	1	3	4
	S_3	21	1	2	4	2
	S_4	24	0	2	2	0

Table 4. Confidence trends for Dow Jones

	Increasing Steady	Decreasing
Correctly classified	119	8
Misclassified	16	24

7 Conclusion

In this paper, we study the use of an incremental algorithm to predict financial markets movement direction. As demonstrated in our empirical analysis, Learn++ is observed to give good results on converting the weakLearner (MLP) into a strong learning algorithm that has confidence in all its decisions. The Learn++ algorithm is observed to assess the confidence of its own decisions. In general, majority of correctly classified instances had very high confidence estimates while lower confidence values were associated with misclassified instances. Therefore, classifications with low confidences can be used as a flag to further evaluate those instances. Furthermore, the algorithm also showed increasing confidences in correctly classified instances and decreasing confidences in misclassified instances after subsequent training sessions. This is a very comforting outcome, which further indicates that algorithm can incrementally acquire new and novel information from additional data.

Acknowledgement

This research was fully funded by the National Research Foundation of the Republic of South Africa.

References

1. Carpenter, G., Grossberg, S., Marhuzon, N., Reynolds, J., Rosen, D.: Artmap: A neural network architecture for incremental learning supervised learning of analog multidi-mensional maps. In: Transactions in Neural Networks. Volume 3., IEEE (1992) 678–713
2. McNelis, P.D., ed.: Neural Networks in Finance: Gaining the predictive edge in the market. Elsevier Academic Press, Oxford-UK (2005)
3. Freund, Y., Schapire, R.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Science (1997)
4. Bishop, C., ed.: Neural Networks for Pattern Recognition. Oxford University Press, Oxford-London (1995)
5. Vilakazi, B., Marwala, T., Mautla, R., Moloto, E.: Online bushing condition monitoring using computational intelligence. WSEAS Transactions on Power Systems **1** (2006) 280–287
6. Littlestone, N., Warmuth, M.: Weighted majority voting algorithm. information and computer science **108** (1994) 212–216
7. Polikar, R., Byorick, J., Krause, S., Marino, A., Moreton, M.: Learn++: A classifier independent incremental learning algorithm. Proceedings of International Joint Conference on Neural Networks (2002)
8. Polikar, R.: Algorithms for enhancing pattern separability, feature selection and incremental learning with applications to gas sensing electronic noise systems. PhD thesis, Iowa State University, Ames (2000)
9. Freund, Y., Schapire, R.: A short introduction to boosting. Japanese Society for Artificial Intelligence **14** (1999) 771–780
10. Polikar, R., Udpa, L., Udpa, S., Honavar, V.: An incremental learning algorithm with confi-dence estimation for automated identification of nde signals. Transactions on Ul-trasonic Ferroelectrics, and Frequency control **51** (2004) 990–1001
11. Grossberg, S.: Nonlinear neural networks: principles, mechanisms and architectures. Neural Networks **1** (1988) 17–61
12. Byorick, J., Polikar, R.: Confidence estimation using the incremental learning algorithm. Lecture notes in computer science **2714** (2003) 181–188
13. McIver, D., Friedl, M.: Estimating pixel-scale land cover classification confidence using nonparametric machine learning methods. Transactions on Geoscience and Remote Sensing **39** (2001)
14. Leung, M., Daouk, H., Chen, A.: Forecasting stock indices: a comparison of classification and level estimation models. (International Journal of Forecasting) 173–190

Pretests for Genetic-Programming Evolved Trading Programs: "zero-intelligence" Strategies and Lottery Trading

Shu-Heng Chen[1] and Nicolas Navet[1,2]

[1] AI-ECON Research Center, Department of Economics, National Chengchi University, Taipei, Taiwan 11623
chchen@nccu.edu.tw

[2] LORIA-INRIA, Campus-Scientifique, BP239, F-54506 Vandoeuvre, France
nnavet@loria.fr

Abstract. Over the last decade, numerous papers have investigated the use of GP for creating financial trading strategies. Typically in the literature results are inconclusive but the investigators always suggest the possibility of further improvements, leaving the conclusion regarding the effectiveness of GP undecided. In this paper, we discuss a series of pretests, based on several variants of random search, aiming at giving more clear-cut answers on whether a GP scheme, or any other machine-learning technique, can be effective with the training data at hand. The analysis is illustrated with GP-evolved strategies for three stock exchanges exhibiting different trends.

1 Motivation and Introduction

The computational intelligence techniques such as genetic programming[1], with their continuous advancement, persistently bring us something positive to expect, and incessantly push the application domain to more challenging issues. However, sometimes, the costs and benefits of using this advanced CI techniques are uncertain. Usually the benefits are not assured, while the costs is immediate. On the one hand, the CI techniques are frequently used as intensive search algorithms, which inevitable are computationally demanding, and take up a great amount of computational resources. On the other hand, whether there is a needle in the sea remains to be dubious.[2] Certainly, if such a needle does not exist at all, the all efforts are made with no avert. Given this asymmetry between costs and benefits, it would be economical, at the first stage, to test the existence of such a needle before a full-fledged version of search is applied. We call this procedure a *pretest*.

[1] Although, in this paper, we only focus on genetic programming, but the general ideas and some specific implementations may also be applicable to other computational intelligence technique used to induce trading strategies.

[2] For example, in the financial application domain, it can be particularly due to the efficient market hypothesis or the no-arbitrage condition.

The pretest procedure proposed here is similar in a sense to the pretests used in econometrics where the estimator of an unknown parameter is chosen on the basis of the outcome of a pretest ([1]). Pretesting, also known as "data-snooping" in finance, serves classically for selecting the right model that will be used later on for forecasting purpose ([2,3]). More broadly, pretesting can be considered as a practice of a sequential decision-making process, which is used when the decision involves a great deal of uncertainty, and the costs of making a wrong decision is huge. [3] In this case, in the first stage, we would like to spend some limited resources in probing to gain some initial information, e.g. the distribution of a very uncertain environment, while in later stages, we make our decision based on the gauged distribution.

The reason behind prestesting is very intuitive, and [4] is the first who applies this idea to the financial application of genetic programming (GP). [4] proposed a measure known as the η statistic. The η statistic is a measure of predictability. Basically, using simple (vanilla) version of GP, one can first gauge the predictability by η. When η is low to zero, it indicates that there is nothing to forecast. So, the use of full-fledged GP is not advised. The virtue of this doing is to distinguish *two kinds of possibilities* when we see a failure of an initial attempts based on simple GP. First, the series itself has nothing to forecast; second, GP has not been used appropriately. Understanding this distinction can result in big differences in our second stage of the decision. For the former case, we may simply give up the further search to avoid a waste. For the latter, we should keep on exploring different deliberations of GP to search for potential gains before a final conclusion can be made. In either case, we have a clear-cut situation. However, when a pretest is absent, we become less conclusive: we are no longer sure whether it is due to the non-existence of the needle, or the improper use of GP.

Unfortunately, in most financial trading applications of GP, a pretest has been largely neglected.[4] We think that this negligence may cause many observed inconclusive results. Typically, what happens is that the results with GP are not very convincing, but the investigators always suggest directions for further improvements, leaving the actual conclusion regarding the effectiveness of GP undecided. Therefore, this study attempts to provide practical pretesting procedure aiming at reducing the number of cases where the conclusion is inconclusive.

Needless to say, there are various ways to implement different pretesting. For example, the η statistic mentioned above can be used as a pretest, as [4] did, but that is mainly applied to forecasting time series. A series being predictable does not necessarily imply that we can develop profitable trading strategies. For example, the fluctuation is not volatile enough to cover the round-trip transaction

[3] The problem of sequential decision making under incomplete knowledge has been studied by researchers in various fields, such as optimal control, psychology, economics, and game theory.

[4] This may not be completely so. In fact, most earlier studies selected the buy-and-hold strategies or a risk-free investment (e.g., treasury bills) as the benchmark. However, the conclusion that "GP performs better than buy-and-hold in a bearish market and worst in a bullish market" is often found in the literature. This shows the limits of choosing buy-and-hold as a pretest. See, for example, [5].

costs. Consequently, literature of forecasting with GP and literature of trading with GP usually are separated. Therefore, in this paper, we attempt to develop pretest procedures more suitable for the trading purpose.

More precisely, we will propose several different styles of pretests, which when put together can help us decide whether there are hidden patterns to discover and whether GP is designed properly to do the job. The essential idea underlying all proposed pretests is to compare the performance of GP with random trading strategies or behavior. However, as we shall see in Section 2, just making trading strategies or trading behavior arbitrarily random is not sufficient to give a fair and informative comparison. To do so, some constraints are expected, and the intriguing point is how to impose these constraints properly.

The rest of the paper is organized as follows. Section 2 provides a detail formulation of four pretests. The first three concerns the trading strategies, whereas the last one concerns the trading behavior. Normally, trading behavior comes from trading strategies, and they cannot be separated; but, when randomness is introduced, difference between the two can arise. In particular, in the vein of algorithmic complexity, random trading strategies can imply trading behavior actually using knowledge, while random trading behavior presumably exclude such possibility. We, therefore, intentionally distinguish the two by called the former *zero-intelligence strategies*, and the latter *lottery trading*. Section 3 discuss how to use these proposed tests together to make a better judgement given the initial results we have. Section 4 illustrates the proposed pretests based on the real detail and the experimental designs detailed in the appendix. Section 5 makes the concluding remarks.

2 Pretests : Description and Rationale

In this section, we describe a series of 4 pretests and discuss their purpose and implementation. Out of the 4 pretests, we highlight that 2 are of particular interest and, as shown in section 3, enable us to gain complementary knowledge on the data under study and on the efficiency of the GP implementation. In the following, we consider GP with a validation stage before the actual testing on the out-of-sample data. Validation means that the best rules induced on the training interval are further selected on unseen data, the validation period, before being applied out-of-sample. The validation step is a device to fight overfitting[5] that has been widely used in earlier GP work (see for instance [7,8]). Note that our pretest proposals remain valid for GP without validation step except that pretest 2 replaces pretest 1, which requires validation.

2.1 GP Versus Equivalent Intensity Random Search

The basic idea is here to compare the outcome of GP with an *equivalent intensity random search*. We say that two search algorithms are equivalent in terms

[5] The actual effectiveness of validation is still an open question, see [5] and [6].

of search intensity if their execution leads to the evaluation of the same number of different trading strategies on the training data. For instance, let us consider GP with the parameters chosen for this study: a population of 500 individuals evolved over 100 generations. In first approximation, the equivalent random search (ERS) would consist in evaluating 50,000 randomly created solutions. In practice, search algorithms sometimes rediscover identical solutions over the course of their execution. This can be detected by keeping track of all created individuals since the beginning of the execution, and doing so useless fitness evaluations can be skipped, which actually saves computing time when the fitness function is rather time-consuming as it is in our context. Since, computationally speaking, what is preponderant is the fitness evaluation, and since the extent to which GP re-discovers the same individuals is very dependent upon the implementation, we impose that our definition of equivalent search intensity only accounts for unique individuals, *i.e.* individuals which require evaluation. We consider two solutions to be different if their expression is *syntactically different*[6], in our context, if the trees representing the programs are different.

The three following pretests compare GP with a random search with and without training and validation stage. In the latter search technique, the biologically inspired evolution process of GP is simply replaced by the creation of solutions at random. Since with random search the strategies do not benefit from the "intelligence" resulting from the evolution or learning process, we dub randomly created solutions *zero-intelligence trading strategies.*

For each pretest i, we formulate the null hypothesis $\mathcal{H}_{i,0}$ that GP does not outperform the technique it is compared with at pretest i, where the alternative hypothesis is denoted $\mathcal{H}_{i,1}$.The experiments will provide us with the answer on whether $\mathcal{H}_{i,0}$ should be rejected in favour of $\mathcal{H}_{i,1}$ or not.

Pretest 1: GP Versus Equal Search Intensity Random Search with Training and Validation Stage. The implementation of the random search strategy is straightforward: parameters of GP are set in such a way that only the initial generation, where individuals are created at random, is used. The size of the initial population is adjusted so that the resulting search intensity is identical to the one of the regular GP.

[6] Two individuals can be syntactically different while being equivalent in the sense that they lead to equivalent trading decisions, the equivalence could thus be also defined in terms of semantics. With symbolic simplification using rewriting rules and interval arithmetic on the function arguments, one could detect that some syntactically different individuals are in fact semantically identical. However, there is no way to make sure that all duplicates will be detected and the implementation of this procedure would be so complex and time consuming at run time that, in our opinion, a definition based on semantics would be of little practical interest. Alternatively, the equivalence in search intensity could be defined in terms of equivalent computing time, however there is such a difference of complexity between a full-fledged GP implementation and random search that it is hard to imagine how we can ensure that the two implementations have been optimized in a similar manner, while a better implementation of GP for instance may lead to an opposite conclusion.

– **Hypothesis $\mathcal{H}_{1,0}$ cannot be rejected:** the first explanation that can be envisaged is that, GP or not, there is nothing essential to be learned from the past. It that case GP would strongly "overfits" the training data, possibly explaining that its out-of-sample performance is worse than with a random search. This can be due by the market being efficient or because the training interval is very dissimilar to the out-of-sample[7]. Another explanation is that the GP machinery is not working properly, for instance due to a wrong choice of the function/terminal sets, because the parameters are inappropriate (e.g. too low search intensity), or the genetic operators unable to create better-than-random individuals.
– **Hypothesis $\mathcal{H}_{1,0}$ is rejected in favour of $\mathcal{H}_{1,1}$:** there may be something to learn from the past and GP, with the chosen parameters, may be effective in that task.

Rejecting $\mathcal{H}_{1,0}$ is of course a first indication of the efficiency of GP but we cannot rule out the case where there would nothing useful to learn on the data at hand and GP would beat random search by mere luck. We will see in Section 3, that further investigation may provide additional evidence to answer that question.

Pretest 2: GP Versus Equal Search Intensity Random Search with Training But Without Validation Stage. Here, the best random solutions on the training interval are applied directly to the out-of-sample period. With regards to pretest 1, pretest 2 could give us some insight about how effective is validation as a device to fight against overfitting. However, since overfitting is unlikely to occur with random solutions, the rationale of using pretest 2 is unclear and it will not be further considered in this study. A more direct and effective way to evaluate the effect of the validation stage is simply to compare regular GP with and without validation[8].

Pretest 3: GP Versus Equal Search Intensity Random Search Without Training and Without Validation Stage. In pretest 3, selection of the strategies on the training set is removed: a large number random strategies are created and applied directly out-of-sample. The performance is evaluated as the average performance (e.g. average total return) over the set of random strategies. Comparing the outcome of pretest 3 with regards to pretest 1 and regular GP tells us something about how effective is the selection process, the extent to which a top performing rule on the training and validation sets will keep on performing well out-of-sample. If strategies selected by GP or random search on the training and validation intervals have some predictive ability out-of-sample, it provides use with some evidence that there is something to learn from the past. It is worth to point out that the randomness of the strategies is here constrained by the GP language: rules can only be made with GP functions/terminals organized according to the typing scheme. For instance, it is possible that the GP

[7] In [5], numerous experiments have highlighted that when training and out-of-sample data sets are very "dissimilar", for instance if the market exhibits an opposite trend, then there is little chance that GP performs well out-of-sample.

[8] For instance, as it is done in [5].

language is not expressive enough to represent a rule consisting in buying and selling every other period[9]. In the rest of this study, we will consider pretest 4, presented in Section 2.2, that is similar in spirit to prestest 3, but is more random in the sense that it does not possess the bias in randomness induced by the GP language.

2.2 GP Versus Lottery Trading

We call *lottery trading* a strategy that would consist in making the investment decision at random on the basis of the outcome of a random variable. In its simplest form, the random variable would follow a Bernoulli distribution where the parameter p expresses the probability to take a long position and $1 - p$ the probability to be out of the market.

In our context, this requires refinement since we are interested in profitability and profitability takes into account transaction costs. So, to allow a fair comparison with GP, we should make sure that the expected number of transactions for lottery trading is the same as for GP. We call the expected number of transactions per unit of time the *frequency of a trading strategy.* Another important characteristic of a trading strategy is what we term its *intensity, i.e.* the number of periods where a position[10] "in the market" is held, over the length of the trading interval. We should also enforce lottery trading to have the same expected intensity as GP to avoid misleading results, for instance, the case where, given its frequency, the intensity of lottery trading is not sufficient to cover the transaction costs with the volatility of the market under study.

One denotes by F_{GP} and I_{GP} respectively the average frequency and average intensity observed for the set of GP evolved rules applied on the testing interval over all GP runs, N_{GP} is the number of transactions leading to F_{GP}. For the experiments made in the following, a sequence of investment decisions S_{LT} resulting from lottery trading is generated at random according to the following procedure:

- the intensity for lottery trading, I_{LT} , is uniformly chosen in $[I_{GP} \cdot (1 - \alpha), \min(1, I_{GP} \cdot (1 + \alpha))]$ with $0 \leq \alpha \leq 1$. In a first step, S_{LT} is made of the '0' positions (*i.e.* out of the market) followed by the block of '1' positions (*i.e.* in the market) corresponding to I_{LT},
- the number of transactions N_{LT} is uniformally chosen in the set of integer values that are even[11] in interval $[N_{GP} \cdot (1 - \alpha), N_{GP} \cdot (1 + \alpha)]$. The block of

[9] Period refers to the granularity of time used for trading, for instance, one second or one day.

[10] Implicitly, we consider here the trading of a single instrument, e.g. an index, where 2 decisions are possible at each time period: be in or be out of the market without short selling, or with short selling as implemented in [5], hold a long position or a short position. The concept remains valid where one can be holding a long position, a short position or be out of the market. One can also define the intensity and the frequency of a strategy for each instrument traded.

[11] N_{LT} has to be even since a "buy" transaction is followed by a sell transaction and no positions are left open.

'1' is subdivided at random in $N_{LT}/2$ sub-sequences and each sub-sequence is inserted at random inside the block of '0'. This design avoids the problem of overlapping '1' sub-sequences that occurs with other schemes.

We formulate the pretest comparing GP and lottery trading and denote by $\mathcal{H}_{4,0}$ the null hypothesis that GP does not outperform lottery trading.

Pretest 4: GP Versus Lottery Trading. Obviously, if GP is not able to outperform lottery trading, it gives strong evidence that GP will not be good at evolving effective trading strategies with the data at hand. In section 3, we shall discuss this point in more details.

3 What Does Pretest Tell Us ?

The outcomes of the pretests provide us with answers to the two following questions: is there something essential to learn on the training data that can be of interest for the out-of-sample period ? Does the GP implementation shows some evidence of effectiveness in that task ? Clearly, before actually trading with GP evolved programs, these two questions must be answered with reasonable certainty; the rest of this section explains how pretests may help in that regard.

3.1 Question 1: Is There Something to Learn ?

Null hypothesis $\mathcal{H}_{4,0}$ corresponding to pretest 4 has been presented in Section 2.2. We introduce pretest 5 that will be used in conjunction with pretest 4.

Pretest 5: Equivalent Intensity Random Search with Training and Validation Versus Lottery Trading. Here, we compare lottery trading to a random search with training and validation, and a search intensity equivalent to the one used for GP in pretest 4. Null hypothesis $\mathcal{H}_{5,0}$ is that the equivalent intensity random does not outperform lottery trading on the out-of-sample data. Depending on the validity of $\mathcal{H}_{4,0}$ and $\mathcal{H}_{5,0}$, we can draw the conclusions that are summarized in Table 1.

In case 1, best solutions on the training intervals, obtained with 2 different search algorithms, do not perform better than lottery trading on the out-of-sample period. This suggest to us than there is nothing to learn. In case 2, GP

Table 1. Information drawn from the outcomes of pretest 4 and pretest 5 ($\neg$**R** means that the null hypothesis $\mathcal{H}_{i,0}$ cannot rejected while **R** means that the hypothesis is rejected in favour of the alternative hypothesis)

	$\mathcal{H}_{4,0}$	$\mathcal{H}_{5,0}$	**Interpretation**
case 1	$\neg$**R**	$\neg$**R**	there is evidence that there is nothing to learn
case 2	**R**	$\neg$**R**	there may be something to learn (weak certainty)
case 3	**R**	**R**	there is evidence that there is something to learn
case 4	$\neg$**R**	**R**	there may be something to learn (weak certainty)

outperforms lottery trading but random search does not; it is possible that there is something to learn but that the selected random rules do not have a sufficient predictive ability. Anyway, this lead us to a less certain conclusion than in case 3 where both search techniques outperform lottery trading. Finally case 4 is a special case where random search performs better than lottery trading but GP does not. The whole evolution process of GP has thus a detrimental effect and a possible explanation is that GP induced solutions overfit the training data.

3.2 Question 2: Is the GP Machinery Working Properly?

The second question we ought to ask is whether GP is effective. Of course, this cannot be answered with the data at hand if pretests 4 and 5 have shown that there is nothing to be learned (case 1 in Table 1). In addition, in case 4 of Table 1, we already know that GP is not efficient since, by transitivity, it is outperformed by the random search based algorithm. Thus, the only two cases where one really needs to proceed to further examination are case 2 and case 3. The validity of null hypothesis $\mathcal{H}_{1,0}$, which can be tested with pretest 1, gives a helpful insight into the answer: only if $\mathcal{H}_{1,0}$ should be rejected we can conclude that GP shows some real effectiveness. We would like to stress that rejecting $\mathcal{H}_{1,0}$ is far from implying profitability, but beating a mere random search algorithm on a difficult problem with an infinite search space, is the bare minimum one can expect from GP.

4 Experiments

The aim of the experiments is to evaluate the extent to which the pretests proposed are reliable. The methodology adopted here is to check if the outcomes of the pretests are consistent with results already published in the literature. We call GP1 the GP implementation developed for this study and GP2 the software[12] used in [5], which will constitute our benchmark. The GP control parameters, identical to the one used in [5], are summarized in Table 1 (Appendix A).

The stock indexes from 3 stock markets are used: TSE 300 (Canada), Nikkei Dow Jones (Japan) and Capitalization Weighted Stock Index (Taiwan). They have been chosen among the 8 markets studied in [5] because they exhibit the main evolution patterns that can be found in the set of 8 markets. The aim of GP is to induce the most profitable strategy, measured by the accumulated return, for trading the stock exchange index. The use of short selling is possible. We adopt what is done classically in literature in terms of data-preprocessing and use normalized data that is obtained by dividing each day's price by a

[12] Although both programs have been developed by members of the AI-ECON Research Center, they have not been written by the same persons and do not share a single line of code. Furthermore GP2, which is based on the *Open-Beagle* library (see `http://beagle.gel.ulaval.ca/`), implements strongly-typed GP.

250-day moving average[13]. In a way similar to what is done usually, we subdivide the whole dataset into three sections: *training*, *validation* and *out-of-sample* test period. For each considered stock index, 3 different out-of-sample test periods of 2 years (*i.e.* 1999-2000, 2001-2002, 2003-2004) follow a 3 year validation and a 3 year training period. In the following, the term market refers to a stock exchange during a specific out-of-sample period. For instance, Canada-1 (C1 for short) is the market corresponding the TSE 300 during the out-of-sample period 1999-2000. Hypothesis testing is performed with the *Student's t-test* at a 95% confidence level. The samples for statistics are constituted of the results of 50 GP runs, 50 runs of equivalent search intensity random search with training and validation (ERS) and 100 runs of lottery trading (LT).

In 4 out of the 9 markets (*i.e.* C3, J2, T1, T3), there is evidence that there is something to learn from the training data (case 1 in Table 1). This is consistent with [5] where GP2 performs outstandingly on these 4 markets (respective total return: 0.34, 0.17, 0.52, 0.27). On markets C1, J3 and T2, pretests 4 and 5 suggest to us that there is nothing to learn (case 3). Except for C1, GP2 also performs poorly (-0.18 for J3 and -0.05 for T2). Finally, in the 3 markets where GP1 is shown to beat ERS ($\mathcal{H}_{1,0}$ is rejected in favor of $\mathcal{H}_{1,1}$ for J1, J2 and T1), GP results are very good : both GP1 and GP2 produce positive returns and outperform the buy-and-hold strategy.

Although more comprehensive tests are to be performed, the experiments conducted here show some preliminary evidence that the proposed pretests possess some predictive ability. Indeed, when the outcome is "nothing to learn", GP performs very poorly (except in one case out of three). When pretests suggest that there is something to learn, at least one implementation did good and when GP1 is more efficient than random search (*i.e.* ERS), GP2 from [5] is efficient too. In the light of the pretests, we should also conclude if our GP implementation (*i.e.* GP1) is more efficient than ERS, it is only slightly more efficient since one would expect more cases where GP beats LT and not LT. This suggest that GP1 is only able to take advantage of "simple" regularities in the data.

5 Conclusions

The main purpose of this paper is to enrich the earlier research on Genetic Programming (GP) induced market-timing decisions by proposing pretests aiming to shed light on the GP results. Actually in the literature, the results of applying GP for market-timing decisions are typically not very convincing but the investigators always suggest the possibility of further improvements. If the investigators can first convince that there is something to learn and that GP is suitable for that task, then their conclusion would be less vague and uncertain. We propose here a series of pretests, where GP is tested against a random behavior (*lottery trading*) and against strategies created at random (*zero-intelligence*

[13] See [5] for a discussion about how non-normalized data affects the performance of GP.

strategies), that aim to answer these two crucial questions. Of course there is the risk of getting a wrong pretest result and the possible reasons why GP may have failed should be thoroughly investigated before drawing conclusion. But, at the end, analyzing the results in the light of the pretests should help to draw more fine-grained conclusion.

Acknowledgment. This research was conducted when the second author (Nicolas Navet) was visiting researcher at the AI-ECON Research Center, National Chengchi University (NCCU), Taipei, Taiwan. The financial support from the AI-ECON Research Center as well as NCCU is greatly acknowledged. The authors would like also to acknowledge the grant from National Science Concil #95-2415-H-004-002-MY3.

References

1. J.A. Giles and D.E.A. Giles: Pre-test Estimation and Testing in Econometrics: Recent Developments. Journal of Economic Surveys **7**(2) (1993) 145–97
2. D. Danilov and J.R. Magnus: Forecast Accuracy After Pretesting with an Application to the Stock Market. Journal of Forecasting **23** (2004) 251–274
3. R. Sullivan and A. Timmermann and H. White: Data-Snooping, Technical Trading Rule Performance, and the Bootstrap. Journal of Finance **54** (1999) 1647–1692
4. M.A. Kaboudan: A Measure of Time Serie's Predictability Using Genetic Programming Applied to Stock Returns. Journal of Forecasting **18** (1999) 345–357
5. S.-H. Chen and T.-W. Kuo and K.-M. Hoi: Genetic Programming and Financial Trading: How Much about "What we Know". In: Handbook of Financial Engineering. Kluwer Academic Publishers (2006) Forthcoming.
6. S.-H. Chen and T.-Z. Kuo: Overfitting or Poor Learning: A Critique of Current Financial Applications of GP. In Springer-Verlag, ed.: Proceedings of the Sixth European Conference on Genetic Programming (EuroGP-2003). Number 2610 in LNCS (2003) 34–46
7. F. Allen and R. Karjalainen: Using Genetic Algorithms to Find Technical trading rules. Journal of Financial Economics **51** (1999) 245–271
8. C. Neely and P. Weller and R. Dittmar: Is Technical Analysis in the Foreign Exchange Market Profitable? A Genetic Programming Approach. Journal of Financial and Quantitative Analysis **32**(4) (1997) 405–427

A Genetic Programming Settings

Program GP2 implements strongly typed GP with the set of functions and terminals described in Table 1. The parameters here are basically identical to the ones in [5] (program GP1) except when fine-tuning GP2 have highlighted that better results may be obtained with different parameters. Precisely, we make use of more elitism, the size of the tournament selection is set to 5 and numerical mutation is implemented.

Population size	500
Number of generations	100
Maxim tree depth	10
Function set	+,-,*,/,norm,average,max,min,lag,and, or, not, >,<,if-then-else,true,false
Terminal set	price, real and integer ephemeral constants
Value range for real constants	[-1,1]
Value range for integer constants	[0,1000]
Offsprings created by:	
crossover	50%
standard mutation	20%
swap mutation	15%
reproduction	10%
ephemeral constant mutation	5%
Initialization	ramp-half-and-half
Evolution scheme	generation-by-generation replacement strategy
Elitism	25 best individuals kept for next generation
Selection scheme	tournament selection of size 5
Fitness function	accumulated return
Transaction costs	0.5%
Validation	
number of best trees saved	1 individual per run is saved for validation

Fig. 1. GP control parameters

Currency Options Volatility Forecasting with Shift-Invariant Wavelet Transform and Neural Networks

Fan-Yong Liu[1] and Fan-Xin Liu[2]

[1] Financial Engineering Research Center, South China University of Technology, Wushan, Guangzhou, P.R. China
fanyongliu@163.com
[2] Department of Physics, Huazhong University of Science and Technology, Wuhan, P.R. China

Abstract. This paper describes four currency options volatility forecasting models. These models are based on shift-invariant wavelet transform and neural networks techniques. The *à trous* algorithm is used to realize the shift-invariant wavelet transform. Wavelets provide a decomposition of the volatility in a nonlinear feature space. Neural networks are used to infer future volatility from the feature space. The individual wavelet domain forecasts are recombined by different techniques to form the accurate overall forecast. The proposed models have been tested with the USD/Yen options volatility market data. Experimental results show that wavelet prediction scheme has the best forecasting performance on testing dataset among four models, with regards to the least error values. Therefore, wavelet prediction scheme outperforms the other three models and avoids effectively over-fitting problems.

1 Introduction

Basically the price of an option consists of two parts, the intrinsic value and the time value. The time value represents the uncertainty until expiration, the risk of the underlying asset and the riskless return of the currencies. A lot of factors influence the time value, but the volatility between the involved currencies is the most important factor. A high volatility increases the risk of the option and the uncertainty about future price movements. The volatility is the most important input to all options pricing models. Besides, volatility is a very important quantity for value-at-risk calculations used in portfolio risk management.

However, volatility is quite unsettling, measuring volatility is in itself more of an art than a science. There is no clear agreement as to what measure of future volatility should be used as an input to the options pricing formulae. Therefore, it is very vital to forecast accurately the future volatility by modeling.

Traditionally, short term forecast techniques use statistical models. The Autoregressive moving average model is among the most popular ones of dynamic models. However, these models are basically linear methods, which have limited ability to capture nonlinearities in currency options volatility series.

Artificial intelligence methods for forecasting have shown ability to give better performance in dealing with the nonlinearity and other difficulties in modeling of the time series. Neural networks (NN) have been practiced recently in the area of time series forecasting due to their flexibilities in data modeling. In particular, Zapart [1] proposes an alternative way of looking at stochastic volatility models based haar wavelet and neural networks and their integration with conventional options pricing methods. Bai-Ling Zhang et al [2] proposed a hybrid neural-wavelet scheme to predict electricity demands in short term. This study is partly motivated by these contributions and shares much of the underlying analytical reasoning with the work recently reported in their papers.

In this paper, we study neural network models combined with wavelet transformed data. These strategies approximate a time series at different levels of resolution using multiresolution decomposition. An autocorrelation shell representation (ASR, [3,4]) technique is employed to reconstruct signals after wavelet decomposition. With the help of this technique, a time series can be expressed as an additive combination of the wavelet coefficients. These techniques are then applied to build forecasting models to predict currency options volatility as from the data obtained from foreign exchange (FX) markets.

The main contributions of this study include: a multiresolution prediction scheme set up on the ASR technique is applied to forecast currency options volatility, the performance comparison between four different volatility forecasting models is performed, and the model based on *à trous* transform avoids the over-fitting problems which occurs often in neural networks models.

This study consists of four sections. Section 2 provides a brief introduction to model design. Section 3 contains the experimental data, the criteria of the prediction performance, and result discussion. Section 4 draws the conclusions.

2 Model Design

In this section we propose the forecasting models of volatility and their application to the currency options.

2.1 Wavelet Transform and Autocorrelation Shell Representation

For a chosen mother wavelet function ψ a function f can be expanded as:

$$f(t) = \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} w_{jk} 2^{j/2} \psi(2^j t - k) \tag{1}$$

where the functions $\psi(2^j t - k)$ are all orthogonal to each other. The coefficient w_{jk} gives information about the behavior of the function f concentrating on the effects of scale around 2^{-j} near time $k \times 2^{-j}$. This wavelet decomposition of a function is closely related to a similar decomposition (the discrete wavelet transform, DWT) of a signal observed in discrete time.

In time series analysis, DWT often suffers from a lack of translation invariance. This problem can be tackled by means of a redundant or non-decimated wavelet transform [4,5]. A redundant transform based on a n-length input time series has a n-length resolution scale for each of the resolution levels of interest. The *à trous* algorithm is used to realize the shift-invariant wavelet transforms [6]. Such transforms are based on the ASR technique by dilations and translations of the autocorrelation functions of compactly supported wavelets.

By definition, the autocorrelation functions of a compactly supported scaling function $\phi(x)$ and the corresponding wavelet $\psi(x)$ are as follows:

$$\phi(x) = \int_{-\infty}^{\infty} \phi(y)\phi(y-x)dy, \quad \psi(x) = \int_{-\infty}^{\infty} \psi(y)\psi(y-x)dy. \tag{2}$$

The set of functions $\{\widetilde{\psi}_{j,k}(x)\}_{1\leq j\leq J, 0\leq k\leq N-1}$ and $\{\widetilde{\phi}_{J,k}(x)\}_{0\leq k\leq N-1}$, can be called an autocorrelation shell, where:

$$\widetilde{\psi}_{j,k}(x) = 2^{-j/2}\psi(2^{-j}(x-k)), \quad \widetilde{\phi}_{J,k}(x) = 2^{-J/2}\phi(2^{-J}(x-k)). \tag{3}$$

The filters $P = \{p_k\}_{-L+1\leq k\leq L-1}$ and $Q = \{q_k\}_{-L+1\leq k\leq L-1}$ is defined as:

$$\tfrac{1}{\sqrt{2}}\phi(\tfrac{x}{2}) = \sum_{k=-L+1}^{L-1} p_k\phi(x-k), \quad \tfrac{1}{\sqrt{2}}\psi(\tfrac{x}{2}) = \sum_{k=-L+1}^{L-1} q_k\phi(x-k). \tag{4}$$

Using the filters P and Q, the pyramid algorithm for expanding into the autocorrelation shell can be obtained as:

$$c_j(k) = \sum_{l=-L+1}^{L-1} p_l c_{j-1}(k+2^{j-1}l), \quad w_j(k) = \sum_{l=-L+1}^{L-1} q_l c_{j-1}(k+2^{j-1}l). \tag{5}$$

These shell coefficients obtained from (5) can then be used to directly reconstruct the signals. Given smoothed signal at two consecutive resolution levels, the detailed signal can be derived as:

$$w_j(k) = \sqrt{2}c_{j-1}(k) - c_j(k). \tag{6}$$

Then the original signal $c_0(k)$ can be reconstructed from the coefficients $\{w_j(k)\}_{1\leq j\leq J, 0\leq k\leq N-1}$ and residual $\{c_J(k)\}_{0\leq k\leq N-1}$:

$$c_0(k) = 2^{-J/2}c_J(k) + \sum_{j=1}^{J} 2^{-j/2}w_j(k) \tag{7}$$

for $k = 0, 1, \cdots, N-1$, where $c_J(k)$ is the final smoothed signal.

To make more precise predictions, the most recent data shall be used and the previous data is penalized with forgetting factors. The time-based *à trous* filters similar to that of [7] are used to deal with the boundary condition.

2.2 Automatic Relevance Determination

When applying neural networks to time series forecasting, it is important to decide an appropriate size of time-window of inputs. The Automatic Relevance Determination (ARD) method [8,9] gives a systematic way for choosing a suitable length of past windows to train the neural networks. In the forecasting model, ARD is used to choose a short-term history for higher temporal resolution and a long-term history for lower temporal resolution. By this way, substantial information on both the 'detailed' and 'general' history of the time series can be effectively exploited.

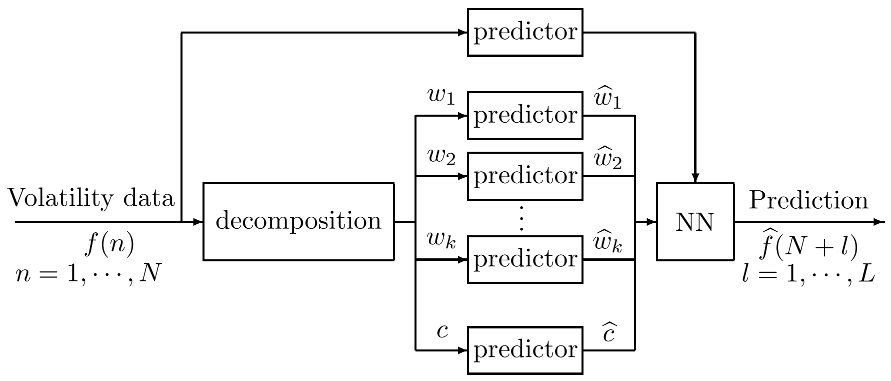
ARD simply approximates the posterior distribution over weights by a Gaussian distribution. The optimization of the regularization parameters is interleaved with the training of the neural network weights. These parameters are divided into classes $\{c\}$, with independent scales $\{\alpha_c\}$. For a network with one hidden layer, the weight classes are: one class for each input; one class for the biases; and one class for each output. Assuming a Gaussian prior for each class, we define $E_{W(c)} = \sum\limits_{i \in c} w_i^2/2$, then the ARD model uses the prior of equation

$$P = (\{w_i\}|\{\alpha_c\}, \mathcal{H}_{\mathcal{ARD}}) = \frac{1}{\Pi Z_{W(c)}} exp(-\sum_c \alpha_c E_{W(c)}). \tag{8}$$

The evidence framework can be used to optimize all the regularization constants simultaneously by finding their most probable value, i.e. the maximum over $\{\alpha_c\}$ of the evidence, $P = (D|\{\alpha_c\}, \mathcal{H}_{\mathcal{ARD}})$.

2.3 Wavelet-Neural Forecasting Model

The proposed prediction model is shown in Fig. 1. Given the time series $f(n), n = 1, \cdots, N$, the aim is to predict the l-th sample ahead, $f(N + l)$. For each value of l a separate prediction architecture is trained accordingly. The hybrid scheme basically involves three stages [2]. First, the time series is decomposed into different scales by autocorrelation shell decomposition; second, each scale is predicted by a separate NN; and at the third stage, the next sample of the original time series is predicted by another NN using the different scale's prediction.



$w_1, \cdots, w_k$ are wavelet coefficients, and c is residual series.

Fig. 1. The forecasting system

The time-based *à trous* transform as described above provides a simple but robust approach. This approach based on ASR is realized by applying (6) and (7) to successive values of t. For example, given currency options volatility series of $N + L$ values, we hope to extrapolate into the future with 1 to L subsequent values. By the time-based *à trous* transform, we simply carry out a wavelet transform on values $x_1, \cdots, x_N$. The last values of the wavelet coefficients at time-point $t = N$ are kept because they are the most critical values for prediction. Repeat the same procedure at time point $t = N+1, N+2, \cdots, N+L$ repeatedly. We empirically determine the number of resolution levels J, mainly depending on the inspection of smoothness of the residual series for a given J. Prior to forecasting, we get an over complete, transformed and normalized dataset.

Secondly, a predictor is allocated for each resolution level and the wavelet coefficients $w_i^j(t)$; $j = 0, \cdots, J; i = 1, \cdots, N$ are used to train the predictor. All networks used to predict the wavelet coefficients of each scale are of the same feed forward multi-layer perceptron(MLP) with D input units, one hidden layer with K sigmoid neurons, and one linear output neuron. In the proposed models, each network is trained by back propagation algorithm using the Scale Conjugate Gradient method and a weight decay regularization of the form $(1/\lambda)\sum_i w_i^2$ [2].

The procedure for designing neural network structure essentially involves selecting the input, hidden and output layers. ARD is used to empirically decide the number of inputs in each resolution level. In all of the experiments performed, the number of hidden neurons is estimated as half of the sum of inputs plus outputs. The number of training vectors available dictates an upper limit on the weight number. A rough guideline recommends that the number of training vectors should be ten times or more the number of weights.

Thirdly, the predicted results of all the different scales $\widehat{w}_{N+i}^j(t), j = 0, \cdots, J$ are appropriately combined. This paper discussed and compared four methods.

In the first method, we simply applied the linear additive reconstruction property of the *à trous* transform, as expressed in (7). The fact that the reconstruction is additive allows the predictions to be combined in an additive manner. This is denoted as *à trous* wavelet prediction scheme in the sequel.

A hybrid strategy can also be empirically applied to determine what should be combined to provide an overall prediction. In proposed first hybrid method, the predicted results of all the different scales are linearly combined by a single-layer perceptron. In order to improve the prediction accuracy, a MLP with the same structure as for wavelet coefficients prediction is employed for currency options volatility series and the corresponding prediction results are incorporated into the third stage, as shown in Fig. 1.

In experiments, we also applied a third stage MLP in place of the simple perceptron. The number of hidden neurons is also estimated as half of $\{\#inputs + \#outputs\}$. We denote this as second hybrid method for the combination of prediction results from the second stage. This method uses the second stage MLP for recombining wavelet coefficients prediction.

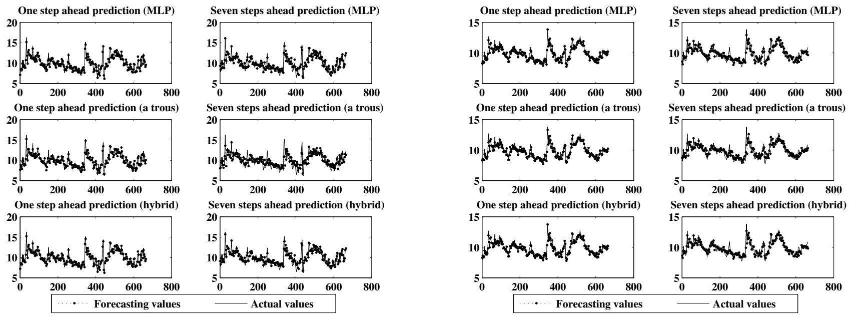
For comparison purpose, a plain MLP was also trained and tested for original time series, denoted as MLP, without any wavelet preprocessing involved.

3 Experimental Results

In this section, the proposed forecasting model is tested with the weekly and monthly USD/Yen options volatility data. The market data covers the year from 2002 to 2006, and are taken from the British Banker's Association website.The number of data patterns is 1000.The training data is taken as the data between 1 and 700, while the testing data covers the period between 701 and 960.

3.1 The Currency Options Volatility Forecasting

The simulation was performed based on historical data of volatility in FX markets. The hybrid model for decomposition and forecast employs six sub-NNs and a seventh NN for summation of the decomposed signals. Examples of forecasting profiles from the feed forward perceptron (MLP), *à trous* wavelet and the second hybrid system, together with actual volatility, are given in Fig. 2 and Fig. 3. As shown in figures, the forecasts are accurate at a satisfactory level, which reveals the effectiveness of the proposed. The predictions from the two hybrid schemes are similar, so we only illustrated results from the second one.



(a) weekly volatility forecasting (b) monthly volatility forecasting

Fig. 2. Forecasting the currency option volatility on training data set

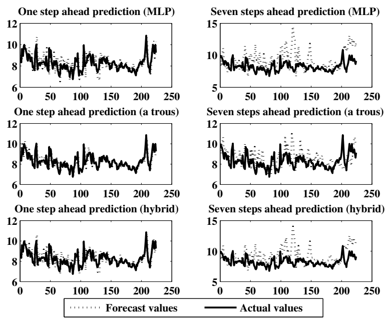
3.2 Evaluating the Prediction Performance

The final step in the design procedure is the assessment of the forecasting performance of the introduced models. Forecasting errors have considerable implications for the forecasting performance of models. Various error metrics between the actual and forecasted volatility have been defined in the literature [10]. This paper adopted normalized mean squared error (NMSE) as the error measure. It is defined as:

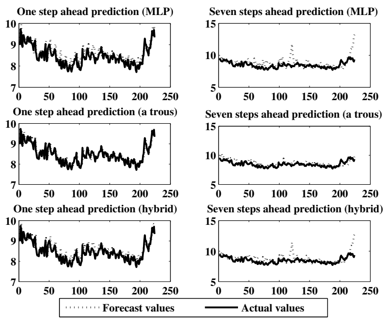
$$NMSE = \frac{1}{N}\sum_{k=1}^{N}(f_k - \widehat{f_k})^2 \tag{9}$$

where f_k and $\widehat{f_k}$ is the actual and forecasted volatility at time k, respectively.

NMSE is more suitably applied to evaluate the forecasting performance of all four models over the forecasting range of interests. The NMSE values from



(a) weekly volatility forecasting



(b) monthly volatility forecasting

Fig. 3. Forecasting the currency option volatility on a testing data set

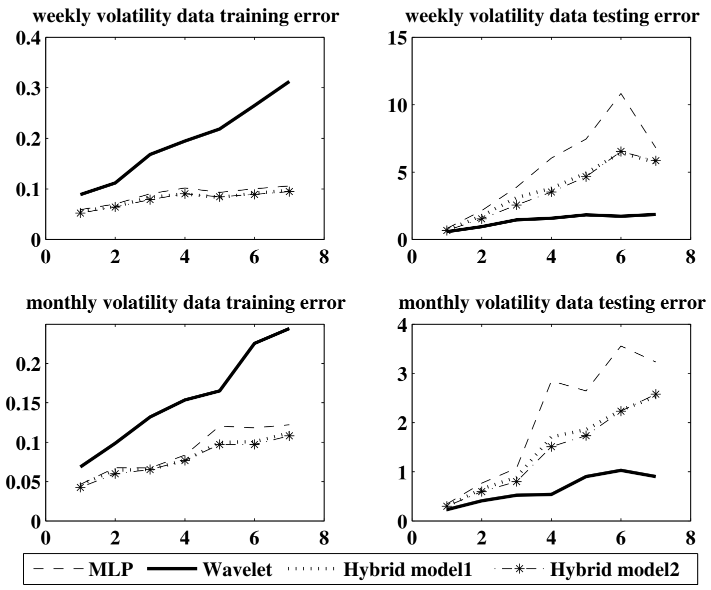


Fig. 4. The normalized mean squared error from different models with two data type

the different forecasting models are presented in Fig. 4. This figure shows that the wavelet only model is the best forecasting performance on testing dataset among four models, with regards to the least NMSE value. On the other hand, the other three modes occur over-fitting problems.

4 Conclusions

Currency options volatility short term forecast is important for option pricing and risk management in the FX markets. Proper volatility forecast helps the market participants to maximize their profits and/or reduce their possible losses. Traditional statistics based linear regression methods need modification to capture the more and more nonlinearities in volatility signals under the market conditions. This paper introduced a hybrid neural-wavelet scheme to predict currency options volatility in short term. Wavelet decomposition techniques are discussed and used to gain deeper insight into the volatility data series, and, therefore, get better prediction results. Different techniques are used to implement the prediction model considering its different components. ARD method dealing with neural network length of training data window is also discussed.

The proposed techniques have been tested with the weekly and monthly USD/Yen options volatility market data series with promising results. Different approaches for volatility forecast are also performed on the same volatility series for comparison to show the effectiveness of the forecast models. According to Fig. 4, wavelet only model is the best forecasting performance on testing dataset among four models, with regards to the least NMSE values. Therefore, wavelet only model outperforms the other three models in forecasting volatility.

Under the FX market situation, the currency options price signal is more appealing to market players. The further research is study how to apply the volatility forecasting values into pricing currency options.

References

1. Zapart, C.: Stochastic Volatility Options Pricing with Wavelets and Artificial Neural Networks. Quantitative Finance **2(6)** (2002) 487–495
2. Bai-Ling, Z., Zhao-Yang, D.: An Adaptive Neural-Wavelet Model for Short Term Load Forecasting. Electric Power Systems Research **59** (2001) 121–129
3. Beylkin, G., Satio, N.: Wavelets, their Autocorrelation Functions and Multiresolution Representation of Signals. IEEE Trans. Signal Processing **7** (1997) 147–164
4. Satio, N., Beylkin, G.: Multiresolution Representations Using the Auto-correlation Functions of Compactly Supported Wavelets. IEEE Trans. Signal Processing **41(12)** (1993) 3584–3590
5. Coifman, R.R., Donoho, D.L.: Translation-Invariant De-noising. In: Antoniades, A., Oppenheim, G.(Eds.): Wavelets and Statistics. Lecture Notes in Statistics. Springer-Verlag, Berlin Heidelberg New York (1995) 125–150
6. Sari-Sarraf, H., Brzakovic, D.: A shift-invariant discrete wavelet transform. IEEE Trans. Signal Processing **45(10)** (1997) 2621–2630
7. Aussem, A., Campbell, J., Murtagh, F.: Wavelet-based Feature Extraction and Decomposition Strategies for Financial Forecasting. Journal of Computational Intelligence in Finance **6** (1998) 5–12
8. MacKay, D.J.C.: Bayesian Non-linear Modelling for the Energy Prediction Competition. ASHRAE Transcations **4** (1994) 448–472
9. MacKay, D.J.C.: Bayesian Non-linear Modelling for the Prediction Competition. ASHRAE Transcations **100** (1994) 1053–1062
10. Makridakis, S., Wheelwright, S.C., Hyndman, R.J.: Forecasting, Methods and Applications. 3rd edn. Wiley, New York (1998)

Trend-Weighted Fuzzy Time-Series Model for TAIEX Forecasting

Ching-Hsue Cheng, Tai-Liang Chen, and Chen-Han Chiang

Department of Information Management,
National Yunlin University of Science and Technology,
123, section 3, University Road, Touliu, Yunlin 640, Taiwan, R.O.C.
{chcheng, g9320817, g9323746}@yuntech.edu.tw

Abstract. Time-series models have been used to make reasonably accurate predictions in the areas of weather forecasting, academic enrolment and stock price etc… We propose a methodology which incorporates trend-weighting into the fuzzy time-series models advanced by S.M. Chen and Hui-Kuang Yu. By using actual trading data of Taiwan Stock Index (TAIEX) and the enrolment data of the University of Alabama, we evaluate the accuracy of our trend-weighted, fuzzy, time-series model by comparing our forecasts with those derived from Chen's and Yu's models. The results indicate that our model surpasses in accuracy those suggested by Chen and Yu.

1 Introduction

Each day individual investors, stock fund managers and financial analysts attempt to predict price activity in stock market on the basis of either their professional knowledge or stock analyzing tools. Higher accuracy is most concerned, because more profit will be made if more accurate predictions are given. So, they have, perennially, strived to discover ways to predict stock price accurately.

For more than one decade, different fuzzy time-series models have also been applied to solve various domain problems, such as financial forecasting [3], [5], [16], university enrolment forecasting [1], [12], [13], temperature forecasting, etc... As Dourra (2002) notes, it is common practice to "deploy fuzzy logic engineering tools in the finance arena, specifically in the technical analysis field, since technical analysis theory consists of indicators used by experts to evaluate stock price [2]."

In this paper, a trend-based, fuzzy, time-series model is proposed to improve the forecast accuracy in stock market. In this model, several factors such as the fuzzy relationships of trend weighted, a reasonable universe of discourse, a reliable length of intervals and past patterns of stock prices are all considered together for forecasting. Moreover, three refined processes are employed in the forecasting algorithm. Using one year period of the trading data from the Taiwan Stock Index and, the enrolment of the University of Alabama, as the data sets for training and testing, the results demonstrate that the proposed model outperforms other fuzzy time-series models.

The remaining content of this paper is organized as follows: Section 2 introduces the related literature of fuzzy time-series model; section 3, demonstrates the proposed model and algorithm; section 4 evaluates the model's performance; and section 5 concludes the paper.

2 Related Works

This section briefly reviews the related literature, including two sections: literature reviews of time-series model and fuzzy time-series definitions and algorithm.

2.1 Literature Reviews of Fuzzy Time-Series Model

Fuzzy theory was originally developed to deal with the problems involving human linguistic terms [18], [19], [20]. Time-series methods had failed to consider the application of this theory until fuzzy time-series was defined by Song and Chissom [11]. In 1993, Song and Chissom proposed the definitions of fuzzy time-series and methods to model fuzzy relationships among observations [11]. In the following research, they continued to discuss the difference between time-invariant and time-variant models [13]. Besides these researchers, Chen (1996) proposed another method to applied simplified arithmetic operations in forecasting algorithm rather than the complicated max-min composition operations presented in Song and Chissom' [1].

In time-series model, when unexpected conditions happen, the fluctuations can not be recorded into the historical data immediately. This would probably results in terrible inaccurate forecast by using the out-of-date data. To deal with the problem, a group decision-making method was employed to integrate the subjective forecast values of all decision makers. Fuzzy weighted method was then combined with subjective forecast values to produce the aggregated forecast value.

Huarng (2001) pointed out that the length of intervals affects forecast accuracy in fuzzy time-series and proposed a method with distribution-based length and average-based length to reconcile this issue [6]. The method applied two different lengths of intervals to Chen's model and the conclusions showed that distribution-based and average-based lengths could improve the accuracy of forecast. Although this method has excellent performance, it creates too many linguistic values to be identified by analysts. According to Miller (1956), establishing linguistic values and dividing intervals would be a trade off between human recognition and forecasting accuracy [8].

It becomes apparent that the major drawback of these methods is the lack of consideration in determining a reasonable universe of discourse and the length of intervals. Moreover, the researchers find that the neglected information, which indicates the patterns of trend changes in history, should be considered in the processes of forecasting. To reconcile these problems above, a new methodology is hereby proposed.

2.2 Fuzzy Time-Series Definitions and Algorithm

Over the past fourteen years, many fuzzy time-series models have been proposed by following Song and Chissom's definitions [11]. Among these models, Chen's model is very conventional one because of easy calculations and good forecasting performance [1]. Therefore, Song and Chissom's definitions and the algorithm of Chen's model are used for illustrations as following:

Definition 1: fuzzy time-series

Let $Y(t)(t=\ldots,0,1,2,\ldots)$, a subset of real numbers, be the universe of discourse by which fuzzy sets $f_j(t)$ are defined. If $F(t)$ is a collection of $f_1(t), f_2(t)\ldots$ then $F(t)$ is called a fuzzy time-series defined on $y(t)$.

Definition 2: fuzzy time-series relationships

Assuming that $F(t)$ is caused only by $F(t-1)$, then the relationship can be expressed as: $F(t)=F(t-1)*R(t,t-1)$, which is the fuzzy relationship between $F(t)$ and $F(t-1)$, where * represents as an operator. To sum up, let $F(t-1)=A_i$ and $F(t)=A_j$. The fuzzy logical relationship between $F(t)$ and $F(t-1)$ can be denoted as $A_i \to A_j$ where A_i refers to the left-hand side and A_j refers to the right-hand side of the fuzzy logical relationship. Furthermore, these fuzzy logical relationships can be grouped to establish different fuzzy relationships.

The Algorithm of Chen's model

Step 1: Define the universe of discourse and intervals for rules abstraction. Based on the issue domain, the universe of discourse can be defined as: U = [*starting*, *ending*]. As the length of interval is determined, U can be partitioned into several equal length intervals.

Step 2: Define fuzzy sets based on the universe of discourse and fuzzify the historical data.

Step 3: Fuzzify observed rules.For example, a datum is fuzzified to A_j if the maximal degree of membership of that datum is in A_j.

Step 4: Establish fuzzy logical relationships and group them based on the current states of the data of the fuzzy logical relationships.

For example, $A_1 \to A_2$, $A_1 \to A_1$, $A_1 \to A_3$, can be grouped as: $A_1 \to A_1, A_2, A_3$.

Step 5: Forecast. Let $F(t-1)=A_i$ .Case 1: There is only one fuzzy logical relationship in the fuzzy logical relationship sequence. If $A_i \to A_j$, then $F(t)$, forecast value, is equal to A_j.Case 2: If $A_i \to A_i, A_j, \ldots, A_k$, then $F(t)$, forecast value, is equal to $A_i, A_j, \ldots, A_k$.

Step 6: Defuzzify. Apply "Centroid" method to get the results of this algorithm. This procedure (also called center of area, center of gravity) is the most often adopted method of defuzzification.

3 Trend-Weighted Fuzzy Time-Series Model

In this section, we propose a research model (Fig.1) of trend-weighted, fuzzy, time-series and its algorithm. From our review of the literature, there are two major drawbacks: (1) The lack of consideration in determining a reasonable universe of discourse and the length of intervals; and (2) Many researchers neglect the information, which indicates patterns of trend changes in the past history. In order to reconcile these problems, three refined processes are factored into the model: (1) To define a reliable length of intervals for linguistic values; (2) To classify recurrent fuzzy relationships into three different types of trends and assign a proper weight to individual fuzzy relationships; and (3) To modify the forecasting equation of Chen's model and assign a adaptive value, *alpha* (α), to make the forecast results more reliable.

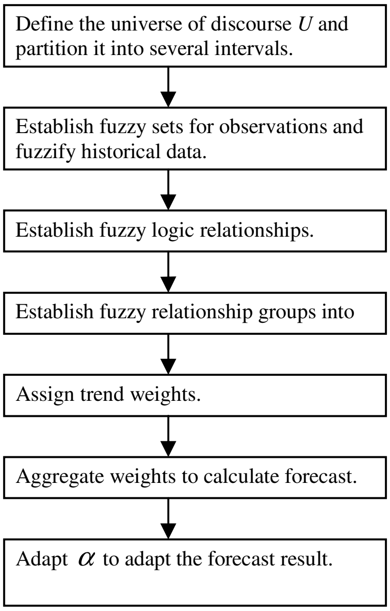

Define the universe of discourse U and partition it into several intervals.
Establish fuzzy sets for observations and fuzzify historical data.
Establish fuzzy logic relationships.
Establish fuzzy relationship groups into
Assign trend weights.
Aggregate weights to calculate forecast.
Adapt α to adapt the forecast result.

Fig. 1. Research model for trend-weighted fuzzy time-series

Table 1. Assign weights to different trends

($t = 1$) $A_1 \to A_1$	No change	Assign weight 1
($t = 2$) $A_1 \to A_2$	Up trend	Assign weight 1
($t = 3$) $A_2 \to A_1$	Down trend	Assign weight 1
($t = 4$) $A_1 \to A_1$	No change	Assign weight 2
($t = 5$) $A_1 \to A_1$	No change	Assign weight 3
($t = 6$) $A_1 \to A_3$	Up trend	Assign weight 1

* t denotes time point

Initially, in the first refined process, the universe of discourse should be partitioned into seven linguistic values [8], and if the data amount of a given linguistic value is larger than the average amount, then the original linguistic value should be further partitioned in half. Because the data occur more frequently in the linguistic value, using once-divided linguistic value to present the data is supposed to be less reliable than twice-divided. However, it would be undesirable to create too many linguistic values and, thereby, ignoring the meaning of fuzzy application.

The second comes from the belief of the researchers that classifying these relations into three different types of trends should enhance the performance of prediction. Traditionally, fuzzy relationship weights are determined either based on knowledge which could be elicited from domain experts or their chronological order. Since each fuzzy relationship will reoccur, from the researcher's perspective, classifying them into different trends and converting the counts of trends to incremental weights is reasonable for making more accurate predictions, hence, the trend-weighted method. The details describing the assignment of weights are listed in Table 1. For example, it is clear that among the Fuzzy Logical Relationships (FLRs), when t = 5 (t denotes time point), then it is assigned the highest value of 3, which means that the probability of its appearance in the near future is 3 times higher than in any of the other cases. The merits of the trend-weighted model are that they can foresee the cycles and events which will eventually occur and relate the fuzzy relationship in a more reasonable manner.

The third is to modify forecasting equation with a proper *alpha* (α) value. Here the value represents the confidence level of the investors represented in the whole data set, ranging from 0~1 but not equal to 0. If the investor is very confident in the predicted variant, then α is assigned 1; conversely, if the investor is cautious, 0.1 may be assigned. By way of summarization, a detailed algorithm for the proposed model is illustrated below:

Step 1: Define the universe of discourse and partition it into intervals. By the problem used in forecasting, the universe of discourse for observations is defined as: U = [*starting*, *ending*]. Then the average datum that should be in each linguistic value may be calculated. The linguistic value, which the amount of the data falling in is larger than the average amount of all linguistic values, should further be split into smaller linguistic values by dividing them into two.

Step 2: Establish fuzzy sets for observations. Each linguistic observation A_i can be defined by the intervals: $u_1, u_2, \ldots, u_n$. Each A_i can be represented as following equation (1). And the value, k_j, is determined by the situation as follows: if $j= i-1$, then $k_j = 0.5$; if $j = i$, then $k_j = 1$; if $j= i+1$,then $k_j = 0.5$; elsewhere $k_j = 0$;

$$A_i = \sum_{j=1}^{n} k_j / u_j \tag{1}$$

Step 3: Establish fuzzy relationships. Two consecutive fuzzy sets $A_i(t-1)$ and $A_j(t)$ can be established into a single FLR as $A_i \to A_j$

Step 4: Establish fuzzy relationship groups into corresponding trends. The FLRs with the same LHSs (left hand sides) can be grouped to form a FLRG. For example, $A_i \to A_j$, $A_i \to A_k$, $A_i \to A_m$ can be group as $A_i \to A_j, A_k, A_m$.

Step 5: Assign trend weights. The FLRs are grouped into the trend to which they belong. For example, $A_1 \to A_2$, will be grouped into the "up trend," $A_1 \to A_1$ into the "no-change trend," and $A_2 \to A_1$, into the "down trend". These weights should be standardized to obtain the weight matrix, $W(t) = [W_1', W_2', \ldots, W'_j]$.The standardized weight matrix equation is defined in equation (2).

$$W(t) = [W_1', W_2', \ldots, W'_j] = \left[\frac{W_1}{\sum_{k=1}^{j} W_k}, \frac{W_2}{\sum_{k=1}^{j} W_k}, \ldots, \frac{W_k}{\sum_{k=1}^{j} W_k} \right] \tag{2}$$

For example, if the forecast data is Table 1, the weights are specified as follows: $W_1 = 1$, $W_2 = 1$, $W_3 = 2$, $W_4 = 3$, $W_5 = 1$. By equation (2), the weight matrix, $W(t)$, is determined as following equation (3) :

$$W(t)=\left[\frac{1}{1+1+2+3+1},\frac{1}{1+1+2+3+1},\frac{2}{1+1+2+3+1},\frac{3}{1+1+2+3+1},\frac{1}{1+1+2+3+1}\right] \tag{3}$$

Step 6: Calculate forecast value. From step 5, we can obtain the standardized weight matrix. Hereby, apply equation (4) to generate the forecast value ($L_{df}(t-1)$ is deffuzified matrix, $W_n(t-1)$ is weight matrix).

$$Forecast\,(t) = L_{df}\,(t-1)\cdot W_n(t-1) \tag{4}$$

Step 7: Apply α to adapt the forecast value. The adapted-forecast equation (5) is generated from the modified forecast equation (4).

$$Adapted\ \_\ forecast\ (t) = Actual\ (t-1) + \alpha(Forecast\ (t) - Actual\ (t-1)) \tag{5}$$

4 Verifications and Comparisons

To illustrate the forecasting performance of the model, one period of the trading data from Taiwan Stock Exchange Capitalization Weighted Stock Index (TAIEX) and an open data set, the enrolment of the University of Alabama, are used to perform it.

Table 2. Comparisons of the forecast results for university enrolments with different models

Year	Actual Enrolment data	Song & Chissom [1993]	Sullivan& Woodall [1994]	Chen [1996]	Lee [1996] (w=2)	Proposed Model
1971	13055					
1972	13563	14000	13500	14000		13680.5
1973	13867	14000	14500	14000	13860	13731.3
1974	14696	14000	14500	14000	13964	13761.7
1975	15460	15500	15231	15500	14710	15194.6
1976	15311	16000	15563	16000	15452	15374.8
1977	15603	16000	15563	16000	15311	15359.9
1978	15861	16000	15500	16000	15603	16410.3
1979	16807	16000	15500	16000	15861	16436.1
1980	16919	16813	16684	16833	16830	17130.7
1981	16388	16813	16684	16833	16919	17141.9
1982	15433	16789	15500	16833	16388	15363.8
1983	15497	16000	15563	16000	15417	15372.1
1984	15145	16000	15563	16000	15497	15378.5
1985	15163	16000	15563	16000	15145	15343.3
1986	15984	16000	15563	16000	15163	15345.1
1987	16859	16000	15500	16000	15984	16448.4
1988	18150	16813	16577	16833	16862	17135.9
1989	18970	19000	19500	19000	18122	18915.0
1990	19328	19000	19500	19000	18970	18997.0
1991	19337	19000	19500	19000	19091	19032.8
1992	18876	*	*	19000	19101	19033.7
Average Error		3.22%	2.66%	3.11%	2.95%	2.087%
MSE		423027	386055	407507	377728	192084.3

Note: * denotes No answer

4.1 Forecasting for University Enrolment

The open data set, the enrolment of the University of Alabama, is used to verify the researchers' model. The comparisons with different models, Chen's model, Song and Chissom's, Sullivan and Woodall's, Lee's and the proposed model (Using $\alpha = 0.9$ to generate the forecast results), are listed in Table 2.

Table 3. Comparisons of the forecast results for TAIEX with different models

Date	Actual	Chen's Model	Forecast Error	Yu's Model	Forecast Error	Proposed Model	Forecast Error
2000/11/2	5,626.08	5300	5.80%	5340.0	5.08%	5463.85	2.88%
2000/11/3	5,796.08	5750	0.80%	5721.7	1.28%	5644.80	2.61%
2000/11/4	5,677.30	5450	4.00%	5435.0	4.27%	5797.80	2.12%
2000/11/6	5,657.48	5750	1.64%	5721.7	1.13%	5690.90	0.59%
2000/11/7	5,877.77	5750	2.17%	5721.7	2.66%	5673.06	3.48%
2000/11/8	6,067.94	5750	5.24%	5760.0	5.07%	5871.32	3.24%
2000/11/9	6,089.55	6075	0.24%	6062.0	0.44%	6042.47	0.77%
2000/11/10	6,088.74	6075	0.23%	6062.0	0.43%	6061.92	0.44%
2000/11/13	5,793.52	6075	4.86%	6062.0	4.64%	6061.19	4.62%
2000/11/14	5,772.51	5450	5.59%	5435.0	5.85%	5795.50	0.40%
2000/11/15	5,737.02	5450	5.00%	5435.0	5.26%	5776.59	0.69%
2000/11/16	5,454.13	5450	0.08%	5435.0	0.35%	5744.65	5.33%
2000/11/17	5,351.36	5300	0.96%	5340.0	0.21%	5409.92	1.09%
2000/11/18	5,167.35	5350	3.53%	5350.0	3.53%	5317.42	2.90%
2000/11/20	4,845.21	5150	6.29%	5150.0	6.29%	5151.81	6.33%
2000/11/21	5,103.00	4850	4.96%	4850.0	4.96%	4861.89	4.72%
2000/11/22	5,130.61	5150	0.38%	5150.0	0.38%	5093.90	0.72%
2000/11/23	5,146.92	5150	0.06%	5150.0	0.06%	5118.75	0.55%
2000/11/24	5,419.99	5150	4.98%	5150.0	4.98%	5213.56	3.81%
2000/11/27	5,433.78	5300	2.46%	5340.0	1.73%	5459.32	0.47%
2000/11/28	5,362.26	5300	1.16%	5340.0	0.42%	5391.60	0.55%
2000/11/29	5,319.46	5350	0.57%	5350.0	0.57%	5327.23	0.15%
2000/11/30	5,256.93	5350	1.77%	5350.0	1.77%	5288.71	0.60%
2000/12/1	5,342.06	5250	1.72%	5250.0	1.72%	5232.44	2.05%
2000/12/2	5,277.35	5350	1.38%	5350.0	1.38%	5309.05	0.60%
2000/12/4	5,174.02	5250	1.47%	5250.0	1.47%	5250.81	1.48%
2000/12/5	5,199.20	5150	0.95%	5150.0	0.95%	5157.82	0.80%
2000/12/6	5,170.62	5150	0.40%	5150.0	0.40%	5180.48	0.19%
2000/12/7	5,212.73	5150	1.20%	5150.0	1.20%	5154.76	1.11%
2000/12/8	5,252.83	5250	0.05%	5250.0	0.05%	5192.66	1.15%
2000/12/11	5,284.41	5250	0.65%	5250.0	0.65%	5228.75	1.05%
2000/12/12	5,380.09	5250	2.42%	5250.0	2.42%	5257.17	2.28%
2000/12/13	5,384.36	5350	0.64%	5350.0	0.64%	5343.28	0.76%
2000/12/14	5,320.16	5350	0.56%	5350.0	0.56%	5347.12	0.51%
2000/12/15	5,224.74	5350	2.40%	5350.0	2.40%	5289.34	1.24%
2000/12/16	5,134.10	5250	2.26%	5250.0	2.26%	5203.46	1.35%
2000/12/18	5,055.20	5150	1.88%	5150.0	1.88%	5121.89	1.32%
2000/12/19	5,040.25	5450	8.13%	5405.0	7.24%	5050.88	0.21%
2000/12/20	4,947.89	5450	10.15%	5405.0	9.24%	5037.42	1.81%
2000/12/21	4,817.22	4950	2.76%	4950.0	2.76%	4954.30	2.85%
2000/12/22	4,811.22	4850	0.81%	4850.0	0.81%	4836.70	0.53%
2000/12/26	4,721.36	4850	2.72%	4850.0	2.72%	4831.30	2.33%
2000/12/27	4,614.63	4750	2.93%	4750.0	2.93%	4750.42	2.94%
2000/12/28	4,797.14	4650	3.07%	4650.0	3.07%	4654.37	2.98%
2000/12/29	4,743.94	4750	0.13%	4750.0	0.13%	4818.62	1.57%
2000/12/30	4,739.09	4750	0.23%	4750.0	0.23%	4770.74	0.67%
Average error (Testing)			2.43%		2.36%		1.76%

From the Comparisons in Table 2, the average forecasting error of Song and Chissom's model is 3.22%, with an MSE (Mean Square Error) of 423027; Sullivan and Woodall model, the average error is 2.66%, MSE is 386055; Chen's model, the average error is 3.11%, MSE is 407507; Lee's model, the average error is 2.95%, MSE is 377728; and proposed model, the average error is 3.11%, MSE is 407507; It is obvious that our model has a smaller MSE and less average error than the listing models.

4.2 Forecasting for TAIEX

The data from TAIEX during 2000/11/1~2000/12/30 are used for forecasting. The comparisons of Chen's model, Yu's model and the forecast results from the proposed model are listed in Table 3. (Using $\alpha = 0.9$ to generate the forecast results).

From the Comparisons in Table 3, the forecasting error of Chen's model ranges from 0.05% to 10.15%; Yu's model ranges from 0.05% to 9.24%; and proposed model ranges from 0.15% to 6.33%. It is quite obvious that the worst case of our model is much better than either Chen's or Yu's. Furthermore, the average error of Chen's model is 2.43%, Yu's model is 2.36%, and proposed model is 1.76%. Based on these comparisons, the proposed model has a smaller prediction error interval and less average error than the other two models.

5 Conclusions and Future Research

This paper has proposed a trend-weighted, fuzzy, time-series model to overcome the problems mentioned in the literature. Base on the two cases in the verification section the researchers conclude that the refined processes would lead to better performance than the cited models in forecasting. For each step of the process, we can further draw independent conclusions: (1) The second-divided linguistic values seem to be more reliable than once-divided linguistic value, and can detail more closely the distribution of the data in that interval; (2) Classifying recurrent fuzzy relationships and assigning proper weights to them make more reasonable descriptions for the past patterns and more accurate predictions for the future; and (3) the results shows that the forecast value, with reasonable *alpha* value adaptation of the user's opinion, can make more precise adjustments to match the past trends in the data set. However, there is still room for testing and improving the hypothesis of this model as follows:

1. Using the other periods of TAIEX, stocks and financial materials as data sets to evaluate the accuracy and performance of the model.
2. Simulating trading, and sum up the profits of these trades to evaluate the profit making ability.
3. Applying different types of data sets to test the model.
4. Reconsidering the factors affecting the behavior of the stock markets, such as trading volume, news and finical reports which might impact it in the future.

References

1. Chen, S.M.: Forecasting enrollments based on fuzzy time-series, Fuzzy Sets and Systems, 81, (1996) 311-319.
2. Dourra, Hussein and Pepe Siy.: Investment using technical analysis and fuzzy logic, Fuzzy Sets and Systems, 127, (2002) 221-240.

3. Faff, R.W., R.D. Brooks and Ho Yew Kee.: New evidence on the impact of financial leverage on beta risk: A time-series approach, North American Journal of Economics and Finance, 13, (2002) 1-20.
4. Huarng, Kunhuang.: Effective lengths of intervals to improve forecasting in fuzzy time-series, Fuzzy Sets and Systems, 123, (2001) 387-394.
5. Huarng, Kunhuang and Hui-Kuang Yu.: A Type 2 fuzzy time-series model for stock index forecasting, Physica A, 353, (2005) 445-462.
6. Huarng, K.: Heuristic models of fuzzy time-series for forecasting, Fuzzy Sets and Systems, 123, (2001) 137-154.
7. Hwang, J.R., S.M. Chen and C.H. Lee.: Handling forecasting problems using fuzzy time-series, Fuzzy Sets Systems, 100, (1998) 217-228.
8. Miller, George A.: The magical number seven, plus or minus two: some limits on our capacity of processing information, The Psychological Review, 63, (1956) 81-97.
9. Ross, Timothy J.: Fuzzy Logic with Engineering Applications. New York: McGraw-Hill, 1995.
10. Shin, Hyungwon and So Young Sohn.: Segmentation of stock trading customers according to potential value, Expert Systems with Applications,27, (2004) 27-33.
11. Song, Q. and Chissom, B.S.: Fuzzy time-series and its models, Fuzzy Sets and Systems, 54, (1993) 269-277.
12. Song, Q. and B.S. Chissom.: Forecasting enrollments with fuzzy time-series – Part Ⅰ, Fuzzy Sets and Systems, 54, (1993) 1-10.
13. Song, Q. and B.S. Chissom.: Forecasting enrollments with fuzzy time-series – Part Ⅱ, Fuzzy Sets and Systems, 62, (1994) 1-8.
14. Tsaur, Ruey-Chyn, Jia-Chi O Yang and Hsiao-Fan Wang.: Fuzzy Relation Analysis in Fuzzy Time-series Model, Computers and Mathematics with Applications, 49, (2005) 539-548.
15. Wang, Y.F.: Predicting stock price using fuzzy grey prediction system, Experts Systems with Applications, 22, (2002) 33-39.
16. Yu, Hui-Kuang.: Weighted fuzzy time-series models for TAIEX forecasting, Physica A, 349, (2004) 609-624.
17. Yu, Hui-Kuang.: A refined fuzzy time-series model for forecasting, Physica A, 346, (2005) 657-681.
18. Zadeh, L.A.: The concept of a linguistic variable and its application to approximate reasoning Ⅰ, Information Science, 8, (1975) 199-249.
19. Zadeh, L.A.: The concept of a linguistic variable and its application to approximate reasoning Ⅱ, Information Science, 8, (1975) 301-357.
20. Zadeh L.A.: The concept of a linguistic variable and its application to approximate reasoning Ⅲ, Information Science, 9, (1976) 43-80.

Intelligence-Based Model to Timing Problem of Resources Exploration in the Behavior of Firm

Hsiu Fen Tsai[1,*] and Bao Rong Chang[2]

[1] Department of International Business
Shu-Te University, Kaohsiung, Taiwan
soenfen@mail.stu.edu.tw
Tel.: +886-7-6158000 ext. 3315; Fax: +886-7-6158001
[2] Department of Computer Science and Information Engineering
National Taitung University, Taitung, Taiwan
brchang@nttu.edu.tw

Abstract. We have insight into the importance of resource exploration derived from the quest for sustaining competitive advantage as well as the growth of the firm, which are well-explicated in the resources-based view. However, we really do not know when the firm will seriously commit to this kind of activities. Therefore, this study proposes an intelligence-based model using quantum minimization (QM) to tune a composite model of adaptive neuron-fuzzy inference system (ANFIS) and nonlinear generalized autoregressive conditional heteroscedasticity (NGARCH) such that it constitutes the relationship among five indicators, the growth rate of long-term investment, the firm size, the return on total asset, the return on common equity, and the return on sales. In particularly, this proposed approach outperforms several typical methods such as auto-regressive moving-average regression (ARMAX), back-propagation neural network (BPNN), or adaptive support vector regression (ASVR) for this timing problem in term of comparing their achievement and the goodness of fit. Consequently, the preceding methods involved in this problem truly explain the timing of resources exploration in the behavior of firm. Meanwhile, the performance summary among methods is compared quantitatively.

1 Introduction

When we think about the firm as a collective of resources [1], it drives the different aspects of research directions to answer two fundamental strategic questions: the sources of competitive advantage and the growth trajectory of firm. There is fairly general agreement that the accumulation of heterogeneous resources can explain the success of firm for a period of time [2][3][4][5][6][7]; it also shapes the path of firm's growth [1][8]. However, resources or capabilities, like product, have life cycles [9]. Thus, researchers always remind us the importance of exploring new resources due to the pressure of external changing environment [1][8][10].

[*] Corresponding author.

We know the importance of resource exploration derived from the quest for sustaining competitive advantage as well as the growth of the firm that are well-explicated in the resources-based view. It is worth understanding that the idea of balance between exploration and exploitation will be solely achieved under the assumption of calculated rationality. However, we in fact do not know when the firm takes it into account and commits itself to the exploring activities. In each occasion of decision-making, decision makers are constrained by bounded rationality [11][12][13] which is raised from two facts: (a) managers have limited absorptive capacity, and (b) managers acquired finite information subjected to the external changing environment. Therefore, the managers might miss or postpone the exact timing of exploring activities because of their inability on controlling the future uncertainty under the situation of limited rationality. Accordingly, the reinforcement of the precedent tendency mentioned above will be stressed by the conservative personality of managers. All of these will leads the managers to persist on the exploiting the existing resources rather than exploring new ones.

So, we concern the timing problem, namely when the risky attitude of managers will be shifted from risk-avoiding exploiting activities to relative risk-taking exploring activities. In the basis of the prospect theory [14], we argue that the turning point will be triggered by the negative prospects. That is, when the firm is framed by positive performance, it will incline the managers to utilize the existing resources and neglect the need of exploring new ones. On the other hand, when the firm suffers from loss or decline in performance, it will reverse the risky attitude of decision makers to approach risk-taking considerably that will ignite more exploring activities. And then, we can observe that the trajectory of the growth of firm is emerging with the exploration and the following exploitation and so on [1][15]. In the mean time, we also proposed that large firm holds much more resources than small one [16] in this case that leads to the large firm's 'value function' [14] is flatter than small firm. Therefore, large firm has revealed low risk-aversion so as to potentially proceed to higher exploring activities in positive frames; in contrast, small firm with the emergence of high risk-seeking in negative frames will then undertake more exploring activities.

There are five indicators that are the growth rate of long-term investment, the firm size, the return on total asset, the return on common equity, and the return on sales. The relationship among these indicators indeed can be used to analyze the timing of resources exploration in the behavior of firm. Several quantitative methods, such linear time-series models with single-output, multi-input structure as AR, MA, ARX, ARMA, and ARMAX [17][18], are applicable to the simulation of the dynamics of the interaction between five indicators. Once a trained structure is built, it interprets the timing of resources exploration in the behavior of firm based on the coefficients with respective to explanatory variables. However, a trained ARMAX [19] do not get the least mean-absolute-percent-error for the timing problem. Thus, four remarkable nonlinear models, adaptive neuro-fuzzy inference system (ANFIS) [20], back-propagation neural network (BPNN) [21], adaptive support vector regression (ASVR) [22], and quantum minimization [23] tuning ANFIS/NGARCH composite model [24] are also provided in this study so that the performance comparison among methods for the timing problem is compared quantitatively.

2 Methods

2.1 ANFIS Inference Model

Adaptive neuro-fuzzy inference system (ANFIS) [20] is a remarkable Sugeno-type fuzzy inference machine and we herein apply and treat it as a non-periodic short-term predictor to forecasting the flow of data packets among hosts. A single Sugeno rule is represented as the diagram in Fig. 1. As a matter of fact, a fuzzy rule with crisp input signals x and y is formulated on Eq. (1).

$$\textit{If Input}1 = x \textit{ and Input}2 = y, \textit{ then Output } z_i = ax + by + c\ ,\ \ i = 1,2,...,N_r\ , \tag{1}$$

where the output signal z_i is designated as a linear function of input signals x and y. Two fuzzy membership function values $F_1(x)$ and $F_2(y)$, with respective to two input signals x and y, implement a "AND" function and result in a weight value w_i that is so-called firing strength as expressed on Eq. (2),

$$w_i = AND(F_{i1}(x), F_{i2}(y))\ ,\ \ i = 1,2,...,N_r\ . \tag{2}$$

The final output Z_{final}, during the defuzzification, is evaluated by a weighted average through N fuzzy rules as listed below.

$$Z_{final} = \sum_{i=1}^{N_r} w_i z_i \Big/ \sum_{i=1}^{N} w_i\ . \tag{3}$$

Autoregressive moving-average (ARMA) [18] and artificial neural network (ANN) [21] are frequently employed to forecast time series, but not suitable for modeling the short-term structure with fewer data because they need a lot of data to train their structure in-depth. ANFIS is capable of building the most recent few data to fit the short-term dynamics. However, the occurrence of extreme outlier causes ANFIS output a big residual error when volatility clustering effect [6] happens. Therefore, we introduce a nonlinear conditional heteroscedasticity (NGARCH) [25], which can resolve volatility clustering effect, to compensate ANFIS outputs, and then this composite model is tuned optimally by quantum minimization (QM). Accordingly, we denote this intelligence-based composite model as QM tuning ANFIS/NGARCH approach.

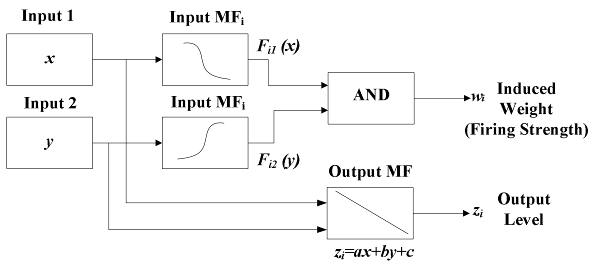


Fig. 1. A single Sugeon fuzzy rule is conducted by adaptive neuro-fuzzy inference system (ANFIS) wherein MF denotes the fuzzy membership function

2.2 ARMAX/NGARCH Composite Model

The ARMAX [19] encompass autoregressive (AR), moving-average (MA), and regression (X) models, in any combinations as expressed below.

$$y_{armax}(t) = C^{armax} + \sum_{i=1}^{r} R_i^{armax} y(t-i) + e_{resid}(t) + \sum_{j=1}^{m} M_j^{armax} e_{resid}(t-j) + \sum_{k=1}^{N_x} \beta_k^{armax} \mathbf{X}(t,k) , \quad (4)$$

where C^{armax} = a constant coefficient, R_i^{armax} = autoregressive coefficients, M_j^{armax} = moving average coefficients, $e_{resid}(t)$ = residuals, $y_{armax}(t)$ = responses, β_k^{armax} = regression coefficients, $\mathbf{X}$ = an explanatory regression matrix in which each column is a time series and $X(t,k)$ denotes a element at the t th row and k th column of input matrix.

The NGARCH [25] consists of nonlinear time-varying conditional variances and Gaussian innovations. Its mathematical formula is shown as follows.

$$\sigma_{ntvcv}^2(t) = K^{ng} + \sum_{i=1}^{p} G_i^{ng} \sigma_{ntvcv}^2(t-i) + \sum_{j=1}^{q} A_j^{ng} \sigma_{ntvcv}^2(t-j) \left[\frac{e_{resid}(t-j)}{\sqrt{\sigma_{ntvcv}^2(t-j)}} - C_j^{ng} \right]^2 \quad (5)$$

with constraints

$$\sum_{i=1}^{p} G_i^{ng} + \sum_{j=1}^{q} A_j^{ng} < 1 , \ K^{ng} > 0 , \ G_i^{ng} \geq 0, \ i = 1,...,p , \ A_j^{ng} \geq 0, \ j = -1,...,q$$

where K^{ng} = a constant coefficient, G_i^{ng} = linear-term coefficients, A_j^{ng} = nonlinear-term coefficients, C_j^{ng} = nonlinear-term thresholds, $\sigma_{ntvcv}^2(t)$ = a nonlinear time-varying conditional variance and $e_{resid}(t-j)$ = j-lag Gaussian distributed residual in ARMAX.

2.3 Quantum Minimization (QM)

Quantum-based minimization (QM) that makes optimization task work out associated with probability of success at least 1/2 within an unsorted database is realized by quantum minimum searching algorithm [23]. A quantum exponential searching algorithm [26] is called by quantum minimum searching algorithm to be as a subroutine to serve a fast database searching engine.

2.3.1 Quantum Exponential Searching Algorithm

As reported in [26], we assume in this section that the number t of solutions is known and that it is not zero. Let $A = \{i \mid F(i) = 1\}$ and $B = \{i \mid F(i) = 0\}$.

Step 1: For any real numbers k and l such that $tk^2 + (N-t)l^2 = 1$, redefine

$$|\Psi(k,l)\rangle = \sum_{i \in A} k \mid i\rangle + \sum_{i \in B} l \mid i\rangle .$$

A straightforward analysis of Grover's algorithm shows that one iteration transforms $|\Psi(k,l)\rangle$ into

$$\left| \Psi\left(\frac{N-2t}{N}k + \frac{2(N-t)}{N}l, \frac{N-2t}{N}l - \frac{2t}{N}k \right) \right\rangle .$$

Step 2: This gives rise to a recurrence similar to the iteration transforms in Grover's algorithm [27], whose solution is that the state $|\Psi(k_j, l_j)\rangle$ after j iterations is given by

$$k_j = \frac{1}{\sqrt{t}}\sin((2j+1)\theta) \ \ and \ \ l_j = \frac{1}{\sqrt{N-t}}\cos((2j+1)\theta) \ .$$

where the angle θ is so that $\sin^2\theta = t/N$ and $0 < \theta \le \pi/2$.

2.3.2 Quantum Minimum Searching Algorithm

We second give the minimum searching algorithm [23] in which the minimum searching problem is to find the index i such that $T[i]$ is minimum where $T[0,...,N-1]$ is to be an unsorted table of N items, each holding a value from an ordered set.

Step 1: Choose threshold index $0 \le i \le N-1$ uniformly at random.

Step 2: Repeat the following stages (2a and 2b) and interrupt it when the total running time is more than $22.5\sqrt{N} + 1.4\lg^2 N$. Then go to stage (2c).

(a) Initialize the memory as $\sum_j \frac{1}{\sqrt{N}} |j\rangle|i\rangle$. Mark every item j for which $T[j] < T[i]$.

(b) Apply the quantum exponential searching algorithm [26].

(c) Observe the first register: let i' be the outcome. If $T[i'] < T[i]$, then set threshold index i to i' .

Step 3: Return i

This process is repeated until the probability that the threshold index selects the minimum is sufficiently large.

3 Intelligence-Based Approach to Problem

In this case, the signal difference (or deviation) of Eq. (6) can provide precious information about the short-run dynamics of the currently applied data sequence. A signal deviation $\delta o(k)$ is defined as the backward difference between two consecutively adjacent observations, $o(k)$ and $o(k-1)$, as

$$\delta o(k) = o(k) - o(k-1) \ . \tag{6}$$

A single-step-look-ahead prediction, as shown in Fig. 2, can be arranged by adding the most recent predicted signal deviation $\delta\hat{o}(k+1)$ of Eq. (7) to the observed current output $o(k)$. The summation results in a predicted output $\hat{o}(k+1)$ at the next period as expressed in Eq. (8) [24]. The function h in Eq. (7) represents a predictor that includes a data preprocessing unit, a QM-ANFIS/NGARCH system and a summation unit, as shown in Fig. 3. A data preprocessing unit is used to calculate signal deviations of Eq. (6) as

$$\delta\hat{o}(k+1) = h(o(k), o(k-1),..., o(k-s), \delta o(k), \delta o(k-1),..., \delta o(k-s)) \tag{7}$$

$$\hat{o}(k+1) = o(k) + \delta\hat{o}(k+1) \tag{8}$$

Let's turn back and examine again QM-ANFIS/NGARCH system as shown in Fig. 3. In order to construct an ANFIS-based prediction, the most recent predicted deviation $\delta\hat{o}(k+1)$ at next period is assigned as the output of the QM-ANFIS/NGARCH system. As shown in Fig. 2, the most recent observations and their deviations, $\{o(k), o(k-1),..., o(k-s), \delta o(k), \delta o(k-1),..., \delta o(k-s)\}$, have been specified as inputs of the QM-AFNG system. Based on the QM-ANFIS/NGARCH structure, one can form the function p of the ANFIS output, $\delta\hat{o}_{anfis}(k+1)$, and the square-root of NGARCH's output, $\hat{\sigma}_{\delta o}(k+1)$, as presented below and shown in Fig. 3.

$$\delta\hat{o}_{qm-afng}(k+1) = p(\delta\hat{o}_{anfis}(k+1), \hat{\sigma}_{\delta o}(k+1)) \tag{9}$$

A weighted-average function is assumed to combine both $\delta\hat{o}_{anfis}(k+1)$ and $\hat{\sigma}_{\delta o}(k+1)$ to attain a near-optimal result $\delta\hat{o}_{qm-afng}(k+1)$.

$$\begin{aligned} \delta\hat{o}_{qm-afng}(k+1) &= w_{anfis} \cdot \delta\hat{o}_{anfis}(k+1) + w_{ngarch} \cdot \hat{\sigma}_{\delta o}(k+1) \\ s.t. \qquad & w_{anfis} + w_{ngarch} = 1 \end{aligned} \tag{10}$$

Here, the linear combination of two nonlinear functions, $\delta\hat{o}_{anfis}(k+1)$ and $\hat{\sigma}_{\delta o}(k+1)$, can also optimally approximate an unknown nonlinear target $\delta\hat{o}_{qm-afng}(k+1)$. The reason for the approach of Eq. (10) is that individual nonlinear function implemented by using soft-computing is fast and effective, speeding convergence and reducing computational time. This proposed approach is called QM-ANFIS/NGARCH as shown in Fig. 3.

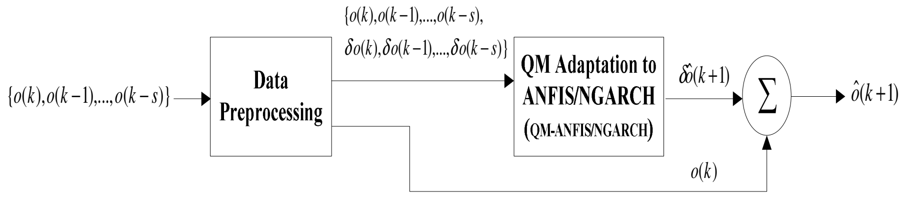

$\{o(k),o(k-1),...,o(k-s)\}$
Data Preprocessing
$\{o(k),o(k-1),...,o(k-s),$
$\delta o(k),\delta o(k-1),...,\delta o(k-s)\}$
QM Adaptation to ANFIS/NGARCH (QM-ANFIS/NGARCH)
$\delta\hat{o}(k+1)$
$\sum$
$\hat{o}(k+1)$
$o(k)$

Fig. 2. Diagram of QM Tuning ANFIS/NGARCH outputs

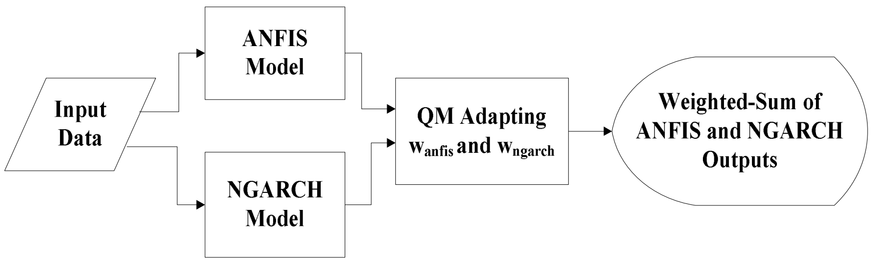

Input Data
ANFIS Model
NGARCH Model
QM Adapting w_{anfis} and w_{ngarch}
Weighted-Sum of ANFIS and NGARCH Outputs

Fig. 3. Prediction using QM-ANFIS/NGARCH system

4 Empirical Simulation and Discussions

A collection of data about five indicators, (i) the growth rate of long-term investment (GRLTI), (ii) the firm size (FS), (iii) the return on total asset (ROA), (iv) the return on common equity (ROE), and (v) the return on sales (ROS), from TEJ [28] including 30 corporations have been cited herein for explaining the timing of resources exploration in the behavior of firm to fit in with the real world dynamics in changing environments. In order to accomplish data manipulation easier, data preprocess is required to transform indicator GRLTI linearly with appropriate bias, and a natural logarithm applied to indicator FS. The first phase designed as training/learning stage for modeling linear structure of ARMAX as well as nonlinear structure of BPNN and ASVR, which is of the posterior analysis from observed 383 historical data for a period of 10 years from 1995 to 2004. Next, the second phase, the prior validation stage proceeded to simulate the empirical results for examining the system performance employing interpolation from trained model. Estimated ARMAX with a bias of scalar (bias= 2000.5) simulated from computer:

$$\begin{aligned} & y(t) = 0.3023 \cdot y(t-1) + 0.1651 \cdot y(t-2) + 0.04129 \cdot y(t-3) \\ & + 0.2023 \cdot y(t-4) + 0.04312 \cdot u_1(t-3) - 0.004114 \cdot u_2(t-2) \\ & - 0.0023 \cdot u_3(t-1) - 0.0056 \cdot u_4(t-3) + e(t) - 0.2227 \cdot e(t-1) \\ & - 0.1648 \cdot e(t-2) \end{aligned} \tag{11}$$

Thus, we can construct an estimated ARMAX model as expressed in Eq. (11). In this estimated ARMAX model, we can check directly from Eq. (11) to explain the current GRLTI is related to it's the most recent four lags of GRLI, as well as coupled to the second lag of FS, the second lag of ROA, the second lag of ROE, and the second lag of ROS. We can interpret that in the respect of autoregressive GRLTI is definitely auto-correlated to a few of most recent historical (the past) GRLTI. Furthermore, there are three indicators ROA, ROE, and ROS playing the roles to affect the current GRLTI negatively. In other words, increasing on ROA, ROE, or ROS will depress the current GRLTI. This development can meet the typical theory in the prospect theory perspective [14]. Nevertheless, an indicator FS can promote the current GRLTI such that the larger FS is, the higher GRLTI will be. It is also noted that strictly speaking the residual terms indicated by $e(t)$, $e(t-1)$, and $e(t-2)$ in MA part have small values usually. The MA part cannot affect GRLTI significantly, even though they have relatively larger coefficients with respect to those terms, $e(t)$, $e(t-1)$, and $e(t-2)$ as shown in Fig. 4. Obviously, MA part is trivial in this ARMAX model as a result of small residuals; in contrast, AR and X part of estimated ARMAX model are used to determine GRLTI predominantly and their corresponding coefficients is displayed as shown in Fig. 5. As a matter of fact, the most recent lags of GRLTI are essentially related to the performance of GRLTI, and secondly we must also take FS into account when we examine the changes in GRLTI.

The performance criteria [29] based on mean square error (MSE), mean absolute deviation (MAD), and mean absolute percent error (MAPE) will be used to compare the performance of empirical simulation among models (ARX, ARMAX, ANFIS, BPNN, ASVR, and QM-ANFIS/NGARCH) as listed in Table 1. Nonlinear models get lower MSE, MAD, and MAPE than the linear ones, ARX and ARMAX. This is because MAPE for ARX and ARMAX always cannot less than 0.3% even varying with different biases. It implies that the accuracy of empirical simulation is not enough for

ARX and ARMAX. On the contrary, nonlinear models like ANFIS, BPNN and ASVR obtain higher accuracy in empirical simulation. Furthermore, as listed in Table 1, the goodness of fit for the proposed methods is also tested by Q-test [30], and null hypothesis do not be rejected due to all p-value greater than the level of significance (5%). In other words, all of trained structures are significantly for this timing problem. As for model validation, QM-ANFIS/NGARCH has attained the best Akaike information criterion (AIC) and Bayesian information criterion (BIC) [18], which implies the best reliability for the problem. However, interactions between indicators do not be resolved yet due to none of exact input/output equation to represent any nonlinear structure ANFIS, BPNN, ASVR, or QM-ANFIS/NGARCH. Fortunately, linear structure ARMAX built a representative equation in which we check the correlation between input and output indicators, respectively. That is, this equation can give us new insight into the timing of resources exploration in the behavior of firm.

Table 1. The performance comparison is shown among six approaches

Criteria	ARX	ARMAX	ANFIS	BPNN	ASVR	QM-ANFIS/NGARCH
MSE	0.0023	0.00003991	0.0000092	0.0000073	0.0000065	0.00000086
MAD	0.0101	0.00045029	0.00024341	0.00021757	0.00065275	0.00016435
MAPE	0.0644	0.0038	0.0021	0.0019	0.0016	0.0011
p-VALUE	0.2089	0.9997	0.7414	0.9062	0.9067	0.9331
AIC	-123.5	-107.9914	-215.1323	-344.5377	-1035.3	-1292.1
BIC	-121.9	-92.1992	-199.3402	-328.7455	-1019.5	-1274.3

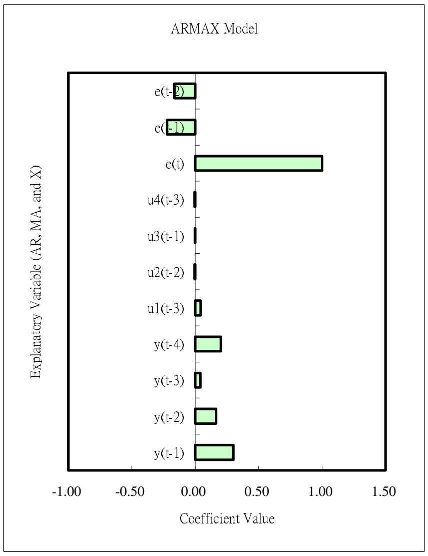


Fig. 4. The coefficients of AR, MA and X parts of ARMAX are displayed

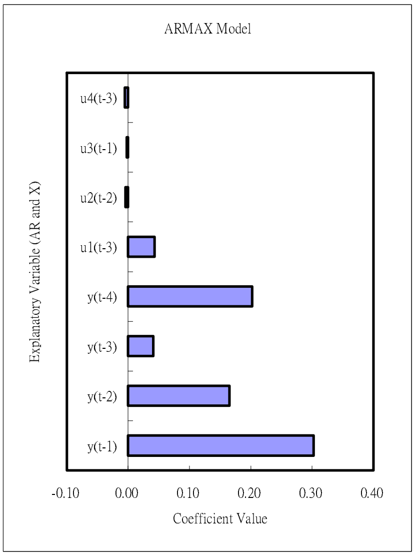


Fig. 5. The coefficients of AR and X parts of ARMAX are emphasized here

5 Conclusions

The following statements summarize the accomplishment of the proposed methods, including intelligence-based models like ANFIS, BPNN, ASVR, and QM-ANFIS/NGARCH. The resulting ARMAX model explains the growth rate of long-term investment, which can help decision-maker to explore new resources due to the pressure of external changing environment. This dynamics also can be considered as the timing when the risky attitude of managers will be shifted from risk-avoiding to relative risk-taking exploring activities. In ARMAX model, the firm size affects the growth rate of long-term investment positively whereas the return on total asset, the return on common equity, and the return on sales influence the growth rate of long-term investment negatively. Clearly, the nonlinear intelligence-based model QM-ANFIS/NGARCH gets the satisfactory results, and improves the goodness of fit better than the linear structure of ARMAX or the nonlinear structure of ANFIS, BPNN or ASVR. However, the nonlinear intelligence-based models, including ANFIS, BPNN, ASVR, and QM-ANFIS/NGARCH, cannot tell us the exact impact to the growth rate of long-term investment from the other individual factors because those are hidden in the nonlinear system.

Acknowledgements

This work is partially supported by the National Science Council, Taiwan, Republic of China, under grant number **NSC 93-2218-E-143-001**.

References

1. Penrose, E.: The Theory of the Growth of the Firm**.** Oxford University Press, Oxford (1959)
2. Rumelt, R. P.: Toward a Strategic Theory of the Firm. In Competitive Strategic Management. 556-570. Lamb RB (Ed), Prentice-Hall, Englewood Cliffs NJ (1984)
3. Barney, J. B.: Types of Competition and the Theory of Strategy: Towards an Integrative Framework. Academy of Management Review. 11 4 (1986) 791-800
4. Barney, J. B.P: Firm Resources and Sustained Competitive Advantage. Journal of Management. 17 1 (1991) 99-120
5. Barney, J. B.: Looking Inside for Competitive Advantage. Academy of Management Executive. 9 4 (1995) 49-61
6. Dierickx, I., Cool, K.: Asset Stock Accumulation and Sustainability of Competitive Advantage. Management Science. 35 12 (1989) 1504-1514
7. Peteraf, M. A.: The Cornerstones of Competitive Advantage: A Resourced-Based View. Strategic Management Journal. 14 (1993) 179-191
8. Wernerfelt, B.: A Resource-Based View of Firm. Strategic Management Journal. 5 (1984) 171-180
9. Helfat, C. E., Peteraf M. A.: The Dynamic Resource-Based View: Capability Lifecycles. Strategic Management Journal. 24 (2003) 997-1010
10. March, J. G.: Exploration and Exploitation in Organizational Learning. Organization Science. 2 (1991) 71-87

11. Cyert, R. M., March J. G.: A Behavioral Theory of the Firm. Prentice-Hall, New Jersey (1963)
12. Levinthal D. A., March J. G.: The myopia of learning. Strategic Management Journal. (1986-1998) 14 (1993) 95-112
13. Simon, H. A.: Administrative Behavior: A Study of Decision-Making Process in Administrative Organization. Free Press, New York (1997)
14. Kahneman, D., Tversky A.: Prospect Theory: An analysis of Decision Under Risk. Econometrica. (1979) 263-291
15. Rothaermel, F. T., Deeds D. L.: Exploration and Exploitation Alliances in Biotechnology: A System of New Product Development. Strategic Management Journal. 25 3 (2004) 201-221
16. Schumpeter J. A.: The Theory of Economic Development. Transaction Publishers. New Brunswick, New Jersey (1934)
17. Hamilton, J.D.: Time Series Analysis. Princeton University Press, New Jersey (1994)
18. Box, G. E. P., Jenkins, G. M., Reinsel, G. C.: Time Series Analysis: Forecasting & Control, Prentice-Hall, New Jersey (1994)
19. Bowerman, B. L., O'Connell, R. T.: Forecasting and Time Series: An Applied Approach, Duxbury Press, Belmont (1993)
20. Jang, J.-S. R.: ANFIS: Adaptive-Network-based Fuzzy Inference Systems. IEEE Transactions on Systems, Man, and Cybernetics. 23 3 (1993) 665-685
21. Haykin, S.: Neural Network: A Comprehensive Foundation, 2nd Ed.. Prentice Hall, New Jersey (1999)
22. Chang, B. R.: Compensation and Regularization for Improving the Forecasting Accuracy by Adaptive Support Vector Regression. International Journal of Fuzzy Systems. 7 3 (2005) 110-119
23. Durr, C., Hoyer, P.: A Quantum Algorithm for Finding the Minimum. http://arxiv.org/abs/quant-ph/9607014
24. Chang, B. R.: Applying Nonlinear Generalized Autoregressive Conditional Heteroscedasticity to Compensate ANFIS Outputs Tuned by Adaptive Support Vector Regression. Fuzzy Sets and Systems. 157 13 (2006) 1832-1850
25. Gourieroux, C.: ARCH Models and Financial Applications. Springer-Verlag, New York (1997)
26. Boyer, M., Brassard, G., Hoyer P., and Tapp A.: Tight Bounds on Quantum Searching. Fortschritte Der Physik (1998)
27. Grover, L. K.: A Fast Quantum Mechanical Algorithm for Database Search. In Proc.28th Ann. ACM Symp. Theory of Comp., ACM Press (1996) 212-219
28. TEJ database, Taiwan Economic Journal Co. Ltd.,Taiwan, (2004) http://www.tej.com.tw/
29. Diebold, F. X.P: Elements of Forecasting. South-Western, Cincinnati (1998)
30. Ljung, G. M., and Box G. E. P.: On a Measure of Lack of Fit in Time Series Models. Biometrika. 65 (1978) 67-72

Application of ICA in On-Line Verification of the Phase Difference of the Current Sensor

Xiaoyan Ma and Huaxiang Lu

Neural Network Laboratory, Institute of Semiconductors, Chinese Academy of Sciences,
P. O. Box 912, Beijing, 100083, China
xyma@semi.ac.cn

Abstract. The performance of the current sensor in power equipment may become worse affected by the environment. In this paper, based on ICA, we propose a method for on-line verification of the phase difference of the current sensor. However, not all source components are mutually independent in our application. In order to get an exact result, we have proposed a relative likelihood index to choose an optimal result from different runs. The index is based on the maximum likelihood evaluation theory and the independent subspace analysis. The feasibility of our method has been confirmed by experimental results.

Keywords: Independent Component Analysis (ICA), Independent Subspace Analysis, Relative Likelihood Index, Current Sensor.

1 Introduction

The on-line monitoring of power equipment for high-voltage insulation is the key technique to improve the security of the power supply. Because the current leakage is very small in normal state, the current sensor, which is used to monitor the parameters of the current, must have high accuracy. However, the performance of most monitoring equipment may become worse in application. Because most current sensors work in a strong electromagnetic environment, they tend to be affected by the electromagnetism. Also, the temperature and the humidity of the atmosphere can change the current sensor's phase difference. Further more, in the measurement circuit, the linear amplification of the current also has an influence on the phase difference. In our country, the range of the phase difference of the current sensor used in the primary substation is 0.75^{o} with 20% rating current. In this paper, we focus on finding an on-line method to verify the phase difference of the current sensor.

Because it's a linear system, the frequency of the current will not change after processing. We input a test sine current I_t with fixed frequency and phase to the current sensor. I_t is assumed to be independent of the sensor's original input I_x, which is the leaking out current of the power equipment. Then separating the response of I_t from the sensor's output I_o, we can get I_t'. The phase difference of the current sensor can be determined by comparing I_t' with I_t. With proper separating algorithm, this method can achieve high accuracy.

The traditional method of separating a sine component with a given frequency from a mixed signal is FFT analysis. But, because of the spectral leakage and the noise interference, it is often difficult to get an acceptable result. It may be also not feasible to use least-squares estimation, for there is little information about the system's original output.

Because Independent Component Analysis (ICA) is capable of finding the underlying sources [1], and in our case the test signal is independent of another signal, we can use ICA to solve this problem. However, the basic ICA model requires that the number of observed mixed signals is not less than the number of source signals. It is also required that the observed signals should be additive mixtures of the source signals with different weights. Thus we firstly take measures to change it into the basic ICA model. It is described in detail in Section 2.

Since there are intrinsic limitations of ICA [2], the separation results of the same mixed signals may be somewhat different in different runs. Previous researches, such as [3], [4], have studied the reliability of the ICA estimation. But they mainly focus on the independence of the source components which are all mutually statistical independent. It is not suitable for us. Because some components are not completely independent after we change it into the basic ICA model. In Section 3, we have proposed a relative likelihood index to indicate the separation phase errors of different runs in order to choose an optimal one as the final separation result. In Section 4, the feasibility of our method is illustrated by experiments. And the factors affecting the accuracy of the result are discussed. Conclusions are drawn in the last section.

2 Signal Separation Method

2.1 The ICA Model

Generally Independent Component Analysis (ICA) consists in recovering N unknown sources $\{s_n(t)\}_{n=1}^{N}$ from M instantaneous mixtures $\{x_m(t)\}_{m=1}^{M}$, and we can use this matrix notation for the instantaneous linear mixture model

$$\begin{bmatrix} x_1(t) \\ \dots \\ x_M(t) \end{bmatrix} = A \begin{bmatrix} s_1(t) \\ \dots \\ s_N(t) \end{bmatrix} + \begin{bmatrix} n_1(t) \\ \dots \\ n_M(t) \end{bmatrix} \tag{1}$$

Where A is an M×N unknown mixing matrix and $n_k(t)$ are additive noise signals which will always be assumed to be mutually de-correlated and de-correlated from all sources. In basic ICA model, it is required $M \geq N$.

In order to verify the phase difference of the current sensor, we need to separate the response of the test current signal I_t from the mixed output I_o using ICA. This can be modeled as follows: the testing sine current I_t is denoted as $s_t(t)$ with fixed frequency f_0 and phase θ_0. The mixed output current of the current sensor I_o is denoted as $x(t)$, which is the additive mixture of the original output signal $g_o(t)$ and the test signal's response $s_o(t)$. Because of the linearity of the system, $s_o(t)$ must be a sine signal with the same frequency as $s_I(t)$. It is also restricted that $g_o(t)$ hardly has the component with frequency f_0 in its spectrum. So, we obtain the output vector $x(n), n = 1, 2...N$ by sampling.

$$x(n) = s_o(n) + g_o(n) + noise(n) \tag{2}$$

Our focus is on how to separate $s_o(n)$ from $x(n)$ with exact phase. We can take the sum of $g_o(n)$ and $noise(n)$ as one component. They are emitted by different physical sources from the test signal, and can be considered to be independent of the test signal. Because the frequency of $s_o(n)$ is f_0, we can project test signal $s_o(n)$ into the subspace which is generated by two vertical vectors $\sin(2\pi f_0 t_0 n)$ and $\cos(2\pi f_0 t_0 n)$ to determine its phase and amplitude. Therefore, $x(n)$ can also be expressed as

$$x(n) = a_1 \sin(2\pi f_0 t_0 n) + a_2 \cos(2\pi f_0 t_0 n) + g(n) \tag{3}$$

Where $g(n) = g_o(n) + noise(n)$, a_1 and a_2 are unknown mixing weights.

In order to satisfy the basic ICA requirements, $x_1(n)$ and $x_2(n)$ are generated by subtracting $\sin(2\pi f_0 t_0 n)$ and $\cos(2\pi f_0 t_0 n)$ from $x(n)$ respectively. Although $\sin(2\pi f_0 t_0 n)$ and $\cos(2\pi f_0 t_0 n)$ are not completely independent, they still can be separated by ICA explained in [9]. Then applying FastICA [1] to the generated mixture matrix $X = (x(n), x_1(n), x_2(n))^T$, three latent sources can be estimated. They are $g(n)$, $\sin(2\pi f_0 t_0 n)$ and $\cos(2\pi f_0 t_0 n)$. Meanwhile, the mixing matrix A in Eq.1 is also estimated.

2.2 Simulation to Test the Model

To test the preceding method, artificial source signals are generated in MATLAB with length of 8K samples: a 50 Hz square wave with 18.42 dB SNR as the current sensor's original output $g(n)$ and a 62.5 Hz test sine signal $s_o(n)$ with phase $\pi/4$. We sampled the sum of those signals with a 50 us interval and obtained the mixed signal $x(n)$ shown in Fig. 1. The normalized separation results are shown in Fig. 2.

Then the estimated test signal (shown in Fig.3) is the sum of the sine and cosine components with the weights estimated in A (Eq.1). The phase error of the estimated signal can be obtained by comparing the zero points of the real signal with the estimated signal. The errors of 10 different runs of FastICA on the same mixed signal are shown in Table 1. The results are somewhat different and some results' phase errors are too large. Considering the requirement of the application, we have to find an index to choose an optimal result with a smaller error from several runs. It will be discussed in the next section.

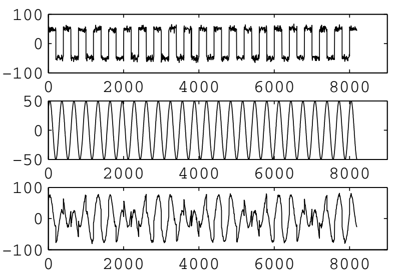


Fig. 1. The source signals and the mixed signal. Top: The source signal $g(n)$ with 18.42 dB SNR. Middle: The source signal $s_o(n)$. Bottom: The mixed signal $x(n)$.

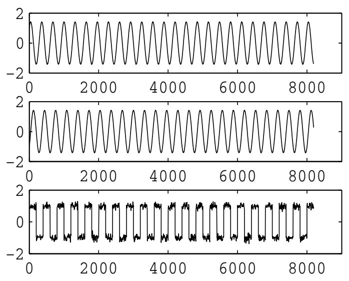


Fig. 2. The normalized separation results. Top: The cosine component. Middle: The sine component. Bottom: The sensor's original output $g(n)$.

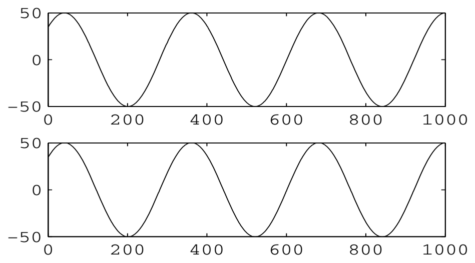


Fig. 3. The comparison of the real test signal (*top*) with the estimated signal (*bottom*)

Table 1. The phase errors (^{o}) of the separation results of 10 runs on the same mixed signal

Time	1	2	3	4	5	6	7	8	9	10
error	0.0018	0.6539	0.6556	0.0018	0.0248	0.3132	0.0018	0.0018	0.4869	0.0018

3 The Evaluation of the Separation Results

Figures in Table 1 indicate that the separation results of the same mixed signal are different in different runs. This may be induced by several reasons. The first reason is the statistical character of ICA. Because the initial mixing matrix A is generated randomly, the algorithm may find a local extremum of the objective function. The second reason, as for this case, is that the sine and cosine components in the source signals are not independent although they are uncorrelated. However, the subspace generated by the two signals is independent of the other source component. The third reason is the intrinsic limitations to the accuracy of the estimation of the mixing matrix [2].

A major problem of evaluating the reliability of the estimation is that the real signal is unknown. In general, ICA can be illustrated by a probability density matching problem [5], [7], which, in fact, turns out to be equivalent to mutual information minimization, and maximum likelihood estimation [6]. So, we can derive a relative likelihood index from maximum likelihood evaluation. In theory the indexes should be in reverse order in relation to their corresponding phase errors' absolute value.

Previous study [1] shows that based on using the well-known result on the density of a linear transform, the density p_x of the mixture vector X (defined in Eq.1) can be formulated as

$$p_x(X) = |\det B|\, p_s(S) = |\det B| \prod_i p_i(s_i) \tag{4}$$

where $B = A^{-1} = (b_1, b_2 ... b_n)^T$, $s_i = b_i^T X$ and the p_i denote the densities of the independent components. Then the likelihood [1] can be obtained as the product of this density evaluated at the T points. This is denoted by L and considered as a function of B :

$$L(B) = \prod_{t=1}^{T} \prod_{i=1}^{n} p_i(b_i^T X(t)) |\det B| \tag{5}$$

We can denote the sum over the sample index t by an expectation operator, and divide the likelihood by T to obtain the log-likelihood

$$\frac{1}{T} \log L(B) = E\left\{ \sum_{i=1}^{n} \log p_i(b_i^T X) \right\} + \log |\det B| \tag{6}$$

As for the estimation of the densities, it should be divided into two cases. On one hand, for the independent components, it has been verified (chapter 9 in [1]) that it is enough to use just *two* approximations of the density of an independent component. They are *supergaussian and subgaussian, and are estimated by these formulae Eq.7 and Eq.8 respectively [1].* The motivation for these formulae is that $\tilde{p}_i^+$ is a supergaussian density, because the logcosh function is close to the absolute value that would give the laplacian density. The density given $\tilde{p}_i^-$ by is subgaussian, because it is like a gaussian logdensity.

$$\log \tilde{p}_i^+(s) = \alpha_1 - 2 \log \cosh(s) \tag{7}$$

$$\log \tilde{p}_i^-(s) = \alpha_2 - \left[s^2 / 2 - \log \cosh(s) \right] \tag{8}$$

Where α_1 and α_2 are positive parameters that are fixed so as to make these two functions logarithms of probability densities. s is a normalized vector with zero-mean, and $E(s^2) = 1$. The Theorem 9.1 in [1] shows the maximum likelihood estimator is locally consistent, if the assumed densities p_i fulfill

$$E\{s_i g_i(s_i) - g_i'(s_i)\} > 0 \tag{9}$$

Where $g_i(s_i) = \frac{\partial}{\partial s_i} \log \tilde{p}_i(s_i) = \frac{\tilde{p}_i'(s_i)}{\tilde{p}_i(s_i)}$. Computed by Eq.9, the nonpolynomial moment $2E\left\{ -\tanh(s_i)s_i + (1 - \tanh(s_i)^2) \right\}$ can be obtained to determine which one of the two approximations is better. If this nonpolynomial moment is positive, the Eq.7 should be used, otherwise the Eq.8 should be used.

On the other hand, the density estimation in the independent subspace which is generated by sine and cosine components is different from the above. The principle of invariant-feature subspaces states that we can consider an invariant feature as a linear subspace in a feature space. The value of the invariant, higher-order feature is given by (the square of) the norm of the projection of the given data point on that subspace, which is typically spanned by lower-order features [1]. So, the densities of the components inside a subspace can be evaluated by

$$p_j(s_i, i \in S_j) = p(\sum_{i \in s_j} s_i^2) \qquad (10)$$

Where S_j denotes the j th subspace and s_i denotes the i th component inside one subspace. Then the relative likelihood index can be formulated as

$$RL = E\left\{\log p_1(g(n)) + \log p_2(s_{\sin}^2 + s_{\cos}^2)\right\} + \log\left|\det B\right| \qquad (11)$$

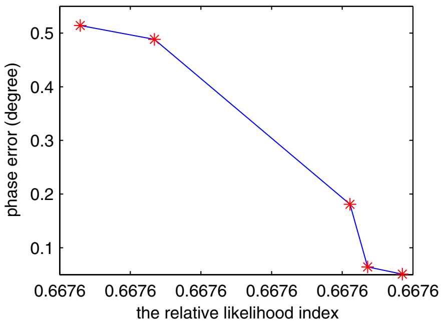


Fig. 4. The relationship of the phase error with the relative likelihood index

To test the relationship of the relative likelihood index with the estimated phase error, we run FastICA 10 times on the same mixed signal. The relationship of the phase error (absolute value) and the relative likelihood index is shown in Fig. 4. It indicates that the phase errors converge at 5 points (marked by red *). It is because the results converge at 5 local extremums of the objective function in FastICA. With the increasing index, the phase error becomes smaller. It accords with our expectation. Therefore, we can obtain an optimal result by choosing the one with the largest index from 10 FastICA runs.

4 The Results of the Experiments

To test the performance of our algorithm, we apply it to mixed signals with 2 different noise levels, 18.42 dB and 27.96 dB. With each noise level, we did 100 experiments. In every experiment, we ran FastICA 10 times on the same mixed signal. Then we chose the result with the maximum relative likelihood index as the final separation result. Fig. 5 shows the final phase errors of the separation results of 200 experiments. It shows that the results' phase errors of mixed signals with 27.96 dB SNR are much smaller than that with 18.42 dB. In both situations, the absolute values of the phase errors are smaller than 0.36^o, which can meet the application's requirement.

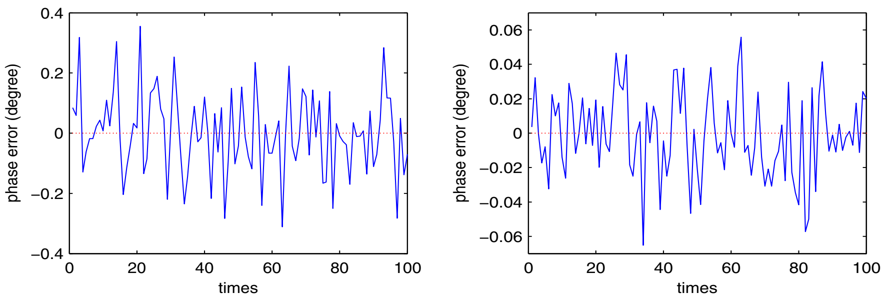


Fig. 5. The phase errors of final separation results of 200 experiments. Left: The original SNR is 18.42 dB. Right: The original SNR is 27.96 dB.

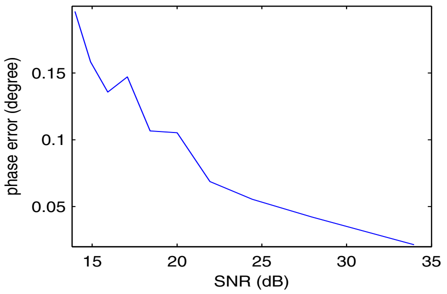


Fig. 6. The averaged absolute value of the phase error against the SNR

Table 2. The phase errors' absolute values of the results with different noise levels. Probatility denotes the probability of the index choosing an optimal result from 10 runs in 100 experiments

SNR (dB)	Experiment times	min (^{o})	max (^{o})	mean (^{o})	Probability(%)
33.98	100	0.0002	0.0659	0.0214	92
27.96	100	0.0003	0.0734	0.0220	82
24.44	100	0.0004	0.1743	0.0555	84
21.94	100	0.0008	0.1549	0.0687	72
18.42	100	0.0001	0.3532	0.0976	68
15.92	100	0.0011	0.5591	0.1357	64

To test our separation algorithm's performance with different noise levels, we apply it to mixed signals with different SNR. With each SNR we did 100 experiments. The averaged absolute values of the results' phase errors with each SNR are shown in Fig. 6. Some of the final phase errors of our experiments are given statistically in Table 2. The phase error becomes larger with decreasing SNR. But, if the SNR is lower than 15 dB, the error may become too large. In this case we should denoise the mixed signal to reduce the error.

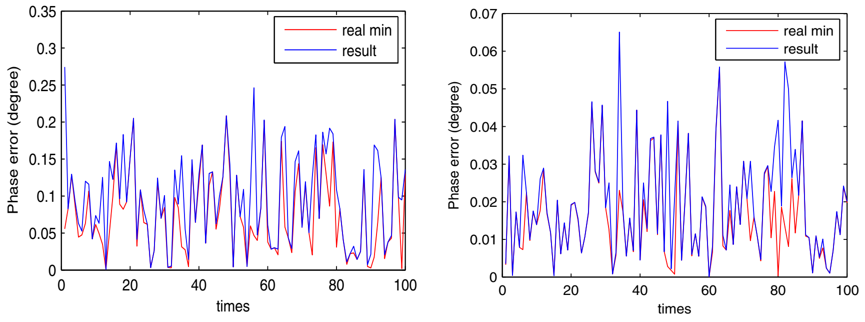


Fig. 7. The comparison of the real minimum phase error of 10 runs with the error of the final separation result chosen by the index. Left: The original SNR is 18.42 dB. Right: The original SNR is 33.98 dB.

To test the validity of the relative likelihood index, we apply it to mixed signals with 2 different noise levels, 18.42 dB and 33.98 dB. With each noise level we did 100 experiments. In every experiment, we ran FastICA 10 times on the same mixed signal, and then chose the final separation result by the relative likelihood index. We have also recorded the real minimum phase error of the 10 results in each experiment. Fig. 7 shows the comparison of the real minimum phase error with the phase error of the final separation result in every experiment. Figures illustrate that in most experiments the final separation result's phase error is the smallest of the 10 results. Because the probability density estimation in the index may be not exact, the result chosen by the index is not always the optimal one. The probability of the index choosing an optimal result from 10 runs is higher than 70% with SNR higher than 18.42 dB (shown in Table 2). It increases with the increasing SNR.

In our work, we have also investigated the factors which may affect the accuracy of the result. First, it is the deficiencies of the ICA algorithm which have been discussed at the beginning of the Section 3.

Second, the manner we sample the observed signals also has a great influence. We obtained 8K samples from the same signals as Section 2.2 with 10 different intervals. With each interval we did 5 experiments. The absolute values of the errors are shown in Table 3 and the averaged absolute values of the phase errors are illustrated in the left of Fig. 8. It shows that for the periodic signal, with the same number of samples, the error becomes smaller with the increasing interval. It may be due to the increased length of sample time. To test the length of sample time's influence, we obtained mixed signals in the same length of sample time with different sample intervals. The number of samples decreases with the increasing interval. With each interval we did 10 experiments, the phase errors' absolute values are shown in Table 4. The averaged absolute values of the phase errors are illustrated in the right of Fig. 8. It illustrates that with the same length of sample time, no matter what the sample interval is, the alterations to the phase errors are very small. So, we can conclude that the length of sample time is a main factor to influence the results. If the length of sample time is longer, the phase error will become smaller.

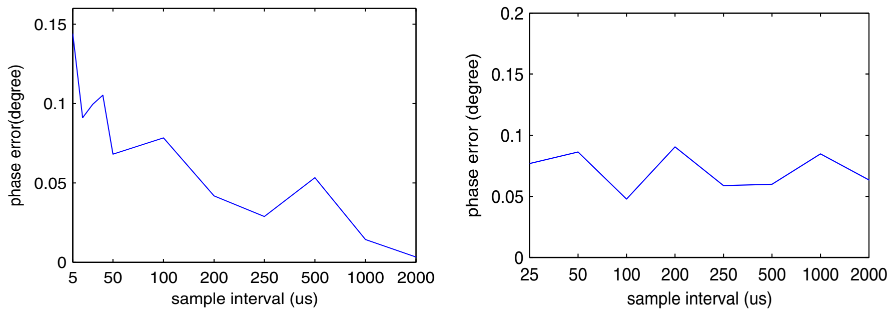


Fig. 8. The averaged absolute value of the phase error against the sample interval. Left: The numbers of samples are all 8K with every interval. As the intervals are different, the lengths of sample time are different. Right: The lengths of sample time are identical with every interval. Because the intervals are different, the numbers of samples are different.

Table 3. The phase errors (^{o}) of the separation results with different sample intervals (The numbers of mixed signal samples in every experiment are identical)

interval (us)	5	30	50	100	200	250	500	1000	2000
1	0.1885	0.0664	0.0707	0.0920	0.1168	0.0212	0.0037	0.0399	0.0037
2	0.0959	0.1484	0.0030	0.0430	0.0051	0.0123	0.0437	0.0060	0.0071
3	0.2106	0.0886	0.0185	0.2691	0.0322	0.0725	0.0350	0.0022	0.0005
4	0.0476	0.1122	0.2335	0.0375	0.1075	0.0022	0.0883	0.0075	0.0046
5	0.1779	0.0819	0.1379	0.0660	0.0236	0.0036	0.0008	0.0118	0.0043

Table 4. The phase errors (^{o}) of the separation results with different sample intervals (The lengths of sample time of the mixed signal in every experiment are identical)

interval (us)	25	50	100	200	250	500	1000	2000
1	0.0447	0.1885	0.0296	0.0813	0.0119	0.0047	0.1645	0.0250
2	0.1088	0.0751	0.0104	0.2469	0.0372	0.1318	0.0350	0.0289
3	0.0187	0.0089	0.0348	0.1066	0.0700	0.0859	0.1562	0.0315
4	0.0089	0.0907	0.1746	0.0705	0.0798	0.0931	0.1339	0.0203
5	0.0701	0.0123	0.0112	0.0390	0.2083	0.0204	0.0800	0.0211
6	0.0812	0.2237	0.0297	0.1437	0.0270	0.1421	0.0113	0.0328
7	0.0593	0.1015	0.0331	0.1128	0.0616	0.0395	0.0837	0.0012
8	0.0479	0.0391	0.0813	0.0119	0.0619	0.0069	0.0201	0.0677
9	0.1763	0.0882	0.0388	0.0361	0.0173	0.0053	0.1587	0.1936
10	0.1515	0.0344	0.0339	0.0559	0.0126	0.0680	0.0050	0.2108

5 Conclusions

In this paper, we apply ICA to verify the phase difference of the current sensor in power equipment. Considering the background, we have made an ICA model of the signals and focus on separating a sine component from the mixed signal. Our algorithm is that every time we run FastICA 10 times on the same mixed signal. Then we choose the separation result which has the maximum relative likelihood index as the final separation result. The relative likelihood index we proposed is based on the maximum likelihood evaluation theory and independent subspace analysis. Experimental results have verified that with the SNR higher than 18.42 dB, we can obtain a result with maximum phase error 0.3532^{o}, which satisfies the requirement of the primary substation. We have discussed the potential factors affecting the accuracy of the result in detail. This separation algorithm can be applied to other linear systems. However, it still has some limitations; for example, the error becomes larger with the increasing noise. Further studies are necessary to improve it.

Acknowledgements

This paper is supported by the *National Nature Science Foundation* of China (No. 60576033).

References

1. Hyvärinen, A., Karhunen, J., Oja, E.: Independent Component Analysis. John Wiley & Sons. Inc., New York (2001)
2. Cardoso, J.F.: On the performance of orthogonal source separation algorithms. In: Proc. EUSIPCO, Edinburgh 9 (1994) 776-779
3. Meinecke, F., Ziehe, A., Kawanabe, M., Müller, K.R.: A resampling approach to estimate the stability of one-dimensional or multidimensional independent components. In: IEEE Trans. Biomed, Eng 49 (2002) 1514-1525
4. Himberg, J., Hyvärinen, A.: Icasso: software for investigating the reliability of ICA estimates by clustering and visualization. Processing of the Workshop on Neural Networks and Signal Processing, Toulouse, France (2003)
5. Attias, H., Schreiner, C.E.: Blind source separation and deconvolution: The dynamic component analysis algorithm. In: Neural Computation, vol. 10 (1998) 1373-1424
6. Cardoso, J.F.: lnfamax and maximum likelihood for source separation. In: IEEE Signal Processing Letters, vol. 4. 4 (1997) 112-114
7. Choi, S., Cichocki, A.: Correlation matching approach to source separation in the presence of spatially correlated noise. In: Proc. IEEE ISSPA. Vol. 1 (2001) 272-275
8. Gribonval, R., Benaroya, L., Vincent, E., Févotte, C.: Proposals for performance measurement in source separation. 4th International Symposium on Independent Component Analysis and Blind Signal Sepration, Nara, Japan April (2003)
9. Stögbauer, H., Kraskov, A., Astakhov, S.A., Grassberger, P.: Least-dependent-component analysis based on mutual information. In: Physical Review E 70, 066123 (2004) 1-17

Neural Networks Based Automated Test Oracle for Software Testing

Ye Mao[1], Feng Boqin[1], Zhu Li[2], and Lin Yao[1]

[1] School of Electronic & Information Engineering, Xi'an Jiaotong University,
Xi'an 710049, China
xjtuyemao@163.com
[2] School of Software, Xi'an Jiaotong University,
Xi'an 710049, China

Abstract. A lot of test cases must be executed in statistical software testing to simulate the usage of software. Therefore automated oracle is needed to automatically generate the expected outputs for these test cases and compare the actual outputs with them. An attempt has been made in this paper to use neural networks as automated test oracle. The oracle generates the approximate output that is close to expected output. The actual output from the application under test is then compared with the approximate output to validate the correctness. By the method, oracle can be automated. It is of potential application in software testing.

1 Introduction

The software engineering community has turned attention to statistical software testing recently [1, 2, 3]. The main idea is that the reliability of software depends greatly on the manner in which the software is used. The importance of a failure is largely determined by the likelihood of encountering it. Therefore software is tested according to the model which highlights the critical usage and a lot of test cases must be executed to simulate statistically the usage of the software. However it needs a lot of time and is often error-prone to manually generate the expected outputs for these test cases and compare the actual outputs of the application under test (AUT) with them. As a result automated test oracle is needed in statistical software testing to automatically generate the expected output and compare the actual output with it. However there are very few techniques developed to automate oracle. In most cases, a tester is assumed to provide the expected output of the software, which is specified by a table of pairs [4], logical expressions to be satisfied by the software [5], or temporal constraints that must not be violated during software execution [6]. Schroeder uses the input-output (IO) relationships of the software to identify unique combinations of the inputs which influence outputs. By this information a lot of test cases which including the inputs and expected outputs can be automatically generated [7]. However determining all IO relationships manually is rather difficult. When testing software with graphical user interfaces (GUI) by capture/replay tool [8, 9], expected outputs are saved for comparison while recording test scripts or inserted into the test scripts manually. Memon presents planning method to generate expected outputs [10]. It needs to construct GUI model and set conditions for every operator manually. Chen tests software by finite state machines (FSM), expected outputs are manually included in the model [11]. Specification language Z uses predicate logic to specify an operation as a relation between the input and output [12]. Test oracle can be generated from Z specification [13]. However it needs that the user's requirements of the software are represented by Z specification language. To implement more automatic oracle, Aggarwal use neural networks (NN) based approach to generate expected output [14] for triangle classification problem [15]. By the experiment they made the conclusion that NN can be used as test oracle with reasonable degree of accuracy for classification problem in software testing.

We propose in this paper that the relationship from inputs to outputs of AUT is in nature a function. When the function is continuous, an automated oracle is proposed. NN is used to implement the oracle. To appeal of the NN approach lies in its ability to approximate a function in any precision without the need to have knowledge of that function [16-19]. Experiment has been conducted to validate the effectiveness of the proposed method.

In the next section, we briefly describe the theory of multilayer NN. Automated oracle based on NN is proposed in section 3. Section 4 presents the results of the experiment. The conclusions and future directions are presented in section 5.

2 Multilayer Neural Networks for Function Approximation

Multilayer NN has been established to be effective method to approximate continuous or other kinds of functions defined on compact sets in R^n [16, 17]. It can be used to learning the relationship from inputs to outputs by training on the samples. Once training process finishes, it can be given any input and produce an output by the relationship learned. The output generated by NN can be arbitrarily close to the expected output owing to the generalization capabilities of the networks. Back propagation based on gradient descent in error is the most popular training algorithm for multilayer NN [17, 18, 19]. In the algorithm the network is initialized with a random set of weights, and then trained from a set of input and output pairs, i.e. training samples. Training process stops when the training error is acceptable or a predefined number of epochs pass. Once trained, network weights are kept and used to approximate the function. When training NN, the weight update rule can be expressed as follows:

$$\Delta w_{ji}(n) = \alpha \Delta w_{ji}(n-1) + \eta \delta_j(n) y_i(n) \qquad (1)$$

where α is a positive number called the momentum constant, $\Delta w_{ji}(n)$ is the correction which is applied to the weight connecting the output of neuron i to the input of neuron j at the n^{th} iteration, η is the learning rate, $\delta_j(n)$ is the local gradient at the n^{th} iteration, and $y_i(n)$ is the function signal appearing at the output of neuron i at the n^{th} iteration. The training error can be the sum over output units of the squared difference between the expected outputs t_k given by a teacher and the actual output z_k :

$$J(w) = \frac{1}{2}\sum_{k=1}^{c}(t_k - z_k)^2 = \frac{1}{2}\|t - z\|^2 \tag{2}$$

where t and z are the target and the network output vectors of length c respectively and w are the weights in the network. Details about the back propagation algorithm can be found in references [18, 19].

3 Automated Test Oracle Based on Neural Networks

3.1 Model of the Automated Oracle

Function testing involves executing an AUT and examining the output, which is implemented by oracle. General model of oracle is as Fig.1. In the model expected outputs are generated from inputs and then compared with actual outputs from AUT. If they are not same, it implies a failure. The process of generating expected outputs and comparing is traditionally done manually. Testers compute the expected outputs from program specifications or their knowledge of how a program should operate. Expected outputs are then compared with actual outputs by tester's knowledge to determine if a failure occurs.

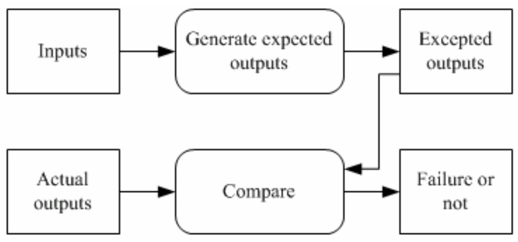


Fig. 1. General oracle model

The relationship from the input to output of the software is in nature a function. Let $x = (x_1, x_2, \cdots, x_n)$ and $y = (y_1, y_2, \cdots, y_m)$ be input and output vectors respectively. Therefore the relationship can be represented by $y = f(x)$, where f implement the specification of the software. When $x \in R^n$, $y \in R^m$, and f is continuous, the function f can be approximated by NN after training. In this paper, we propose the automated oracle in this situation. Each component in y is also a function of x, that is $y_j = f_j(x)$. Let $D(x_i)$ be the set of all possible values of x_i and $D(x)$ be the set of all possible values of x. Therefore $D(x)$ includes every possible combination of the value from $D(x_i)$ and $|D(x)| = \prod_i |D(x_i)|$. Let x^i be a data item in $D(x)$ and $y^i = f(x^i)$, then (x^i, y^i) is a test case. Automated test oracle can generate y^i from x^i and compare it with the actual output automatically.

The model of the automated oracle is as Fig. 2. In the model approximate outputs are automatically generated according to inputs. Approximate output is not as same as expected output. But it can approach expected output in any precision. Comparison process now became as follows:

$$|\eta - a|_\varepsilon = \begin{cases} 0 & if |\eta - a| \le \varepsilon \\ |\eta - a| - \varepsilon & otherwise \end{cases} \tag{3}$$

where η and a are the actual output and approximate output respectively, ε is test precision. Indicator ε controls the criterion if the actual value η is right or not. We can adjust ε between precision and test cost. In experiment, we will describe the effect of the ε. If $|\eta - a|_\varepsilon = 0$, it means the actual output is right within precision ε. Otherwise, if $|\eta - a|_\varepsilon > 0$, it means that a failure occurs because the actual output is not right within precision ε. To generate the approximate outputs, the relationship from the input to output of the AUT should be learned from a small set of training samples of the specification. Let $D(y)$ be co-domain of the AUT. Then the training samples are:

$$S = \{(x^i, y^i) \mid x^i \in D^{'}(x), D^{'}(x) \subset D(x), y^i = f(x^i), y^i \in D(y)\} \tag{4}$$

where y^i is the expected output. The approximate outputs are then automatically generated for $\forall x \in D(x)$ by the relationship learned.

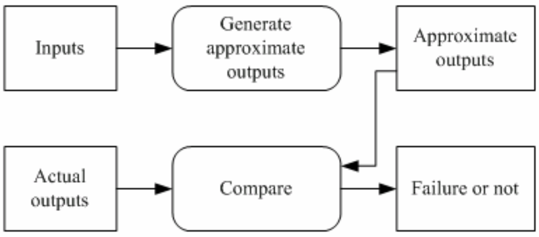


Fig. 2. Automated oracle model

3.2 Automated Oracle Based on Neural Networks

NN can approach any continuous function in theory. This feature is used to generate the approximate output in the automated oracle. To implement automated oracle, two processes must be automated. One is to generate approximate output by NN and the other is to compare the actual output from AUT with the approximate output. Automated oracle can be summarized in the following steps.

Step 1: Manually generate sample set S in equation (4) from the specification of the AUT.
Step 2: Construct NN and initialize the weights. Set training parameters.
Step 3: Train NN by back propagation algorithm in the training set $S^{'} \subset S$.
Step 4: When the stopping criterion is satisfied, keep the weight and go to step 5.
Step 5: Obtain approximate output a^i from NN for $\forall x^i \in D^{'}(x)$. Set test precision $\varepsilon = \max_i(abs(y^i - a^i))$.
Step 6: Get the input of the AUT and input it to NN. Obtain the approximate output a from NN.
Step 7: Get the actual output η from the AUT and compare it with the approximate output a according to equation (3). Determine if there is failure or not by the result of comparison.
Step 8: Repeat step 6 and 7 to test the AUT in other inputs.
Step 9: If it is needed to test in different precision, go to step 2. Otherwise, the process finishes.

4 Experiments

The goal of the experiment is whether the method proposed is effective to expose the failure of the AUT. It includes whether NN can be used to generate the approximate output that is close to the expected output, whether the value of ε can be computed from samples, and whether the method can expose failure of the AUT.

The AUT in the experiment is an application with GUI as Fig. 3. It has three input variables and two output variables. Input and output vector are $x = (x_1, x_2, x_3)$ and $y = (y_1, y_2)$ respectively. The relationship between input and output is:

$$y_1 = x_1 \sin(3x_1) + \cos(x_1)e^{1-x_1^2} \tag{5}$$

$$y_2 = \frac{x_1^2 + x_2^2 + x_3^2 + 3x_2x_3 + x_1x_3}{15} \tag{6}$$

where $x_1, x_2, x_3 \in [0,3\pi]$. The output variable verified in the experiment is y_1 in equation (5). It can be generalized to more complicated and real situations. We manually seed faults into the AUT. The faults seeded in the AUT will cause failures as table 1. It shows that the faults will cause the actual output of the AUT different from the expected output. Manually generate set S with 200 samples from the specification of the AUT. The set S is used for training NN and compute the value of ε . The variable x_1 is dropped from interval $[0,3\pi]$ evenly and the variable y_1 is computed manually. NN is constructed and training parameters are set as table 2. We select samples from S evenly to form training samples S' whose size of is 100. Use S' to train NN. The training process is as Fig. 4. It shows the performance achieves the goal quickly after 192 epochs pass.

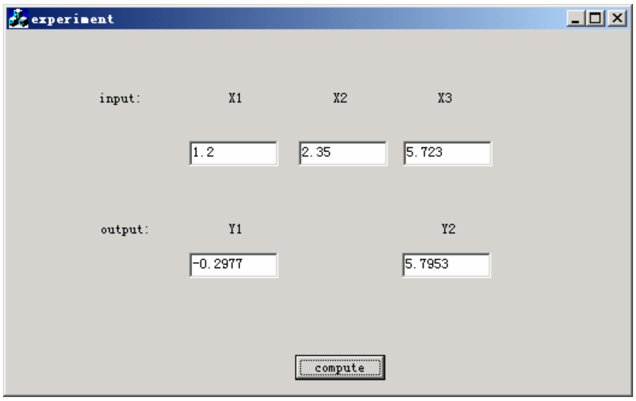


Fig. 3. The example of AUT

Table 1. Five failures caused by faults seeded in the AUT (x_1, η, and y_1 are the inputs of AUT, actual outputs, and expected outputs)

Failure ID	x_1	η	y_1
1	1.1	0.3900	0.1942
2	1.57	-1.8732	-1.5698
3	3.683	-3.5776	-3.6777
4	5.9	-5.3631	-5.3842
5	7.31	0.4367	0.4467

Table 2. Neural networks architecture and training parameters

Network architecture	Training parameters
The number of layers	2
The number of units on the layers	Input: 1; Hidden: 13; Output: 1
Transfer functions	logsig in 1st layer, purelin in 2st layer
Training function	trainlm
Learning function	learngdm
Performance function	mse
Initial weights and biases	The Nguyen-Widrow method
epochs	10000
goal	0.001
Adaptive learning rate	0.1

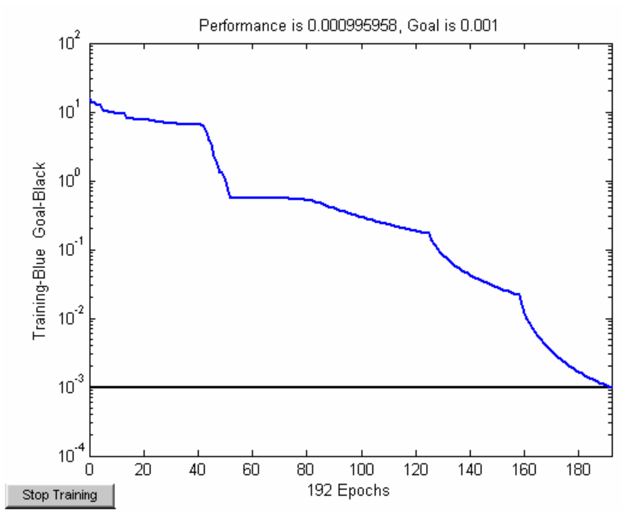


Fig. 4. The process of training the NN

After NN is trained, we test the performance of approximation. The expected output y_1, approximate output a obtained from trained NN, and difference $\zeta = y_1 - a$ is as Fig.5. From it we can see the difference is small enough when the training goal is set to 0.001. It shows that NN can be used to generate the approximate output that is close to the expected output.

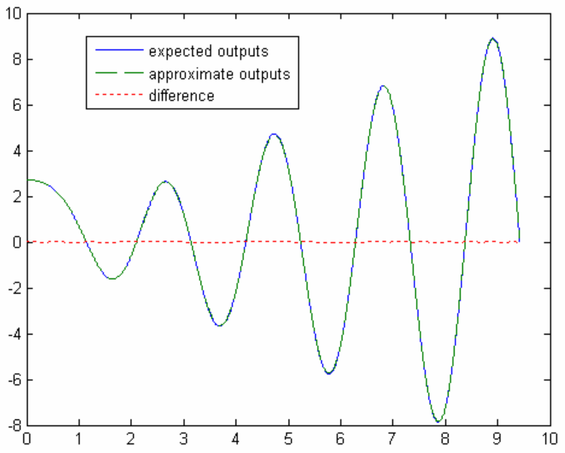


Fig. 5. Plot of the expected outputs, approximate outputs and their difference when the training goal is 0.001

Test precision ε is computed in this step. The trained NN is used to generate approximate output a^{i} for $\forall x^{i} \in D^{'}(x)$, $i = 1, \cdots, 200$. Test precision ε is set as $\max_i(|y_1^i - a^i|)$. The result value of ε is 0.1863 because the values of $(y_1^i - a^i)$ is in the interval $[-0.1863, 0.0757]$. As a result if the actual output obtained from the AUT is in the interval $[a - 0.1863, a + 0.1863]$, there is no failure. Otherwise, a failure is exposed. We now check if the method can expose the failures in table 1. The result is in table 3. In the table $|\eta - a|_\varepsilon$ is computed by equation (3). A failure is exposed if $|\eta - a|_\varepsilon$ is larger than 0. Table 3 shows failure 1 and 2 can be exposed successfully when training goal is set to 0.001 because $|\eta - a|_\varepsilon > 0$. Failure 3, 4, and 5 can not be exposed. It is because the difference between the actual output and the approximate output is below the test precision ε.

Table 3. The failures that can be exposed when ε is 0.1863 (x_1, η, and a are input, actual output, and approximated output. $|\eta - a|_\varepsilon > 0$ means a failure is exposed).

Failure ID	x_1	η	y_1	a	$\|\eta - a\|_\varepsilon$
1	1.1	0.3900	0.1942	0.1943	0.0094
2	1.57	-1.8732	-1.5698	-1.5730	0.1139
3	3.683	-3.5776	-3.6777	-3.6690	0
4	5.9	-5.3631	-5.3842	-5.3693	0
5	7.31	0.4367	0.4467	0.4870	0

When we change the training goal, different test precision can be achieved (table 4). It shows we can set goal according to different demand of precision in software testing. If a precision ε of 0.0165 is needed, we can set training goal under 5e-005. If the actual output is in the interval $[a - 0.0165, a + 0.0165]$, where a is the approximate output obtained from NN, it means there is no failure. Otherwise, it means a failure occurs. All failures in table 1 can be exposed in this situation. It means the method proposed can expose failure effectively.

Table 4. Precision ε achieved under different training goal ($\min$ and $\max$ are min and max difference between expected and approximate outputs)

Goal	$\min$	$\max$	ε	Goal	$\min$	$\max$	ε
0.01	-0.3694	0.1852	0.3694	5E-04	-0.0818	0.0408	0.0818
0.005	-0.2542	0.1378	0.2542	1E-04	-0.0789	0.0294	0.0789
0.001	-0.1863	0.0757	0.1863	5E-05	-0.0149	0.0165	0.0165

In the experiment, the value of ε is obtained from 200 samples. We now obtain the value from more samples to see if the value will change obviously. The result is in table 5. The meaning of each column is same as table 4. It shows that the value of ε

computed from 200 samples is effective in this experiment because it is same as the one computed from more samples where the number is 100000.

Table 5. The comparison of the value ε obtained from different samples ($\min$, $\max$, and ε are obtained when the number of samples is 200, $\min^{'}$, $\max^{'}$, and $\varepsilon^{'}$ are obtained when the number of samples is 100000)

Goal	$\min$	$\max$	ε	$\min^{'}$	$\max^{'}$	$\varepsilon^{'}$
0.01	-0.3694	0.1852	0.3694	-0.3694	0.1864	0.3694
0.005	-0.2542	0.1378	0.2542	-0.2542	0.1378	0.2542
0.001	-0.1863	0.0757	0.1863	-0.1863	0.0758	0.1863
5E-04	-0.0818	0.0408	0.0818	-0.0818	0.0409	0.0818
1E-04	-0.0789	0.0294	0.0789	-0.0789	0.0297	0.0789
5E-05	-0.0149	0.0165	0.0165	-0.0149	0.0165	0.0165

5 Conclusions

In statistical software testing a lot of test cases should be executed to simulate statistically the usage model of the AUT. However it is difficult to manually generate expected outputs for these test cases and compare the actual outputs of the AUT with them. As a result an automated test oracle is proposed in this paper to solve the problem. The oracle can be applied when the relationship from the input to output of the AUT is a continuous function. From above results we conclude that NN can be used to implement the automated test oracle in reasonable precision. It can generate the approximate output for AUT and the precision can be adjusted by training parameter. As a result, we can test AUT in the precision needed. By the method, we need not generate all expected output from AUT manually. It can save a lot time and labor in software testing. It is shown in the experiment that the precision ε is important to expose failure and it can be computed from samples generated manually. However it is not verified that if it is effective when the relationship from the input to output become more complicated. As a result we will do the experiment in more complicated relationships.

Acknowledgment

This work was supported in part by the National High Technology Development Plan of China (863) under grant no. 2003AA1Z2610.

References

1. Sayre, K.: Improved techniques for software testing based on Markov chain usage models, PhD. thesis, University of Tennessee, Knoxville, USA (1999)
2. Bertolini, C., Farina, A.G., Fernandes, P., Oliveira, F.M.: Test case generation using stochastic automata networks: quantitative analysis, In: Proc. of the second International Conf. on Software Engineering and Formal Methods, IEEE Press (2004) 251-260

3. Beyer, M., Dulz, W., Zhen, F.: Automated TTCN-3 test case generation by means of UML sequence diagrams and Markov chains, In: Proc. of the 12th Asian Test Symposium, Piscataway : IEEE Press (2003) 102-105
4. Peters, D., Parnas, D.L.: Generating a test oracle from program documentation, In: Proc. of the International Symposium on Software Testing and Analysis (1994) 58-65
5. Bousquet, L., Ouabdesselam, F., Richier, J., Zuanon, N.: Lutess: a specification-driven testing environment for synchronous software, In: Proc. of the 21th International Conf. on Software Engineering, ACM Press (1999) 267-276
6. Dillon, L.K., Ramakrishna, Y.S.: Generating oracles from your favorite temporal logic specifications, In: Proc. of the 4th ACM SIGSOFT Symposium on the Foundations of Software Engineering, ACM Software Engineering Notes, vol.21 (1996) 106-117
7. Schroeder, P.J., Faherty, P., Korel, B.: Generating expected results for automated black-box testing, In: Proc. of the 17th IEEE International Conf. on Automated Software Engineering, IEEE Press (2002) 139-148
8. Ostrand, T., Anodide, A., Foster, H., Goradia, T.: A visual test development environment for GUI systems, ACM SIGSOFT Software Engineering Notes, vol.23, no.2 (1998) 82-92
9. Chen, W.K., Tsai, T.H., Chao, H.H.: Integration of specification-based and CR-based approaches for GUI testing, In: Proc. of the 19th International Conf. on Advanced Information Networking and Applications, vol.1 (2005) 967-972
10. Memon, A., Nagarajan, A. Xie, Q.: Automating regression testing for evolving GUI software, Journal of Software Maintenance and Evolution: Research and Practice, vol.17, no.1 (2005) 27-64
11. Chen, J. Subramaniam, S.: Specification-based testing for GUI-based applications, Software Quality Journal, vol.10, no.3 (2002) 205-224
12. Hierons, R.M.: Testing from a Z specification, Software Testing, Verification, and Reliability, vol.7 (1997) 19-33
13. McDonald, J. Strooper, P.: Translating object-Z specifications to passive test oracles, In: Proc. of the 2th International Conf. on Formal Engineering Methods, IEEE Press (1998) 165-174
14. Aggarwal, K.K., Singh, Y., Kaur, A., Sangwan, O.P.: A neural net based approach to test oracle, ACM SIGSOFT Software Engineering Notes, ACM Press, vol.29, no.3 (2004) 1-6
15. Ramamoorthy G.V., Ho S.F., Chen W.T.: On the automated generation of program test data, IEEE Trans. Software Engineering, vol.SE-2 (1976) 293-300
16. Chen, T., Chen, H.: Approximations of continuous functionals by neural networks with application to dynamic systems, IEEE Trans. Neural Networks, vol.4, no.6 (1993) 910-918
17. Chen, D.S., Jain, R.C.: A robust back propagation learning algorithm for function approximation, IEEE Trans. Neural Networks, vol.5, no.3 (1994) 467-479
18. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification, second edition, John Wiley & Sons (2001)
19. Fausett, L.: Fundamentals of neural networks: architectures, algorithms, and application, Prentice Hall: Englewood Cliffs, New Jersey (1994)

Tool Wear Condition Monitoring in Drilling Processes Using Fuzzy Logic

Onder Yumak and H. Metin Ertunc

Kocaeli University Mechatronic Engineering Department, Kocaeli/Turkey

Abstract. During the era of the rapid automation of the manufacturing processes, the automation of the metal cutting and drilling process, which is one of the most crucial stages in the industrial process, has become inevitable. The most important difficulty in the automation of machining process is time and production loss that occurs as a result of tool wear and tool breakage. In this study, a fuzzy logic based decision mechanism was developed to determine tool wear condition by using cutting forces. The statistical parameters of the cutting forces collected during the drilling operation have been determined as variables for the membership functions of the fuzzy logic decision mechanism. The system developed in this study, successfully determined the tool wear condition in drilling processes.

Keywords: Fuzzy Logic; Tool Wear Condition; Decision mechanism.

1 Introduction

Unmanned metal cutting systems have widely started to use in manufacturing in 1990's, and this phenomena have forced to develop new automated tool condition monitoring systems [1]. Machines used in unmanned manufacturing systems are unable to determine tool wear or breakage automatically, therefore continuing to manufacturing leads to wrong production and low product quality, hence production costs would increase significantly. Late replacement of worn tool, may cause damage or dimensional errors on workpiece. On the other hand, in case of an early replacement of tool or determining tool condition by direct measurement from workpiece, the manufacturing process will encounter with downtime frequently and hence production capacity will decrease. Modern CNC systems can be programmed according to experimental tool life, thus the appropriate replacement time of tool can be provided. But experimental conditions doesn't match always with the working conditions. Therefore, as a result of early replacement of a workable tool or late replacement of a worn tool, time or production loss may occur. In addition, due to complex structure of tool wear mechanism, unpredictable breakages may occur at any time. Consequently, the purpose of such production systems must be provide replacement of a tool after maximum utility obtained from the tool. This purpose requires sensing of critical tool wear level that necessitates tool replacement at the next cutting process. There are two main techniques used to determine tool wear condition. One of them is "direct method" which requires measurement from workpiece directly and the other is "indirect method" which realized by measuring cutting parameters from medium during cutting process. As mentioned above, downtime of manufacturing process in direct measurement leads production loss. Therefore, researchs on this issue has been focused on tool condition monitoring methods without direct contact with tool. Different sensing methods has been developed for this goal. When the measured parameters which are sensitive to tool wear analyzed by using different methods, they can give meaningful results about tool wear. Many different parameters was used in past researchs. Most used parameters can be arranged as cutting forces, vibration, sound, acoustic and ultrasonic vibration, current, power and temperature. All of the parameters can have advantages and also can have disadvantages with respect to others. Many data analyses methods have been used for extracting information about tool wear state from obtained data. Formerly, researchs started with time series and frequency domain analyses and then more successful results achieved by using different methods together or separately such as pattern recognition [2], statistical analyses [3], fourier transform [4], wavelet transform [5], fuzzy logic and artificial neural Networks [5,6,7]. As such in cutting parameters, analyses methods have also sufficiency and practicability at different degrees with respect to others. Some analyses methods can be realized easily and quickly, but also they maybe less sensitive to tool wear. On the other hand, complex methods provide more information but they are hard to realize or not economic.

Regardless which method was used, realized process is always basically the same. The goal is analysing and classifying of data obtained by different sensors. This classification explains tool condition, but parametric data contains needless information that can be called as noise together with information about tool wear. Used method comes into prominence due to ability of extracting beneficial information. As mentioned before both cutting parameters and analyses methods have advantages or disadvantages according to each other. Therefore, various researchs have been being made lately on every kind of parameters and methods by researchers to increase predictability rate of tool failure.

In this study, as a primary goal a fuzzy logic based decision mechanism was developed to determine tool wear condition for drilling process which is one of the most used material removal process at industry by using cutting forces. Feed force (thrust) measurements which obtained from a project [8] were used as cutting parameters. The statistical parameters such as mean, RMS, standard deviation and maximum value of the cutting forces collected during the drilling operation under different number of revolutions and feed rate have been determined as variables for the membership functions of the fuzzy logic decision mechanism. The system developed in this study, successfully determined the tool wear condition in drilling processes as sharp, workable or worn. Although this work focuses on tool wear condition monitoring for drilling operations, the proposed techniques and decision algorithm introduced in this paper can be applied to other cutting process.

In addition, an Adaptive Neuro Fuzzy Inference System (ANFIS) has been composed to determine tool wear state by using same parameters. Results for ANFIS system and comparison of two different methods have been proposed on last section of this study.

2 Fuzzy Logic

The theory of Fuzzy Logic was introduced by Lotfi A. Zadeh in 1965 as an alternative way of data processing against classical logic [9]. The main reform coming along with fuzzy logic theory is the partial thruthness and partial wrongness instead of the states absolute thruthness and absolute wrongness existed in classical logic. In other words, fuzzy logic has admitted to using multivalued variables based on necessity instead of bivalued variables. Although initially this theory wasn't proposed as a control method, subsequently rapidly progressed after 70's and then it has become a research area that arouses interest among the researchers. Conventional control systems perform the process control based on a mathematical model which is hard to build. Control procces can only be handled by an expert in case of building a matematical model is impossible. Fuzzy logic based systems eliminate the mathematical model necessity and brings expert experience and intuition instead of model. Thus the control process becomes simple and automation possibility occurs for some areas which couldn't be controlled without an expert.

2.1 Fuzzy Thinking

Human thought and reasoning system uses quality instead of quantity to define uncertainties. In other words, gives linguistic meanings to magnitudes instead of numerical meanings and uses linguistic (fuzzy) variables. Human brain has ability to produce perfect behavior through giving weightness to perceived stimulant based on their qualities. Certainly, this ability improves with experience. Fuzzy logic is a method which enables modeling this feature of human brain.

Human brain finds solutions to problems by using simple 'IF condition THEN consequence' approach. For example, measuring of light intensity is unnecessary to decide turning on light. Instead of that 'IF it is dark THEN light must be turned on' behavior rule is simple and enough. As the control problems become complex the number of conditions which will effect to decision may arise and their kinds may change. Conditions can be combined by conjunctions such as AND-OR. For instance, a driver uses the rule 'IF speed $>$ 90 km/h OR forward vehicle is slow THEN speed must be decreased' and this rule requires to use both numerical value on the speedometer and the quality of speed of vehicle on forward. In other words both numerical and linguistic variables can be used together.

Fuzzy logic based control systems has a lot of advantages. Fuzzy logic doesn't require numerical precision. As the control conditions change it can be tuned up simply according to new conditions. Most important advantage of fuzzy logic based systems is it's ability of performing simple, cheap and fast solutions to control necessity by modeling human behavior even in case of building a mathematical model is impossible.

3 The Adaptive Neuro-Fuzzy Inference System (Anfis)

Fuzzy logic and neural network technology were proved as powerful method for the modeling process parameters and outputs when the mathematical model for the process couldn't be constructed. Both methods have their disadvantage. In fuzzy logic, rules and membership functions can only be determined by experts and they couldn't be determined and adjusted automatically. On the other hand neural networks couldn't process the fuzzy information. Hence none of them is the best way for process modeling. So, combination of these two techniques may provides a better way for modeling.[10]

The ANFIS is a multilayer feedforward network and it uses neural network learning algorithms and fuzzy reasoning to model relations between given input/output data sets. Due to its ability to combine verbal power of a fuzzy system with the numeric power of a neural system, it has been shown to be prosperous in modeling a lot of processes. ANFIS has ability of learning, constructing, expensing and classifying. It has the advantage of allowing the extraction of fuzzy rules from numerical data or expert knowledge and constitutes a rule base. In addition it can adapt complicated reasoning of human brain to fuzzy systems. The main drawback of the ANFIS is the time requirement for training and determining parameters which may took much time [11].

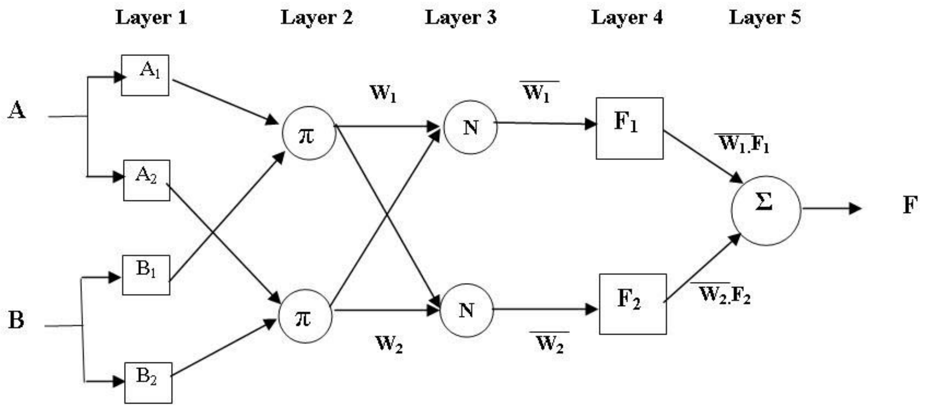


Fig. 1. ANFIS architecture

As mentioned before, fuzzy systems use rules and membership functions based on expert experience to make a decision. ANFIS architecture enables to associate inputs with outputs by itself. Thus system can learn membership functions and rules automatically without expert knowledge. Figure 1 shows ANFIS architecture. Rules for this ANFIS can be written as following form:

If (A is A_1) and (B is B_1) then $f_1 = p_1A + q_1B + r_1$
If (A is A_2) and (B is B_2) then $f_2 = p_2A + q_2B + r_2$

As shown in figure 1, some of the nodes are circular which have fixed transfer function and the others are square which have adaptive parameters. Adaptive parameters change as the network learns the relations between input and output of the network. A and B represent inputs and A1, A2 , B1, B2 represent fuzzy sets (linguistic values) of input variables. Layer 1 is the fuzzify layer which provides transition between numerical and linguistic values and determines membership degrees. Layer 2 is the product layer which products its inputs and generates one output which represents result of two rules. Layer 3 is the normalization layer which calculates firing strength of rules. Layer 4 is defuzzification layer which calculates fuzzy rule results by using a lineer function such as F=pA+qB+r where p, q and r are the lineer parameters of sugeno inference system and they can change in order to learning. Finally layer 5 is the total output layer which realizes mathematical summing and the output of this layer represents ANFIS output.

4 Related Studies

A study which aims to determine tool condition can be realized through 3 stages. 1) Collecting data by using sensors 2) Signal processing and extraction of useful information 3) Classification. Fuzzy logic is one of the most used techniques in classification stage. Li, Tso and Wang were studied to determine the relation beetween spindle motor current and tool wear by analyzing motor current using wavelet and fuzzy logic [12]. Yao, Li and Yuan, successfully determined the tool condition by using acoustic emission and motor current signals through wavelet transform and fuzzy-neuro approach [5]. Bicudo, Sokolowski, Oliveira and Dornfeld achieved high accuracy ratio by modeling based on artificial neural Networks and fuzzy logic [6]. Relation between chip size and RMS value of cutting force has been modelled based on fuzzy logic by Sokolowski and Kosmol [13]. Li [14], Ko and Cho [2] and Fang [15] have determined tool wear condition by using fuzzy expert systems, fuzzy pattern recognition and fuzzy set theory respectively. Mehrabi proposed the fuzzy set theory as optimal method in his study which directed towards determining tool wear condition by using multi-sensor method [16]. Messina has provided high accuracy about tool wear condition by using neuro-fuzzy approach [7].

5 Experimental Results

Used data in this study provided from cutting force measurement (thrust force) at vertical direction (z axis) during drilling operation on a CNC machine. Data can be separated to 3 different group. Table 1 shows cutting conditions for used 16 tools.

Table 1. Cutting Conditions

Group	Tool number	Revolution (RPM)	Feed (mm/min)	Depth (mm)
1	1-6	1800	450	10
2	7-11	1600	450	10
3	12-16	1600	300	20

5.1 Fuzzy System Results

As mentioned before, statistical parameters of measured signals such as standart deviation, mean, maximum and RMS values were calculated for each tool. These parameters have been used as fuzzy variables of fuzzy decision system. Different combinations of variables were tested. Universe of discourse were separated to 3 different fuzzy set as sharp, workable and worn for each used parameters.

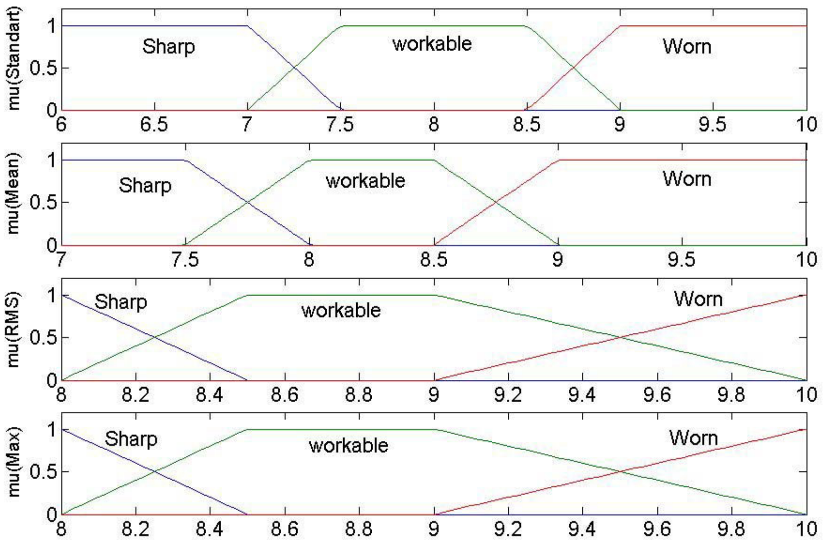


Fig. 2. Membership Functions of Statistical Parameters

Membership function is one of the most important element which effects on system performans. Different membership functions were tested in this study. Figure 2. shows trapezoidal membership functions for each statistical parameters.

Another important element which effects on system performans is rule base. Rule base represents control targets and behaviour of an expert. Constituted decision system in this study classifies tool wear condition on 3 classes as sharp, workable and worn. Different rules were written and tested. Table 2 shows the rule base used in this study. Columns show states of Mean and Standart Deviation, rows show states of RMS and Max values and each cell have a decision about tool wear according to states of statistical parameters.

Table 2. Rule Table

		Mean / Standart Deviation								
	States	**SHRP / SHRP**	**SHRP / WRK**	**SHRP / WRN**	**WRK / SHRP**	**WRK / WRK**	**WRK / WRN**	**WRN / SHRP**	**WRN / WRK**	**WRN / WRN**
	SHRP/SHRP	**SHRP**	**SHRP**	**SHRP**	**SHRP**	**WRK**	**WRK**	**SHRP**	**WRK**	**WRK**
	SHRP/WRK	**SHRP**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**
	SHRP/WRN	**SHRP**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRN**
Rms / Max	**WRK/SHRP**	**SHRP**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRN**
	WRK/WRK	**SHRP**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRN**
	WRK/WRN	**SHRP**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRN**
	WRN/SHRP	**SHRP**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRN**
	WRN/WRK	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRK**	**WRN**
	WRN/WRN	**WRK**	**WRK**	**WRN**	**WRK**	**WRK**	**WRN**	**WRN**	**WRN**	**WRN**

SHRP, WRK, WRN: Represents sharp, workable and worn tool states respectively

For simplify, three sample rules were written below.

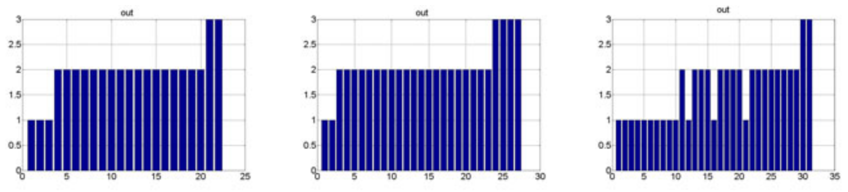
Rule 1:
IF mean is sharp AND S.Deviation is sharp AND Rms is sharp AND Max is worn THEN tool is sharp

Rule 2:
IF mean is sharp AND S.Deviation is sharp AND Rms is worn AND Max is worn THEN tool is workable

Rule 3:
IF mean is worn AND S.Deviation is worn AND Rms is worn AND Max is sharp THEN tool is worn

Developed system has been tested for 16 tool which has 3 different cutting conditions. Obtained decision results for 3 tool which are belong to 3 different group shown in figure 3.

Horizontal axis represents the number of hole drilled with tool and vertical axis shows system decision. Decisions 1,2 and 3 represents sharp, workable and worn tools respectively. For example, for tool 6, system generated sharp decision



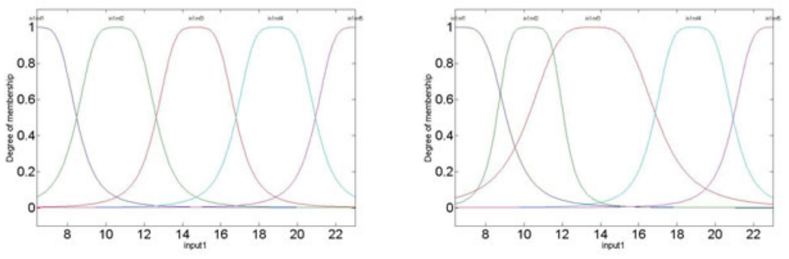
(a) Tool 6 (b) Tool 11 (c) Tool 14

Fig. 3. Fuzzy System Results

at first 3 hole, and workable decision at next 17 hole and finally worn decision at last 2 hole about tool. System has made mistake between holes 11-21 with generating unstable workable and sharp decisions for tool 14 .

5.2 ANFIS Results

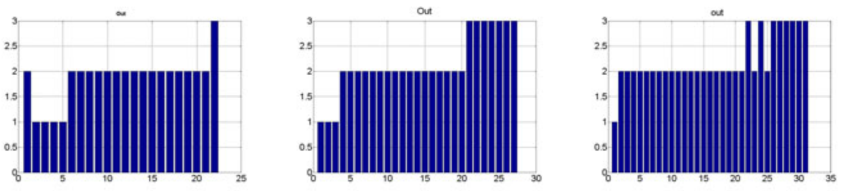
Statistical parameters of cutting forces has been used as inputs of the ANFIS system as were in developed fuzzy system. Some of the parameters were used for training of ANFIS and others for testing the system. Figure 4 shows an example initial membership function before training and final membership function after training.



(a) Initial membership function (b) Trained membership function

Fig. 4. Example Membership Functions

For training ANFIS, different tool parameters used and tested but best results accomplished when the training and testing data belong the same group of tools while fuzzy system can provide more general modeling regardless which tool group used. After training the ANFIS system, cutting parameters of three different group of tool have been used for testing.



(a) Tool 6 (b) Tool 11 (c) Tool 14

Fig. 5. ANFIS System Results

For simplify of comparison, results of same tools 6, 11 and 14 which given in figure 3 as fuzzy decisions, were displayed in figure 5 as ANFIS decisions.

As seen in figure 3 and 5, results for ANFIS and Fuzzy system resembles each other. Table 3 shows comparison between ANFIS and Fuzy decisions for tool number 11.

Table 3. Comparison Between ANFIS and Fuzzy Decisions

Tool 6	Hole Number																										
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**
Fuzzy Decision	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3
Anfis Decision	1	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3	3

1, 2, 3: Represents sharp, workable and worn tool states respectively

As shown in table 3, Fuzzy and ANFIS systems have generated four different decision about tool wear state at hole numbers 3, 21, 22 and 23. In this case, it couldn't be decided which result is more reliable because there is not clear information about tool wear state where the results are different . On the other hand, ANFIS results of tool 14 shown in figure 5-c, seems more accurate than fuzzy results because ANFIS results are more stable while fuzzy system generated unstable decisions between holes 11 and 21 as mentioned before.

6 Conclusions

In this study two decision making mechanism have been developed to determine tool wear condition by using statistical parameters of cutting forces collected during drilling process based on fuzzy logic and the neuro-fuzzy system (ANFIS). Both systems results were similar and successful. Altough results are similar it can be said that fuzzy system results are reliable on all tool data nearly but ANFIS results reliable only if the system trained by the same tool group of data with testing data. This results may show that learning ability of ANFIS couldn't be substituted with expert knowledge used in fuzzy systems or structure of tool wear mechanism isn't suitable enough for learning algorithm of ANFIS. Finally, it can be thought that if there is an expert knowledge the best way is using it, but in other case ANFIS can model a system successfully and eliminates expert knowledge necessity of fuzzy systems.

References

1. Smith Gregory C, Lee Samson S, "A method for detecting tool wear on a CNC lathe using a doppler radar detector", Springer-Verlag London Limited, 2004
2. Ko, T. J., and Cho, D. W., "Tool Wear Monitoring in Diamond Turning by Fuzzy Pattern Recognition", ASME J. Eng. Ind., 116, No. 2, 1994.
3. Huang P.T, Chen J.C, Chou C.Y, A Statistical Approach in Detecting Tool Breakage in end Milling Operations, Journal of Industrial Technology, 1999
4. Chen, W, Chen, J. C, Gemmill, D. "Analysis of an effective sensing location for detecting tool breakage in end milling operations. Journal of Industrial Technology", 1998
5. Yingxue Y, Xiaoli L, Zhejun Y, "Tool wear detection with fuzzy classification and wavelet fuzzy neural network", International Journal of Machine Tools and Manufacture, 1999

6. Bicudo, L.A., Sokolowski, A., Oliveira, J.F., Dornfeld, D.A., Thermal compensation of grinding machines using neural Networks, Fourth International Conference on Monitoring and Automatic Supervision in Manufacturing, Warsaw, 1995
7. Omez S. Mesina, Reza Langari, A Neuro-Fuzzy System for Tool Condition Monitoring in Metal Cutting, 2001
8. Ertunç H.M, Özdemir E, Oysu C, Sönmez M, Kandilli İ, Delme İşlemlerinde Kesici Takim Aşinmasini Gerçek Zamanda Gözlemleyen Bir Sistem Geliştirilmesi, TUBITAK Project, Project No: Misag-148, 2002 (in Turkish)
9. Zadeh, L., Fuzzy sets, Information Control 8, 1965
10. Gao Daoming, Chen Jie, ANFIS for high-pressure waterjet cleaning prediction, Surface and Coatings Technology xx, 2006
11. Fi-John Chang, Ya-Ting Chang, Adaptive neuro-fuzzy inference system for prediction of water level in reservoir, Advances in Water Resources 29 (2006) (1–10)
12. Xiaoli Li, Shiu Kit Tso, Real-Time Tool Condition Monitoring Using Wavelet Transforms and Fuzzy Techniques IEEE Transactıons On Systems, Man, And Cybernetıcs Part C: Applıcatıons And Revıews, Vol. 30, No. 3, 2000
13. Sokolowski, A., Kosmol, J. Feature selection for burr height estimation Fifth International Conference on Monitoring and Automatic Supervision in Manufacturing,1998
14. Li, S, Elbestawi M.A, Du R.X, "Fuzzy Logic Approach for Multi-Sensor Process Monitoring in Machining", Winter Annual Meeting of the American Society of Mechanical Engineers, 1992
15. Fang, X. D, Expert System-Supported Fuzzy Diagnosis of Finish-Turning Process States Int. J. Mach. Tools Manuf., 35, 1995
16. M.G. Mehrabi and E. Kannatey-Asibu, Jr., Mapping Theory: A New Approach to Design of Multi-Sensor Monitoring of Reconfigurable Machining Systems (RMS), 2001

Fault Diagnosis in Nonlinear Circuit Based on Volterra Series and Recurrent Neural Network

Haiying Yuan and Guangju Chen

School of Automation Engineering, University of Electronic Science and Technology of China, 610054, Chengdu, China
yhying@sohu.com

Abstract. The neural network diagnosis method based on fault features denoted by frequency domain kernel in nonlinear circuit was presented here. Each order frequency domain kernel of circuit response under all fault states can be got by vandermonde method; the circuit features extracted was preprocessed and regarded as input samples of neural network, faults is classified. The uniform recurrent arithmetical formula of each order frequency-domain kernel was given, the Volterra frequency-domain kernel acquisition method was discussed, and the fault diagnosis method based on recurrent neural network was showed. A fault diagnosis illustration verified this method. The fault diagnosis method showed the advantages: no precise circuit model is needed in avoiding the difficulty in identifying nonlinear system online, less computation amount, high fault diagnosis efficiency.

1 Introduction

The research on nonlinear system and fault diagnosis had gained some important achievement [1], [2], but the inherence complexity in nonlinear system results in slow research development of fault diagnosis in nonlinear analog circuit. if frequency domain characteristic of nonlinear system is got under continuous actuating signal, The fault recognition is simple, But continuous actuating problem in nonlinear system is unsolved yet, under non-contiguous actuating signal, frequency spectrum obtained in any system only is a subset of whole frequency spectrum, how to distinguish actual work states of system by these insufficient information is difficult in fault diagnosis theory and method based on nonlinear frequency domain response analysis. In 1995, the General Frequency Response Function (GFRF) of nonlinear system got under non-continuous actuating signal was classified by neural network to judge fault and fault sorts in diagnostic system; the thought is presented in document [3], [4], prior study showed: the method is high efficiency obviously, the Volterra model variables and GFRF model variables are regarded as input sample of neural network, the output are work state codes of diagnostic object. The large computation amount aroused by neural network with large scale input nodes is unsolved well, which made fault diagnosis efficiency low. Aim to this problem, the kernel main component analysis method was introduced into feature extraction of GFRF model, the fault feature vectors dimension compressed greatly, the circuit is diagnosed by classified function of neural network, a high efficient method is provide for fault diagnosis in nonlinear circuit based on GFRF model. Three key steps in fault diagnosis method on frequency spectrum analysis in nonlinear system were introduced: frequency spectrum acquisition, frequency spectrum feature extraction, fault diagnosis based on frequency spectrum feature.

2 The Measurement in Volterra Frequency Domain Kernel

In essence, fault diagnosis is a traditional pattern recognition problem, the constructing and choosing of fault feature is the key to diagnosis technology; good fault feature can improve diagnosis efficiency and veracity, frequency domain kernel described essential characteristic of nonlinear system, so, the unique Volterra kernel can be regarded as system features in fault diagnosis.

The time domain response is a transient process, it is difficult to measure accurately, and so, transfer function $H(S)$ is always measured in frequency domain. When the computation value of each order frequency kernel $H_n(S_1,\cdots,S_n)$ under all possible faults is same to the measurable value $H_n'(S_1,\cdots,S_n)$ under fault occurrence, the right fault diagnosis conclusion can be got. From uniqueness theorem of complex analysis, the comparison in two function between $H_n(S_1,\cdots,S_n)$ and $H_n'(S_1,\cdots,S_n)$ just the same as that between $H_n(j\omega_1,\cdots,j\omega_n)$ and $H_n'(j\omega_1,\cdots,j\omega_n)$.

For computing and measuring each order frequency domain kernel in nonlinear network $H_n(j\omega_1,\cdots,j\omega_n)$, multi-frequency sinusoidal signal can be regarded as test input signal, but in stable response of nonlinear network measured actually, some frequency component always include the contribution from different order kernels, that is, the different order kernel all include some same frequency elements, which overlapped with each other in response, the amplitude and phase in ω frequency of response only can be obtained in frequency spectrum analysis to output signal, but the element magnitude in each order kernel is un-acquirable, there is no way to define the value belonged to each order kernel at all[5], then, the key to high order transfer function measurement in nonlinear network is separating each high order transfer function from whole response, which realized by homogeneous characteristic of Volterra response, it roots in vandermonde method, showed as follows:

The n order Volterra kernel with n homogeneous characteristic, in network described by Volterra series, if the response is $y(t)=\sum_{n=1}^{\infty} y_n(t)$ aroused by input $u(t)$, $y_n(t)$ showed the response aroused by n order kernel, if the input increases to $au(t)$, the response is:

$$y = a^1 y_1(t) + a^2 y_2(t) + \cdots + a^n y_n(t) + \cdots \tag{1}$$

When $u(t)$ choose some definite wave signals, the magnitude changed, the formula (1) can be regarded as multinomial expanded formula of y to a, each order response component y_n is the polynomial coefficient. e.g. the primary N Volterra frequency domain kernel in system need to measure, if $N=3$, three same waveform, different magnitude input $a_1u(t), a_2u(t), a_3u(t)$ taken, the corresponding output Y_1, Y_2, Y_3 will be measured, the equation can be got:

$$\begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \end{bmatrix} = \begin{bmatrix} a_1 & a_1^2 & a_1^3 \\ a_2 & a_2^2 & a_2^3 \\ a_3 & a_3^2 & a_3^3 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} \tag{2}$$

Here $e_i(i=1,2,3)$ include measured error and truncation error generated by over four order kernel, the matrix in formula (2) called vandermonde matrix, when $a_i(i=1,2,3)$ is unequal to each other, the inverse matrix existed, so, when $e_i=0(i=1,2,3)$ taken, y_1, y_2, y_3 can be got. The vandermonde method only fits for feeblish nonlinear network (high order kernel decaying quickly).

The disadvantage is sensitive to measure error, increasing measure time can overcome it, a group over-determined equation (equation numbers are more than unknown quantity numbers y_n) can be got, seeking for each order solution of this group equation by least square method. In theory, each order kernel can be measured accurately by choosing multi-frequency signal and vandermonde method properly. Seeking for the solution to over-determined equation, the linear parameter of over-determined equation estimation method resolved based on neural network showed in document [6], the computation speed increase greatly, diagnosis realized online.

In the frequency domain kernel measurement of fault network with multi-frequency signal, the magnitude, frequency, phase of several different frequency sinusoidal signal need be controlled accurately, superposition as actuating signal $u(t)$ of measured fault circuit, the frequency spectrum function $Y(\omega)$ can be got by circuit response $y(t)$ Fourier transformed, high order frequency domain kernel measured can be computed out with frequency spectrum analysis results by vandermonde method, the state features of measured circuit can be obtained, through compression transform and normalized process, which are regarded as input sample of neural network for pattern classify and fault diagnosis of system.

3 Fault Diagnosis Based on Inner Recurrent Neural Network

The research applications on nonlinear system analysis and fault diagnosis based on neural network get more attention. BP network is difficult to deal with system with stronger real-time characteristic, so, feedback signal and deviation unit are added to form inner recurrent neural network (IRN), which describe nonlinear dynamic behavior in system by inner state feedback, specially, the experience knowledge is convenience for introduce into learn process, which improve learn speed greatly. An inner recurrent neural network model with deviation unit and error inverse transfer algorithm method were presented here, it is applied in simulation analysis to fault diagnosis.

The recurrent neural network model composed by three layer is showed: input layer node, hide layer node and output layer node, two deviation nodes are added to hide layer and output layer respectively, the hide layer node receive not only output signal from input layer, but also one-step delay output signal from node-self in hide node, it is called associate node. Supposing NH and NI as hide node numbers and input node numbers (except for deviation node), $I_j(k)$ is the jth input of recurrent neural network with deviation at time k , $x_j(k)$ is output of the jth hide layer node , $y(k)$ is output vector of recurrent neural network with deviation unit, the IRN network can be described by mathematical formula as follows:

$$y(k)=\sum_{j=1}^{NH} WO_j x_j(k)+WO_{bias} \tag{3}$$

$$x_j(k)=\sigma(S_j(k)) \tag{4}$$

$$S_j(k)=\sum_{i=1}^{NH} WR_{ij}x_i(k-1)+\sum_{i=1}^{NI} WI_{ij}I_i(k)+WI_{bias} \tag{5}$$

In formula, $\sigma(\cdot)$ is nonlinear activation function of hide layer node, WI, WR, WO is weigh coefficient from input layer to hide layer, recurrent signal and weigh coefficient from hide layer to output layer respectively, WI_{bias}, WO_{bias} are weigh coefficient of deviation unit added to hide layer and output layer. From formula (5), the output of hide layer node can be regarded as dynamic system states, here, the IRN network is the state space expression for nonlinear dynamic system, the hide node can storage input/output information. The research and application on IRN get progressively, Su apply IRN for nonlinear system modeling successfully, Ku&Lee, Narendra adopt IRN model in nonlinear system identification and control, the effect is satisfied.

The usual fault diagnosis model based on neural network (BP network) include three layer, input layer: receiving all fault information and phenomenon from actual system; middle layer: the fault information got from input layer is transferred into target resolve way by inner learn and process; output layer: according to input faults form, the fault decision can be made by adjusting weigh coefficient W_{ij}. In short, the fault diagnosis on neural network model is training node link weigh value which is stable convergence with samples, sample symptom parameters diagnosed are input into network; computing the actual output of network, according to output magnitude sequence to define fault sorts. each frequency domain kernel under all possible faults can be computed out (own to quick series decaying, only fore three order kernel are computed enough) then, they are normalized to form train sample sets, then, the train samples are input into neural network for training, just as a fault dictionary is set in neural network [2], the neural network classifier trained can classify test samples of actual each order kernel in fault network, fault automation inquiry and diagnosis are completed in fault dictionary.

4 Diagnosis Circuit Illustration

The fault diagnosis in nonlinear circuit verifies this method presented. Firstly, the GFRF estimation in nonlinear system under non-continuous estimation signal, the feature parameters of diagnosis system were extracted and preprocessed, they form input/output sequence with fault pattern diagnosis to construct sample set. The Volterra frequency domain kernel was provide for neural network as input sample, the output of neural network is corresponding to present work states of system, choosing proper network structure, train network with train sample, testing the network performance with test sample. The fault feature vectors are input into neural network trained to diagnose faults and fault sorts system.

4.1 Volterra Series Distinguish and Feature Extraction of Nonlinear System

When nonlinear system operate at a predefined work point, self-adaptation linear model can approach it with high precision, when system jump to other work point, the network can transfer to new work point correspondingly. Volterra series model can complete features extraction of actual system in definite time, seeking for GFRF estimation from it to estimate Minimum Square error in nonlinear system behavior. For guarantying Volterra series model can track current system work point in nonlinear system as soon as possible, the sample rate must be enough quick. whereas, for obtaining enough more information for system modeling in reason, network needs least cognitive time, adding small volume noise to decrease time possible, the noise made nonlinear system increase more operation point dynamically in very short time, the learning is more quicker. Of course, the noise must be small enough to ensure normal work of system.

A weak non-linear circuit with nonlinear capacitor showed in figure 1, linear resistance and voltage source, supposing sinusoidal signal with multiform non-commensurability frequency is added into circuit, the transfer function between excitation voltage $v_s(t)$ and response current $i(t)$.

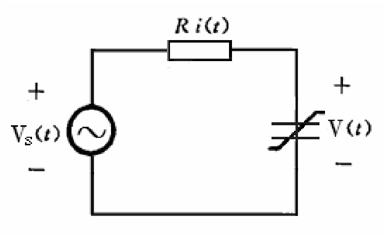


Fig. 1. Weakly nonlinear circuit

Under small-signal excitation and weakly nonlinearity, the direct component is very small, it can be ignored completely, the actual influence from direct component is the value of offset current and voltage which deviate from static state work point little, if the deviation value is bigger, Volterra series analysis method is invalid.

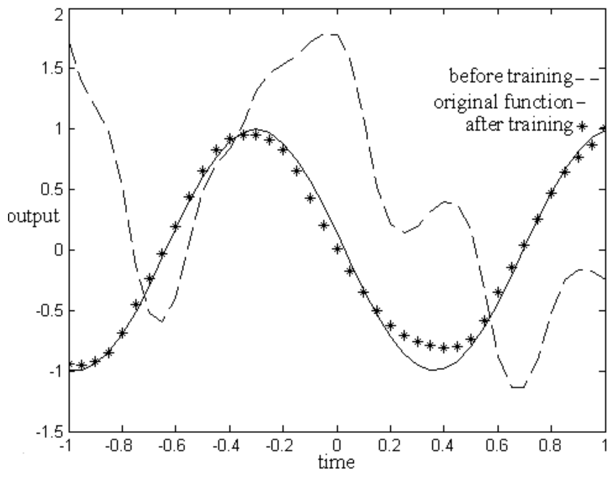


Fig. 2. The identification effect of Volterra series

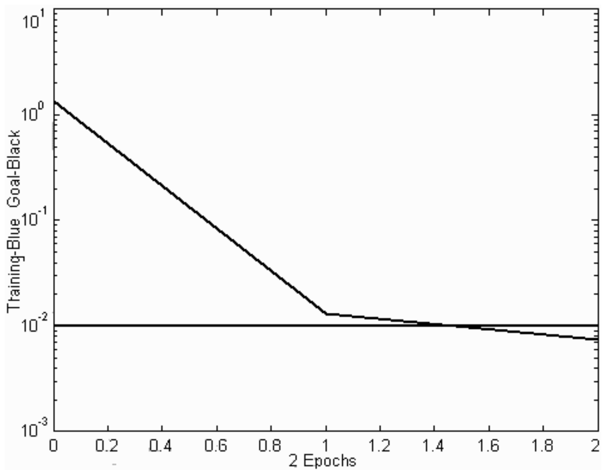


Fig. 3. Identification error

Nonlinear Volterra frequency domain kernel was estimated by neural network [7], train time is 50 identity time, the distinguish error is less than 0.01, other parameter adopted default value. The output contrast waveform of Volterra series in distinguish process of nonlinear system showed in figure 2, by this denotation, at fast sample rate, GFRF model can distinguish nonlinear system in high precision. in figure 3, the distinguish process of nonlinear system is very fast, passing two identity time, Mean Square Error changed from 12.0426/0.01 to 0.00233035/0.01, the error grads change form 299.229/1e-010 to 2.21394/1e-010, the distinguish error satisfy need.

4.2 The Fault Diagnosis Based on Neural Network

The nonlinear system is analyzed by Volterra series firstly, the fault symptom function extracted is compressed and normalized, which are divided into train sample and test sample, take thirty times computation value of each order kernel, twenty as train sample, ten as test sample, the neural network designed was trained with train sample, adjust weigh value, network satisfy system error precision; the neural network trained was freeze, test code is input into network, which is under recollection state, the work state at each test point are judged to diagnose faults with these recollection results [8].

The fault classification is realized by recurrent neural network with deviation unit [9]. the input of IRN have three neutron corresponded to three order Volterra frequency domain kernel respectively, five neurons in output layer corresponding to five bits of one work state respectively, the corresponding output bit as 1 showed the fault class which is corresponded to this bit occurrence. Six neutron cells are in hide layer, the configuration of other associate node and deviation unit are set properly. train samples showed in table 1, test codes are regarded as network input, fault codes are regarded as network output, the learn rate is 1.5 in first layer and second layer respectively, deviation learn rate of input is 1.0, learn to the eighth step, the precision is superior to 0.05, the recollection results shows in table 2, the illustration results showed the fast error convergence in inner recurrent neural network.

Analysis: the circuit response in time domain is difficult to measure and the waveform is difficult to compile into fault dictionary, so, it is a feasible thought in fault diagnosis based on Volterra frequency kernel; under weakly nonlinear system and network, the high order frequency domain kernel attenuate quickly, the fore tree

Table 1. Test and fault code

Fault sequence	Test code	Fault code
1	111	00000
2	010	10000
3	100	01000
4	110	00100
5	101	00010
6	011	00001

Table 2. Recollection result of train mode

Test code					
111	010	100	110	101	011
Fault code					
Bit1	Bit1	Bit1		Bit1	Bit1
0.0001	0.0001	0.0000		0.0000	0.0001
0.9931	0.0000	0.0002		0.0001	0.0000
0.0000	0.9931	0.0002		0.0001	0.0000
0.0000	0.0000	0.9939		0.0002	0.0001
0.0001	0.0001	0.0000		0.9917	0.0001
0.0001	0.0001	0.0000		0.0000	0.9958

order frequency domain kernel was adopted for fault diagnosis; the feature vector in nonlinear system is compressed by main component analysis, the convergence and classified speed of neural network improve greatly, the fault diagnosis in nonlinear system online was realized.

5 Conclusions

The fault diagnosis method based on frequency spectrum in nonlinear circuit was researched here, the frequency response component in nonlinear analog circuit is regarded to construct research object, the nonlinear system was identified by Volterra frequency domain kernel, the frequency spectrum in nonlinear system was analyzed by GFRF analysis to extract state features in nonlinear analog circuit, the IRN network with deviation node was adopted to make fault decision, an diagnosis illustration verify frequency spectrum estimation and fault diagnosis based on neural network with nonlinear Volterra frequency domain kernel.

Acknowledgment

The Natural Science Funding Council of China under Grant (No. 60372001; No. 90407007); the Ph.D. Programs Foundation of Ministry of Education of China under Grant (No. 20030614006).

References

1. L. Jiao, "Fault Diagnosis in Nonlinear Circuit and System: a New Theory and Method," Science in China (Series A) 1988, vol. 6, pp.649-657(in Chinese).
2. C. Han, et al., "Identification of Nonparametric GFRF Model for a Class of Nonlinear Dynamic Systems," Control Theory and Applications 1999, vol. 16, no. 6, pp.816-825.
3. D. Aiordachioaie, E. Ceanga, "Estimation, Compression and Classification of Volterra Kernels with Application to Process Diagnosis," Proceedings of the IEEE International Symposium on Computer-Aided Control System Design, 1999, pp.170-175.
4. C. Han., et al., "Identification of Nonparametric GFRF Model for a Class of Nonlinear Dynamic Systems," Control Theory and Applications 1999, vol. 16, no. 6, pp.816-825.
5. O. Chual, Y. NGC, "Frequency domain analysis of nonlinear systems, formulation of transfer function," IEEE Electronic circuits and systems, 1979, vol. 3, no. 11, pp.257-269.
6. A. Billings, K. A. Tsang, "Spectral analysis for nonlinear systems part II: interpretation of nonlinear frequency response functions Mechanical systems and signal processing, 1989, vol. 3, no. 4, pp. 345-350.
7. Andrzej cichocki, Rolf Unbehauen, "Neural network for solving systems of linear equations and related problems," IEEE Transactions on circuit and systems, 1997, vol. 39, no. 2, pp.785-802.
8. H. Yuan, G. Chen, "A Method for Fault Diagnosis in Nonlinear Analog Circuit Based on Neural Network," Conference proceedings of the Seventh International Conference on Electronic Measurement & Instruments, Beijing, 2005, vol. 8, pp.66-70.
9. J.A. Starzyk, M.A. El-Gamal, "Artificial neural network for testing analog circuits," IEEE International Symposium on Circuits and Systems, 1990, vol. 3, pp.1851-1854.

Manufacturing Yield Improvement by Clustering

M.A. Karim, S. Halgamuge, A.J.R. Smith, and A.L. Hsu

Department of Mechanical and Manufacturing Engineering,
University of Melbourne, VIC 3010, Australia
makarim@gmail.com, saman@unimelb.edu.au, ajrs@unimelb.edu.au, alhsu@unimelb.edu.au

Abstract. Dealing with product yield and quality in manufacturing industries is getting more difficult due to the increasing volume and complexity of data and quicker time to market expectations. Data mining offers tools for quick discovery of relationships, patterns and knowledge in large databases. Growing self-organizing map (GSOM) is established as an efficient unsupervised data-mining algorithm. In this study some modifications to the original GSOM are proposed for manufacturing yield improvement by clustering. These modifications include introduction of a clustering quality measure to evaluate the performance of the programme in separating good and faulty products and a filtering index to reduce noise from the dataset. Results show that the proposed method is able to effectively differentiate good and faulty products. It will help engineers construct the knowledge base to predict product quality automatically from collected data and provide insights for yield improvement.

Keywords: Data mining, Self-organising map, Clustering quality, Filtration of noisy data, Yield improvement.

1 Introduction

The volume of manufacturing data is growing at an unprecedented rate, both in the number of features and objects (instances). Manufacturing databases usually comprise of process control, process step and quality control data. In many applications, data is automatically generated (by sensors) and therefore the number of objects can be very large. The combination of upstream information (supplier information for example) and manufacturing & product quality data results in a large number of features in the dataset. Traditional techniques of dealing with quality problems are the use of statistical process control (SPC) and design of experiments (DOE). However, these techniques fail to extract underlying features from complex data [1]. Moreover, these methods are highly time-consuming. To shorten design cycle times, it is very important for designers to have a tool that can quickly and efficiently analyze manufacturing data, estimate product performance, predict the effects of design changes and manufacturing variations, and determine optimum design parameters for new products.

Data mining appears to offer a solution to the shortcomings of traditional methods and offers tools for the discovery of patterns, associations and statistically significant structures and events in data. Thus, it has the potential for providing the underpinning technology for decision support tools.

2 Unsupervised Clustering Method

Cluster analysis is a technique that processes data that do not have known class labels, or where the analyst opts not to use them. Data within a cluster are highly similar to one another, but are very dissimilar to data in other clusters. Each cluster that is formed can be treated as a class of data, on which class or concept characterisation and discrimination can be performed.

2.1 SOM and GSOM

A popular and well-accepted self-organising method of neural network analysis is Self-Organising Maps (SOM) [2]. SOM, when two-dimensional topological constraints are defined, are useful for providing visualisation of high-dimensional data by projecting it onto a two-dimensional network. However, the two-dimensional SOM are rectangular grids of neurons that have predefined sizes and are specified by widths and heights. This can potentially hinder the real representation of input space topology [3].

Consequently, several modifications to SOM have been proposed to overcome the problem of predefined grids. One of the most recent variants, called the growing self-organising map (GSOM) was proposed to let the algorithm determine the size of the feature map [3-6]. Hsu et al. [7] proposed a hierarchical clustering algorithm using multiple layers of GSOM that automatically suggests an appropriate number of clusters that are present in the data. A significant difference between standard SOM and GSOM is that SOM is not able to grow but GSOM grows according to its own growing criterion. A parameter of growth, the growth threshold (GT), is defined as:

$$GT = -D \times \ln(SF) \tag{1}$$

where D is the dimensionality of data and SF is the user defined spread factor that takes values between 0 and 1 with 0 representing minimum and 1 representing maximum growth. The size of the output cluster map will depend on the value of growth threshold.

2.2 Proposed Modifications

The GSOM algorithm was developed for clustering and class discovery and its primary applications were for biological datasets. Manufacturing datasets are complex due to the large number of processes, diverse equipment set and nonlinear process flows and often comprise of hundreds of process control, process step and quality control data. To deal with complexity of manufacturing data, the following two modifications are proposed for GSOM developed.

Evaluating Clustering Results

With manufacturing data, it may not be straightforward to obtain a good clustering. It might be necessary to change different variables and monitor the change of ‘quality’ of the cluster. A clustering quality measure, CQ, has been proposed as a benchmark [8] to evaluate the results of running simulations with different parameter changes. Mathematically, clustering quality can be expressed as:

$$CQ = \sum_i \max\left\{ \frac{\frac{g_i}{n_i} - \frac{G}{N}}{1 - \frac{G}{N}} * \frac{n_i}{N}, 0 \right\} + \sum_i \max\left\{ \frac{\frac{b_i}{n_i} - \frac{B}{N}}{1 - \frac{B}{N}} * \frac{n_i}{N}, 0 \right\} \tag{2}$$

where, B is total number of faulty products, G is total number of good products, N is total number products (B+G), b_i is number of faulty products in neuron i, g_i is number of good products in neuron i, and n_i = number of products in neuron i ($b_i + g_i$).

A CQ of 1 would mean a perfect separation of good and faulty products and a CQ of 0 would mean no separation at all. This proposed CQ takes the complete cluster map into account and can be generated automatically as an objective quantifier of the programme to separate good and faulty products from the data provided

Filtration of Noisy Data

It was mentioned above that manufacturing databases are usually large and complex and may have substantial amounts of noise. Noise reduction from the data is one of the primary conditions for obtaining good clustering. Manufacturing data may contain many categorical variables, which are comprised of letters or a combination of numbers and letters. These categorical data should be transformed to numerical data to allow their use in data mining programmes. One of the ways to transform categorical data to numerical data is to use 0 and 1 as shown in Table 1 for a simple case of 3 categories. In Table 1, the first column is the original dataset. A new column is created for each categorical variable and a value of 1 or 0 is provided for the existence and non-existence of that variable in the respective row. In this process each column with categorical data will be expanded to many columns depending on the number of different categorical values in that column. This makes the dataset very large and noisy.

Table 1. Expansion of Categorical Data

Original data	Expanded data		
	XS21C	PG201	P4812
XS21C	1	0	0
PG201	0	1	0
P4812	0	0	1

Investigation of sample manufacturing datasets has revealed that there are many categorical variables that do not have any affect on separating good and faulty products. For example a categorical variable may have an equal distribution among good and faulty products. These variables, when expanded, only add noise to the dataset. To filter out these unnecessary categorical variables, a method is proposed to delete them. Mathematically the constraint can be expressed as:

$$FI = \left(\frac{X_{ig}}{N_g} - \frac{X_{if}}{N_f} \right) \leq \alpha \tag{3}$$

where, FI = Filtration Index, X_{ig} = number of good products having a particular categorical variable , X_{if} = number of faulty products having a particular categorical variable, N_g = number of good products in the sample, N_f = number of faulty products in the sample, α = user defined constraint limit.

If the Filtration Index for a variable is close to zero, it can be considered that the variable is equally distributed among good and faulty products and hence should not be considered for analysis. For example, if a dataset has 50 good products and 25 faulty products and among them 10 good products and 5 faulty products have the same variable, then FI will be [(10/50)-(5/25)] = 0. As the variable is similarly distributed among the good and faulty products, it will not have any effect on clustering. However, it is not expected that a variable would have FI value exactly zero. In practice, the user has to define the value (or range of values) for α depending on the characteristics of the dataset. Although the Filtration Index has been proposed for categorical variables, it can similarly be used for numerical variables.

3 Results and Discussions

The method proposed has been applied to several manufacturing datasets. The results demonstrate the effectiveness of the methodology. Details of the simulation results are described in the following sections.

3.1 Manufacturing Process for Industrial Conveyor Belts

Manufacturing process data for industrial conveyor belts reported by Hou et al. [9] was used in this study. For the conveyor belt manufacturing process, the temperature at the rollers, thickness of the belt on both sides and revolution speed of the final product are three key parameters that determine the quality of the finished product. To ensure these parameters are within acceptable ranges, they are monitored and controlled closely. A total of 27 records of these key parameters and faulty product types were used and a sample of the dataset is shown in Table 2. In the last column of the table, different types of faults (1, 3, 7 and 9) are shown. Four good finished products are also included in the table and designated as fault type 0. So, there are five product quality categories. It is expected that GSOM should be able to cluster the different types of faults as separate clusters. In order to use the data in GSOM, the manufacturing parameters are normalized between 0 and 1.

Table 2. Manufacturing Process Parameters and Faulty Product Types

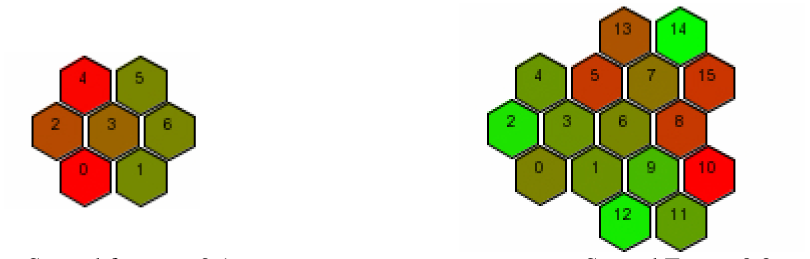
Data	Spee	Thic	Thic	Thic	Tem	Tem	Tem	Tem	Fault
1	3.72	5.58	5.58	5.6	77.8	82.1	80.6	80.6	1
2	3.62	5.58	5.58	5.61	76.7	82.3	81.6	80.6	7
3	3.58	5.79	5.62	5.6	76.6	82.3	81.6	80.6	7
4	3.42	5.67	5.61	5.65	76.6	82.1	80.5	80.6	3
•	•	•	•	•	•	•	•	•	•
23	3.42	5.61	5.58	5.6	76.7	82.2	80.6	80.6	3
24	3.6	5.57	5.58	5.59	76.7	82.2	80.5	80.7	0
25	3.62	5.58	5.6	5.6	76.8	82.1	80.6	80.7	0
26	3.61	5.6	5.61	5.61	76.6	82.2	80.5	80.6	0
27	3.62	5.58	5.61	5.59	76.6	82.1	80.6	80.6	0

Since manufacturing parameters for different quality categories are not significantly different, detection of patterns is challenging. However, as the dataset is small and there are no categorical variables, it is expected that no filtration of data will be required. Simulation results for different runs at different spread factors (SF) are presented in Table 3. Cluster maps at SF 0.1 and 0.9 are shown in Figure 1. Clusters in the cluster maps are numbered for the convenience of analysis. In the programme, the last column of the dataset is considered as the object identifier. So the fault categories (last column in Table 2) will be shown in the clusters of the output map. In Table 3, the first column shows the SFs used in the simulation, the 2nd to 15th columns show the number of products and the fault category clustered in each cluster and the last column shows the calculated clustering quality. For example, data in the 1st row of the 2nd column shows that there are 6 products (shown in parentheses) of failure category 3 in cluster 0. As there is no data in clusters 8 and 11, they are not shown in Table 3.

Table 3. Distribution of Faulty and Good Products in Different Clusters

SF	C 0	C 1	C 2	C 3	C 4	C 5	C 6	C 7	C 9	C 10	C 12	C 13	C 14	C 15	CQ
0.1	3(6)	**0(2) 1(1)**		0(2)	9(7)		**7(6) 1(3)**								0.95
0.3	3(6)	**0(2) 1(1)**		0(2)	9(7)		**7(6) 1(3)**								0.95
0.5	3(6)				7(4)	7(2) 1(2)		0(4) 1(2)	9(7)						0.96
0.9		7(1)	3(2)		3(4)		0(4)			1(4)	7(5)	9(2)	9(3)	9(2)	**1.00**

At lower SFs, good and faulty products are not perfectly separated. It can be seen that at SF 0.1 and 0.3 there is mix up of good and faulty products in clusters 1 and 6 (shown in bold). Clustering quality of the entire cluster is 0.95 in both cases. As SF is increased, CQ increased correspondingly and reaches its maximum at SF 0.9. That means at SF 0.9, good and faulty products are completely separated. However, it does not mean that results at lower SF are meaningless. At SF 0.1, presence of fault category 0 and 1 in cluster 1 and fault category 3 and 6 in cluster 6 indicate that there is something common among fault categories 0 and 1 and also among fault categories 3 and 6.



Spread factor = 0.1 Spread Factor 0.9

Fig. 1. Cluster Map Showing the Influence of the Spread Factor

3.2 Wafer Manufacturing Data

Recorded manufacturing process data can be used to analyse the performance of a process. However, it is difficult to find the causes of any abnormal output or the factors resulting in lower yield rates [9]. To determine whether GSOM could deal with a complex manufacturing dataset, a simulation has been run with a large dataset obtained from Motorola USA. The quality problem of the Motorola wafer fabrication process is described in reference [1].

In the context of analysis of manufacturing quality problems, the focus involves two main aspects - separation of good and faulty products and identifying the reason for yield failure. The present study focuses on the first aspect. The challenge is not solely in clustering, but also to obtain a meaningful and adequate number of clusters. With meaningful clusters, grouped in appropriate numbers, identification of the reasons that contribute significantly to the differentiation of clusters should become a simpler task.

The dataset is the historical wafer data collected for 2500 wafers over a 2-month period. The input database measured 133 parameters by 16,381 entries organized into an Excel file. The data consisted of Wafer Probe Data of 39 wafer probe functional tests, Process Control Data (59 numerical electrical PC measurements probed at 8 sites per wafer) and Process Step Data such as material vendor/lot, wafer boat position, etc. A sample of the dataset is shown in Table 4. The second column in the table is the product ID, the 3rd column is the product reference number and columns C1 to X133 are measured parameters. In the original dataset, there are 59 'C' columns (C1-C59), 38 'K' columns (K60-K98) and 34 'X' columns (X99-X133). The reference number in the 3rd column identifies which product is good and which product is faulty. Reference numbers above 8750 indicate good products and below 8750 indicate faulty product.

Table 4. Motorola Wafer Manufacturing Data (showing dimensionality and layout)

No.	NAME	REF	C1	C2	•••	K60	K61	•••	X132	X133
1	J546040_12_1	9628	1.00E-09	2.21E-02	•••	33	18	•••	R2793	Dec-18-95
2	J546040_12_2	9628	-1.33E-09	2.39E-02	•••	33	18	•••	R2793	Dec-18-95
• •	• •	• •	• •	• •	• •	• •	• •	• •	• •	• •
16381	F606617_21_5	9584	-2.69E-08	2.23E-02	•••	19	18	•••	R285	Feb-26-96

As can be seen in the table, there are many categorical variables especially in the X-columns. This dataset must be converted into a suitable format to use in GSOM. Preprocessing of data includes removal of outliers, transformation of categorical data into numerical values and normalization. It was found that there are some excessively large values in the dataset and if these are not removed, most of the values will become zero after normalization. After removing the outliers, the categorical data are expanded following the procedure suggested in Table 1 and then normalized between 0 and 1. This now presents a significant problem as the 133 columns of the dataset now become 832 columns with 16,381 entries. Considering the computational time and complexity of the dataset, about one quarter of the entries (4000) was considered for simulation purposes.

Simulation was run at SF 0.1 first but a very poor (0.25) CQ was obtained. Clustering quality was improved with higher SF but the maximum CQ was only 0.52, obtained at an SF of 0.95. It is thought that introduction of a large number of attributes (because of the expansion of categorical data) created unnecessary noise in the dataset. It may be necessary and of great insight to study the effect of removing several attributes from the input dataset and to check the impact on the generated maps.

The categorical data from the dataset were reduced using equation (3). First using the constraint IF ≤ 0.05, 163 categorical variables were deleted and the simulation was rerun. A significantly higher CQ of 0.71 was achieved. To test further, more categorical variables were deleted. Using IF ≤ 0.15 a CQ of 0.81 was achieved. As further deletion of variables did not produce any more significant improvement, no further deletion was done. Moreover, if many variables are deleted, some important features may be lost. A summary of the results is presented in Table 5.

Table 5. Improvement of Clustering Quality Changing SF and IF

SF(IF≤ 0)	**0.1**	0.5	0.7	0.8	0.9	**0.95**
CQ	**0.25**	0.40	0.45	0.49	0.51	**0.52**
IF (SF=0.95)	**≤ 0.05**	≤ 0.10	≤ 0.12	≤ 0.13	≤ 0.14	**≤ 0.15**
CQ	**0.71**	0.76	0.78	0.79	0.80	**0.81**

Although higher clustering quality is desired, it may not be possible to obtain high CQ with complex manufacturing data as 100% separation of all good and faulty products is not possible. For example, consider that the desired dimension of a product is 10mm. If the dimension of the finished product is 11mm it will be considered a failure and if the dimension of final product is 15mm it is also a failure but certainly different to the previous failure. It cannot be expected that all products with a dimension 10mm (plus tolerance) be clustered in one group and the rest will be grouped in another cluster. There will certainly be some mixing. In practice, some clusters with 100% good and faulty products and the rest of the clusters with majority either good or faulty products are expected. In this study some clusters with 100% good and faulty products were obtained and other clusters either contained majority of good products or faulty products.

4 Conclusions

The proposed methodology is a significant contribution to the data mining techniques available, especially in dealing with complex datasets (such as manufacturing data). To the best of the authors' knowledge there is no similar methodology in the literature. For example, Hou et al. [9] used a back propagation neural network method to identify faulty product categories. It was shown that the network was able to identify faulty product groups. However, the dataset used in that study is very simple and real manufacturing datasets in no way are so simple and straightforward. Moreover, product quality may depend on many factors [1]. Hou et al. [9] have not suggested how to deal with a complex manufacturing dataset and what to do if a reasonable separation is not achieved.

Application of data mining to manufacturing is relatively limited mainly because of complexity of manufacturing data. The original GSOM algorithm has been proven to be an efficient algorithm to analyse unsupervised DNA data. However, it produced unsatisfactory clustering when used on manufacturing data. Moreover, there was no benchmark to monitor improvement in clustering. The present study has proposed methods to evaluate quality of the clusters produced by GSOM and to remove insignificant variables from the dataset. With the proposed modifications, significant improvement in unsupervised clustering was achieved with simple as well as complex manufacturing data. Possible further improvement in cluster analysis may be achieved by applying evolutionary optimization algorithms [10].

For manufacturing data, the objective is not limited to finding the separation of good and faulty products. The main objective is to find the underlying reason for poor yield. To discover this, it is necessary to create clusters of good and faulty products, as has been done in this study. The technique is being extended to model the failure causes of the lower yielding products.

Acknowledgements. The authors would like to thank Robert M. Gardner, Director Intelligent Systems Research Lab, Motorola USA, for kindly providing the valuable semiconductor wafer data. The Authors also thank Prof. Malcolm Good and The Research Centre for Advanced By-Wire Technologies (RABiT), The University of Melbourne, for allowing the use of their computing facilities.

References

1. Gardner, M., Bieker, J., Elwell, S., Thalman, R., Rivera, E.: Solving Tough Semiconductor Manufacturing Problems using Data Mining. IEEE/SEMI Advanced Semiconductor Manufacturing Conference, Boston, MA (2000) 46-55.
2. Kohonen, T.: Self-Organising Maps. Springer , Berlin (1997).
3. Alahakoon, D., Halgamuge, S. K., Srinivasan, B.: Dynamic Self-Organising Maps with Controlled Growth for Knowledge Discovery. IEEE Transactions on Neural Networks. 11 (2000) 601-614.
4. Alahakoon, D.: Controlling the Spread of Dynamic Self Organising Maps. Journal Neural Computing and Applications, 13(2), (2004) 168-174.
5. Wickramasinghe, L.K., Alahakoon, L.D.,: Dynamic Self Organizing Maps for Discovery and Sharing of Knowledge in Multi Agent Systems. Web Intelligence and Agent Systems: An International Journal, (IOS Press), 3(1)(2005).
6. Hsu, A. L., Halgamuge, S. K.: Enhancement of Topology Preservation and Hierarchical Dynamic Self-Organising Maps for Data Visualisation. Elsevier, 32 (2-3) (2003).
7. Hsu, A. L., Tang, S.-L., Halgamuge, S. K.: An Unsupervised Hierarchical Dynamic Self-Organizing Approach to Cancer Class Discovery and Marker Gene Identification in Microarray Data. Bioinformatics 19 (2003) 2131–2140.

8. Russ, G., Karim, M. A., Islam, A., Hsu, A. L., Halgamuge, S. K., Smith, A. J. R, Kruse, R.: Detection of Faulty Semiconductor Wafers using Dynamic Growing Self Organizing Map. IEEE Tencon, Melbourne, Australia (2005).
9. Hou, T.-H., Liu, W.-L., and Lin, L.: Intelligent Remote Monitoring and Diagnosis of Manufacturing Processes Using an Integrated Approach of Neural Networks and Rough Sets. Journal of Intelligent Manufacturing 14 (2003) 239-253.
10. Ratnaweera, X . A., Halgamuge, S. K., Watson, H. C.: Self-Organizing Hierarchical Particle Swarm Optimizer with time varying acceleration coefficients. IEEE Transactions on Evolutionary Computation (2004).

Gear Crack Detection Using Kernel Function Approximation

Weihua Li[1], Tielin Shi[2], and Kang Ding[1]

[1] School of Automotive Engineering,
South China University of Technology, 510640 Guangzhou, China
{whlee, kding}@scut.edu.cn
[2] School of Mechanical Science and Engineering,
Huazhong University of Science and Technology, 430074 Wuhan, China
tlshi@mail.hust.edu.cn

Abstract. Failure detection in machine condition monitoring involves a classification mainly on the basis of data from normal operation, which is essentially a problem of one-class classification. Inspired by the successful application of KFA (Kernel Function Approximation) in classification problems, an approach of KFA-based normal condition domain description is proposed for outlier detection. By selecting the feature samples of normal condition, the boundary of normal condition can be determined. The outside of this normal domain is considered as the field of outlier. Experiment results indicated that this method can be effectively and successfully applied to gear crack diagnosis.

1 Introduction

Feature extraction methods play an important role in machine condition monitoring and faults diagnosis, from which the diagnostic information can be obtained. The omnipresence of gearbox in rotating machine made the study of gearbox condition monitoring a more interesting subject [1-4]. However, the industrial signal is always complex because the strong random noises in an industrial environment degrade the signal-to-noise ratio greatly. Especially at the first stage of the gear fault such as tooth crack, it is very difficult to detect the weakly fault information immerged in noises.

There are many techniques for extracting indicators of a machine's condition from the vibration signals, such as power spectrum, neural networks and time-frequency distribution. But the choice of features is often arbitrary leading to situations where several features can be providing the same information as well as some features providing no useful information at all. The additional burden of computing these features may increase the learning cost and affect real-time application of the condition monitoring system. So feature selection is helpful to reduce dimensionality, discard deceptive features and extract an optimal subset from the raw feature space [5]. It is critical to the success of faults recognition and classification.

Kernel based methods have recently gained wide attention and enabled new solutions and improved know ones in the field of communication, geophysics, biomedical, text categorization and patter recognition etc [6-10]. The main idea behind kernel methods is to map the input space into a convenient high dimensional feature space *F*, where variables are nonlinearly related to the input space. In the feature space, one can solve the problem in a classical way. It is able to approximate almost any nonlinear functions, and is possible to model the nonlinear dynamics of gearboxes using kernel methods when failures occurring. SVM (support vector machine), KPCA (kernel principal component analysis) and KDA (kernel discriminant analysis) have found many applications in machine fault diagnosis [11-14].

In this paper, KFA (Kernel Function Approximation) based method was used to extract the feature sub-space of gear vibration signals' raw feature space, which was composed of some statistical features. By selecting the feature samples of normal condition, the boundary of normal condition can be determined. The outside of this normal domain is considered as the field of outlier, which can be seen as the samples of gear crack failure.

The paper is organized as follows: in Section 2, an overview of feature vector selection and kernel function approximation is introduced. The gear failure experiment is described in Section 3, and the proposed method is used to classify different faults. Section 4 concludes with a discussion of the results.

2 Kernel Function Approximations for Outlier Detection

According to SVM and other kernel methods such as KPCA, if the kernel matrix **K** of dot products is well defined, the map function is constructed. By kernel representations, we just need to compute the value of dot product in F without having to carry out the map ϕ. The fascinating idea of using a kernel approach is that we can construct an optimal separating hyperplane in the feature space without considering this space in an explicit form. For improving computational efficiency and extend the classical linear algorithms, Baudat proposed a new approach called kernel function approximation [15], in which the kernel tricks is used to extract a relevant data set into the feature space and then the data sets are projected onto the subspace of the selected feature vectors where classical algorithms are applied for classification or regression. Here we adopt this method to realize the normal data domain description and to detect the machine failure.

2.1 Feature Vector Selection

Consider the input space X contains of M condition samples, and every sample represents a feature vector of machine condition at that time. Suppose we map the data into a high dimensional space $\boldsymbol{F}$ by a possibly nonlinear map

$$[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M] \xrightarrow{\varphi} [\varphi(\mathbf{x}_1), \varphi(\mathbf{x}_2), \cdots, \varphi(\mathbf{x}_M)] \tag{1}$$

These data can be represented in a subspace $\boldsymbol{F}_s$ of $\boldsymbol{F}$, and the dimension of this subspace is lower than M and equal to the ranks of kernel matrix **K**. The purpose of feature vector selection is to choose a basis of the subspace for expressing all the data as a linear combination of these selected vectors in the transformed space $\boldsymbol{F}$, so that classical linear methods can be used to detect the outliers. The feature vectors are among the samples which represent the machine condition.

Let L be the number of feature vectors need to be selected ($L \leqslant M$), the mapping of $\mathbf{x}_i$ simplified as φ_i ($i = 1, \cdots, M$), and the selected vectors noted as $\mathbf{x}_{S_j}$, its mapping as φ_{S_j} ($1 \leqslant j \leqslant L$), here 'S' is used in subscripts to declare the selected feature vectors.

According to the definition of basis vector, for a given set of selected vectors $\mathbf{S} = \{\mathbf{x}_{s_1}, \mathbf{x}_{s_2}, \cdots, \mathbf{x}_{s_L}\}$, after mapping into subspace $\boldsymbol{F}_s$, the mapping of any vector $\mathbf{x}_i$ can be denoted as a linear combination of $\mathbf{S}$

$$\hat{\varphi}_i = \sum_{j=1}^{L} \boldsymbol{a}_i^j \varphi_{s_j} \tag{2}$$

Where It can be formulated as a dot product

$$\hat{\varphi}_i = \varphi_S \cdot \boldsymbol{a}_i \tag{3}$$

where $\hat{\varphi}_i$ is the mapping of $\mathbf{x}_i$ in subspace $\boldsymbol{F}_s$, $\varphi_S = (\varphi_{s_1}, \cdots, \varphi_{s_L})$ is the matrix of selected vectors into F and $\boldsymbol{a}_i = [\boldsymbol{a}_i^1, \boldsymbol{a}_i^2, ..., \boldsymbol{a}_i^L]^{\mathrm{T}}$ is coefficient vector that weighted this matrix.

For a given set S, the key is to find coefficients $\boldsymbol{a}_i$ in order to minimize the difference between $\hat{\varphi}_i$ and φ_i. Consider solve this problem in space $\boldsymbol{F}$, the normalized Euclidean distance can be given by the following ratio

$$\delta_i = \frac{\|\varphi_i - \hat{\varphi}_i\|^2}{\|\varphi_i\|^2} \tag{4}$$

So, the minimum of equation (4) can be expressed with dot products and leads to

$$\min \delta_i = 1 - \frac{\mathbf{K}_{Si}{}^{\mathrm{T}} \mathbf{K}_{SS}^{-1} \mathbf{K}_{Si}}{\mathbf{K}_{ii}} \tag{5}$$

where $\mathbf{K}_{SS}$ is the matrix of dot products of the selected vectors, $\mathbf{K}_{SS} = (\mathbf{k}_{s_p s_q}) = \mathbf{K}(\varphi_{s_p} \cdot \varphi_{s_q})$ ($1 \leq p \leq L$, $1 \leq q \leq L$), $\mathbf{K}_{Si}$ is the vector of dot products between xi and selected vector set **S,** $\mathbf{K}_{Si} = (\mathbf{k}_{s_p i}) = \mathbf{K}(\varphi_{s_p} \cdot \varphi_i)$ ($1 \leq p \leq L$, $1 \leq i \leq M$), and $\boldsymbol{K}_{ii}$ is the matrix of dot products of all condition samples, $\mathbf{K}_{ii} = (\mathbf{k}_{ii}) = \mathbf{K}(\varphi_i \cdot \varphi_i)$ ($i = 1, \cdots, M$)

The problem now is to find the condition samples set $\mathbf{S}$ minimizing equation (5) over all samples $\mathbf{x}_i$

$$\min_S \Big(\sum_{x_i \in X} \Big(1 - \frac{\boldsymbol{K}_{Si}{}^T \boldsymbol{K}_{SS}^{-1} \boldsymbol{K}_{Si}}{\boldsymbol{K}_{ii}}\Big)\Big) \tag{6}$$

For solving this problem, the fitness function J_s is defined as

$$J_S = \frac{1}{M} \sum_{x_i \in X} \Big(\frac{\boldsymbol{K}_{Si}{}^T \boldsymbol{K}_{SS}^{-1} \boldsymbol{K}_{Si}}{\boldsymbol{K}_{ii}}\Big) \tag{7}$$

Obviously, eq.(6) is equivalent to maximize eq.(7), and note that for $\mathbf{x}_i \in \mathbf{S}$, (5) is 0. J_s can be used to descript the fitness of selected vectors, the larger the value of J_s, the selected vectors more representative, and the maximum of J_s is 1.

The selection algorithm is an iterative process, which is sequential forward selection: at each step looking for the sample that, when combined with the previously selected vectors, maximize fitness function J_s. The algorithm stops when $\mathbf{K}_{SS}$ is no longer invertible, which means that **S** is a basis for expressing data in subspace $\boldsymbol{F}_s$. Besides, the stop criterion can also be that the fitness or the number of selected vectors reaches a given value.

2.2 Kernel Function Approximation

In classical function approximation, for a given function f, the observation can be denoted as

$$y = f(x, a_0) + \xi \tag{8}$$

where x is the input data, y is the output data, a_0 is the parameter unknown and ξ is the noise.

The goal of function approximation is to compute the parameter a_0 by the observation value, so the output $\hat{y}_i$ can be estimated. Function approximation is mainly used in the problems of regression, which is closely related to classification. Especially for outlier detection, this classification task into 2 classes is a particular regression problem for which the output y is a binary value such that y=1 where x is belong to the class of normal condition, otherwise, y=0. Therefore, function approximation can be used to solve classification problems.

Once the feature vectors are obtained, we can transform all of the samples into the subspace $\boldsymbol{F}_s$, and the projection of a sample $\mathbf{x}_i$ can be denoted as $\mathbf{z}_i$ by dot products

$$\mathbf{z}_i = (\boldsymbol{\varphi}_i \cdot \boldsymbol{\varphi}_S) = \boldsymbol{\varphi}_S{}^{\mathrm{T}} \boldsymbol{\varphi}_i \tag{9}$$

A set of data $(\mathbf{x}_i, \mathbf{y}_i)$ transformed to $(\mathbf{z}_i, \mathbf{y}_i)$ after projection, and the problem of function approximation is to determine $\mathbf{z}_i$ is outlier or not according to the output $\mathbf{y}_i$.

The estimation of the function using linear regression into subspace $\boldsymbol{F}_s$, minimizes the MSE error of the learning set **H** and results in an optimal solution

$$\hat{\mathbf{y}}_i^{\mathrm{T}} = \mathbf{z}_i^{\mathrm{T}} \mathbf{A} + \boldsymbol{\beta}^{\mathrm{T}} \tag{10}$$

where $\mathbf{A} = (\mathbf{Z}_H{}^{\mathrm{T}} \mathbf{Z}_H)^{-1} \mathbf{Z}_H{}^{\mathrm{T}} \mathbf{Y}_H$, $\mathbf{Z}_H = (\mathbf{z}_i^{\mathrm{T}})_{i \in H}$, $\mathbf{Y}_H = (\mathbf{y}_i)_{i \in H}$, and $\boldsymbol{\beta}$ is a vector that can be included in the estimation of **A** by adding a constant component in each vector $\mathbf{z}_i$. For more detail, please refer to [15].

The process of kernel function approximation can be described in the Fig 1.

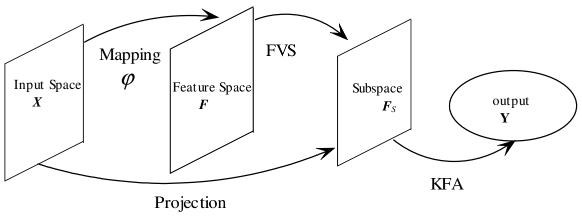


Fig. 1. The Procedure of Kernel Function Approximation

3 Experiments

In this section, we will apply the proposed approach as an unsupervised learning method to gear vibration signal analysis and compare the results with that of supervised one.

3.1 Gearbox Failure Experiments

The experiments were conducted on an auto-gearbox (Type/Manufacturer: 6J90T/Shaanxi Auto-gear Factory, China), and the transmission diagram is shown in Fig.2. The gearbox vibration signals were measured externally on the gearbox bearing case using an acceleration sensor B&K 4396 and amplified by a charge amplifier B&K 2626 to monitor the operating condition of the gearbox. The total testing time of the gear fatigue experiments was up to 182 hours. During the testing process, the gearbox's running condition underwent three different stages naturally. At first, the gearbox's operating condition was normal. Then a crack in one tooth root arose and propagated gradually. Lastly one tooth of the meshing gear was broken and the testing experiment was terminated. It was found that one tooth of the meshing gear with 24 teeth was broken when the gearbox was stripped. For more details of the experiment one may refer to reference [16]. There are distinct differences between the vibration signals of the gearbox with a broken tooth and those of the other two operating conditions, however there is only a slight difference between the vibration signals of the normal condition and the condition with a cracked tooth. Therefore, we will concentrate our efforts on the analysis of the vibration signals of latter two conditions.

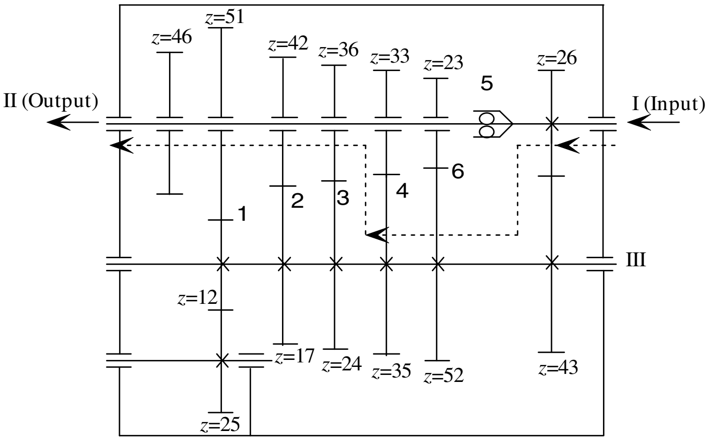


Fig. 2. The transmission diagram of 6J90T auto-gearbox is shown as above. The gear transmission realizes six forward shifts and one backward shift. The third forward gear shift (it is the most commonly used shift in practice, which is shown in dash line) was used in the experiments.

In the experiment, the input power is about 193.96kW, the input torque of shaft I is about 883Nm, the input speed is about 1270rpm, and the sampling rate was 12.5 kHz. So, the rotating frequencies of the three shafts are: 21.17Hz for shaft I; 8.53Hz for shaft II and 12.80Hz for the middle shaft (III). The meshing frequency of the first gear-pair is f_{z1}=550Hz and the meshing frequency of the second gear-pair is f_{z2}=307Hz.

Fig. 3 (a) and (b) show the time domain acceleration signal of normal condition gearbox and cracked tooth gearbox. It is difficult to describe the difference between the two signals in time domain.

Fig.4 (a) and (b) show the magnitude spectra of the discrete Fourier transform (DFT) of the vibration signals shown in Fig 3 (a) and (b) respectively. From Fig. 4 it is seen that the normal gearbox vibration consists of mainly the meshing vibration and their higher order harmonics. Moreover, the dominant components are the first meshing frequency (550Hz) and its higher order harmonics (1100Hz and 1650Hz). The meshing vibration at the second meshing frequency (307Hz) and the corresponding order frequencies are relatively weaker than that of the first meshing frequency. When there is a tooth crack in one of the gears, the vibration magnitude of the second frequency has a significant increase (Fig. 4(b)), as the meshing-stiffness decreased. Besides, as shown in Fig.4(b), it can be seen that a new component at frequency 857Hz (307Hz+550Hz) appears, which indicates the existence of the nonlinearity caused by the coupling of the two meshing frequencies.

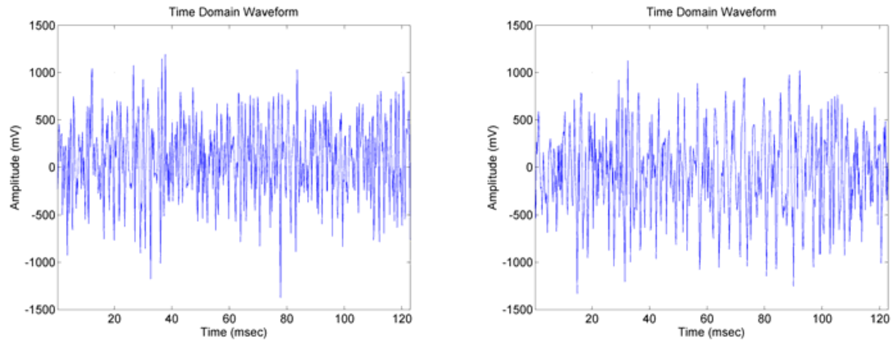


Fig. 3. Time domain vibration signals: (a) Normal gearbox (b) Cracked tooth gearbox

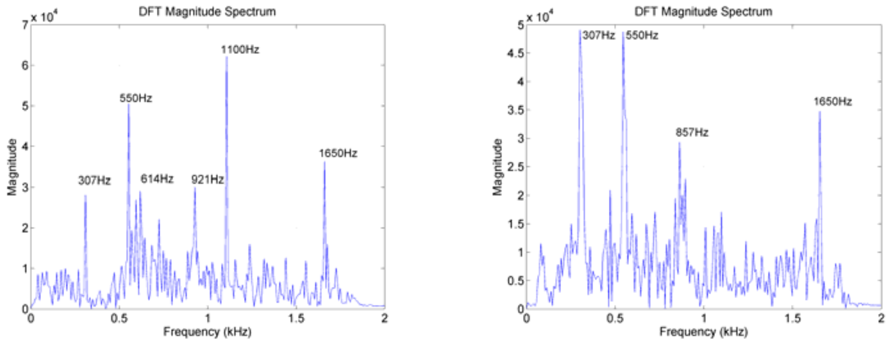


Fig. 4. DFT magnitude spectra: (a) Normal gearbox (b) Cracked tooth gearbox

However, in addition to the two pairs of meshing gears, there are also many other gears that not in meshing condition and supporting rolling bearings, they also contribute to the gearbox vibration. These additional components and measuring noise make the signal-to-noise ratio of vibration signals very low. Therefore, it is not adequate to determine that there is a tooth crack failure within the gearbox.

3.2 Gear Crack Detection Using KFA

It is known that the time domain of the vibration signals provides a useful feature set for gear faults diagnosis. Many feature parameters have been defined in the pattern recognition field. Here only 11 time domain features, including *maximum value*, *minimum value*, *standard deviation*, *absolute mean value*, *crest factor*, *impulse factor*, *clearance factor*, *root mean square value*, *kurtosis*, *skewness* and *variance*, were used as raw feature sets for further analysis. These parameters are non-dimensional and are used to construct "input space" in this work.

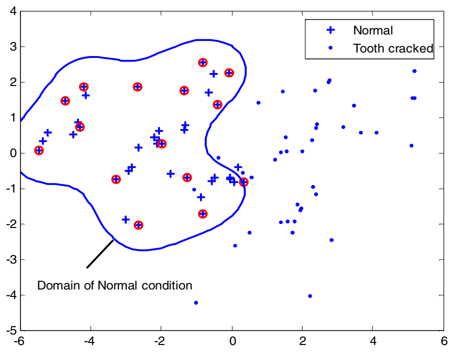


Fig. 5. Gear tooth crack detection using unsupervised KFA learning method with a Gaussian kernel function $K(\mathbf{x},\mathbf{y}) = \exp\left(-\|\mathbf{x}-\mathbf{y}\|^2 / 2\right)$ and $\sigma = 1$

There were 80 raw vibration signals, each having 1024 samples, measured under normal and tooth cracked conditions of the gearbox (40 per condition) for the investigation. Firstly, 11 features mentioned above of each raw signal were computed and normalized; in total 80 11-D raw feature data were obtained. In views of visualization the result, all the feature data first were processed with principal component analysis to reduce the dimensions to 2-D. Then selecting feature vectors of 40 feature samples of normal condition, the criterion of feature vector selection was that the number of vectors reaches 20 or the fitness value reaches 0.95. When 15 feature vectors have been selected, the fitness value reached 0.950256, which means that the subspace for gear crack detection can be constructed by these 15 feature vectors. Projecting all the feature samples onto the subspace, using KFA to descript the normal condition domain as an unsupervised learning method, the samples which located out of the boundary can be thought as outliers, and the detection result is shown in Fig.5, in which we use the symbol '+' to represent data of normal condition and '•' tooth cracked data respectively. The data labeled with 'o' are the selected feature vectors, and the correct detection rate of this method reaches 95%.

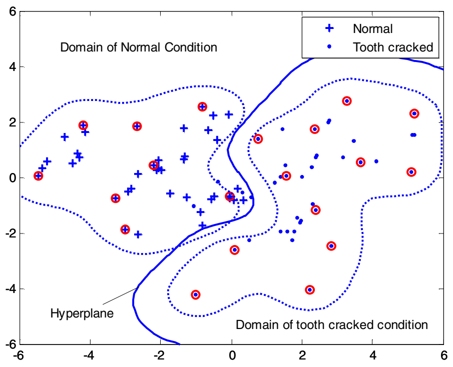


Fig. 6. Gear condition classification using supervised KFA learning method with a Gaussian kernel function $K(\mathbf{x},\mathbf{y}) = \exp\left(-\|\mathbf{x}-\mathbf{y}\|^2/2\right)$ and $\sigma = 1$

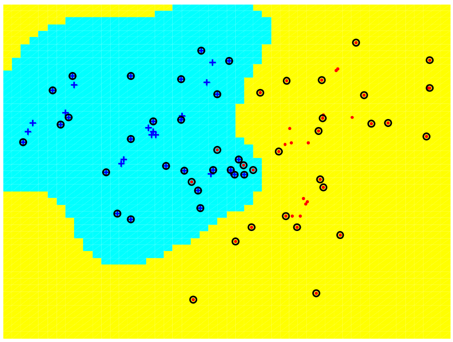


Fig. 7. Gear condition classification using SVM learning method with a Gaussian kernel function $K(\mathbf{x},\mathbf{y}) = \exp\left(-\|\mathbf{x}-\mathbf{y}\|^2/2\right)$ and $\sigma = 1$

The gear conditions classification result using supervised KFA learning method is shown in Fig.6. Here we need to select feature vectors of two classes, the criterion is that the number of feature vectors reaches 20 or the fitness value reaches 0.95. When 20 feature vectors have been selected, in which there are 8 vectors of normal condition and 12 vectors of tooth cracked condition, the fitness value reached 0.912566, and the correct classification rate is 95% too. It can be seen that the result of unsupervised KFA performed as well as that of supervised one.

Fig.7 shows the classification result of gear conditions with SVM method, here OSU_SVM toolbox is used to get the result and the same kernel function used in KFA is adopted [17]. The number of support vectors is 51, and the correct rate is 95%. Obviously, the number of feature vectors is less than that of support vectors, which means that KFA can give slightly better results than the SVM does considering the computation load and speed.

It should be pointed out that parameter σ of Gaussian kernel influences the domain description, especial for the domain boundary. Different value of σ will lead to different number of feature vectors and different boundary of normal data domain, and in this work we determine this value by testing and experience, which is related to the selection of kernel functions.

4 Conclusions

Machine failure detection involves modeling the normal behavior of a system hence enabling detection of any divergence from normality or highlighting abnormal features of machine damage. In this paper an approach to outlier detection based on KFA is presented, in which feature vectors are selected to construct the feature subspace and determine the normal data domain, and then all the samples are projected onto this subspace, samples outside of the normal data domain are outliers. Experiment results on industrial gearbox vibration signals indicate that the proposed method is sensitive to select feature vectors from raw feature sets and is capable of detecting early machine failure such as gear crack. It has great potential for machine fault diagnosis in practice.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (Grant No. 50475095), Natural Science Foundation of Guangdong Province (Grant No. 05300143 and 04020082) and Open Research Foundation of State Key Lab of Vibration, Shock and Noise (Grant No.VSN-2004-03).

References

1. P Chen, T Toyota and Z He: Automated Function Generation of Symptom Parameters and Application to Fault Diagnosis of Machinery Under Variable Operating Conditions. IEEE Transaction on Systems, Man, and Cybernetics, Vol.31, (2001)775-781
2. Ypma A: Learning Methods for Machine Vibration Analysis and Health Monitoring, Ph.D. Dissertation. Delft University of Technology (2001)
3. W Li,T Shi, G Liao and S Yang: Feature extraction and Classification of gear faults using principal component analysis. Journal of Quality in Maintenance Engineering, Vol.9, (2003) 132–143
4. J Addison, S Wermter, K McGarry and J Macintyre: Methods for integrating memory into neural networks in condition monitoring. Proc. International Conference on Artificial Intelligence and Soft Computing, Alberta, Canada, (2002) 380-384
5. Richard O.Duda, Peter E.Hart, and David G. Stork: Pattern Classification, 2nd edn, John Wiley & Sons, New York (2001).
6. K. R. Muller, S. Mika, G. Rätsch, et al: An Introduction to Kernel-Based Learning Algorithms. IEEE Trans. on Neural Networks, Vol. 12, (2001) 181-201
7. D Tax and R Duin: Support vector domain description. Pattern Recognition Letters, 20(11-13), (1999) 1191–1199.
8. Schölkopf B, Smola AJ, Müller KR: Nonlinear Component Analysis as A Kernel Eigenvalue Problem. Neural Computation, Vol.10, (1998)1299-1319

9. G Baudat and F Anouar: Generalized Discriminant Analysis Using a Kernel Approach. Neural Computation, Vol.12, (2000)2385-2404
10. C Campbell and K.P. Bennett: A linear programming approach to novelty detection. Advances in NIPS, Vol. 14, MIT Press, Cambridge, MA, (2001)
11. D Tax, A Ypma, R. Duin: Support vector data description applied to machine vibration analysis. In: M Boasson, J Kaandorp, J Tonino, M Vosselman(eds.), Proc. 5th Annual Conference of the Advanced School for Computing and Imaging, Heijen, (1999)398-405
12. D Tax, A Ypma, R. Duin: Pump Failure Detection Using Support Vector Data Descriptions. Proceedings of Third Symposium on Intelligent Data Analysis IDA'99, Lecture Notes in Computer Science, Vol.1642. Springer-Verlag, Amsterdam, (1999) 415-426
13. Li WH, Shi TL and Yang SZ: Mechanical Fault Classification Using Nonlinear Discriminant Analysis. Journal of Vibration Engineering, Vol.18, (2005) 133-138
14. Li WH, Liao GL and Shi TL: Kernel Principal Component Analysis and its Application in Gear Fault Diagnosis. Chinese Journal of Mechanical Engineering, Vol.39, (2003) 65-70
15. Baudat G and Anouar F: Kernel-based Methods and Function Approximation. Proceedings of International Joint Conference on Neural Networks, Washington D.C. (2001) 1244-1249.
16. G Zhang: Gear Vibration Signature Analysis and Fault Diagnosis, Master thesis. Xi'an Jiaotong University. Xi'an (1993)
17. J Ma, Y Zhao, S Ahalt: OSU SVM Classifier Matlab Toolbox (version 3.00), Ohio State University, Columbus, USA, (2002). http://www.ece.osu.edu/~maj/osu svm/

The Design of Data-Link Equipment Redundant Strategy

Li Qian-Mu[1,2], Xu Man-Wu[1], Zhang Hong[2], and Liu Feng-Yu[2]

[1] Nanjing University, State Key Laboratory for Novel Software Technology,
210093 Nanjing, China
{liqianmu, liu.fengyu}@126.com
[2] Nanjing University of Science and Technology, Computer Science Department,
210094 Nanjing, China
{liqianmu, njust}@126.com

Abstract. A framework model proposed in this paper is a data-link Equipment Redundant Strategy based on reliability theory. The strategy combined with the normal maintenance could greatly improve the performance of the network system. The static-checking and policy of authentication mechanism ensure the running network without any error. The redundant equipments are independent but are capable of communication with each other when they work their actions. The model is independent of specific application environment, thus providing a general-purpose framework for fault diagnosis. An example is given to express the calculating method.

1 Introduction

One of the important characteristics of informationized weapons is that the platforms are horizontally implemented that they can be merged into information network system to accomplish information sharing. As a result, the efficiency of the platforms can be significantly upgraded. Traditionally, the land-based operation platform takes tank, war chariot, cannon and guided missile as representative and the sea-based operation platform takes naval ships and submarines as representative, and the sky-based operation terrace takes plane and helicopter as representative. But all of the above platforms could become informationized weapons with high-tech only when they are implemented with superiority in modern information technology as well as superiority in traditional firepower.

Therefore, more and more attention is being paid to the task of making a “data chain” that mixes each platform, information resources optimization, efficient deployment and use of armed forces, and the task is now being put into practical use. The “data chain” is becoming the “binder” and “multiplier” of armed forces in future, making every commanding and controlling system and computer systems on operation platforms consist into a tactic data transmission/exchange and information processing network, so that all the concerned data and complete battle information can be provided to the commanders and warriors.

Nowadays, the researchers have focused on the performance of data link networks, so only little progress has been made on user’s safeguard. But the performance of the user’s safeguard strategy is a direct factor into the reliability and cost of data link network, including the whole procedure from fault discovering to trouble shooting. The reliability of data link network is based not only on the design of system structure and redundancy but also the strategy adopted. Research [1] shows that the fault tolerable network with redundant strategy is a better choice on performance, cost and reliability.

Base on this, the Datalink Equipment Redundant Strategy (**DERS**) strategy is put forwarded in this paper, analyzing the state transmission mechanism of tactics, building a mathematical model based on the reliability theory. And the theoretical proof of it is given while the availability of this strategy is proved by the simulation.

2 Datalink Equipment Redundant Strategy

2.1 Basic Principle

In a redundant datalink network, θ represents the time cost on a normal safeguard, λ represents MTTR, μ represents MTBF, c represents the equipment quantity in this network, d represents equipment quantity that are under normal synchronously safeguard. Obviously $0 \le d \le c$. Let $\xi = c\mu$, $\tau = (c-d)\mu$. Assuming $1 \le d \le c$, when a trouble is shot, if the quantity of the equipments with trouble is less than $c-d$, then d equipments start synchronous safeguard, other $c-d$ equipments are not able to enter the safeguard state even if they are idling. After d equipments accomplish the safeguard procedure, if the quantity of the equipments with trouble is still less than $c-d$, d equipments start another safeguard. Otherwise, stop the safeguard procedure. When the safeguard procedure terminates synchronously, j equipments are under trouble, $c-d < j < c$, $j-c+d$ equipments (among the redundant equipments returning from safeguard) are working, and $c-j$ equipments are idling.

2.2 Analysis of DERS Model

The state shifts of **DERS** strategy model is shown in Fig. 1. After a trouble is shot at state（ $c-d$ +1, 1）, the state shifts to state（ $c-d$, 0）, and d redundant equipments start the safeguard procedure now. Let $L_v(t)$ represent the quantity of the equipments with trouble at time t, and

$$J(t) = \begin{cases} 0 & \text{d equipments are under safeguard procedure at time t} \\ 1 & \text{no equipment is under safeguard at time t} \end{cases},$$

then（ $L_v(t)$ ， $J(t)$ ）is a tow-dimensional Malcov procedure, and a state space exists:

$$\Omega = \{(k,0) \mid 0 \le k \le c-d\} \bigcup \{(k,j) \mid k > c-d, j = 0 or 1\} . \quad (1)$$

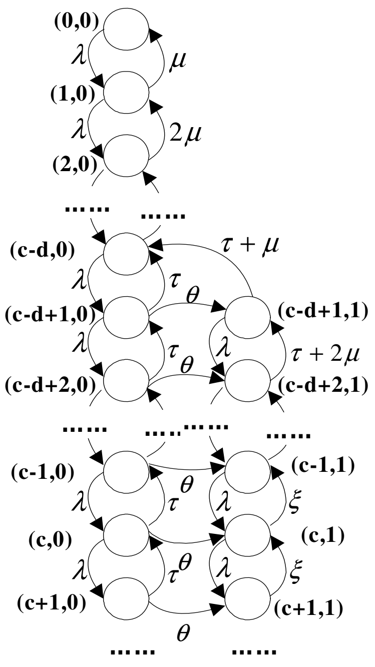


Fig. 1. State Shifts of **DERS** Strategy Model

Generated elements in the procedure are as below:

$$Q=\begin{bmatrix} A_0 & C_0 & & & & & & \\ B_1 & A_1 & C_1 & & & & & \\ & \ddots & \ddots & \ddots & & & & \\ & & B_{c-1} & A_{c-1} & C_{c-1} & & & \\ & & & B & A & C & & \\ & & & & B & A & C & \\ & & & & & \ddots & \ddots & \ddots \end{bmatrix} \tag{2}$$

In it,

$$A_k=\begin{cases} -(\lambda+k\mu), & 0\le k\le c-d \\ \begin{pmatrix} -(\lambda+(c-d)\mu+\theta) & \theta \\ 0 & -(\lambda+k\mu) \end{pmatrix}, & c-d<k\le c \end{cases} . \tag{3}$$

$$B_k = \begin{cases} k\mu, & 1 \le k \le c-d \\ \begin{pmatrix} (c-d)\mu \\ (c-d+1)\mu \end{pmatrix}, & k = c-d+1 \\ \begin{pmatrix} c-d & 0 \\ 0 & k\mu \end{pmatrix}, & c-d+1 < k \le c \end{cases} . \tag{4}$$

$$C_k = \begin{cases} \lambda, & 0 \le k < c-d \\ (\lambda, 0), & k = c-d \\ \lambda I, & c-d < k \le c \end{cases} . \tag{5}$$

$$A = \begin{pmatrix} -[\lambda + (c-d)\mu + \theta] & \theta \\ 0 & -(\lambda + c\mu) \end{pmatrix}. \tag{6}$$

$$B = \begin{pmatrix} (c-d)\mu & 0 \\ 0 & c\mu \end{pmatrix}. \tag{7}$$

$$C = \lambda I . \tag{8}$$

Assuming $\rho = \lambda/(c\mu)$, when ρ <1,（ L_v, J ） represents the stable limit of the procedure（ $L_v(t)$， $J(t)$ ）.

According to state balance equation[1] $\pi Q = 0$, in it $\pi = (\pi_0, \pi_1, \cdots, \pi_i, \cdots)$, $\pi_k = (\pi_{k1}, \cdots, \pi_{kj})$, $\pi_{ki} = \lim_{t\to\infty} P\{L_v(t) = k, J(t) = i\}, (k,i) \in \Omega$ 。We get

$$\pi_{k-1} C + \pi_k A + \pi_{k+1} B = 0 . \tag{9}$$

The distribution of（ L_v， J ）could be represented as

$$\begin{cases} \pi_{j0} = K \dfrac{1}{j!} (\dfrac{\lambda}{\mu})^j & 0 \le j \le c-d \\ \pi_j = K(\beta_{j0}, \beta_{j1}) & c-d < j \le c \\ \pi_{j1} = K\beta_{c1}\rho^{j-c} + K\beta_{c0} \dfrac{\theta r}{c\mu(1-r)} \sum_{i=0}^{j-c-1} r^i \rho^{j-c-1-i} & j > c \end{cases} . \tag{10}$$

In it, $\beta_{j0} = \dfrac{1}{(c-d)!} (\dfrac{\lambda}{\mu})^{c-d} r^{j-(c-d)}$, $c-d+1 \le j \le c$;

$$\beta_{j1} = \frac{1}{j!} (\frac{\lambda}{\mu})^j \frac{\theta r}{\lambda(1-r)} [1 + \frac{1}{(c-d)!} \sum_{i=1}^{j-(c-d)-1} (c-d+1)! (\frac{r\mu}{\lambda})]^i , \quad c-d+1 \le j \le c$$

$$K = \{ \sum_{j=0}^{c-d} \frac{1}{j!} (\frac{\lambda}{\mu})^j + \sum_{j=c-d+1}^{c-1} (\beta_{j0} + \beta_{j1}) + (\beta_{c0}, \beta_{c1})(I-R)^{-1} e \}^{-1}$$

Assuming $R^k = \pi^k (\pi_0)^{-1}$, we get (11) from (9)

$$R^2 B + RA + C = 0 . \tag{11}$$

Proposition 1: Let r 和 r^* represent the two solutions of the quadratic equation $(c-d)\mu z^2 - [\lambda + (c-d)\mu + \theta]z + \lambda = 0$, 0< r <1, r^* >1. When ρ <1, matrix equation (11) has its minimal non- negative solution.

$$R = \begin{bmatrix} r & \dfrac{\theta r}{c\mu(1-r)} \\ 0 & \rho \end{bmatrix} . \tag{12}$$

Proof: As A, B, C are all above-triangle matrices, assuming $R = \begin{bmatrix} r_{11} & r_{12} \\ 0 & r_{22} \end{bmatrix}$. With the substitution of R into the matrix equation $R^2 B + RA + C = 0$, we get

$$\begin{cases} (c-d)\mu r_{11}^2 - [\lambda + (c-d)\mu + \theta] r_{11} + \lambda = 0 \\ c\mu r_{22}^2 - (\lambda + c\mu) r_{22} + \lambda = 0 \\ c\mu r_{12}(r_{11} + r_{22}) + \theta r_{11} - (\lambda + c\mu) r_{12} = 0 \end{cases}$$

As r is $(c-d)\mu z^2 - [\lambda + (c-d)\mu + \theta]z + \lambda = 0$ the minimal non-negative solution in (0, 1), we could assume $r_{11} = r$. According to the second formula, assuming $r_{22} = \lambda/(c\mu)$. Substituting the above results into the third formula, we can get $r_{12} = \dfrac{\theta r}{c\mu(1-r)}$. Obviously, $\max(r,\rho) < 1$ exists when and only when ρ <1。The proposition is proved.。

Proposition 2: When the queue consisted of equipments with trouble exists in the network, the probability of not able to work normally is $\dfrac{\sigma}{1-\rho}K$, in it, $\sigma = \beta_{c1} + \dfrac{\theta r}{c\mu(1-r)^2}\beta_{c0}$.

Proof: The probability of the existence of the queue consisted of equipments with trouble is

$$P\{L_v \geq c, J = 1\} = \sum_{k=c}^{\infty} \pi_{k1} = K\beta_{c1} \sum_{k=c}^{\infty} \rho^{k-c} + K \frac{\theta r}{c\mu(1-r)} \beta_{c0} \sum_{j=c+1}^{\infty} \sum_{k=0}^{j-c-1} r^k \rho^{j-c-1-k}$$

$$= \frac{K}{1-\rho}\left(\beta_{c1} + \frac{\theta r}{c\mu(1-r)^2}\beta_{c0}\right) = \frac{\sigma}{1-\rho} K .$$

The proposition is proved.

3 Simulation of the DERS Model

In order to prove the availability of this system and its algorithm, we use the NS2 and its ETR data sampling lib, provided by Berkley, as the experiment environment to take this simulation. In the environment of this simulation experiment, the interval of radio frequency is 25 KHz; data transmission rate is 5 Kbps; there are 5 redundant equipments in each network segment, in which only two equipments are allowed to be in the state of self-safeguard. And we assume that the normal safeguard takes 0.2 minutes, MTBF is 30 seconds and MTTR is 120 seconds, i.e. c=5, d=2, θ=0.2, λ=2, μ=0.5. We get

$$A=\begin{pmatrix} -[\lambda+(c-d)\mu+\theta] & \theta \\ 0 & -(\lambda+c\mu) \end{pmatrix}=\begin{pmatrix} -3.7 & 0.2 \\ 0 & -4.5 \end{pmatrix}$$

$$B=\begin{pmatrix} (c-d)\mu & 0 \\ 0 & c\mu \end{pmatrix}=\begin{pmatrix} 1.5 & 0 \\ 0 & 2.5 \end{pmatrix}; \quad C=\lambda I=\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$$

Assuming r is the root of the equation $1.5z^2-3.7z+2=0$ between (0, 1), We get r=0.7667, $\rho=\dfrac{\lambda}{c\mu}=0.8<1$.

According to (12), we get $R=\begin{pmatrix} 0.7667 & 0.2628 \\ 0 & 0.8 \end{pmatrix}$.

According to (10), we can get the average length of the queue consisted of equipments with troubling in this segment is $E(L_q)=8.11$. Then we can get the MTTR is $E(W)=\dfrac{1}{\lambda}E(L_q)=4.06$ minutes by using formula Little[1].

More researches on the relationship among each parameter in the redundant configuration can be done by using this model. Let's take a common problem on redundant requirement as an example: try to find out the number of redundant equipments needed to reduce the MTTR to less than 2 minutes.

Proof: Calculate the values of $E(L_q)$ and $E(W)$ as below by fixing the values of λ, μ, θ and changing the value of c. The relationship between E(W) and c is shown in fig. 2.

Table 1. Average Idling Time on Different Redundant Equipment Number

c	5	6	7	8
$E(W)$	4.0560	2.6620	1.9240	1.4200
$E(L_q)$	8.1120	5.3240	3.8480	2.8400

So we can get the result that we should add two more equipments in order to make $\overline{W} = E(W) < 2$.

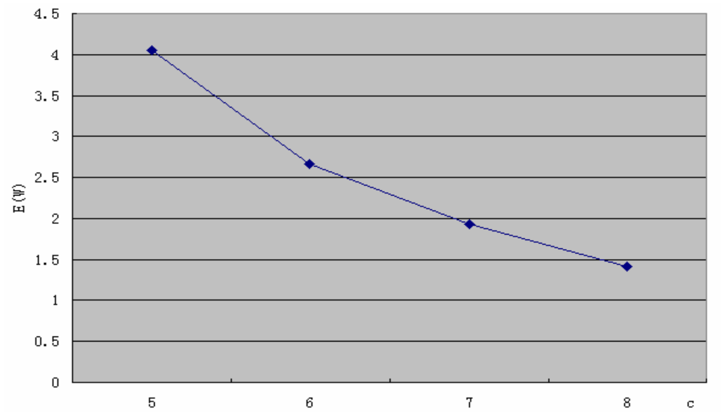


Fig. 2. Relationship between the Number of Redundant Equipments and Average Idling Time

4 Conclusion

We have proposed a new redundant strategy mathematic model DERS to solve the problem on Datalink Equipment Redundant Strategy in this paper. Based on this model, we can solve the problem of redundant management efficiently. Also, this strategy can minimize the cost of equipments' safeguard procedure and normal running on the premise of the normal running of the network. The simulation experiment has proved the availability of DERS strategy.

References

1. Othmar K. Network Troubleshooting. Berlin: Agilrnt Technologies, 2002.
2. Jun Z, Hong L, etc. A New Analysis Method on the Hidden Terminals in Self—Organized TDMA VHF Datalink. Electric Journal. 2001, 12 (29) :128-132
3. QIAN Fang, JIA Yan, HUANG Yan, etc. A Dynamic And Fault-Tolerable Algorithm on Redundant Service Performance Improvement. Journal of Software. 2001, 12(6):523-528
4. Jacobson V. Congestion Avoidance and Control. IEEE/ACM Transaction Networking, 2000, 6(3):314-329
5. Caserri C, Meo M. A new approach to model the stationary behavior of TCP connections. In: Proc IEEE INFOCOM2000, Tel Aviv, Israel, CA: IEEE Computer Society, 2000
6. Veres A, Boda M. The chaotic nature of TCP congestion control. In: Proc INFOCOM2000, Tel Aviv, Israel, CA:IEEE Computer Society 2000
7. Floyd S, Fall K. Promoting the use of End-to-End congestion control in the Internet. IEEE/ACM Transaction Networking, 2002, 7(4): 458-472
8. Harris B, Hunt R. TCP/IP security threats and attack methods. Computer Communications, 2004,22(10): 885-897

9. Christoph L, Schuba, Ivan V, Krsul. Analysis of denial of service attack on TCP. Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy 2000
10. Hamann T, Walrand J. A new fair window algorithm for ECN capable TCP(New-ECN). In: Proc IEEE INFOCOM2000, Tel Aviv, Israel, CA: IEEE Computer Society, 2000
11. ChenDan, HeHua-Can, WangHui. A new t-norm and its application in fuzzy control. In: Proc WCICIA'2000, Hefei, China, 2000. 1284-1289
12. Yager R. OWA neurons: A new class of fuzzy neurons. In:Proc IEEE-FUZZ, 1992. 2316-2340
13. Glorennec Pierre-Yves. Neuro-fuzzy logic. In: ProcIEEE-FUZZ, 2003. 512-518
14. Witold Pedrycz. Fuzzy neural networks and neuro computations. Fuzzy Sets and Systems. 1992, 56(1):1-28
15. Witold Pedrycz. Logic-based neurons: Extensions, uncertainty representation and development of fuzzy controllers. Fuzzy Sets and Systems, 2004, 66(1): 251-266
16. Bailey S A, Chen Ye-Hwa. A two layerd network using the OR/AND neuron. In: Proc IEEE-FUZZ, 1999. 1566-1571
17. Averkin A N. Decision making based on multivalued logic and fuzzy logic, architectures for semiotic modeling and situation analysis in large complex systems . In: Proc ISIC Workshop, Monterey, 2001. 871-875
18. He Hua-Can et al. Generalized logic in experience thinking. Science in China(E), 1996, 39(3): 225-234
19. Buckley J J, Siler W. A new t-norm. Fuzzy Sets and Systems,2004,16(1):283-290

Minimizing Makespan on Identical Parallel Machines Using Neural Networks

Derya Eren Akyol and G. Mirac Bayhan

Department of Industrial Engineering, Dokuz Eylul University,
35100 Bornova-Izmir, Turkey
{derya.eren, mirac.bayhan}@deu.edu.tr

Abstract. This paper deals with the problem of minimizing the maximum completion time (makespan) of jobs on identical parallel machines. A Hopfield type dynamical neural network is proposed for solving the problem which is known to be NP-hard even for the case of two machines. A penalty function approach is employed to construct the energy function of the network and time evolving penalty coefficients are proposed to be used during simulation experiments to overcome the tradeoff problem. The results of proposed approach tested on a scheduling problem across 3 different datasets for 5 different initial conditions show that the proposed network converges to feasible solutions for all initialization schemes and outperforms the LPT (longest processing time) rule.

1 Introduction

This paper considers the problem of scheduling independent jobs on identical parallel machines with the objective of minimizing makespan. The problem can be described as follows: we are given n jobs each of which is to be executed on one of the identical parallel m machines during a fixed processing time without preemption.

So far, much research work has been performed on identical parallel machine scheduling problems and a detailed survey of applications is provided by [4,15]. Different objective functions can be defined for the problem but makespan minimization has been one of the most widely studied objectives in the literature.

In the standard three-field problem classification notation of [8], the problem is denoted in the scheduling literature as $P||Cmax$ where P represents the identical parallel machines, C_{max} denotes the makespan. It is assumed that the processing times are positive and that $1<m<n$.

Even though traditional techniques such as complete enumeration, dynamic programming, integer programming, and branch and bound were used to find the optimal solutions for small and medium sized problems, they do not provide efficient solutions for the problems with large size and the problem is known to be NP-hard.

Since no efficient polynomial algorithm exists to find the optimal solution, heuristic methods were developed to obtain near optimal solutions. Although, efficient heuristics can not guarantee optimal solutions, they provide approximate solutions as good as the optimal solutions. The LPT rule of Graham [7], one of the most popular heuristics, has been shown to perform well for the makespan criterion.

This rule arranges jobs in descending order of processing times, such that $p_1 \geq p_2 \geq \ldots \geq p_n$, and then successively assigns jobs to the least loaded machine. The MULTIFIT algorithm, a classical constructive heuristic developed by [5], determines the smallest machine capacity to find a feasible solution using the LPT scheme. This is achieved by solving heuristically a series of bin packing problems. Although MULTIFIT is not guaranteed to perform better than LPT, it has been shown that it has a worst case bound better than LPT.

Besides a large number of approaches such as mathematical programming, dispatching rules, expert systems, and neighborhood search used to model and solve scheduling problems, over the last decades, there has been an explosion of interest in using Artificial Neural Networks (ANNs) for the solution of various scheduling problems. After the success of the use of Hopfield and Tank's network [12] in solving the Traveling Salesman Problem, a variety of optimization problems including scheduling problems are solved using Hopfield type networks. However, the Hopfield model gets easily trapped in local minimum states and stochastic approaches are more successfully able to improve solution quality since they attempt to embed the principles of simulated annealing into the Hopfield network to escape from local minima. Replacing sigmoidal activation function with a stochastic decision type activation function, adding noise to the weights of the network or to the biases of the network are some of the main methods used to embed stochasticity into the Hopfield network [17]. Boltzmann machine [10], Gaussian machine [2] and mean field annealing [16] approaches were obtained by embedding stochastic properties into the Hopfield network.

The aim of this research is to explore the use of ANNs in solving the identical parallel machine scheduling problem for minimizing the makespan. To the best of our knowledge, this study will be the first attempt to solve the considered problem using neural networks. A dynamical gradient network is developed to attack the problem and a penalty function approach is used to construct the energy function. The idea of overcoming the tradeoff problem encountered in using the penalty function approach motivated us to use time varying penalty coefficients during simulation experiments.

The results of proposed approach tested on a scheduling problem for 5 initialization schemes show that the proposed network outperforms the LPT rule in all the datasets with a 100 % convergence rate. The rest of the paper is organized follows. In Section 2, a mixed integer programming (MIP) formulation for the identical parallel machine scheduling problem is given and the proposed dynamical network is explained in Section 3. The computational results are provided in Section 4 and the conclusions with future research directions are given in Section 5.

2 Formulation of the Problem

The problem under study can be formulated using a mixed integer programming formulation. The objective is to find an appropriate allocation of jobs to machines that would optimize the performance criterion, makespan, *Cmax*.

The notations for the problem are as given below:

n: number of jobs
m: number of machines
J_i : job i, $i \in N=\{1,...,n\}$
M_j : machine j, $j \in M=\{1,...,m\}$
p_i: processing time of J_i
C_i: completion time of J_i
C_{max}: makespan, the maximum completion time of all jobs
$C_{max} = \max\{C_1, C_2, ...,C_n\}$

$$x_{ij} = \begin{Bmatrix} 1 & \textit{if job i is assigned to machine j} \\ 0 & \textit{otherwise} \end{Bmatrix}$$

A MIP formulation of the minimum makespan problem can be defined as follows:

min C_{max}
subject to

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad 1 \le i \le n \tag{1}$$

$$C\mathbf{max} - \sum_{i=1}^{n} p_i x_{ij} \ge 0 \qquad 1 \le j \le m \tag{2}$$

The first constraint given in (1) ensures that each job is assigned to exacly one of the machines. The second constraint given in (2) ensures that the makespan is at least the completion time of each machine.

3 Design of the Proposed Dynamical Gradient Network

This section describes how dynamical gradient networks can be used to solve the considered problem given in the previous section. The proposed approach is an extension of the original formulation given in [11,12].

3.1 The Network Architecture

In the proposed dynamical network, we will have two types of neurons: a continuous type neuron to represent real valued variable Cmax, and discrete types of neurons to represent binary valued variables $X_{11},\ldots, X_{1m;}$ $X_{21},\ldots, X_{2m}$; $X_{n1},\ldots,X_{nm}$. The input to the neuron for job i and resource j is represented by UX_{ij} and UCmax represents the input to the neuron representing Cmax. The dynamics of the gradient net will be defined in terms of these input variables.

While the output of the neuron for job i and resource j is denoted by VX_{ij}, the output of the neuron representing Cmax is denoted by VCmax. We use a piecewise linear type activation function for neuron Cmax and the activation function for discrete neurons will take the usual sigmoidal form with slopes λ_X. In other words, we use a log-sigmoid function to convert discrete neurons to continuous ones.

3.2 The Energy Function

The energy function for this network is constructed using a penalty function approach. That is the energy function E consists of the objective function *Cmax* plus a penalty function to enforce the constraints. For the problem considered, the penalty function *P(X, Cmax)* will include three penalty terms: P1, P2 and P3.

In other words, the first penalty term P1 tries to ensure that each job is allocated to one only one machine and will yield a positive penalty value if any of the following constraints are violated.

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad 1 \le i \le n \tag{3}$$

Therefore, P1 $= \sum_{i=1}^{n} (\sum_{j=1}^{m} X_{ij} - 1)^2$. P2 adds a positive penalty if the solution does not satisfy any of the inequality constraints given in (4).

$$C\mathbf{max} - \sum_{i=1}^{n} p_i x_{ij} \ge 0 \qquad 1 \le j \le m \cdot \tag{4}$$

We can write P2 as follows:

P2$= \sum_{j=1}^{m} v(\sum_{i=1}^{n} p_i X_{ij} - C_{\max})$ where *v* represents the penalty function.

Here, $v(\varepsilon) = \varepsilon^2 \ for\, all \ \varepsilon > 0\, and\ v(\varepsilon) = 0\ for\ all\ \varepsilon \le 0$ [20].

The binary constraints X$_{ij}$ ϵ{0,1}will be captured by P3 which adds a positive penalty if they are violated.

P3 $= \sum_{i=1}^{n}\sum_{j=1}^{m} X_{ij}(1 - X_{ij})$ and the total penalty function P (X, Cmax) with all constraints can be written as follows.

$$\min\ B\sum_{i=1}^{n}(\sum_{j=1}^{m} X_{ij} - 1)^2 + C\sum_{j=1}^{m} v(\sum_{i=1}^{n} p_i X_{ij} - C_{\max}) + D\sum_{i=1}^{n}\sum_{j=1}^{m} X_{ij}(1 - X_{ij})$$

The resulting energy function can thus be written as:

$$\min\ AC_{\max} + B\sum_{i=1}^{n}(\sum_{j=1}^{m} X_{ij} - 1)^2 + C\sum_{j=1}^{m} v(\sum_{i=1}^{n} p_i X_{ij} - C_{\max})$$
$$+ D\sum_{i=1}^{n}\sum_{j=1}^{m} X_{ij}(1 - X_{ij})$$

where A, B, C and D are positive penalty coefficients.

3.3 The Dynamics

The dynamics for the proposed network are obtained by gradient descent on the energy function. The motion equations can be written as follows.

$$\begin{aligned}\frac{dUC_{\max}}{dt} &= -\frac{\partial E}{\partial C_{\max}} \\ &= -A-(-1)C\sum_{j=1}^{m} \upsilon'[\sum_{i=1}^{n} P_i X_{ij} - C_{\max}]\end{aligned} \quad (5)$$

$$\begin{aligned}\frac{dUX_{ij}}{dt} &= -\frac{\partial E}{\partial X_{ij}} \\ &= -2B[\sum_{k=1}^{m} X_{ik} - 1] - C(Pi)\upsilon'[\sum_{l=1}^{n} p_l X_{lj} - C_{\max}] - D(1-2X_{ij})\end{aligned} \quad (6)$$

where $\eta_{C\max}$ and η_X are positive coefficients which will be used to scale the dynamics of the network, and v' is the derivative of the penalty function v.

$$\upsilon'(\varepsilon) = 2\varepsilon \textit{ for all } \varepsilon \rangle 0 \textit{ and } \upsilon'(\varepsilon) = 0 \textit{ for all } \varepsilon \le 0$$

Since the computation is performed in all neurons at the same time, the network operates in a fully parallel mode.

The solution of equations of motion may be performed via the use of Euler's numerical integration method. The states of the neurons are updated at iteration *k* as follows.

$$UC_{\max}{}^{k} = UC_{\max}{}^{k-1} + \eta_{C\max} \frac{dUC \max}{dt} \quad (7)$$

$$UX_{ij}{}^{k} = UX_{ij}{}^{k-1} + \eta_X \frac{dUX_{ij}}{dt} \quad (8)$$

Neuron outputs are calculated by V=g (U), where g (.) is the activation function, U is the input and V is the output of the neuron.

VCmax=g(UCmax) = UCmax for UCmax ≥ 0; otherwise VCmax=0 (a piecewise linear function)

$VX_{ij} = g(UX_{ij}) = \text{logsig}(\lambda_X \times UX_{ij})$ (a log-sigmoid function)

where λ_X is the slope of the activation function and logsig(n) = 1 / (1 + exp(-n)).

3.4 Parameter Selection

We need to determine some parameters in order to simulate the proposed network for solving the problem described by the dynamics given in Section 3.3. These are the penalty parameters A, B, C and D; the activation slopes λ_X; the step sizes $\eta_{C\max}$, η_X and the initial conditions.

Since there is no theoretically established method for choosing the values of the penalty coefficients for an arbitrary optimization problem, the appropriate values for these coefficients can be determined empirically. In other words, simulation runs are conducted, and optimality and/or feasibility of the resulting equilibrium points of the system are observed. The network can be initialized to small random values, and then synchronous or asynchronous updating of the network will allow a minimum energy state to be attained. In order to ensure smooth convergence, step size must be selected carefully [20].

The dynamics of the proposed Hopfield-like gradient network will converge to local minima of the energy function E. Since the energy function includes four terms, competing to be minimized, there are many local minima and a tradeoff exists among the terms. The penalty parameters that result a feasible and a good solution, which minimizes the objective function should be found.

Determining the appropriate values of the penalty parameters, network parameters and initial states are critical issues associated with gradient type networks. Obviously, tradeoff problem will exist among the penalty terms to be minimized, in solving scheduling problems represented by many constraints. In the last years, some problems of Hopfield like NNs in solving optimization problems are observed. While several authors modified the energy function of the Hopfield network to improve the convergence to valid solutions [1,3,18], others employed different penalty parameters to the same formulation [9,13,14]. Recently, time based penalty parameters are proposed to overcome the tradeoff problems encountered in using penalty function approach. [19] used monotonically time-varying penalty parameters for solving convex programming problems. [6] proposed linearly increasing time-varying penalty parameters for solving clustering problems. In this paper, we propose to use time varying penalty parameters that take zero values as initial values and then they are increased in a linear fashion in a stepwise manner to reduce the feasible region. We update all the neurons synchronously since better simulation results are obtained for this problem.

The proposed gradient network algorithm can be summarised by the following pseudo-code.

```
Step 1. Construct an energy function for the considered problem using a penalty function approach.
Step 2. Initialize all neuron states to random values.
Step 3. Select the slope of the activation function ($\lambda$) and step sizes ($\eta$).
Step 4. Determine penalty parameters
Step 4.1. Select C (the coefficient of the inequality constraint) and assign zero as initial value to other penalty parameters A, B and D. If the constraint associated with parameter C is satisfied, proceed to Step 4.2 otherwise go back to Step 4.1.
Step 4.2. Select D (a higher value than C to increase the effect of equality constraint), and use the predetermined value of C (without taking into consideration of the effect of parameter A and B) to check whether both of the constraints associated with these terms are satisfied. If yes go to step 4.3, otherwise to step 4.4.
Step 4.3. Select B (a higher value than D), assign 1 to A, and use the predetermined values of C, D together with B to check whether all of the constraints associated with these terms are satisfied. If yes go to step 5, otherwise to step 4.4.
Step 4.4. Increase the value of parameter whose associated constraint is not satisfied.
Step 5. Repeat n times:
    Step 5.1. Update U using equations (7) and (8), and then compute V by V=g (U).
Step 6. If the energy has converged to local minimum proceed to step 7, otherwise go back to step 5.
Step 7. Examine the final solution to determine feasibility and optimality.
Step 8. Adjust parameters A, B, C, D if necessary to obtain a satisfactory solution, reinitialize neuron states and repeat from step 5.
```

4 Simulation Results

In order to evaluate the performance of the proposed gradient network in terms of solution quality, a simulation experiment was conducted. A 10-job 3-machine identical parallel machine scheduling problem was considered. The initial conditions of the network and the processing times of jobs were chosen randomly from uniform distribution in an interval [0,1], and [1,3], respectively.

Following the steps of the proposed methodology, we firstly try to satisfy the inequality constraints by penalizing them and run the simulations without considering any other constraints. For the first 2000 iterations, the best initial value of the penalty parameter C is determined as 8. Satisfying these inequality constraints after 2000 iterations, we can proceed to the next phase. In the next phase (for iterations from 2001 to 4000), one of the equality constraints (binary constraints) is taken into consideration, and its associated parameter D is chosen as 20, a value greater than C. The predetermined value of C, 8, is used to penalize the inequality constraint. Since both of the constraints are satisfied, it is decided to proceed to the next phase where we run the simulations for iterations from 4001 to 5000. In this phase, the aim is to satisfy all of the constraints. Using the predetermined values of C and D, the penalty parameter B belonging to the assignment constraint is chosen as 100 (a value greater than other parameters). Parameter A belongs to the original objective function and it is not penalized, and we assign 1 to A. After running simulations with all these 4 penalty terms, the feasibility and optimality of the final solution are checked. It is realised that except the inequality constraint, being violated with a small percentage error, all of the constraints are satisfied. Thus, it is decided to enhance the weight of this constraint, and the value of its parameter, C, is increased to 600. An optimal solution is obtained at iteration 5100. All of the constraints were met satisfactorily, and the cost value is found as 7.35. Empirically determined values of penalty parameters used during the solution of the problem are shown in Table 1.

Table 1. Penalty parameter values in four phases of simulation

Penalty Coef. / *Iterations*	A	B	C	D
1:2000	0	0	8	0
2001:4000	0	0	8	20
4001:5000	1	100	8	20
5001:5100	1	1	600	1

The proposed network was run for 5 different initial conditions on 3 different datasets and in Table 2, the results are compared with the results of the LPT rule and with the optimum solutions obtained by Lingo (version 8.0), a linear programming solver, in terms of Best Cmax (cost of the best solution obtained by the gradient network), Avg. Cmax (cost of the average solution obtained by the gradient network), Worst Cmax (cost of the worst solution obtained by the gradient network), and % deviations. Columns (6) and (7) represent the % deviations of the proposed gradient network solution from the LPT rule solution and from the optimal solution, respectively. The % deviations reported in Columns (6) and (7) are given by

$$\%\ deviation\ from\ LPT = \frac{Avg.C\max(Gradient\ network) - C\max(LPT)}{C\max(LPT)} * 100\%$$

$$\%\ deviation\ from\ the\ optimal = \frac{Avg.C\max(Gradient\ network) - C\max(optimal)}{C\max(optimal)} * 100\%$$

where Avg. Cmax(Gradient network) is the average gradient network solution of the 5 runs, Cmax(LPT) is the LPT solution and Cmax(optimal) is the optimal solution obtained by the linear programming solver. The percentage of times, which resulted in a feasible solution by the network, was also displayed in the last columns of these tables. It is obvious that the negative % deviation values from the LPT dispatching rule represent the % improvement realized by the gradient network.

Table 2. Results for m=3, n=10 over 3 problems

Gradient Network			LPT (4)	Optimum (5)	Deviation (%) from the LPT solution (6)	Deviation (%) from the optimal solution (7)	Percent Feasibility of Computed Solutions (8)
Best Cmax (1)	Avg. Cmax (2)	Worst Cmax (3)					
7.35	7.46	7.49	7.63	6.95	-2.2	7.34	100 %
6.99	7.11	7.20	7.41	6.77	-4.04	5.02	100 %
6.74	6.81	6.94	7.36	6.57	-7.47	3.65	100 %

The results given in Table 2 show that for the three datasets in all the initialization schemes, percent feasibility obtained is 100 %. If we consider the first dataset, the best, average and the worst makespan of the 5 feasible solutions are found as 7.35, 7.46 and 7.49, respectively. The average Cmax of the 5 runs is 2.22 % less costly than the result of LPT rule, and 7.34 % more costly than the global optimal solution. Even the worst makespan 7.49 obtained out of the 5 runs outperform the LPT rule result 7.63. Similarly, in all the three datasets, the worst makespans out of the 5 different runs outperform the LPT rule results. Since the initial conditions of the network effect the solution quality, the performance is tested on different initial conditions. From the results obtained, it is seen that besides the convergence to valid schedules, convergence of the proposed network to good quality solutions points out its general applicability in other scheduling environments.

5 Conclusions

In this paper, we have studied the identical parallel machine scheduling problem with the makespan criterion, which is known to be an NP-hard problem. A dynamical neural network that employs time varying penalty parameters is proposed for the solution of the problem. By this way, the tradeoff problem is tried to be overcome. The performance of the proposed network is evaluated on an example scheduling problem and the proposed methodology is explained. The simulation results obtained from the network is compared with the well-known LPT heuristic commonly used to solve the problem under study, and also with the optimal solutions in terms of the solution quality. The simulation experiments demonstrated that the proposed network generated feasible solutions for all the data sets and it found smaller makespan compared to LPT. To the best of our knowledge, there is no previously published article that tried to solve this NP-hard problem using neural networks, so we believe that this study will also make a contribution to the scheduling literature.

Further research can concentrate on analyzing the effects of parameters on the solution quality, selecting the parameters of the network automatically rather than choosing by trial and error, which is one of the shortcomings of neural networks and the performance of the proposed network can be tested on different sizes of scheduling models. Additionally, the implementation of the network in hardware can make a great progress in computational efficiency.

References

1. Aiyer, S.V.B., Niranjan, M., Fallside, F.: A theoretical investigation into the performance of the Hopfield model. IEEE Transactions on Neural Networks. 1 (1990) 204-215
2. Akiyama, Y., Yamashita, A., Kajiura, M., Aiso, H.: Combinatorial Optimization with Gaussian Machines. Proceedings IEEE International Joint Conference on Neural Networks. 1 (1989) 533-540
3. Brandt, R.D., Wang, Y., Laub, A.J., Mitra, S.K.: Alternative Networks for Solving the Travelling Salesman Problem and the List-Matching Problem. Proceedings of the International Conference on Neural Networks. 2 (1988) 333-340
4. Cheng, T., Sin, C.: A State-of-the-Art Review of Parallel-Machine Scheduling Research. European Journal of Operation Research. 47 (1990) 271-292
5. Coffman, E.G., Garey, M.R., Johnson, D.S.: An application of bin-packing to multiprocessor scheduling. SIAM Journal of Computing. 7 (1978) 1-17
6. Dogan, H., Guzelis, C.: (accepted paper) A new approach based on a time varying penalty parameter providing robustness and fuzziness in spherical clustering. IEEE Transactions on Circuits and Systems-II
7. Graham, R.L.: Bounds on multiprocessor timing anomalies. SIAM Journal of Applied Mathematics. 17 (1969) 416-429
8. Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan A.H.G.: Optimization and approximation in deterministic sequencing and scheduling: A survey. Annals of Discrete Mathematics. 5 (1979) 287–326
9. Hedge, S., Sweet, J., Levy, W.: Determination of parameters in a Hopfield/Tank computational network. Proceedings IEEE International Conference on Neural Networks. 2 (1988) 291-298

10. Hinton, G.E., Sejnowski, T.J.: Learning and relearning in Boltzmann machines. In: D.E. Rumelhart, & J.L. McClelland, Parallel Distributed Processing: Explorations in Microstructure of Cognition. Cambridge: MIT Press. (1986)
11. Hopfield, J.: Neurons with graded response have collective computational properties like of two-state neurons. Proceedings of the National Academy of Sciences of the USA. 81 (1984) 3088-3092
12. Hopfield, J., Tank, T.W.: Neural computation of decisions in optimization problems, Biological Cybernetics. 52 (1985) 141-152
13. Kamgar-Parsi, B., Kamgar-Parsi, B.: Dynamical Stability and Parameter Selection in Neural Optimization. Proceedings of International Joint Conference on Neural Networks. 4 (1992) 566-571
14. Lai, W.K., Coghill, G.G.: Genetic Breeding of Control Parameters for the Hopfield/Tank Neural Net. Proceedings of the International Joint Conference on Neural Networks. 4 (1992) 618-623
15. Mokotoff, E.: Paralel Machine Scheduling Problems: A Survey. Asia-Pacific Journal of Operational Research. 18 (2001) 193-242
16. Peterson, C., Anderson, J.R.: A mean-field theory learning algorithm for neural networks. Complex Systems. 1 (1987) 995-1019
17. Smith, K.A.: Neural Networks for Combinatorial Optimization: A Review of More Than a Decade of Research. INFORMS Journal on Computing. 11 (1999) 15-34
18. Van Den Bout, D.E., Miller, T.K.: A Traveling Salesman Objective Function that Works. Proceedings of IEEE International Conference on Neural Networks. 2 (1988) 299-303
19. Wang, J.: A Time-Varying Recurrent Neural System for Convex Programming. Proceedings of IJCNN-91-Seattle International Joint Conference on Neural Networks. (1991) 147-152
20. Watta, P.B., Hassoun, M.H.: A Coupled Gradient Network Approach for Static and Temporal Mixed-Integer Optimization. IEEE Transactions on Neural Networks. 7 (1996) 578-593

Ensemble of Competitive Associative Nets for Stable Learning Performance in Temperature Control of RCA Cleaning Solutions

Shuichi Kurogi[1], Daisuke Kuwahara[1], Hiroaki Tomisaki[1], Takeshi Nishida[1], Mitsuru Mimata[2], and Katsuyoshi Itoh[2]

[1] Kyushu Institute of Technology, Kitakyushu, Fukuoka 804-8550, Japan
[2] Komatsu Electronics Inc., Hiratsuka, Kanagawa 254-8453, Japan

Abstract. For cleaning silicon wafers via the RCA clean, temperature control is important in order to obtain a stable performance, but it is difficult mainly because the RCA solutions expose nonlinear and time-varying exothermic chemical reactions. So far, the MSPC (model switching predictive controller) using the CAN2 (competitive associative net 2) has been developed and the effectiveness has been validated. However, we have observed that the control performance, such as overshoot and settling time, does not always improve as the number of learning iterations increases when using multiple units of the CAN2. So we apply the ensemble learning scheme to the CAN2 for stable control over learning iterations, and we examine the effectiveness of the present method by means of computer simulation.

1 Introduction

The competitive associative net called CAN2, developed for utilizing the conventional competitive and associative schemes [1,2], has been shown effective in several applications such as control, function approximation, rainfall estimation, time series prediction, estimating predictive uncertainty and so on [5]-[11]. The CAN2 is characterized as the net which uses a gradient method for competitive learning, recursive least squares for associative learning, and an exploration heuristic based on an "asymptotic optimality" criterion (see [7]) for overcoming local minima problems of the gradient method. Although local linear models [12,13] also utilize piecewise linear approximation, they use linear models in piecewise regions obtained via the K-nearest neighbors, while the CAN2 utilizes linear models (associative memories) in the piecewise regions obtained via the competitive learning designed for minimizing the mean square error of function approximation. Here, note that the K-nearest neighbors are for minimizing the distance measures between input vectors and the centers of the piecewise regions, and do not relate to the mean square error of function approximation. Thus, the CAN2 is supposed to show better performance in function approximation and its applications.

As one of the applications of the CAN2, we have been dealing with the temperature control of RCA cleaning solutions, where the RCA cleaning introduced

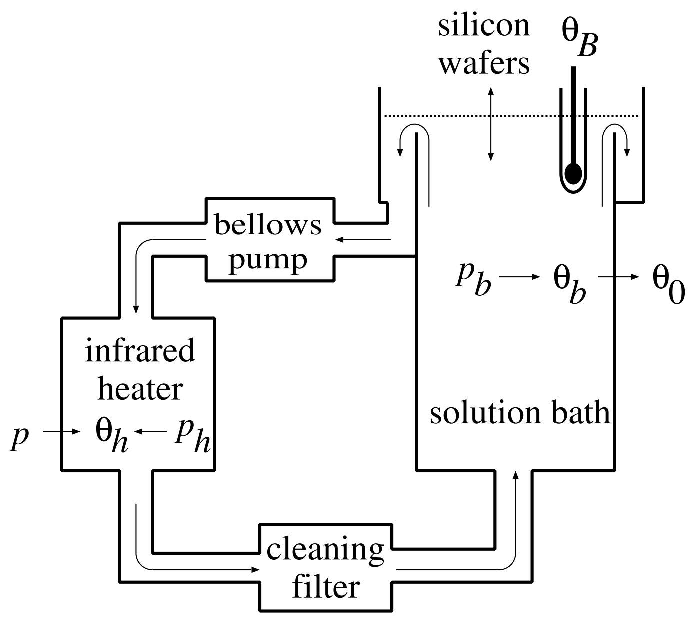


Fig. 1. Schematic diagram of the RCA cleaning system

and developed at Radio Cooperation of America is the industry standard way to clean silicon wafers and the temperature control is important for a stable cleaning performance. Since the RCA cleaning procedure uses corrosive and hazardous chemical solutions such as SPM (sulfuric acid, H_2SO_4, and hydrogen peroxide, H_2O_2, mixture) and so on, several special pieces of equipment are arranged for heating solutions as shown in Fig. 1, where there is a solution bath, a bellows pump, an infrared (IR) heater, and a cleaning filter, which are connected by anti-corrosive recirculation pipes; and the thermal sensor is covered by an anti-corrosive glass tube. Thus, this system involves long and fluctuating time lags and delays, and the mixture of the solutions exposes several exothermic reactions which are nonlinear and time-varying.

In order to control such nonlinear and time-varying plants involving time lags and delays, we have developed a control method called MSPC (model-switching predictive controller) using the CAN2 [3]-[10]. Precisely, the CAN2 in the MSPC learns multiple linear models of the plant dynamics from the input and output data of the plant, and then selects an appropriate linear model at each time of the control phase in order for the GPC (generalized predictive controller) to use the selected linear model. Although we could have obtained good control performance, such as overshoot and settling time, after certain learning iterations, we have observed that the performance does not always improve with the increase of learning iterations and increase of the number of units of the CAN2. One of the reasons for this phenomena is the supposition that the training data, which are obtained through the previous control iteration, may involve some biased

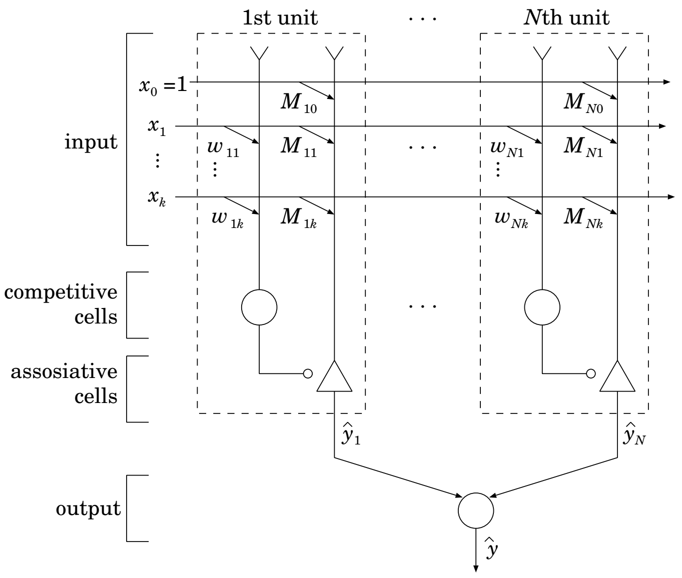


Fig. 2. Schematic diagram of the CAN2

data depending on the previous control trajectory, and the CAN2 may overlearn the data. One of the methods for avoiding overlearning is the cross-validation, with which we have obtained a certain level of improvement [10]. As another method for avoiding overlearning, we in this article try to apply the ensemble method, where the ensemble of the CAN2s may have a higher prediction ability than the single CAN2 because there are a number of researches showing that an ensemble prediction is (not average but) often more accurate than any of the single predictions in the ensemble [14].

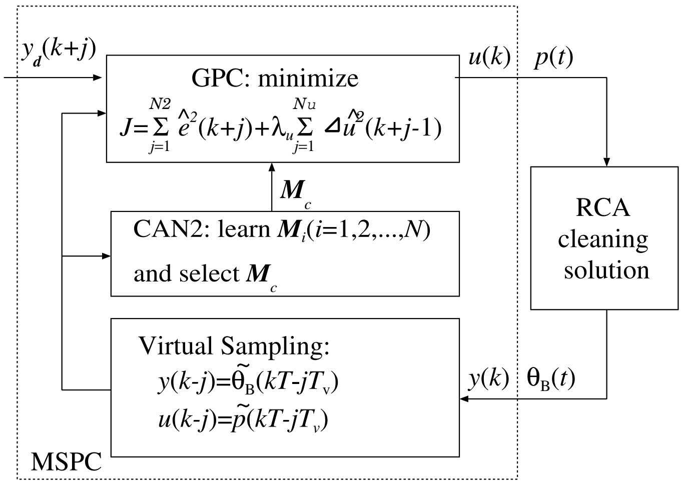
This article is organized as follows; Section 2 gives a brief overview of the CAN2 and the iterations of control and batch learning. Section 3 introduces the CAN2 ensemble for the MSPC to control the plant. Section 4 shows the result of numerical experiments and examines the performance.

2 Single CAN2 for Control

2.1 CAN2 for Approximating the System Dynamics

In Fig. 1, the input power $p = p(t)$ and the bath temperature $\theta_B = \theta_B(t)$ are sampled with a sampling period T_v, and denoted by $u(j) = p(jT_v)$ and $y(j) = \theta_B(jT_v)$ for $j = 1, 2, \cdots, n$, where they actually are sampled by the virtual sampling method [3]. Further, suppose they have the relation given by

$$y(j) = f(\boldsymbol{x}(j)) + d(j) \tag{1}$$



$y_d(k+j)$
GPC: minimize
$J=\sum_{j=1}^{N2}\hat{e}^2(k+j)+\lambda_u\sum_{j=1}^{Nu}\Delta\hat{u}^2(k+j-1)$
$u(k)$
$p(t)$
$\boldsymbol{M}_c$
CAN2: learn $\boldsymbol{M}_i(i=1,2,...,N)$ and select $\boldsymbol{M}_c$
RCA cleaning solution
Virtual Sampling:
$y(k-j)=\tilde{\theta}_B(kT-jT_v)$
$u(k-j)=\tilde{p}(kT-jT_v)$
$y(k)$
$\theta_B(t)$
MSPC

Fig. 3. Schematic diagram of MSPC using the associative matrices $\boldsymbol{M}_i$ ($i = 1, 2, \cdots, N$) learned by the batch CAN2 for controlling the temperature of RCA cleaning solution

where the function $f(\cdot)$ does not change or changes slowly in time, $d(j)$ represents zero-mean noise with the variance σ_d^2, and the input vector $\boldsymbol{x} = \boldsymbol{x}(j) \in X \triangleq \mathbb{R}^{k\times 1}$ is given by

$$\boldsymbol{x}(j) = (1, y(j-1), \cdots, y(j-k_y), u(j-1), \cdots, u(j-k_u))^T \tag{2}$$

where k_y and k_u are positive integers, and $k = 1 + k_y + k_u$.

A CAN2 has N units. The ith unit has a weight vector $\boldsymbol{w}_i \triangleq (w_{i1}, \cdots, w_{ik})^T \in \mathbb{R}^{k\times 1}$ and an associative matrix $\boldsymbol{M}_i \triangleq (M_{i0}, M_{i1}, \cdots, M_{ik}) \in \mathbb{R}^{1\times(k+1)}$ for $i \in I = \{1, 2, \cdots, N\}$ (see Fig. 2). The CAN2 approximates the above function $y = f(\boldsymbol{x})$ for $\boldsymbol{x} = \boldsymbol{x}(j)$ by

$$\widehat{y} \triangleq \widehat{y}_c \triangleq \boldsymbol{M}_c\widetilde{\boldsymbol{x}} \tag{3}$$

where $\widetilde{\boldsymbol{x}} \triangleq (1, \boldsymbol{x}^T)^T \in \mathbb{R}^{(k+1)\times 1}$. The MSPC at every discrete time j selects the model $\boldsymbol{M}_c$ which minimize the approximation error for $l = j-1, j-2, \cdots, j-N_e$, namely c is selected by

$$c \triangleq c(j) \triangleq \arg\min_{i\in I} \sum_{l=j-1}^{j-Ne} \|y(l) - \boldsymbol{M}_i\widetilde{\boldsymbol{x}}(l)\| \tag{4}$$

where N_e is a constant. Note that the weight vector $\boldsymbol{w}_c$ is not used in the above procedure, but used for the batch learning (see [8] for details).

2.2 Control and Batch Learning Iterations

We use the controller called MSPC (see Fig. 3 and [3] for details) involving the CAN2. Here, note that when we apply the MSPC involving the CAN2 to the real

RCA system, we use the CAN2 after learning the data obtained from simulations using the model RCA system. The simulation (shown bellow) is executed by the repetition of control and batch learning iterations as follows; after we set certain initial values to the associative matrices $\boldsymbol{M}_i (i \in I)$ of the CAN2, and then execute the following two procedures repeatedly;

1) **Control procedure:** control the model RCA system by means of the MSPC with the CAN2, from which obtain $D = \{(\boldsymbol{x}(j), y(j) | j = 1, 2, \cdots, n)\}$ consisting of the input and the output of the RCA system as given by (1) and (2)
2) **Batch learning procedure:** execute the batch learning with the dataset D where the $\boldsymbol{M}_i$ are updated to minimize the approximation error on the dataset D (see [8] for details of the batch learning of the CAN2).

2.3 Problems on Conventional Method

In the actual factories, the RCA cleaning is executed repeatedly for cleaning many silicon wafers where the system dynamics varies depending on the environmental conditions, such as the humidity and temperature of the room, the amount of the remaining solution in the pipe used at the previous control, and so on, which cannot be controlled precisely in the current system. A strategy for overcoming this problem, as well as saving cost, is that we use several units for the CAN2 to learn different dynamics at various environmental conditions. Precisely, when we apply the MSPC to the actual RCA system, we are going to use the CAN2 after learning the data obtained from simulations with the model system, whose parameter values are set differently according to the variability of the actual system. However, we have observed a problem on the learning of the CAN2, i.e. the control performance such as overshoot and settling time does not always improve with the increase of learning iterations. In order to overcome this problem, we will try to apply the ensemble of the CAN2s.

3 Ensemble of CAN2s for MSPC

Suppose we have multiple CAN2s, and the lth CAN2 has l units with the weight vectors $\boldsymbol{w}_i^{(l)}$, the associative matrices $\boldsymbol{M}_i^{(l)}$ for $i = 1, 2, \cdots, l$. Further, let $\widehat{y}^{(l)} = \widehat{y}_c^{(l)} = \boldsymbol{M}_c^{(l)} \widetilde{\boldsymbol{x}}$ be the output of the lth CAN2. Then, we use a simple ensemble given by the mean of the outputs of the CAN2s as

$$\widehat{y}^{(l_1,l_2)} = \frac{1}{l_d} \sum_{l=l_1}^{l_2} \widehat{y}^{(l)} = \frac{1}{l_d} \sum_{l=l_1}^{l_2} \boldsymbol{M}_c^{(l)} \widetilde{\boldsymbol{x}} = \widehat{\boldsymbol{M}}^{(l_1,l_2)} \widetilde{\boldsymbol{x}}, \tag{5}$$

where l_1 and l_2 and $l_d \triangleq l_2 - l_1 + 1$ are positive integers, and

$$\widehat{\boldsymbol{M}}^{(l_1,l_2)} = \frac{1}{l_d} \sum_{l=l_1}^{l_2} \boldsymbol{M}_c^{(l)}. \tag{6}$$

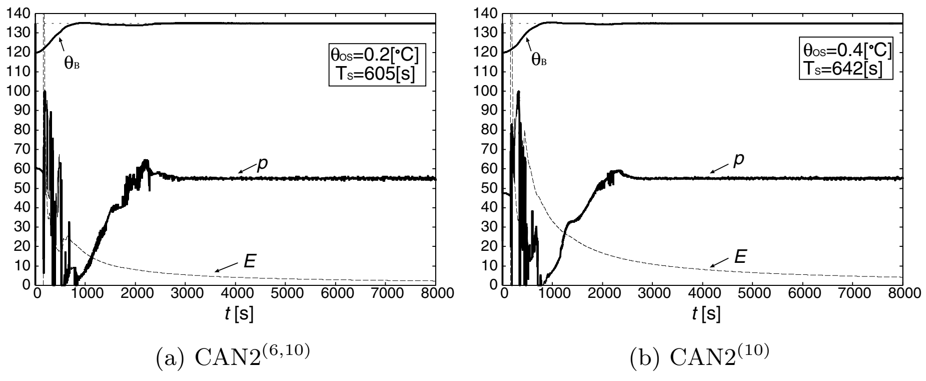


Fig. 4. Examples of the time course of the control at a control iteration with the CAN2 after $L = 30$ times of control and learning iterations, where (a) is of the present method using CAN2$^{(6,10)}$, and (b) is of the conventional method using CAN2$^{(10)}$. The input power relative to the maximum power is represented by p[%], the output temperature is θ_B[°C], the mean square prediction error is E[5000(°C)^2], the overshoot is θ_{OS}[°C], and the setting time t_{S}[s].

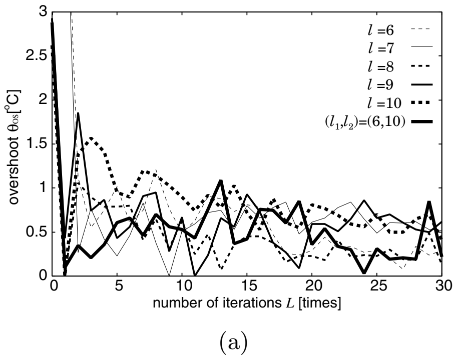
For simple expression of the following, let CAN2$^{(l)}$ denote the single CAN2 with l units, and CAN2$^{(l_1,l_2)}$ be the ensemble of the single CAN2s using $l = l_1, l_1+1, \cdots, l_2$. The above two equations show that the ensemble of the outputs is equivalent to the output using the ensemble of linear models $\widehat{\boldsymbol{M}}^{(l_1,l_2)}$. Since the MSPC utilizes GPC (generalized predictive controller) which requires the linear model of the plant, the model ensemble $\widehat{\boldsymbol{M}}^{(l_1,l_2)}$ is sufficient for the MSPC to calculate the control input $\widehat{u}(j)$ at the current discrete time j or the real time $t = jT_v$.

4 Numerical Experiments

We have examined the present method with the model RCA system whose parameter values are of the SPM, which is one of the most difficult RCA solutions to be controlled. The initial temperature is set $\theta_B = 120$ [°C] at $t = 0$ [s], which also makes the control more difficult, and the set point is 135 [°C] with allowable error ± 2 [°C]. The dimension of the input vector $\boldsymbol{x}(j)$ of Eq.(2) is $k = 4$ where $k_y = 2$ and $k_u = 1$. Since the sampling period is $T = 0.25$ [s], the number of the training data for the CAN2 is about $n = 32,000 = 8,000/0.25$.

4.1 Control Iteration

In order to understand a control iteration, examples of time course of the input power $p = p(t)$ and the output temperature $\theta_B = \theta_B(t)$ in a control iteration after $L = 30$ times of the control and learning iterations are shown in Fig. 4, where (a) and (b) show the results of the present and the conventional methods,



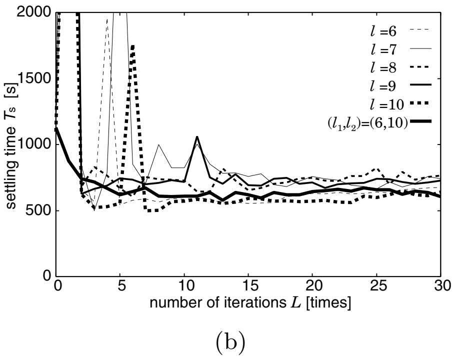


Fig. 5. (a) Overshoot $\theta_{\rm OS}$ and (b) setting time $T_{\rm S}$ achieved by single CAN2s and the CAN2 ensemble versus the number of control and learning iterations

respectively. From Fig. 4, we can see that the temperature θ_B increases even if the input power p is small, around $t = 1000$ [s], which indicates that exothermic chemical reaction is occurring, while p should be kept about 60[%] (of the maximum power) around $t > 2500$ where the exothermic reaction is disappeared. For evaluating the control performance, the overshoot $\theta_{\rm OS}$ and the settling time $T_{\rm S}$ are calculated. Further, for evaluating the prediction error, the mean square prediction error at time t given by

$$E(t) \triangleq \frac{1}{t}\sum_{j=1}^{t} \|e(j)\|^2 \triangleq \frac{1}{t}\sum_{j=1}^{t} \|\widehat{y}(j) - y(j)\| \qquad (7)$$

is also calculated, where $\widehat{y}(j)$ is the prediction by the single CAN2 or the CAN2 ensemble at the discrete time j, and $e(j) \triangleq \widehat{y}(j) - y(j)$ is the prediction error. From Fig. 4, we can see that the present method shows a slightly better performance than the conventional one, but we are interested much more in the stability through the increase of control and learning iterations, which is examined in the next section.

4.2 Stability of Control Performance Through Number of Iterations

First, we examine the effect of the ensemble in control performance. As an example, Fig. 5 shows the overshoot $\theta_{\rm OS}$ and the settling time $T_{\rm S}$ as the control performance achieved by single CAN2s and the CAN2 ensemble with respect to the number of control and learning iterations, L. From Fig. 5(a), we can see that the CAN2 ensemble with $(l_1, l_2) = (6, 10)$ achieves the overshoot $\theta_{\rm OS}$ less than the allowable error 2[°C] for $L \geq 1$, while the single CAN2 with $l = 6$ and 7, respectively, could not achieve the overshoot less than the allowable error at $L = 1$. Further, we can see the CAN2 ensemble achieves the settling time $T_{\rm S}$ stabler than the single CAN2s as shown in Fig. 5(b).

Next, we examine the effect of the number of units in the ensemble. As an example, Fig. 6 shows the control performance achieved by the CAN2 ensembles

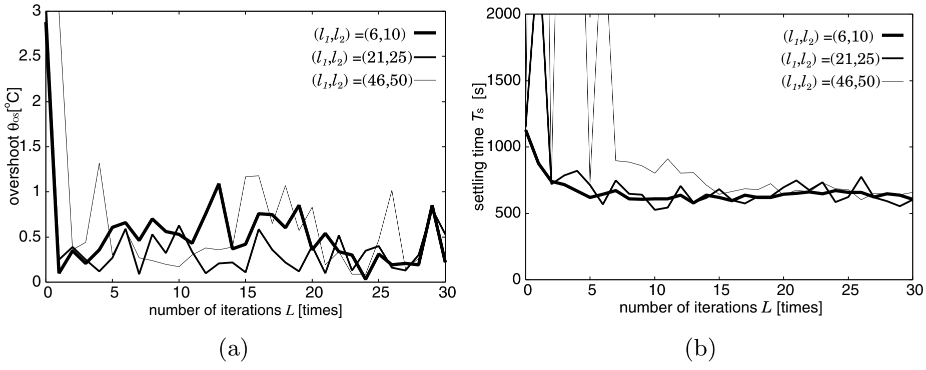


Fig. 6. (a) Overshoot $\theta_{\rm OS}$ and (b) setting time $T_{\rm S}$ of the CAN2 ensembles with five single CAN2s using different number of units

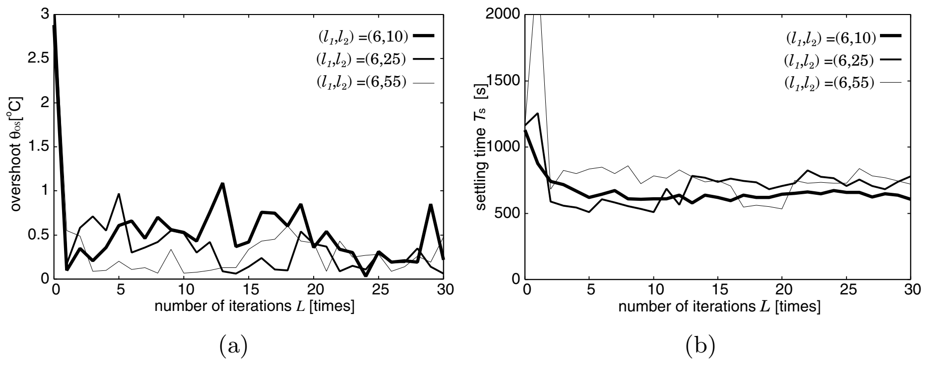


Fig. 7. (a) Overshoot $\theta_{\rm OS}$ and (b) setting time $T_{\rm S}$ of the CAN2 ensembles with increasing number of single CAN2s

with five single CAN2s using different number of units. We can see that the CAN2 ensemble using $(l_1, l_2) = (6, 10)$ is the best among the ones in the figure because it achieves the overshoot in the allowable error for $L \geq 1$ and stable settling time for the increase of the number of the iterations, L.

As another example, Fig. 7 shows the control performance achieved by the CAN2 ensembles with increasing number of units. From the figure, we may say that every CAN2 ensemble shows a good performance in overshoot from the point of view of the allowable error ± 2[°C], and the CAN2 with $(l_1, l_2) = (6, 10)$ seems to be the best in the stability of the settling time.

5 Conclusion

We have introduced the CAN2 ensemble for the stability of control performance through the increase of control and learning iterations. By means of numerical experiments, we have observed that the CAN2 ensemble shows better performance than single CAN2s. We have examined different members for the CAN2 ensemble, and the result indicates that there may be a good set of members. However, we have not yet examined how we can choose good members for the ensemble, this will be for our future research.

This research was partially supported by the Grant-in-Aid for Scientific Research(B) 16300070 of the Japanese Ministry of Education, Science, Sports and Culture.

References

1. Kohonen, T.: Associative Memory. Springer Verlag (1977)
2. Rumelhart, D.E. and Zipser, D.: A feature discovery by competitive learning. ed. Rumelhart, D.E., McClelland, J.L. and the PDP Research Group: Parallel Distributed Processing. The MIT Press, Cambridge, **1** (1986) 151–193
3. Kurogi, S., Nishida, T., Sakamoto, T., Itoh, K. and Mimata, M.: A simplified competitive associative net and a model-switching predictive controller for temperature control of chemical solutions. Proc. ICONIP2000 (2000) 791–796
4. Kurogi, S., Nishida, T., Nobutomo, H., Sakamoto, T., Mimata, M. and Itoh, K.: A thermal model of the RCA cleaning system and adaptive predictive temperature control of cleaning solutions. Trans. SICE of JAPAN (in Japanese) **37**(8) (2001) 754–762
5. Kurogi, S., Nobutomo, H., Sakamoto, H., Fuchikawa, Y., Mimata, M. and Itoh, K.: An analysis of competitive associative net for temperature control of RCA cleaning solutions. Proc. ICONIP2001 (2001) 957–962
6. Kurogi, S. and Nishida, T.: Adaptive and predictive control using competitive associative net for learning and switching of multiple models. Trans. SICE of JAPAN (in Japanese) **37**(3) (2001) 203–212
7. Kurogi, S.: Asymptotic optimality of competitive associative nets for their learning in function approximation. Proc. of the 9th International Conference on Neural Information Processing **1** (2002) 507–511
8. Kurogi, S., Araki, N., Miyamoto, H., Fuchikawa, Y. and Nishida, T.: Temperature control of RCA cleaning solutions using batch learning competitive associative Net. Proc. SCI2004 **5** (2004) 18–23
9. Kurogi, S., Sawa, M., Ueno, T. and Fuchikawa, Y.: A batch learning method for competitive associative net and its application to function approximation. Proc. SCI2004 **5** (2004) 24–28
10. Tomisaki, H., Kurogi, S., Araki, N., Nishida, T., Fuchikawa, Y., Mimata, M. and Itoh, K.: Cross-validation of competitive associative nets for stable temperature control of RCA cleaning solutions. Proc. ICONIP2005 (2005) 166–170
11. Evaluating Predictive Uncertainty Challenge: http : //predict.kyb.tuebingen.mpg.de/pages/home.php
12. Farmer, J.D. and Sidorowich, J.J.: Predicting chaotic time series. Phys. Rev.Lett. **59** (1987) 845–848
13. Chandrasekaran, H. and Manry, M.T.: Convergent design of a piecewise linear neural network. Proc. IJCNN1999 **2** (1999) 1339–1344
14. Opitz, D. and Maclin, R. Popular ensemble methods : an empirical study. Journal of Artificial Intelligence Research **11** (1999) 169–198.

Predication of Properties of Welding Joints Based on Uniform Designed Neural Network

Shi Yu[1], Li Jianjun[2], Fan Ding[1], and Chen Jianhong[2]

[1] Key Laboratory of Non-ferrous Metal Alloys, The Ministry of Education, Lanzhou Univ. of Tech., Lanzhou 730050, China
{shiyu, fanding}@lut.cn
[2] State Key Lab of Gansu New Non-ferrous Metal Materials, Lanzhou Univ. of Tech., Lanzhou 730050, China

Abstract. It is difficult to predict the mechanical properties of welded joints because of non-linearity in welding process and complicated mutual effects in multi composition welding material. Based on these practical problems, the application of neural network technology in predicting mechanical properties of welding joints is developed. The modeling method has been studied and the author puts forward that the parameters of neutral network can be optimized by the method of uniform design. The neutral network model of mechanical properties of welding joints is established on the basis of the data of welding thermal simulation, and the experimental results show that this model can predict the mechanical properties including impact toughness, tensile strength, subdued strength, reduction ratio of area and hardness more accurately. At the same time, using this method can improve estimating precision largely compared with using traditional statistic method. That is, this method provides an effective approach to estimate the mechanical properties of welding joints.

1 Introduction

It is a tough problem to predict the mechanical properties of the heat affected zone (HAZ) of welding joints with different welding specification, because of non-linearity in welding process and complicated mutual effects in multi composition welding material. In order to solve this problem, a great deal of welding procedure qualification reports (PQR) and welding procedure specifications (WPS) are developed in practice, and labor and resource are wasted. It makes production cycle longer and production cost higher. In latest several decades, the neural networks are applied widely in welding field In consideration of its superior characteristics of non-linear mapping and fault-tolerant [1], the author uses the neural network model to predict the mechanical properties of welding joints. In addition, considering the uncertainty of the BP neural network parameters during the period of designing model and the need of reducing the workload of designing neural network as possible, the author brings the method of uniform design into optimizing the parameters and has obtained good results.

2 Constructing and Classify the Sample Data

2.1 Construction of the Sample Data

For setting up the predicting model, large quantities of accurate and dependable sample data are required. In order to obtain these sample data, 17 kinds of steel materials are machined into specimens and executed to simulate the welding heat cyclic process under the different cooling time $t_{8/5}$ in welding thermal simulator. About 159 groups of sample data including tensile strength, yield strength, impact toughness, reduction of area and hardness under the different cooling time $t_{8/5}$ are obtained.

2.2 Classifying of the Sample Data

Because of the differences of alloying composition and heat treatment process, mechanical properties have biggish difference between the 17 kinds of steel materials. If the same model were used to predict the mechanical properties of the 17 kinds of steel materials, it would lessen the predicting accuracy and could not approximate the property functions of steel materials. So it is necessary to divide these materials on the close principle of alloying composition and mechanical properties into four classes: low carbon alloy steel (hot rolling), low-alloy steel (hot rolling, normalizing), low-alloy steel (quenching and tempering), Cr-Mo steel.

3 Selecting the Architecture and Optimizing the Parameters of the Neural Network

3.1 Selection of Architecture of the Network

The BP neural network has a formidable spatial mapping ability and many mature and effective learning algorithms to train the network, so it is chosen as the structure of the predicting model. The network structure is shown in Fig. 1.

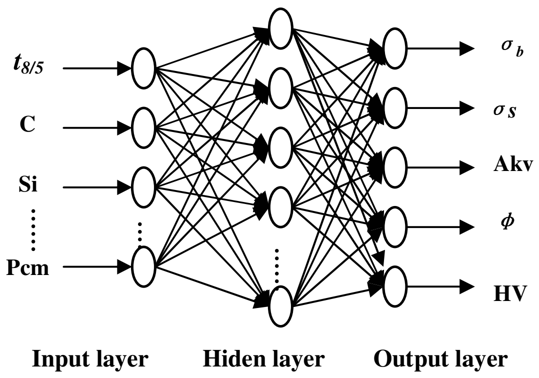


Fig. 1. Neural network mode

According to the metallographic principles and the characteristics of welding process, these fifteen parameters including the welding cooling time $t_{8/5}$, the content of carbon, silicon, manganese, sulfur, phosphorus, chromium, molybdenum, vanadium, titanium, nickel, niobium, boron, aluminum and cold cracking sensitive coefficient P_{cm} are chosen as inputs. The network outputs contains tensile strength (σ_b), yield strength (σ_s), impact toughness (*Akv*), reduction of area (ϕ) and hardness (*HV*). The number of hidden-layer neurons (*HN*) and others parameters will be obtained through optimizing BP network by the uniform design method in section 4.1.

3.2 Optimizing the Parameters of Neural Network

At present, the study of the neural network is still at the stage of grope. There are not mature theories and rules to follow during the design of the neural network for specific application. Usually the network parameters are set by tests based on the experience. But it costs a great deal of time and resources and can not ensure that a satisfied neural network could be found [2]. Common optimizing methods need more experiments and have stronger blindness and larger uniformity error [3]. The method of uniform design that applies the number theory in the mathematical statistic successfully is an optimizing method that adapts larger variety of the factors. Moreover it is a direct design method that does not rely on the concrete problems. The neural network training process essentially is obtaining one group or several groups of appropriate weights by iterative algorithm [4]. Suppose the mapping relation of the neural network is $f(x)$. It is a multivariate function in number field $\boldsymbol{D}$. The purpose of the training process is obtaining the maximum of $f(x)$ and the value x^* which follows the equation (1):

$$f(x^*) = \max_{x \in C^s} f(x) \tag{1}$$

Because $f(x)$ has many peaks, the classical gradient methods can not ensure that $f(x)$ obtains the global optimal solution. The actual method is picking out the best one as the global optimal approximate solution from certain local maximums that are obtained from certain starting spots by the gradient method. The starting spots initialized by uniform design are superior to those spots created randomly because these spots distribute evenly in field $\boldsymbol{D}$.

The experiment is arranged according to the method of uniform design. At first, it needs to choose the appropriate factors and the level numbers according to its goal and the characteristics of the object. Suppose there are $\boldsymbol{S}$ parameters and the level number is n, and then

$$X_{ki} \in [a_k, f_k] \qquad (k \in S, i = 1,2,\ldots,n) \tag{2}$$

$$X_{ki} = (1 - C_{ki}^{1/(s-i+1)}) \prod_{j=1}^{i-1} C_{ki}^{1/(s-j+1)} \qquad (i = 1,2,\ldots,n) \tag{3}$$

The suitable $U_n^*(n^8)$ table used to looking up $\boldsymbol{S}$ that has been assigned and n in the uniform design table. X_{ki} is the element of the table. The level numbers of column elements can be adjusted. Follow equations can obtained from (3).

$$a_k \le (1-C_{ki}^{1/(s-i+1)})\prod_{j=1}^{i-1} C_{ki}^{1/(s-j+1)} \le f_k \qquad (i=1,2,\ldots,n) \tag{4}$$

$$(1-a_k)^s \ge C_{ki} \ge (1-f_k)^s \tag{5}$$

$$(1-a_k C_{k1}^S C_{K2}^{S-1})^{S-2} \ge C_{k3} \ge (1-f_k C_{k1}^S C_{k1}^{S-1})^{S-2} \tag{6}$$

$$\begin{aligned} X_{k1} &= (1-C_{kl}^{1/(S)}) \\ X_{k2} &= (1-C_{k2}^{1/(S-1)})C_{kl}^{1/(S)} \\ X_{K3} &= (1-C_{k3}^{1/(S-2)})C_{kl}^{1/(S)}C_{k2}^{1/(S-1)} \end{aligned} \tag{7}$$

X_{ki} can be obtained from the above formulas and fills in the uniform design table. Then the experiment can be done according to the table. The selection of C_{ki} is vital during the process of selecting independent variables and must be in its scope because it decides the uniformity of the whole design process. The appropriate uniform design table is selected according to the recommendation in the uniform design handbook. Then the factors are put in the uniform design table and their column numbers are selected from the using tables of these above uniform design tables and their level numbers are the same as the index of their classes. Now the experiment has been arranged well.

4 The Results of Experiments

4.1 The Experiment Results of Optimizing BP Network by the Uniform Design Method

The seven parameters including *HN*, μ_0, α, β, *moment*, *dmoment* and ρ are the important performance parameters of the neural network according to the characteristics of the neural network and the study algorithm where *HN* is the number of hidden-layer neurons, μ_0 is the coefficient in the transfer function, α is the training coefficient of weights, β is the training coefficient of biases, *dmoment* is the variance of the *moment* and ρ is the training proportion coefficient.

The table 1 ($U_{28}^*(28^8)$) is selected as the standard from the uniform training design handbook because of the minimal uniformity error of the above seven parameters. n is number of runs, s is number of factors and q is number of levels in $U_n^*(q^s)$ [5]. The step length and the variation range of seven parameters were obtained according to Table 1 and are shown in Table 3. Low carbon alloy steel (hot rolling) adopts the 15×72×5 three-layer BP network. Low-alloy steel (hot rolling, normalizing) adopts the 15×62×5 three-layers BP network. Low-alloy steel (quenching and tempering) adopts the 15×74×5 three-layer BP network. Cr-Mo steel adopts the 15×56×5 three-layer BP network. Then the experiment is done according to this table and Table 2 (the employing table of $U_{28}^*(28^8)$).

Table 1. Table $U_{28}^{*}(28^8)$

	1	2	3	4	5	6	7	8
1	1	7	16	18	20	23	24	25
2	2	14	3	7	11	17	19	21
3	3	21	19	25	2	11	14	17
4	4	28	6	14	22	5	9	13
5	5	6	22	3	13	28	4	9
6	6	13	9	21	4	22	28	5
7	7	20	25	10	24	16	23	1
8	8	27	12	28	15	10	18	26
9	9	5	28	17	6	4	13	22
10	10	12	15	6	26	27	8	18
11	11	19	2	24	17	21	3	14
12	12	26	18	13	8	15	27	10
13	13	4	5	2	28	9	22	6
14	14	11	21	20	19	3	17	2
15	15	18	8	10	10	26	12	27
16	16	25	24	27	1	20	7	23
17	17	3	11	16	21	14	2	19
18	18	10	27	5	12	8	26	15
19	19	17	14	23	3	2	21	11
20	20	24	1	12	23	25	16	7
21	21	2	17	1	14	19	11	3
22	22	9	4	19	5	13	6	28
23	23	16	20	8	25	7	1	24
24	24	23	7	26	16	1	25	20
25	25	1	23	15	7	24	20	16
26	26	8	10	4	27	18	15	12
27	27	15	26	22	18	12	10	8
28	28	22	13	11	9	6	5	4

Table 2. Employing table of $U_{28}^{*}(28^8)$

S	No. of column							D
2	1	4						0.0545
3	1	2	5					0.0935
4	1	2	5	7				0.1074
5	1	2	3	7	8			0.1381
6	1	2	3	5	6	7		0.1578
7	1	2	3	5	6	7	8	0.1550

Table 3. The step length and the variation range of the seven parameters

Parameter	*HN*	*μ0*	*α*	*β*	*moment*	*dmoment*	*ρ*
Step length	2	0.030	0.015	0.005	0.015	0.005	0.020
Variation range	30 ~84	0.100 ~0.940	0.200 ~0.605	0.050 ~0.185	0.500 ~0.905	0.010 ~0.150	1.020 ~1.560

Comparing with empiric design, the uniform design method can make points distribute more uniformly and make data more representative [6]. The results of uniform design experiment are shown in Table 4 and the results of the conventional empiric method are shown in Table 5. From these tables, it is easy find that the BP neural network model optimized by the method of uniform design has lower training error and training times than the model designed by empiric method.

Table 4. The best training results of uniform design

	σ_b	σ_s	*AKV*	ϕ	*HV*
Maximum of absolute error	0.214	0.262	0.449	0.310	0.310
Maximum of relative error (%)	0.863	1.127	2.486	1.470	1.256

Table 5. The best training results of conventional method (empiric design)

	σ_b	σ_s	*AKV*	ϕ	*HV*
Maximum of absolute error	0.363	0.598	1.111	3.819	0.440
Maximum of relative error (%)	1.300	1.959	5.646	5.188	1.505

4.2 The Predicted Results of Neural Network

Some samples reserved as checking samples are used to check the predicting accuracy of the network. Take the low carbon alloy steel (hot rolling) as the example, the predicate results are showed from Fig.2 to Fig.6.

The predicating results show that the neural network has high predicting accuracy of mechanical properties, but the predicting error values are big in some points of tensile strength and impact toughness. If more samples are used to train the model could be a good solution.

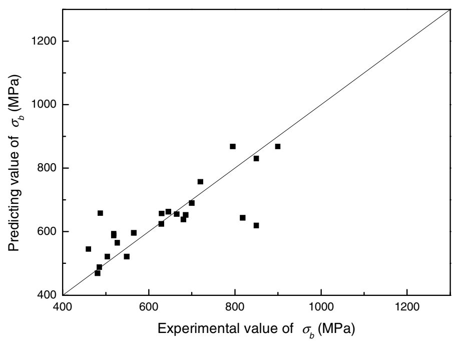


Fig. 2. Predicating results of σ_b

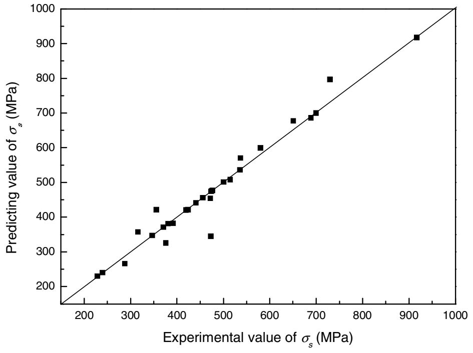


Fig. 3. Predicting results of σ_s

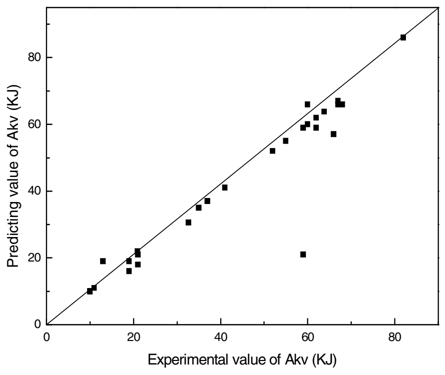


Fig. 4. Predicting results of *Akv*

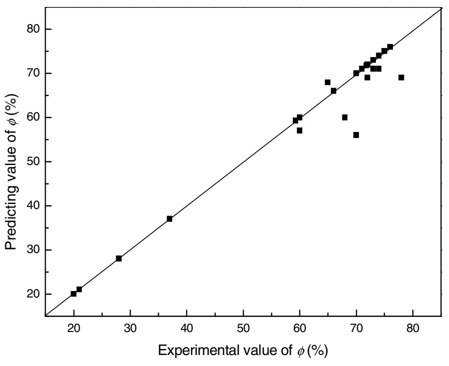


Fig. 5. Predicting results of ϕ

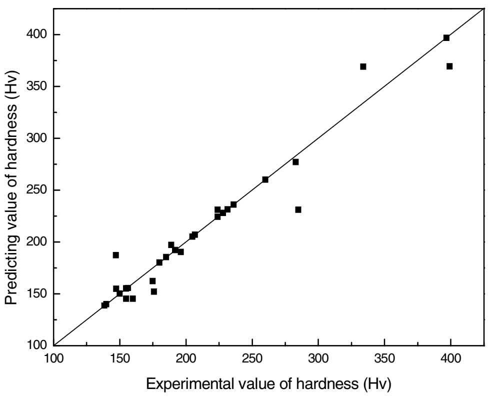


Fig. 6. Predicting results of hardness

4.3 Testing the Fault-Tolerance Ability of the Neural Network

The fault-tolerance ability is very important to the neural network, because all the samples being used to train the network come from the experiments and have some deviations. If the fault-tolerance ability of the neural network were bad, these deviations would be taken in the predicted results. In order to test the fault-tolerance ability of the network, some samples changed manually are inputted in the network and some fluctuations around these samples in the predicted results are found. The test shows that if the deviations are 2~4 times smaller than themselves, their influences on the predicted results can be acceptable.

4.4 Testing the Association Ability of the Neural Network

In order to test the association ability of the network, steels not having been used to train the network are selected as the samples to test the association ability of the network. Take the alloy steel as the example: the steels including HQ60, 15MnV, BHW-35 and STE-460 are used to train the network, and then 14MnVR steel is used to test the association ability of the network. The test results show that the predicting effect of tensile strength and hardness is better and the relative error is respectively 0.88%~13.04% and 0.93%~9.24%, that the predicting effect of yield strength and reduction of area (section shrinkage rate) is good and the relative error is respectively 6.69%~38.96% and 1.36%~48.54%. The predicting effect of impact toughness is bad and the relative error is 31.54%~343.77%. It is because of the large variation range of impact toughness, the strong influence of chemical composition and the relative insufficiency of sample data.

5 Conclusion

- The neural network for predicting the mechanical properties of weld joints has the better abilities to memory, association and fault-tolerant and has the high predicting accuracy. It is feasible to use the neural network to assist making welding craft.
- The neural network model having the reasonable parameter structure can be obtained easily by optimizing the model with the method of uniform design. It will reduce the training cycle and training error of neural network effectively.

References

1. Cook G E. Artificial neural networks applied to arc welding process modeling and control. IEEE Transactions on Industry Applications. vol. 26, no 5, pp824~830, May 1996.
2. Mozer M C,Jordan M L,Petsche T. Advances in neural information processing systems.Cambrideg, May 1997, pp162□168.
3. Tian, G. L. and K. T. Fang (1999). Uniform design for mixture-amount experiments and for mixture experiments under order restrictions. Science in China, Ser. A, vol.42, no.5, pp456~470, May 1999.
4. Fang, K. T., Lu, X., Tang, Y., Yin, J. X. Constructions of Uniform Designs by Using Resolvable Packings and Coverings. Discrete Mathematics, 2004, 2704, pp 25-040.
5. Fang, K. T, D. K. J. Lin, P. Winker and Y. Zhang. Uniform design: theory and applications. Technometrics, 42, 237~248.
6. Fang, K. T., Wang, Y. Number-Theoretic Methods in Statistics. London: Chapman and Hall, 1994.

Applying an Intelligent Neural System to Predicting Lot Output Time in a Semiconductor Fabrication Factory*

Toly Chen

Department of Industrial Engineering and Systems Management, Feng Chia University,
100, Wenhwa Road, Seatwen, Taichung City, Taiwan
tolychen@ms37.hinet.net
http://www.geocities.com/tinchihchen/

Abstract. Output time prediction is a critical task to a wafer fab (fabrication plant). To further enhance the accuracy of wafer lot output time prediction, the concept of input classification is applied to Chen's fuzzy back propagation network (FBPN) approach in this study by pre-classifying input examples with the k-means (kM) classifier before they are fed into the FBPN. Production simulation is also applied in this study to generate test examples. According to experimental results, the prediction accuracy of the intelligent neural system was significantly better than those of four existing approaches: BPN, case-based reasoning (CBR), FBPN without example classification, and evolving fuzzy rules (EFR), in most cases by achieving a 11%~46% (and an average of 31%) reduction in the root-mean-squared-error (RMSE) over the comparison basis – BPN.

1 Introduction

Predicting the output time for every lot in a wafer fab is a critical task not only to the fab itself, but also to its customers. After the output time of each lot in a wafer fab is accurately predicted, several managerial goals can be simultaneously achieved [5]. Predicting the output time of a wafer lot is equivalent to estimating the cycle (flow) time of the lot, because the former can be easily derived by adding the release time (a constant) to the latter. There are six major approaches commonly applied to predicting the output/cycle time of a wafer lot: multiple-factor linear combination (MFLC), production simulation (PS), back propagation networks (BPN), case based reasoning (CBR), fuzzy modeling methods, and hybrid approaches. Among the six approaches, MFLC is the easiest, quickest, and most prevalent in practical applications. The major disadvantage of MFLC is the lack of forecasting accuracy [5]. Conversely, huge amount of data and lengthy simulation time are two shortages of PS. Nevertheless, PS is the most accurate output time prediction approach if the related databases are continuingly updated to maintain enough validity, and often serves as a benchmark for evaluating the effectiveness of another method. PS also tends to be preferred because it allows for computational experiments and subsequent analyses without any actual execution [3]. Considering both effectiveness and efficiency, Chang et al. [4] and

* This work was support by the National Science Council, R.O.C.

Chang and Hsieh [2] both forecasted the output/cycle time of a wafer lot with a BPN having a single hidden layer. Compared with MFLC approaches, the average prediction accuracy measured with the root mean squared error (RMSE) was considerably improved with these BPNs. On the other hand, much less time and fewer data are required to generate an output time forecast with a BPN than with PS. Chang et al. [3] proposed a k-nearest-neighbors based case-based reasoning (CBR) approach which outperformed the BPN approach in forecasting accuracy. Chang et al. [4] modified the first step (i.e. partitioning the range of each input variable into several fuzzy intervals) of the fuzzy modeling method proposed by Wang and Mendel [8], called the WM method, with a simple genetic algorithm (GA) and proposed the evolving fuzzy rule (EFR) approach to predict the cycle time of a wafer lot. Their EFR approach outperformed CBR and BPN in prediction accuracy. Chen [5] constructed a fuzzy BPN (FBPN) that incorporated expert opinions in forming inputs to the FBPN. Chen's FBPN was a hybrid approach (fuzzy modeling and BPN) and surpassed the crisp BPN especially in the efficiency respect.

To further enhance the effectiveness of wafer lot output time prediction, the concept of input classification is applied to Chen's FBPN approach by pre-classifying input examples into different categories before they are fed into the network. The classification mechanism is kM. PS is also applied in this study to generate test examples. Using simulated data, the effectiveness of the intelligent neural system is shown and compared with those of four existing approaches, BPN, CBR, EFR, and FBPN without example classification.

2 Methodology

The intelligent neural system is composed of two parts. In the first part, kM is applied to classify wafer lots that are examples to the FBPN.

2.1 Wafer Lot Classification with kM

Every lot fed into the FBPN is called an example. The procedure of applying kM in forming inputs to the FBPN is now detailed:

1. Examples are pre-classified to m categories before they are fed into the FBPN according to their Euclidean distances to the category centers, which are arbitrarily chosen from those of all examples in the beginning. In this way, lot n is classified to category j with the smallest $d(n, j)$.
2. Each time after all examples are classified, the parameter sets of all category centers are recalculated by averaging those of the examples clustered in the same categories.
3. Example classification is continued until the sum of the average Euclidean distances (SADs) from examples to their category centers in all categories converges to a minimal value.

Examples of different categories are then learned with different FBPNs but with the same topology.

2.2 Output Time Prediction with FBPN Incorporating Expert Opinions

Subsequently, a FBPN is applied for output time prediction within a category. The configuration of the FBPN is established as follows:

1. Inputs: six parameters associated with the n-th example/lot including the average fab utilization (U_n), the total queue length on the lot's processing route (Q_n) or before bottlenecks (BQ_n) or in the whole fab (FQ_n), the fab WIP (WIP_n), and the latenesses ($D_n^{(i)}$) of the i-th recently completed lots. These parameters have to be normalized so that their values fall within [0, 1]. Then some production execution/control experts are requested to express their beliefs (in linguistic terms) about the importance of each input parameter in predicting the cycle (output) time of a wafer lot. Linguistic assessments for an input parameter are converted into several pre-specified fuzzy numbers. The subjective importance of an input parameter is then obtained by averaging the corresponding fuzzy numbers of the linguistic replies for the input parameter by all experts. The subjective importance obtained for an input parameter is multiplied to the normalized value of the input parameter.
2. Single hidden layer: Generally one or two hidden layers are more beneficial for the convergence property of the network.
3. Number of neurons in the hidden layer: the same as that in the input layer. Such a treatment has been adopted by many studies (e.g. [2, 5]).
4. Output: the (normalized) cycle time forecast of the example.
5. Network learning rule: Delta rule.
6. Transformation function: Sigmoid function,

$$f(x) = 1/(1+e^{-x}). \tag{1}$$

7. Learning rate (η): 0.01~1.0.
8. Batch learning.

The procedure for determining the parameter values is now described. A portion of the examples is fed as "training examples" into the FBPN to determine the parameter values. Two phases are involved at the training stage. At first, in the forward phase, inputs are multiplied with weights, summated, and transferred to the hidden layer. Then activated signals are outputted from the hidden layer as:

$$\tilde{h}_j = (h_{j1},\ h_{j2},\ h_{j3}) = 1/1 + e^{-\tilde{n}_j^h}\ , \tag{2}$$

where

$$\tilde{n}_j^h = (n_{j1}^h,\ n_{j2}^h,\ n_{j3}^h) = \tilde{I}_j^h (-) \tilde{\theta}_j^h\ , \tag{3}$$

$$\tilde{I}_j^h = (I_{j1}^h,\ I_{j2}^h,\ I_{j3}^h) = \sum_{all\ i} \tilde{w}_{ij}^h (\times) \tilde{x}_{(i)}\ , \tag{4}$$

and $(-)$ and $(\times)$ denote fuzzy subtraction and multiplication, respectively; $\tilde{h}_j$'s are also transferred to the output layer with the same procedure. Finally, the output of the FBPN is generated as:

$$\tilde{o} = (o_1,\ o_2,\ o_3) = 1/1 + e^{-\tilde{n}^o}, \tag{5}$$

where

$$\tilde{n}^o = (n_1^o,\ n_2^o,\ n_3^o) = \tilde{I}^o(-)\tilde{\theta}^o, \tag{6}$$

$$\tilde{I}^o = (I_1^o, I_2^o, I_3^o) = \sum_{all\ j} \tilde{w}_j^o (\times) \tilde{h}_j . \tag{7}$$

To improve the practical applicability of the FBPN and to facilitate the comparisons with conventional techniques, the fuzzy-valued output $\tilde{o}$ is defuzzified according to the centroid-of-area (COA) formula:

$$o = \mathrm{COA}(\tilde{o}) = (o_1 + 2o_2 + o_3)/4. \tag{8}$$

Then the defuzzified output o is applied to predict the actual cycle time a, for which the RMSE is calculated:

$$RMSE = \sqrt{\sum (o-a)^2 / \text{number of examples}}. \tag{9}$$

Subsequently in the backward phase, the deviation between o and a is propagated backward, and the error terms of neurons in the output and hidden layers can be calculated, respectively, as

$$\delta^o = o(1-o)(a-o), \tag{10}$$

$$\tilde{\delta}_j^h = (\delta_{j1}^h,\ \delta_{j2}^h,\ \delta_{j3}^h) = \tilde{h}_j(\times)(1-\tilde{h}_j)(\times)\tilde{w}_j^o \delta^o . \tag{11}$$

Based on them, adjustments that should be made to the connection weights and thresholds can be obtained as

$$\Delta\tilde{w}_j^o = (\Delta w_{j1}^o,\ \Delta w_{j2}^o,\ \Delta w_{j3}^o) = \eta\delta^o \tilde{h}_j, \tag{12}$$

$$\Delta\tilde{w}_{ij}^h = (\Delta w_{ij1}^h,\ \Delta w_{ij2}^h,\ \Delta w_{ij3}^h) = \eta\tilde{\delta}_j^h (\times)\tilde{x}_i, \tag{13}$$

$$\Delta\theta^o = -\eta\delta^o, \tag{14}$$

$$\Delta\tilde{\theta}_j^h = (\Delta\theta_{j1}^h, \Delta\theta_{j2}^h, \Delta\theta_{j3}^h) = -\eta\tilde{\delta}_j^h . \tag{15}$$

Theoretically, network-learning stops when the RMSE falls below a pre-specified level, or the improvement in the RMSE becomes negligible with more epochs, or a large number of epochs have already been run. In addition, to avoid the accumulation of fuzziness during the training process, the lower and upper bounds of all fuzzy numbers in the FBPN will no longer be modified if Chen's index [5] converges to a minimal value. Then test examples are fed into the FBPN to evaluate the accuracy of the network that is also measured with the RMSE. Finally, the FBPN can be applied to predicting the cycle time of a new lot. When a new lot is released into the fab, the six parameters associated with the new lot are recorded and fed as inputs to the FBPN. After propagation, the network output determines the output time forecast of the new lot.

In addition, the fuzzy-valued output $\tilde{o} = (o_1, o_2, o_3)$ of the FBPN can be thought of as providing a weighted interval forecast for the actual cycle time *a*, and it becomes possible to further reduce the RMSE with such weighted interval forecasts to the following value:

$$RMSE = \sqrt{\sum \min((o_1 - a)^2, (o_3 - a)^2) / \text{number of examples}} \,. \tag{16}$$

3 A Demonstrative Example from a Simulated Wafer Fab

In practical situations, the history data of each lot is only partially available in the factory. Further, some information of the previous lots such as Q_n, BQ_n, and FQ_n is not easy to collect on the shop floor. Therefore, a simulation model is often built to simulate the manufacturing process of a real wafer fabrication factory [1-7]. Then, such information can be derived from the shop floor status collected from the simulation model [3]. To generate a demonstrative example, a simulation program coded using Microsoft Visual Basic .NET is constructed to simulate a wafer fabrication environment with the following assumptions:

1. The distributions of the interarrival times of orders are exponential.
2. The distributions of the interarrival times of machine downs are exponential.
3. The distribution of the time required to repair a machine is deterministic.
4. The percentages of lots with different product types in the fab are predetermined. As a result, this study is only focused on fixed-product-mix cases. However, the product mix in the simulated fab does fluctuate and is only approximately fixed in the long term.
5. The percentages of lots with different priorities released into the fab are controlled.
6. The priority of a lot cannot be changed during fabrication.
7. Lots are sequenced on each machine first by their priorities, then by the first-in-first-out (FIFO) policy. Such a sequencing policy is a common practice in many foundry fabs.
8. A lot has equal chances to be processed on each alternative machine/head available at a step.
9. A lot cannot proceed to the next step until the fabrication on its every wafer has been finished. No preemption is allowed.

The basic configuration of the simulated wafer fab is the same as a real-world wafer fabrication factory which is located in the Science Park of Hsin-Chu, Taiwan, R.O.C. A trace report was generated every simulation run for verifying the simulation model. The simulated average cycle times have also been compared with the actual values to validate the simulation model. Assumptions (1)~(3), and (7)~(9) are commonly adopted in related studies (e.g. [2-5]), while assumptions (4)~(6) are made to simplify the situation. There are five products (labeled as A~E) in the simulated fab. A fixed product mix is assumed. The percentages of these products in the fab's product mix are assumed to be 35%, 24%, 17%, 15%, and 9%, respectively. The simulated fab has a monthly capacity of 20,000 pieces of wafers and is expected to be fully utilized (utilization = 100%). POs with normally distributed sizes (mean = 300 wafers; standard deviation = 50 wafers) arrive according to a Poisson process, and then the corresponding MOs are released for these POs a fixed time after. Based on these assumptions, the mean inter-release time of MOs into the fab can be obtained as (30.5 * 24) / (20000 / 300) = 11 hours. An MO is split into lots of a standard size of 24 wafers per lot. Lots of the same MO are released one by one every 11 / (300/24) = 0.85 hours. Three types of priorities (normal lot, hot lot, and super hot lot) are randomly assigned to lots. The percentages of lots with these priorities released into the fab are restricted to be approximately 60%, 30%, and 10%, respectively. Each product has 150~200 steps and 6~9 reentrances to the most bottleneck machine. The singular production characteristic "reentry" of the semiconductor industry is clearly reflected in the example. It also shows the difficulty for the production planning and scheduling people to provide an accurate due-date for the product with such a complicated routing. Totally 102 machines (including alternative machines) are provided to process single-wafer or batch operations in the fab. Thirty replicates of the simulation are successively run. The time required for each simulation replicate is about 12 minute on a PC with 512MB RAM and Athlon™ 64 Processor 3000+ CPU. A horizon of twenty-four months is simulated. The maximal cycle time is less than three months. Therefore, four months and an initial WIP status (obtained from a pilot simulation run) seemed to be sufficient to drive the simulation into a steady state. The statistical data were collected starting at the end of the fourth month. For each replicate, data of 30 lots are collected and classified by their product types and priorities. Totally, data of 900 lots can be collected as training and testing examples. Among them, 2/3 (600 lots, including all product types and priorities) are used to train the network, and the other 1/3 (300 lots) are reserved for testing.

3.1 Results and Discussions

To evaluate the effectiveness and efficiency of the intelligent neural system and to make some comparisons with four approaches – BPN, CBR, EFR, and FBPN without example classification, all the five methods were applied to five test cases containing the data of full-size (24 wafers per lot) lots with different product types and priorities. In the intelligent neural system, a fixed number of 5 categories were assumed for all cases. The convergence condition was established as either the improvement in the RMSE becomes less than 0.001 with one more epoch, or 1000 epochs have already been run. The minimal RMSEs achieved by applying the five approaches to different cases were recorded and compared in Table 1. As noted in Chang and Liao [5], the *k*-nearest-neighbors based CBR approach should be fairly compared with a BPN trained with only randomly chosen *k* cases. The latter was also adopted as the comparison basis, and the percentage of improvement on the minimal RMSE by applying another approach is enclosed in parentheses following the performance measure. The optimal value of parameter *k* in the CBR approach was equal to the value that minimized the RMSE [5]. According to experimental results, the following discussions are made:

1. From the effectiveness viewpoint, the prediction accuracy (measured with the RMSE) of the intelligent neural system was significantly better than those of the other approaches by achieving a 11%~46% (and an average of 31%) reduction in the RMSE over the comparison basis – BPN. The average advantages over CBR and EFR were 28% and 4%, respectively.

2. The effect of example classification is revealed by the fact that the prediction accuracy of the intelligent neural system was considerably better than that of FBPN without example classification in all cases with an average advantage of 27%.
3. In the case that the lot priority was the highest (super hot lot), the intelligent neural system has the greatest advantage over BPN and CBR in forecasting accuracy. In fact, the cycle time variation of super hot lots is the smallest, which makes their cycle times easy to predict. Clustering such lots seems to provide the most significant effect on the performance of cycle time prediction.
4. As the lot priority increases, the superiority of the intelligent neural system over BPN and CBR becomes more evident.
5. The greatest superiority of the intelligent neural system over EFR happens when the lot priority is the smallest (normal lots).

Table 1. Comparisons of the RMSEs of various approaches

RMSE	A(normal)	A(hot)	A(super hot)	B(normal)	B(hot)
BPN	177.1	102.27	12.23	286.93	75.98
FBPN	171.82(-3%)	89.5(-12%)	11.34(-7%)	286.14(-0%)	76.14(+0%)
CBR	172.44(-3%)	86.66(-15%)	11.59(-5%)	295.51(+3%)	78.85(+4%)
EFR	164.29(-7%)	66.21(-35%)	9.07(-26%)	208.28(-27%)	44.57(-41%)
I. N. S.	157.78(-11%)	54.93(-46%)	9.48(-22%)	197.1(-31%)	42.01(-45%)

4 Conclusions and Directions for Future Research

To further enhance the effectiveness of wafer lot output time prediction, the concept of input classification is applied to Chen's FBPN approach by pre-classifying input examples into different categories before they are fed into the network. The classification mechanism is kM. In this way, similar examples are clustered in the same category. Examples of different categories are then learned with different FBPNs but with the same topology. At last, an intelligent neural system is constructed with two special features tailed to the problem: incorporating expert opinions, and classifying examples. For evaluating the effectiveness of the intelligent neural system and to make some comparisons with four approaches – BPN, CBR, EFR, and FBPN without example classification, production simulation is applied in this study to generate test data. Then all the five methods are applied to five cases elicited from the test data. According to experimental results, the prediction accuracy (measured with the RMSE) of the intelligent neural system was significantly better than those of the other approaches by achieving a 11%~46% (and an average of 31%) reduction in the RMSE over the comparison basis – the BPN. The average advantages over CBR and EFR were 28% and 4%, respectively. The effect of example classification is revealed with an average advantage of 27% over FBPN without example classification.

However, to further evaluate the effectiveness and efficiency of the intelligent neural system, it has to be applied to fab models of different scales, especially a full-scale actual wafer fab. In addition, the intelligent neural system can also be applied to cases with changing product mixes or loosely controlled priority combinations, under which the cycle time variation is often very large. Besides, there are many other neural network methods that classify examples before learning as well. For example, a simple hybrid of a principal component analysis (PCA) neural network or a self-organization feature map (SOFM) and a FBPN can also be applied for the same purpose. These constitute some directions for future research.

References

1. Barman, S.: The Impact of Priority Rule Combinations on Lateness and Tardiness. IIE Transactions 30 (1998) 495-504
2. Chang, P.-C., Hsieh, J.-C.: A Neural Networks Approach for Due-date Assignment in a Wafer Fabrication Factory. International Journal of Industrial Engineering 10(1) (2003) 55-61
3. Chang, P.-C., Hsieh, J.-C., Liao, T. W.: A Case-based Reasoning Approach for Due Date Assignment in a Wafer Fabrication Factory. In: Proceedings of the International Conference on Case-Based Reasoning (ICCBR 2001), Vancouver, British Columbia, Canada (2001)
4. Chang, P.-C., Hsieh, J.-C., Liao, T. W.: Evolving Fuzzy Rules for Due-date Assignment Problem in Semiconductor Manufacturing Factory. Journal of Intelligent Manufacturing 16 (2005) 549-557
5. Chen, T.: A Fuzzy Back Propagation Network for Output Time Prediction in a Wafer Fab. Journal of Applied Soft Computing 2/3F (2003) 211-222
6. Goldberg, D. E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading, MA (1989)
7. Lin, C.-Y.: Shop Floor Scheduling of Semiconductor Wafer Fabrication Using Real-time Feedback Control and Prediction. Ph.D. Dissertation, Engineering-Industrial Engineering and Operations Research, University of California at Berkeley (1996)
8. Wang, L.-X., Mendel, J. M.: Generating Fuzzy Rules by Learning from Examples. IEEE Transactions on Systems, Man, and Cybernetics 22(6) (1992) 1414-1427
9. Weeks, J. K.: A Simulation Study of Predictable Due-dates. Management Science 25 (1979) 363–373

Multi-degree Prosthetic Hand Control Using a New BP Neural Network

R.C. Wang[1], F. Li[1], M. Wu[2], J. Z. Wang[1], L. Jiang[3], and H. Liu[3]

[1] Division of Intelligent and Biomechanical System, State Key Laboratory of Tribology, Tsinghua University, Beijing 100084, China
[2] Northwestern University
[3] Robotics Research Institute, Harbin Institute of Technology, Harbin 150001, China
wangrc@mail.tsinghua.edu.cn

Abstract. A human-like multi-fingered prosthetic hand, HIT hand, has been developed in Harbin Institute of Technology. This paper presents a new pattern discrimination method for HIT hand control. The method uses a bagged-BP neural network based on combing the BP neural networks using bagging algorithm. Bagging has been used to overcome the problem of limited number of training data in uni-model systems, by combining neural networks as weak learners. We compared the results of the bagging based BP network, using four features, with the results obtained separately from these uni-feature systems. The results show that the bagged-BP network improves both the accuracy and stability of the BP classifier.

1 Introduction

Surface electromyography (sEMG) signals are often used as interface for prosthetic devices, since they are the manifestation of electrical stimulations, which motor units receive from the central nervous system (CNS), and indicate the activation level of motor units associated with muscle contractions. Different motions resulting from different modes of muscle activation generate different EMG patterns.

The neural network (NN) is suitable for modeling nonlinear data, and is able to cover the distinction among different conditions. It has been widely used for the EMG-based motion discrimination since 1980s. However, in general, neural networks classifier is unstable. One method for improving stability of neural networks classifier is to construct good ensembles of classifiers [1].

This paper proposed a new BP neural network using bagging with four feature sets. Offline pattern discrimination experiments of the EMG signal were conducted using bagged-BP with four subjects, and the proposed method was compared with BP classifier using uni-feature set.

2 Background

2.1 HIT Hand

HIT hand was (shown in Fig.1) developed in Harbin Institute of Technology (HIT) [2]. The robot hand has 5 fingers and 14 joints just like human hand, and the thumb has 2 joints. All 5 fingers and 14 joints are controlled by only 3 separate electro-motors. Thumb, index finger and middle finger are controlled by each electromotor separately. The ring finger and the little finger of the prosthetic hand act along with middle finger. All other freedom-degrees are developed based on under-actuated adaptive theory. So the robot hand can achieve kinds of general grasp functions such as power grasp, pinch grasp and so on.

Considering the function of HIT hand, in the experiment, we try to discriminate six patterns of finger movements, i.e. thumb extension (TE), thumb flexion (TF), index finger extension (IE), index finger flexion (IF), middle finger extension (ME), and middle finger flexion (MF).

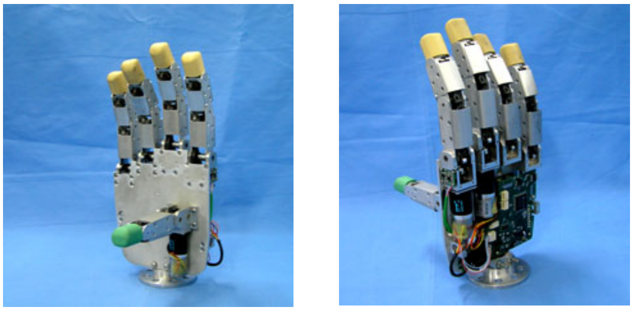


Fig. 1. HIT multi-finger robot hand

2.2 Signal Representation

The performance of time domain feature set (TD), the autoregressive coefficients (AR), cepastral (CEP) coefficients and those based upon the wavelet transform (WT) were used to discriminate the EMG patterns. Data were acquired from four channels of EMG from electrodes placed on extensor digitorum, extensor pollicis brevis, flexor carpi ulnaris and flexor digitorum.

The TD [3] was got by computing the mean absolute value (MAV), zero crossings (ZC), slope sign changes (SSC) and waveform length (WL) of each channel, and combining the four channels' parameters to construct the TD feature vector $\{MAV1, ZC1, SSC1, WL1, \cdots, MAV4, ZC4, SSC4, WL4\}$.

The AR feature set [4] is a vector containing 16 parameters, including the four-order AR coefficients of each channel. The CEP feature set [5] is also a vector containing 16 parameters, containing the four-order cepastral coefficients of each channel.

We made wavelet decomposition at level four, using coiflet4 wavelet. Once the wavelet decomposition vector containing the approximate coefficients and the detail coefficients are obtained they were subject to dimensionality reduction using singular value decomposition respectively, and the standard deviation are also calculated. So for four channel raw EMG signals, by doing wavelet decomposition and dimension reduction, a WT feature set of 40 parameters was got. Table 1 shows the details of the four feature sets.

Table 1. List of created feature set for each hand motion

Parameter title	Parameter number
4th order AR model	4×4
4th order cepstrum coefficients	4×4
Time domain feature set	4×4
Wavelet coefficients	10×4

2.3 Bagged BP Classifier

Bagging predictors [6] is a method for generating multiple versions of a predictor and using these to get an aggregated predictor. The aggregation averages over the versions when predicting a numerical outcome and does a plurality vote when predicting a class. The multiple versions are formed by making bootstrap replicates of the learning set and using these as new learning sets. For unstable procedures, such as neural networks, bagging works well. It can push a good but unstable procedure a significant step towards optimality. In this paper, we bagged BP neural networks on real EMG data and compare the classification rate with those using BP neural networks alone.

The BP neural network contains three layers. The number of nodes in input layer agrees with the dimension of the feature sets. The middle layer has 30 nodes and the output layer has 6 ones representing the six patterns of finger movements. The network was trained by gradient descent with momentum algorithm and adaptive learning rate back-propagation algorithm.

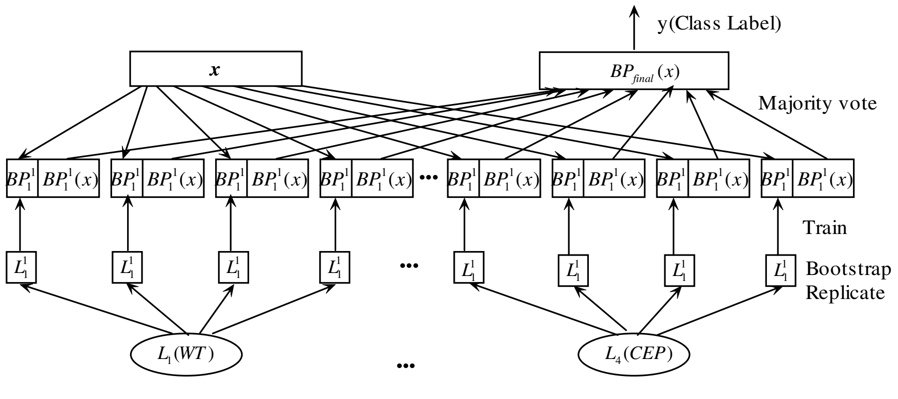


Fig. 2. The structure of the bagged-BP classifier

We created four training sets $L_k (k = 1,2,3,4)$. WT feature sets make up L_1, TD feature sets make up L_2, AR feature sets make up L_3 and CEP feature sets make up L_4. Each training set $L_k (k = 1,2,3,4)$ consists of data $\{(y_n^k, x_n^k), n = 1,2,\cdots 180\}$, where y_n^k is the class label and x_n^k is the feature vector of the n_{th} sample. Take repeated bootstrap samples $L_k^{(B)} (B = 1,2,3,4)$ from L_k, and form individual BP nets $\{BP(x, L_k^{(B)})\}$. Then let

$\{BP(x, L_k^{(B)})\}(k=1,2,3,4; B=1,2,3,4)$ vote to form $BP_{final}(x)$. This procedure is called "bootstrap aggregating" and uses the acronym bagging. The $L_k^{(B)}(B=1,2,3,4)$ forms replicate data sets, each consisting of 120 cases, drawn at random, but with replacement, from $L_k\,(k=1,2,3,4)$. Each $\{(y_n^k, x_n^k), n=1,2,\cdots 180\}$ may appear repeated times or not at all in any particular $L_k^{(B)}(B=1,2,3,4)$. Fig. 2 shows the structure of the bagged BP classifier.

3 Methodology

Data were collected from four normally limbed subjects and ID numbers were assigned according to their names, i.e. ZJD, WJZ, DXN and MTB. The discrimination results of the experiment represent the best performance that an amputee can obtain when using the same classifier.

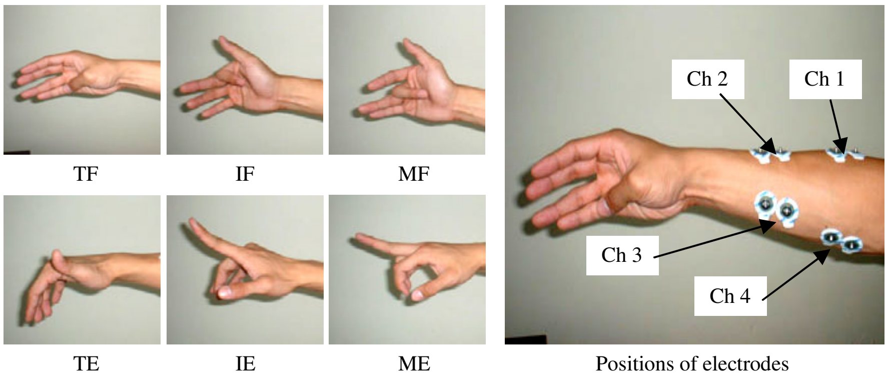


Fig. 3. Subject's six patterns of finger movements which the controller discriminates and the positions of electrodes. The selected four muscles where the electrodes placed on are most correlative with the six motions.

Each subject generated six different classes of finger movements: thumb flexion (SF)/extension (SE), index finger flexion (IF)/extension (IE), and middle finger flexion (MF)/extension (ME). Four channels of EMG signal were recorded from electrodes placed on extensor digitorum, extensor pollicis brevis, flexor carpi ulnaris and flexor digitorum, which are most involved when executing the six patterns of finger movements. Each bipolar channel was acquired using Ag-AgCl electrodes spaced at 2 cm. Each record was 3s in duration (7200 points, sampled at 2400 Hz, prefiltered between 10 and 1000 Hz). In each dataset, 100 patterns were generated in each class, resulting in a total of 600 patterns. These data were divided into training sets of 180 patterns, verification sets of 120 patterns and test sets of 300 patterns, and then subject to feature selection and classification. Fig.3 shows the movements and the positions of the electrodes.

4 Results

Each feature set extracted from the raw EMG signals of four subjects with record length of 256 points (107ms) were put into the BP network for recognition respectively. And the hybrid features extracted from the same raw EMG date were used to train the bagged BP classifier. Tables 2 to table 5 depict the discrimination results of the test data of the four subjects.

From these tables we can see that the discrimination rate of the six motions differ from each other. And for a certain motion, the results also differ using different feature set. The tables also show that the bagged-BP classifier is superior to the BP classifier.

Fig.4 and Fig.5 shows the average discrimination rate of the four subjects when using EMG data with different length. For all subjects, the classification rate improves when the data length increases. It is also obvious that different subjects have different recognition result when the length of EMG data, the method to extract feature and the structure of classifier are the same.

Table 2. The classification results of the test data of ZJD

Classifier	Feature	TF (50)	IF (50)	MF (50)	TE (50)	IE (50)	ME (50)	Accuracy(%)
BP	ar	49	49	18	49	50	50	88.3
	cep	48	50	31	48	50	48	91.7
	wt	49	45	37	49	46	48	91.3
	time	44	43	30	49	46	48	86.7
Bagged-BP	hybrid	49	50	38	50	50	50	95.7

Table 3. The classification results of the test data of WJZ

Classifier	Feature	TF (50)	IF (50)	MF (50)	TE (50)	IE (50)	ME (50)	Accuracy(%)
BP	ar	42	28	33	49	38	42	77.3
	cep	42	43	31	49	27	42	78.0
	wt	39	25	40	49	14	46	71.0
	time	37	30	40	43	19	42	70.3
Bagged-BP	hybrid	45	46	41	49	41	42	88.0

Table 4. The classification results of the test data of DXN

Classifier	Feature	TF (50)	IF (50)	MF (50)	TE (50)	IE (50)	ME (50)	Accuracy(%)
BP	ar	43	40	36	2	49	49	73.0
	cep	47	37	41	0	49	49	74.3
	wt	47	33	44	13	49	49	78.3
	time	47	13	46	41	42	45	78.0
Bagged-BP	hybrid	46	47	46	3	50	49	80.3

Table 5. The classification results of the test data of MTB

Classifier	Feature	TF (50)	IF (50)	MF (50)	TE (50)	IE (50)	ME (50)	Accuracy(%)
	ar	17	12	29	37	8	38	47.0
BP	cep	30	4	24	37	15	37	49.0
	wt	39	20	35	44	8	45	63.7
	time	25	8	31	46	25	39	58.0
Bagged-BP	hybrid	38	29	25	39	29	38	66.0

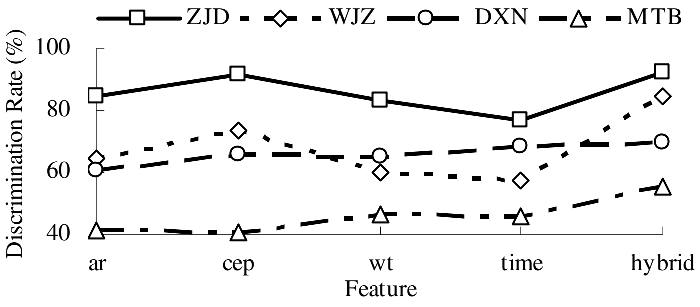


Fig. 4. The test discrimination rate of the four subjects with EMG data of length N=128

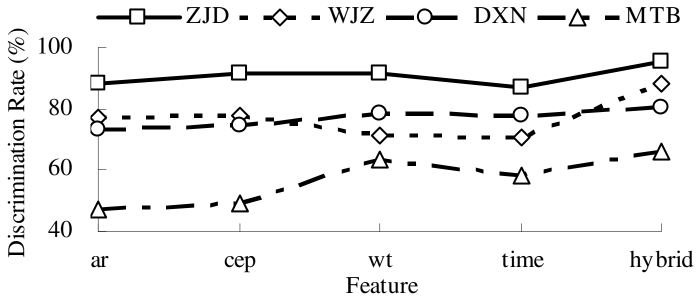


Fig. 5. The test discrimination rate of the four subjects with EMG data of length N=256

5 Discussion

In order to construct a more efficient EMG recognition system, we proposed a new classifier, i.e. bagged BP classifier. The results show that the bagged BP classifier improves accuracy and stability. Since BP neural network is unstable and easily reaches a local minimum of the error, the results of individual neural net may not be satisfying. However, these characteristics of neural network increase the variant of the individual classifier in the ensemble classifier, and bagging improves the efficiency of the weak learner by reducing the differences in the system. So bagging is an effective method to enhance the efficiency of neural network.

sEMG signals have the properties of large variations and subject's dependent. Until now there is no feature set which can give a universal optimal discrimination results among all subject [7, 8, 9, 10]. So we try to use hybrid features and bagged BP classifier to discriminate the different finger movements. The experiment show that hybrid feature improve the classification rate.

We also find that the classification rate varied a lot between subjects. It is because that some subjects act less skillfully and the low proficiency leads to the less desirable results. Therefore, we presume that the more the subjects practice the task, the more desirable results can be observed.

Acknowledgment

The authors wish to acknowledge the support of this work by the NSFC (№50435040), national Hi-Tech Research and Development (863) Program (№2001AA320601).

References

1. Dietterich, T.G.: Machine-learning research. Four current directions. AI Magazine, vol. 18. American Assoc. Artificial Intelligence (1997). 97-136
2. Shi, S. C., Gao, X. H., Jiang, L., Liu, H.: Development of the Underactuated Self-adaptive Robotic Hand. Robot, vol. 26. (2004) 496-501
3. Hudgins, B., Parker, P., Scott, R.N.: A new strategy for multifunction myoelectric control. IEEE Transactions on Biomedical Engineering, vol. 40. IEEE Engineering in Medicine and Biology Society (1993) 82-94
4. Kitagawa, G., Gersch, W.: A smoothness priors time-varying AR coefficient modeling of nonstationary covariance time series. IEEE Transactions on Automatic Control, vol. 30. IEEE Control Systems Society (1985) 48-56
5. Atal, B.S.: Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification. Journal of the Acoustical Society of America, vol. 55. American Institute of Physics Inc., Melville, United States (1974) 1304-1312
6. Breiman,L.: Bagging predictors. Machine Learning, vol. 24. Kluwer Academic Publishers (1996) 123-140
7. Hussein, S.E., Granat, B.A.: Intention detection using a neuro-fuzzy EMG classifier. IEEE Engineering in Medicine and Biology Magazine, vol.21. IEEE(2002) 123-129
8. Englehart, K. ,Hudgins, B. , Parker, P.A., Stevenson, M.: Classification of the myoelectric signal using time-frequency based representations,vol.21. Elsevier(1999)431-438
9. Khalil, M. , Duchene, J.: A dynamic approach for change detection and classification. IEEE Transactions on Biomedical Engineering.vol.47. IEEE(2000)748-756
10. Coatrieux,J.L.,Toulouse,P.,Rouvrais, B., Le Bars, R.: Automatic classification of electromyographic signals. Electroencephalography and Clinical Neurophysiology,vol.55. Elsevier(1983) 333-341

Neural-Network-Based Sliding Mode Control for Missile Electro-Hydraulic Servo Mechanism

Fei Cao, Yunfeng Liu, Xiaogang Yang, Yunhui Peng, and Dong Miao

Xi'an Research Inst. Of High-tech, Hongqing Town, 710025, China
caofeimm@163.com

Abstract. A method investigating a Gaussian radial-basis-function neural network (GRBFNN) with sliding mode control (SMC) for missile electro-hydraulic servo mechanism is presented. Since the dynamics of the system are highly nonlinear and have large extent of model uncertainties, such as big changes in parameters and external disturbance, firstly, SMC is introduced. Since the accurate equivalent control is difficult to reach, a Gaussian radial basis function neural network is utilized. By adjusting the weight on-line, a neural-network-based SMC is developed to estimate the equivalent control of SMC control system. Then the switching control is appended to guarantee the stability of the proposed controller, and a set of fuzzy control rules are used to attenuate chattering phenomenon of the switching control. We apply the control method to the missile electro-hydraulic servo mechanism. Simulation results verify the validity of the proposed approach.

1 Introduction

Electro-hydraulic servo mechanism has been frequently used in the position servo system of a missile thanks to their capability of providing large driving forces or torques, rapid response and a continuous operation [1]. However, electro-hydraulic servo mechanism inherently has many uncertainties and highly nonlinear characteristics, which results from the flow-pressure relationship, oil leakage, and etc. Furthermore, the system is subjected to load disturbances [2]. Consequently, the conventional control approaches based on a linearized model near the operating point of interest may not guarantee satisfactory control performance for the system. Since the variable structure control strategy using the sliding mode can offer many good properties, such as insensitivity to parameter variations, external disturbance rejection [3], sliding mode control (SMC) has been studied by many researchers for the control of electro-hydraulic servo system [4,5,6]. However, it is difficult to guarantee the stability of system as well as to obtain a suitable equivalent control if the nominal mathematics model is unknown in advance [4]. And SMC may suffer from the main disadvantage associated with the chattering control input due to its discontinuous switching control used to deal with the uncertainties [5, 6].

To overcome the problems, the method of neural-network-based SMC is proposed in this paper. Based on the conventional SMC, a Gaussian radial basis function neural network (GRBFNN) [7, 8] is utilized. By adjusting the weight on-line, a neural-network-based SMC is developed to estimate the equivalent control of SMC control system. The switching control is appended to guarantee the stability of the proposed controller. Then a set of fuzzy control rules is constructed to attenuate the chattering phenomenon of the switching control signal. Finally, we apply the control method to the missile electro-hydraulic servo mechanism. Simulation results show the advantages of the proposed approach.

2 Problem Statement and Design of Conventional SMC

For a kind of missile electro-hydraulic servo mechanism, which is a typical electro-hydraulic position servo system [1], Fig.1 shows a structure diagram of missile electro-hydraulic servo system.

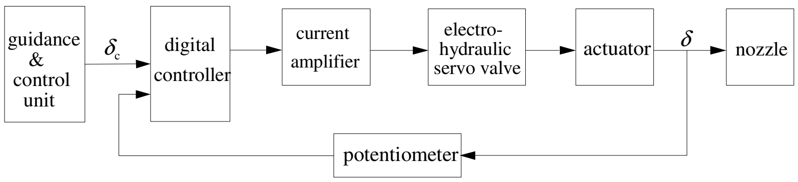


Fig. 1. Structure diagram of missile electro-hydraulic servo system

The closed loop of control system is composed of a digital controller [9], a current amplifier, an electro-hydraulic servo valve, an actuator, and a potentiometer. The objective of the control is to generate the input current such that the angular position of the nozzle is regulated to the desired position. The piston position of the actuator is controlled as follows: Once the voltage input corresponding to the position input δ_c is transmitted to the digital controller, the input current is generated in proportion to the error between the voltage input and the voltage output from the potentiometer. Then the valve spool position is controlled according to the input current applied to the torque motor of the servo valve. Depending on the spool position and the load conditions of the piston, the rate as well as the direction of the flows supplied to each cylinder chamber is determined. The motion of the piston then is controlled by these flows, and then swing angle δ of the nozzle is achieved. At the same time, the piston is influenced by an external disturbance generated from the nozzle. The whole system dynamics model is given by the following derivation equations [1]

$$\begin{aligned} &K_{ui}K_V K_Q u = ARS\delta + p_L(\frac{V_T}{4B}S + K_{ce}) \\ &ARP_L = IS^2\delta + nS\delta + K_\delta\delta + M \end{aligned} \tag{1}$$

where K_{ui} – servo amplifier gain, K_V – servo valve gain, K_Q – valve flow gain; A – pressure area in the actuator, R – effective torque arm of the linkage, V_T – effectivesystem oil volume, $K_{ce} = C_e + K_c$ (C_e – leakage coefficient of cylinder, K_c – valve pressure gain), P_L – load pressure; B – oil effective bulk modulus, n – coefficient of viscous friction, I – moment of inertia, M – load torque,

K_δ − coefficient of position torque, u − input voltage, δ − swing angle of the nozzle, S − Laplace operator.

Choose system state: $\boldsymbol{X} = [x_1 \quad x_2 \quad x_3]^T = [\delta \quad \dot{\delta} \quad \ddot{\delta}]^T$, then the system state-space equation is

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = x_3 \\ \dot{x}_3 = f(\boldsymbol{X}) + gu - d \end{cases} \tag{2}$$

where $a_1 = \dfrac{4BK_\delta K_{ce}}{IV_T}$, $a_2 = \dfrac{4B(AR)^2 + 4BK_{ce}n + K_\delta V_T}{IV_T}$, $a_3 = \dfrac{n}{I} + \dfrac{4BK_{ce}}{V_T}$,

$$f(\boldsymbol{X}) = -a_1 x_1 - a_2 x_2 - a_3 x_3, \quad g = \frac{4BAR}{IV_T} K_Q K_V K_{ui}, \quad d = \frac{4BK_{ce}}{IV_T} M + \frac{1}{I} \cdot \frac{dM}{dt}$$

The parameters a_1, a_2, a_3, g, d are all uncertainties due to the variations of K_Q, B, C_e, K_{ui} and M. It is assumed that δ_d is the desired angle, and has up to 3rd derivative. All state variables are measurable and bounded. The objective is to let the state vector $\boldsymbol{X}$ track $\boldsymbol{X_d}$ under the condition of parameter variations and external disturbances, where $\boldsymbol{X_d} = (\delta_d, \dot{\delta}_d, \ddot{\delta}_d)$. Define the tracking error $e_1 = \delta_d - x_1$, and the error vector

$$\boldsymbol{e} = [e_1 \quad e_2 \quad e_3]^T = [e_1 \quad \dot{e}_1 \quad \ddot{e}_1]^T \tag{3}$$

Thus (2) can be rewritten as

$$\begin{cases} \dot{e}_1 = e_2 \\ \dot{e}_2 = e_3 \\ \dot{e}_3 = \dddot{\delta}_d - f(\boldsymbol{e} + \boldsymbol{X_d}) - gu + d \end{cases} \tag{4}$$

In the conventional SMC design, we usually assume

$$a_i = a_{i0} + \Delta a_i \ , g = g_0 + \Delta g \tag{5}$$

where a_{i0}, g_0 is the nominal parameters of a_i and g □and Δa_i, Δg is the model uncertainty. Let $\alpha_i(t)$, $\beta(t)$ and $r(t)$ are the upper bound function of Δa_i, Δg and d respectively, i.e. $|\Delta a_i| \le \alpha_i(t)$, $|\Delta g| \le \beta(t)$, $|d| \le r(t)$. Take

$$s(\boldsymbol{e}) = c_1 e_1 + c_2 e_2 + e_3 \tag{6}$$

Then the sliding surface is

$$c_1 e_1 + c_2 e_2 + e_3 = 0 \tag{7}$$

where c_1, c_2 are constants and can be properly chosen such that all the roots of (7) are in the open left-half of the complex plane. Take the derivative of (6) and set $\dot{s}(\boldsymbol{e}) = 0$, $d = 0$, then the equivalent control can be obtained as

$$u_{eq} = g_0^{-1}(\sum_{i=1}^{3} a_{i0} x_i + c_1 e_2 + c_2 e_3 + \dddot{\delta}_d) \tag{8}$$

Based on the principle of SMC, the control law consists of two parts. One is the sliding mode equivalent control u_{eq}. Another is the switching control u_h that drives the states toward the sliding surface. From (4), (8), the control law is taken as

$$u = u_{eq} + u_h = u_{eq} + K\,\mathrm{sgn}(s) \tag{9}$$

where $K = \left[\sum_{i=1}^{3} \alpha_i(t)\left|x_i\right| + \beta(t)\left|u_{eq}\right| + r(t) + \eta\right] \Big/ \left[g_0 - \beta(t)\right]$, $\quad \mathrm{sgn}(s) = \begin{cases} 1, s > 0 \\ 0, s = 0 \\ -1, s < 0 \end{cases}$

From the analysis above, we get $s\dot{s} \le -\eta\left|s\right| < 0$, where $\eta > 0$. So under the control law (9), the sliding surface exists and is reachable. However, since the functions $f(\boldsymbol{X})$ and g are uncertain, the accurate equivalent control u_{eq} is difficult to reach. So a GRBFNN is employed to approximate u_{eq}. So the control law consists of the following two parts. One is the estimated equivalent control $\hat{u}_{eq}$ that is constructed by an adaptive mechanism. The function of this term is to force the system state to slide on the sliding surface. Another is the switching control u_h that drives the states toward the sliding surface. Thus, the control law can be represented as

$$u = \hat{u}_{eq} + u_h \tag{10}$$

3 Design of Neural-Network-Based SMC

3.1 GRBFNN Structure

The control design presented in this paper employs GRBFNN to approximate u_{eq}. GRBFNN constitutes a special class of these structures. A hidden neuron in a GRBFNN structure uses a Gaussian nonlinearity as the activation function described in (11). In this definition, i is for ordering the input vector entries, which runs up to m. j indexes the neuron order in the hidden layer. The prototype vector is comprised of the c_{ij} variables, which characterize the centers of the Gaussian functions. The variable σ_{ij} determines how the function α_{ij} spreads over the domain of its input space u_i. The output of jth neuron in the structure is evaluated through the use of (11) and is denoted by w_j.

$$\alpha_{ij}(u_i) = \exp\left\{-\left(\frac{u_i - c_{ij}}{\sigma_{ij}}\right)^2\right\}, \quad w_j = \prod_{i=1}^{m} \alpha_{ij}(u_i) \tag{11}$$

The overall output of the structure depicted is evaluated by a weighted sum of the responses of the neurons contained in the hidden layer and is described by (12), where p_j denotes the weight determining the effect of jth hidden neuron output on the overall network response.

$$y = \sum_{j=1}^{h} w_j p_j = \boldsymbol{W}^T \boldsymbol{P} \tag{12}$$

3.2 The Equivalent Control

In this section, we first construct the neural-network-based SMC, and then show how to develop an adaptive mechanism for obtaining the equivalent control through weight adaptation. Then, we construct the switching control to guarantee system's stability. In this paper, a GRBFNN is employed to approximate the equivalent control. The input variables used in the GRBFNN are integrated into two variables (s and $\dot{s}$) [10]so that the number of input variables could be minimized than those that use state variables. So we have

$$\alpha_{1j}(s)=\exp\left[-\left(\frac{s-c_{1j}}{\sigma_{1j}}\right)^2\right],\ \alpha_{2j}(\dot{s})=\exp\left[-\left(\frac{\dot{s}-c_{2j}}{\sigma_{2j}}\right)^2\right] \tag{13}$$

$$w_j(s,\dot{s})=\alpha_{1j}(s)\cdot\alpha_{2j}(\dot{s}) \tag{14}$$

In order to reduce the computational burden in the control process, the variable c_{ij} , σ_{ij} are pre-arranged according to the maximum variation of s and $\dot{s}$. The weight p_j are on-line adjusted. So the output of the GRBFNN is

$$\hat{u}_{eq}=\boldsymbol{W}^{\mathrm{T}}(s,\dot{s})\boldsymbol{P} \tag{15}$$

where $\boldsymbol{W}(s,\dot{s})=[w_1(s,\dot{s}),w_2(s,\dot{s}),\cdots,w_N(s,\dot{s})]^{\mathrm{T}}$, $\boldsymbol{P}=[p_1,p_2,\cdots,p_N]^{\mathrm{T}}$

The main task of this section is to derive an adaptive law to adjust the weight parameter vector $\boldsymbol{P}$ such that the estimated equivalent control $\hat{u}_{eq}$ can be optimally approximated to the equivalent control of the SMC. Suppose there exists the optimal parameter vector $\boldsymbol{P}^*=[p_1^*,p_2^*,\cdots,p_N^*]^{\mathrm{T}}$ such that $\hat{u}_{eq}$ has minimum approximation error $\varepsilon(s,\dot{s})=u_{eq}^*(s,\dot{s})-u_{eq}(s,\dot{s})$. Thus,

$$u_{eq}^*(s,\dot{s})-u_{eq}(s,\dot{s})=\boldsymbol{W}^{\mathrm{T}}(s,\dot{s})(\boldsymbol{P}^*-\boldsymbol{P}) \tag{16}$$

Define a Lyapunov function candidate:

$$V=(1/2)s^2+(1/2r)g\tilde{\boldsymbol{P}}^{\mathrm{T}}\tilde{\boldsymbol{P}} \tag{17}$$

where $\tilde{\boldsymbol{P}}=\boldsymbol{P}-\boldsymbol{P}^*$ and r is a positive constant. Differentiating (17), we can have

$$\dot{V}=s\dot{s}+(1/r)g\dot{\tilde{\boldsymbol{P}}}^{\mathrm{T}}\tilde{\boldsymbol{P}} \tag{18}$$

Differentiating (6) and substituting (4) into (18), then

$$\dot{V}=s\{c_1e_2+c_2e_3+\dddot{\delta}_d-f(\boldsymbol{X})-g[\hat{u}_{eq}+u_h]+d\}+\frac{1}{r}g\dot{\tilde{\boldsymbol{P}}}^{\mathrm{T}}\tilde{\boldsymbol{P}} \tag{19}$$

Since (8), (19) can be written as

$$\dot{V}=s[-g(\hat{u}_{eq}-u_{eq}+u_h)]+\frac{1}{r}g\dot{\tilde{\boldsymbol{P}}}^{\mathrm{T}}\tilde{\boldsymbol{P}} \tag{20}$$

Substituting (16) into (20), then

$$\begin{aligned}\dot{V}&=s\{-g[\boldsymbol{W}^{\mathrm{T}}(s,\dot{s})\boldsymbol{P}-\boldsymbol{W}^{\mathrm{T}}(s,\dot{s})\boldsymbol{P}^*+\boldsymbol{W}^{\mathrm{T}}(s,\dot{s})\boldsymbol{P}^*-u_{eq}+u_h]\}+1/r\cdot g\dot{\tilde{\boldsymbol{P}}}^{\mathrm{T}}\tilde{\boldsymbol{P}}\\&=s\{-g[\varepsilon(s,\dot{s})+u_h]\}-sg\boldsymbol{W}^{\mathrm{T}}(s,\dot{s})\tilde{\boldsymbol{P}}+\frac{1}{r}g\dot{\tilde{\boldsymbol{P}}}^{\mathrm{T}}\tilde{\boldsymbol{P}}\end{aligned} \tag{21}$$

Choosing the adaptive law

$$\dot{\boldsymbol{P}} = rs\boldsymbol{W}(s,\dot{s}) \tag{22}$$

Since (22), (21) becomes

$$\dot{V} = -gs\varepsilon(s,\dot{s}) - gsu_h \le 0 \tag{23}$$

u_h has the same sign as gs (refer to (26)). So In order to complete the neural-network-based SMC design, the switching control should be taken into account to ensure state trajectory moves toward the sliding surface as well as to guarantee the stability of the control system. So a Lyapunov function candidate is given as

$$V = 0.5s^2 \tag{24}$$

Then differentiate V with respect to time. Substituting (4) and (6) into (24)

$$\begin{aligned} \dot{V} &= s\{c_1e_2 + c_2e_3 + \dddot{\delta}_d - f(\boldsymbol{X}) - g[\hat{u}_{eq} + u_h] + d\} \\ &\le |s|\{|c_1e_2 + c_2e_3 + \dddot{\delta}_d| + |f(\boldsymbol{X})| + g|\hat{u}_{eq}| + |d|\} - sgu_h \end{aligned} \tag{25}$$

Choosing the switching control

$$u_h = \mathrm{sgn}(s)K \tag{26}$$

where $K \ge g_l^{-1}(|c_1e_2 + c_2e_3 + \dddot{\delta}_d| + f_{\max} + D) + |\hat{u}_{eq}|$. Thus, $\dot{V} \le 0$. i.e., the switching control actually achieves a stable control system.

3.3 The Fuzzy Control Law

From (26) it can be seen that the undesirable control input chattering is caused by the discontinuous sign term $\mathrm{sgn}(s)$. The switching control law u_h which guarantees the reachability and existence of the sliding mode is proportional to the uncertainty bound including $f_{\max}$ and D. However, the exact value of the parameter variations and the external load disturbance are difficult to know in advance for practical applications. Therefore, usually a conservative control law with large control gain K is selected. However, it will cause a large amount of chattering phenomenon, which causes high-frequency unmodelled dynamics [11]. Therefore, a fuzzy control law is proposed here, in which a fuzzy inference mechanism is used to facilitate switching control adjustment for minimizing the chattering. The input of the fuzzy controller is s, and the output is u_f to replace the discontinuous sliding switch control u_h. The following rule base is proposed: (1) if s is positive large (PL) , then u_f is negative large(NL); (2)if s is positive small(PS), then u_f is negative small (NS); (3) if s is negative large(NL), then u_f is positive large(PL); (4)if s is negative small(NS), then u_f is positive small(PS).

Choosing sigmoidal membership functions for s and singletons for u_f, we have

$$\mu_{s\_\mathrm{PL}} = \tanh(\frac{s}{\sigma_s}),\ \mu_{s\_\mathrm{PS}} = 1 - \mu_{s\_\mathrm{PL}},\ \mu_{s\_\mathrm{NL}} = -\tanh(\frac{s}{\sigma_s}),\ \mu_{s\_\mathrm{NS}} = 1 - \mu_{s\_\mathrm{NL}} \tag{27}$$

$$\mu_{u_{f\_\mathrm{NL}}} = \begin{cases} 1 & u_f = -\gamma_m \\ 0 & u_f \neq -\gamma_m \end{cases}, \mu_{u_{f\_\mathrm{PL}}} = \begin{cases} 1 & u_f = \gamma_m \\ 0 & u_f \neq \gamma_m \end{cases}, \mu_{u_{f\_\mathrm{NS}}} = \mu_{u_{f\_\mathrm{PS}}} = \begin{cases} 1 & u_f = 0 \\ 0 & \mu_f \neq 0 \end{cases} \tag{28}$$

and using rules (1)-(4), we obtain

$$u_f = \begin{cases} \dfrac{\mu_{s\_\mathrm{PL}} \cdot (-\gamma_m) + \mu_{s\_\mathrm{PS}} \cdot 0}{\mu_{s\_\mathrm{PL}} + \mu_{s\_\mathrm{PS}}} & s > 0 \\ \dfrac{\mu_{s\_\mathrm{NL}} \cdot (\gamma_m) + \mu_{s\_\mathrm{NS}} \cdot 0}{\mu_{s\_\mathrm{NL}} + \mu_{s\_\mathrm{NS}}} & s < 0 \end{cases} \tag{29}$$

In summary, the fuzzy control u_f can be written as follows:

$$u_f = -\gamma_m \tanh(s / \sigma_s) \tag{30}$$

where $\gamma_m = K$, σ_s is a designed parameter.

As discussed above, a GRBFNN has been developed to estimate the equivalent control of SMC system. The adaptive law is applied to adjust the weight parameter vector $\boldsymbol{P}$. In addition, the switching control has also been derived to ensure the requirement of system stability. Finally, four constructed fuzzy control rules are employed to smooth the control law based on the concepts of SMC.

4 Simulation Results and Discussion

For a missile electro-hydraulic servo mechanism (1), the nominal value [1] of some parameters are assumed as $k_{ui} = 5mA/V$, $K_Q = 12cm^3/(s \cdot mA)$, $A = 10cm^2$, $R = 17cm$. Substituting the values into (**2**), we get $a_{10} = 0$, $a_{20} = 8873.64$, $a_{30} = 37.68$, $g_0 = 179425$, $d = 0.86\dot{M} + 9.73M$, where $M = M_{f0} Sgn\dot{\delta} + M_d$, M_{f0} is frictional torque amplitude, M_d is position torque. Assume $\Delta g = 0.2\sin(2\pi t) g_0$, so $g_l = 0.8 \times g_0$; $\Delta a_i = 0.5\sin(2\pi t) a_{i0}$, so $f_{\max} = |1.5 \times a_{20} x_2 + 1.5 \times a_{30} x_3|$, $M_d = 500 + 100\sin 2\pi t$, $M_{f0} = 3000 + 1000\sin 2\pi t$.

The simulation condition and design parameters of neural-network-based SMC are specified as follows:(a) initial values of system state variables $\boldsymbol{X}(0) = [2 \quad 0 \quad 0]^T$; (b) sampling time interval $t = 0.001s$; (c) Desired output $\delta_d(t) = 2\sin 2\pi t$; (d) Choose the poles of the system as described by (5) at $-60, -60$, we can obtain $c_1 = 3600$, $c_2 = 120$; (e) The center of the j^{th} kernel of the neural network for the normalized state(i.e., $s/3000$, $\dot{s}/3000000$) is $c_{ij} \in [-1 \quad -0.5 \quad 0 \quad 0.5 \quad 1]$, the width of nodes for the neural network is $\sigma_{ij} = 0.3$, $i = 1,2$, $j = 1, \cdots, 25$. (f) Parameter in the adaptive law is selected as $r = 0.5$; (g) $\sigma_s = 500$; (h) The initial weight $p_j = [0, 0, \cdots, 0]^T$.

We do simulation research and compare results with that of conventional SMC under the same condition of parameter variations and external disturbances. Simulation results are indicated in Fig. 2 – Fig. 5. Fig. 2 shows the tracking response of the system. Fig. 3 shows the tracking error of the system. Fig. 4 and Fig. 5 show the control input where the controllers are taken as the neural-network-based SMC and conventional SMC.

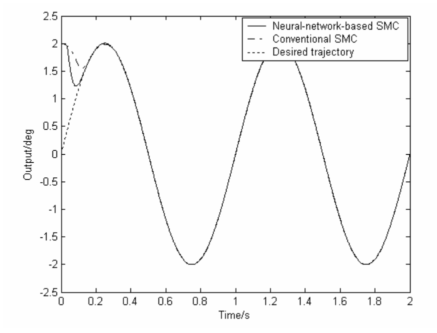


Fig. 2. Tracking response of system

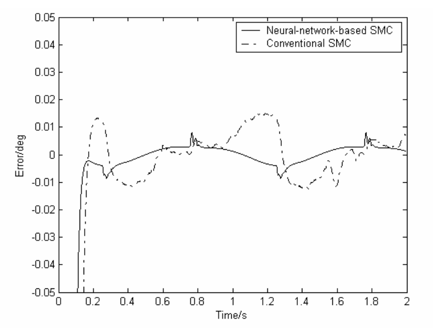


Fig. 3. Tracking error of system

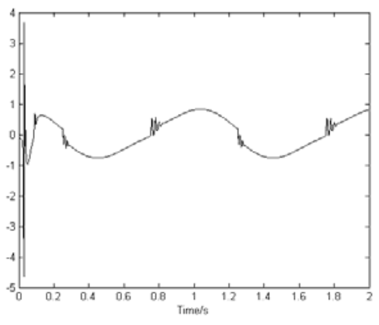


Fig. 4. Control with neural-network-based SMC

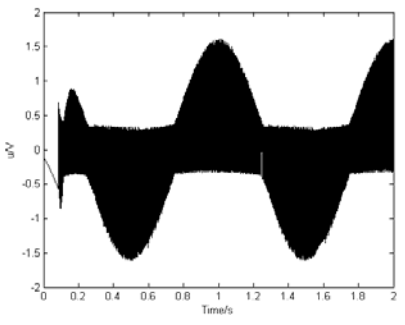


Fig. 5. Control with Conventional SMC

Simulation analysis: From the simulation results, we can conclude that: 1) If the controller is the conventional SMC, the tracking error is small and there are serious high frequency chattering in the control input due to the sign function in the switching control. 2) If the controller is the neural-network-based SMC, chattering phenomenon is attenuated, the control input is smooth and the strength of the control signal can also be significantly reduced. The transient deviation of control input, which is depicted in Fig.4, is induced owing to the random initialization of the weight of the GRBFNN especially under the occurrence of uncertainties. The tracking error is smaller than that with conventional SMC because adjusting weight can effectively deal with the parametric uncertainty and external disturbances of the system. Good tracking result is obtained. It is robust to the uncertainties and the external disturbance in the missile electro-hydraulic servo mechanism.

5 Conclusions

In this paper, a neural-network-based SMC method is applied to control missile electro-hydraulic servo mechanism. The input variables used in the controller are integrated into two variables (s and $\dot{s}$) so that the number of control input could be minimized than those that use state variables. By adjusting the weight on-line, a GRBFNN is developed to estimate the equivalent control of SMC control system. The system is stable in the sense of Lyapunov under a given common Lyapunov function. In general, the switching control depends on the bounds of system and it is usually chosen to be large to ensure the stability of the fuzzy control system, which induces chattering phenomenon. Therefore, four constructed fuzzy control rules are employed to smooth the control law based on the concepts of SMC. Simulation results from the missile electro-hydraulic servo mechanism are given to demonstrate the applicability and the effectiveness of the proposed approach.

References

1. Zhu, Z. H.: Thrust Vector Control Servo System. Beijing, Astronautics press (1995)
2. Wang, Z. L.: Control on Modern Electrical and Hydraulic Servo. Beijing, Beijing University of Aeronautics and Astronautics press (2004)
3. Hung, J. Y., Gao, W. B., and Hung, J. C.: Variable Structure Control: A Survey. IEEE Trans. Ind. Electron., Vol.40, No.2 (1993) 2-22
4. Duan, S. L., An, G. C.: Adaptive Sliding Mode Control for Electro-hydraulic Servo Force Control Systems. Chinese Journal of Mechanical Engineering, Vol.38. No.5 (2002) 109-113
5. Mohamed, A. G.: Variable Structure Control for Electro-hydraulic Position Servo System. The 27th Annual Conference of the IEEE Industrial Electronics Society,(2001) 2195-2198
6. Liu, Y. F.: Research on Variable Structure Robust Control for Electro-hydraulic Servo System. Journal of Second Artillery Engineering Institute, Vol.19, No.4(2005) 12-14
7. Efe, M. O., Kaynak, O.: Sliding Mode Control of Nonlinear System Using Gaussian Radial Basis Function Neural Networks. 0-7803-7044-9/01, IEEE, (2001) 474-479
8. Zha, X., Cui, P.Y.: Sliding Mode Control for Uncertain Nonlinear Systems Using RBF Neural Networks. Lecture Notes in Computer Science, Springer Berlin-Heidelberg, Vol.3498, (2005) 21-29
9. Liu, Y. F., Miao, D.: 1553B BUS and Its Application in Electro-hydraulic Servo System. Machine Tool & Hydraulics, Vol.38. No.9 (2004) 106-108
10. Guan, P., Liu, X. J., Liu, J. Z.: Adaptive Fuzzy Sliding Mode Control for Flexible Satellite. Engineering Application of Artificial Intelligence, Vol. 18, No.3 (2005) 451-459
11. Raksoglu, K. A., Sundreeshan, M. K.: A Recurrent Neural Network-based Adaptive Variable Structure Model Following Control of Robotic Manipulators. Automatica, Vol.31, (1995) 1495-1507

Turbulence Encountered Landing Control Using Hybrid Intelligent System

Jih-Gau Juang[1] and Hou-Kai Chiou

Department of Communications and Guidance Engineering
National Taiwan Ocean University
Keelung City, Taiwan, ROC
jgjuang@mail.ntou.edu.tw[1]

Abstract. During a flight, take-off and landing are the most difficult operations in regard to safety issues. Aircraft pilots must not only be acquainted with the operation of instrument boards but also need flight sensitivity to the ever-changing environment, especially in the landing phase when turbulence is encountered. If the flight conditions are beyond the preset envelope, the automatic landing system (ALS) is disabled and the pilot takes over. An inexperienced pilot may not be able to guide the aircraft to a safe landing at the airport. This paper proposes an intelligent aircraft automatic landing controller that uses recurrent neural network (RNN) controller with genetic algorithm (GA) to improve the performance of conventional ALS and guide the aircraft to a safe landing.

1 Introduction

China Airlines Flight 642 had a hard landing at Hong Kong International Airport on 22 August 1999. The lifting wing was broken during the impact, which killed 3 passengers and injured 211 people. After 15 months of investigation, a crash report was released on November 30, 2000. It showed that the crosswind-correction software 907 on the Boeing MD-11 had a defect. Boeing also confirmed this software problem later and replaced nearly 190 MD-11 crosswind-correction software programs with the 908 version. Although this accident was attributed to bad weather and software problem, the ALS was focus of the safety issue. The first ALS was made in England during 1965. Since then, most aircraft have had this system installed. The ALS relies on the Instrument Landing System (ILS) to guide the aircraft into the proper altitude, position, and approach angle during the landing phase. According to Federal Aviation Administration (FAA) regulations, environmental conditions considered the determination of dispersion limits as being: headwinds up to 25 knots, tailwinds up to 10 knots, crosswinds up to 15 knots, moderate turbulence, and wind shear of 8 knots per 100 feet from 200 feet to touchdown [1]. If the flight conditions are beyond the preset envelope, the ALS is disabled and the pilot takes over. An inexperienced pilot may not be able to guide the aircraft to a safe landing at the airport. It is therefore desirable to develop an intelligent ALS that expands the operational envelope to include safer responses under a wider range of conditions. The goal of this paper is to show that the proposed intelligent ALS can relieve human operators and guide the aircraft to a safe landing in a wind-turbulence environment.

Most of the improvements in the ALS system have been on the guidance instruments, such as GNSS Integrity Beacons, Global Positioning System, Microwave Landing System, and Autoland Position Sensor [2]-[5]. By using improvement calculation methods and high accuracy instruments, these systems provide more accurate flight data to the ALS and can help to make landings smoother. However, these researches do not include weather factors such as wind turbulence. Recently, some researchers have applied some intelligent concepts such as neural networks, fuzzy system, particle swarm optimization, GA, and hybrid systems to flight control to increase the flight controller's adaptively to different environments [6]-[10]. Among these intelligent concepts, neural network techniques are used most because of their adaptively and robustness for unmodeled system and the hardware implementation capability. This paper proposes an intelligent aircraft automatic landing controller that uses a RNN controller with GA to improve the performance of conventional ALS. This controller can guide the aircraft to a safe landing and make the controller more robust and adaptive to the ever-changing environment.

2 Aircraft Landing Analysis

In a normal landing process, the pilot descends from the cruising altitude to an altitude of approximately 1200ft. The pilot then positions the aircraft so that the aircraft is on a heading towards the runway centerline. When the aircraft approaches the outer airport marker, which is about 4 nautical miles from the runway, the glide path signal is intercepted (as shown in Fig. 1). As the aircraft descends along the glide path its pitch, attitude, and speed must be controlled. The aircraft maintains a constant speed along the flight path. The descent rate is about 10ft/sec and the pitch angle is between -5 to +5 degrees. Finally, as the aircraft descends to 20 to 70 feet the glide path control system is disengaged and a flare maneuver is executed. The vertical descent rate is decreased to 2ft/sec so that the landing gear may be able to dissipate the energy of the impact at landing. The pitch angle of the aircraft is then adjusted to between 0 to 5 degrees for most aircraft, which allows a soft touchdown on the runway surface.

A simplified model of a commercial aircraft that moves only in the longitudinal and vertical plane is used in the simulations for implementation ease [9]. The motion equations of the aircraft are given as follows:

$$\dot{u} = X_u(u-u_g) + X_w(w-w_g) + X_q q - g\left(\frac{\pi}{180}\right)\cos(\gamma_0)\theta + X_E\delta_E + X_T\delta_T \quad (1)$$

$$\dot{w} = Z_u(u-u_g) + Z_w(w-w_g) + \left(Z_q - \frac{\pi}{180}U_0\right)q + g\left(\frac{\pi}{180}\right)\sin(\gamma_0)\theta + Z_E\delta_E + Z_T\delta_T \quad (2)$$

$$\dot{q} = M_u(u-u_g) + M_w(w-w_g) + M_q q + M_E\delta_E + M_T\delta_T \quad (3)$$

$$\dot{\theta} = q \quad (4)$$

$$\dot{h} = -w + \frac{\pi}{180}U_0\theta \quad (5)$$

where u is the aircraft longitudinal velocity (ft/sec), w is the aircraft vertical velocity (ft/sec), q is the pitch rate (deg/sec), θ is the pitch angle (deg), h is the aircraft altitude (ft), δ_E is the incremental elevator angle (deg), δ_T is the throttle setting (ft/sec), γ_o is the flight path angle (-3deg), and g is the gravity (32.2 ft/sec²). The parameters X_i, Z_i and M_i are the stability and control derivatives.

To make the ALS more intelligent, reliable wind profiles are necessary. Two spectral turbulence forms modeled by von Karman and Dryden are mostly used for aircraft response studies. In this study the Dryden form [9] was used for its demonstration ease. The model is given by

$$u_g = u_{gc} + N(0,1)\sqrt{\frac{1}{\Delta t}}\left(\frac{\sigma_u\sqrt{2a_u}}{s+a_u}\right) \tag{6}$$

$$w_g = N(0,1)\sqrt{\frac{1}{\Delta t}}\left(\frac{\sigma_w\sqrt{3a_w}(s+b_w)}{(s+a_u)^2}\right) \tag{7}$$

where

$$u_{gc} = -u_{wind510}[1+\frac{\ln(h/510)}{\ln(51)}],\ \sigma_u = 0.2|u_{gc}|,$$

$$L_w = h,\ a_u = \frac{U_o}{L_u},\ a_w = \frac{U_o}{L_w},\ b_w = \frac{U_o}{L_w\sqrt{3}},$$

$$L_u = 100h^{1/3} \text{ for } h > 230,\ L_u = 600 \text{ for } h \le 230,$$

$$\sigma_w = 0.2|u_{gc}|(0.5+0.00098\times h) \text{ for } 0 \le h \le 500,\ \sigma_w = 0.2|u_{gc}| \text{ for } h > 500.$$

The parameters are u_g is the longitudinal wind velocity (ft/sec), w_g is the vertical wind velocity (ft/sec), U_0 is the nominal aircraft speed (ft/sec), $u_{wind510}$ is the wind speed at 510 ft altitude, L_u and L_w are scale lengths (ft), σ_u and σ_w are RMS values of turbulence velocity (ft/sec), Δt is the simulation time step (sec), *N(0,1)* is the Gaussian white noise with zero mean and unity standards deviation, u_{gc} is the constant component of u_g, and h is the aircraft altitude (ft). Fig. 2 shows a turbulence profile with a wind speed of 30 ft/sec at 510 ft altitude.

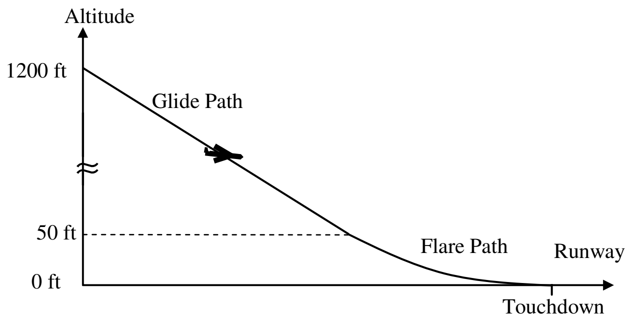


Fig. 1. Glide path and flare path

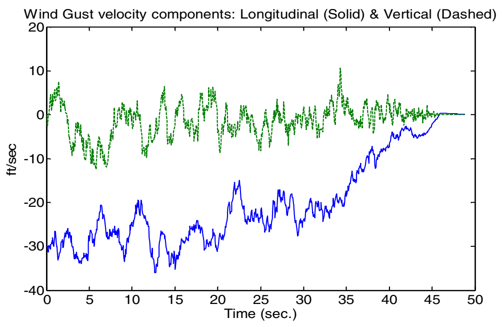


Fig. 2. Turbulence profile

3 Aircraft Landing Control

A simplified structure of aircraft landing controller, which is a PID type controller, is shown in Fig. 3. Its inputs consist of altitude and altitude rate commands along with aircraft altitude and altitude rate. Via PID controller we can obtain the pitch command θ_c. The pitch autopilot is then controlled by pitch command as shown in Fig. 4. Detail

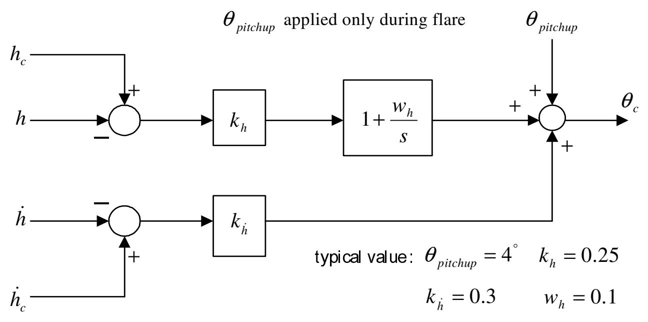
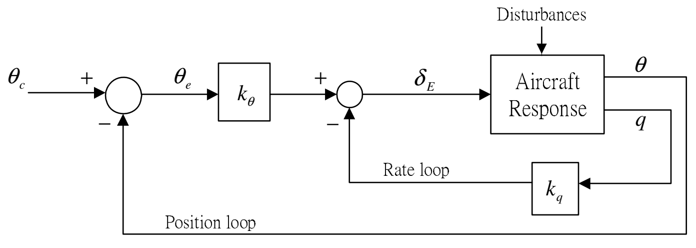

$\theta_{pitchup}$ applied only during flare
$\theta_{pitchup}$
h_c
h
$\dot{h}$
$\dot{h}_c$
k_h
$1+\frac{w_h}{s}$
$k_{\dot{h}}$
θ_c
typical value: $\theta_{pitchup} = 4^\circ$ $k_h = 0.25$
$k_{\dot{h}} = 0.3$ $w_h = 0.1$

Fig. 3. PID-Controller



typical value-Glide Slope : $k_\theta = 2.8$ Flare : $k_\theta = 11.5$

$k_q = 2.8$ $k_q = 6.0$

Fig. 4. Pitch Autopilot

descriptions can be found in [9]. In order to enable aircraft to land more steady when aircraft arrives to the flare path, a constant pitch angle will be added to the controller.

Genetic Algorithm (GA) is a global search method [11], which can search best population by using evolutionary computation. Therefore, it has been widely used to solve optimal problem recently. It can search many points at the same time, so not apt to fall into local optimal solution. The control parameters of the pitch autopilot were derived by traditional control theory, which can not bear various wind turbulence. When the wind turbulence is too strong, the ALS can not control aircraft to land safely. In here we reselected the aircraft's control parameters by using a hybrid RNN-GA control scheme which improved the ability of turbulence resistance of the ALS.

Williams and Zipser [12] proposed a real-time recurrent learning (RTRL) algorithm in 1989, which is used to train RNN (as shown in Fig. 5). RNN is used to approximate and control a nonlinear system through an on-line learning process [13]-[14], and the weights of the networks need to be updated using a dynamical learning algorithm during the control process. In the training schemes of RNN, most algorithms were developed in accordance with different problems. The learning algorithms adjust the weights of the network, so that the trained network will exhibit the required properties. The most general algorithms used for RNN are usually the gradient descent learning algorithms. By evaluating the gradient of cost function with respect to the weights of the networks, it is possible to iteratively adjust the value of the weights in the direction opposite to the gradient. It is well known that there are feedback paths or recurrent links in RNN, *i.e.*, the current output depends upon the past outputs of the network. Furthermore, the level of error depends not only on the current parameter set, but also on the past parameter set. Obviously, it is necessary to consider these dependencies in the learning schemes. In the course of learning, the adjustment of the relevant parameters of the network will be updated by sequential mode; therefore, this can be applied to on-line learning. The characteristics of the RNN are fast learning, short execution time, fast convergence, and the network relatively stabilizes with plasticity.

Fig. 5 shows the structure of the RNN. The inputs for the RNN controller are: altitude, altitude command, altitude rate, and altitude rate command. The output of the controller is the pitch command. The activation of the hidden layers is sigmoid function, which can be used in nonlinear space transformation. Weights updating rules are given as follows.

$$net_j(k) = \sum_{j=1}^{n} w_{ij}^{x} \cdot x_j^{c}(k) + w_i^{u} \cdot u(k) \tag{8}$$

$$x_i(k) = f(net_j(k)) \tag{9}$$

$$x_j^{c}(k) = x_j(k-1) \tag{10}$$

$$net_i(k) = \sum_{i=1}^{n} w_i^{y} \cdot x_i(k) \tag{11}$$

$$y(k) = f(net_i(k)) \tag{12}$$

$$\Delta w_i^{y} = \eta \cdot \left[y_d(k) - y(k) \right] \cdot \frac{\partial y(k)}{\partial net_i(k)} \cdot x_i(k) \tag{13}$$

$$\Delta w_i^{u} = \eta \cdot \left[y_d(k) - y(k) \right] \cdot \frac{\partial y(k)}{\partial net_i(k)} \cdot w_i^{y}(k) \cdot \frac{\partial x_i(k)}{\partial net_j(k)} \cdot u(k) \tag{14}$$

$$\Delta w_{ij}^{x} = \eta \cdot \left[y_d(k) - y(k) \right] \cdot \frac{\partial y(k)}{\partial net_i(k)} \cdot w_i^{y}(k) \cdot \frac{\partial x_i(k)}{\partial net_j(k)} \cdot x_j(k-1) \tag{15}$$

$$\frac{\partial y(k)}{\partial net_i(k)} = y(k) \cdot [1 - y(k)] \tag{16}$$

$$\frac{\partial x_i(k)}{\partial net_j(k)} = x_i(k) \cdot [1 - x_i(k)] \tag{17}$$

where y_d is the desired pitch command, η is the learning rate, other parameters are defined in Fig. 5.

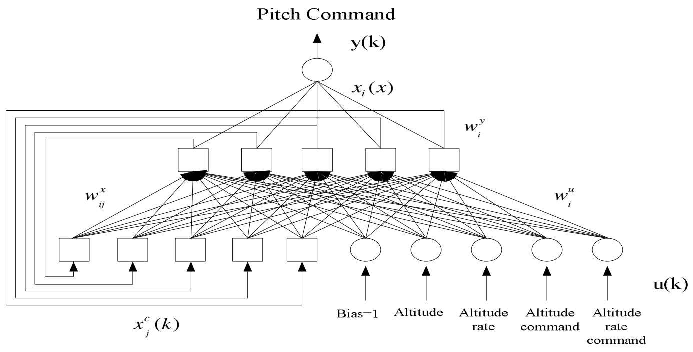


Fig. 5. RNN Structure

Fig. 6 shows the RNN learning structure. The four inputs of the aircraft are altitude, altitude command, altitude rate, and altitude rate command which will input to PI-controller and RNN at the same time. The output of the PID-controller is the expected pitch command, and the output of the network is the pitch command after learning. The network will amend itself through the error between these two commands. After training, the RNN controller will replace PID controller as the pitch autopilot. Feedforward and feedback control gains of the pitch autopilot are selected by GA with different strength of turbulence. The wind turbulence strength increases progressively during the process of parameter search. The purpose of this procedure is to search more suitable control gains for *kθ* and *kq* in glide path and flare path. The control parameters, *kθ* and *kq*, of the glide and flare paths are the chromosomes that need to be searched. The design of fitness function is to consider different intensity of the wind turbulence as fitness function values. This method makes the aircraft adapt itself to wind turbulence with a wider range. Fig. 7 shows the flow chart of the RNN-GA learning process. In GA search, the evolutionary computations are shown below.

(1) Reproduction: Roulette wheel selection [11]

(2) Crossover: modified Adewuya method [15]

Step1: Randomly choose a gene from each individual of a matching pair in parent generation, $p_{m\alpha}$ and $p_{n\alpha}$, as crossover positions.

$$pattern_1 = \left[p_{m1} \quad p_{m2} \quad ..... \quad p_{m\alpha} \quad ..... \quad p_{ms} \right]$$
$$pattern_2 = \left[p_{n1} \quad p_{n2} \quad ..... \quad p_{n\alpha} \quad ..... \quad p_{ns} \right]$$

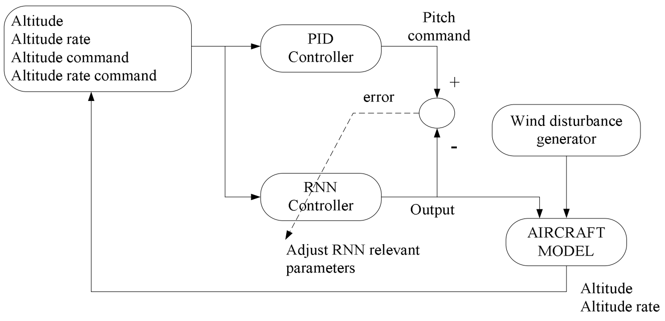


Fig. 6. The RNN learning structure

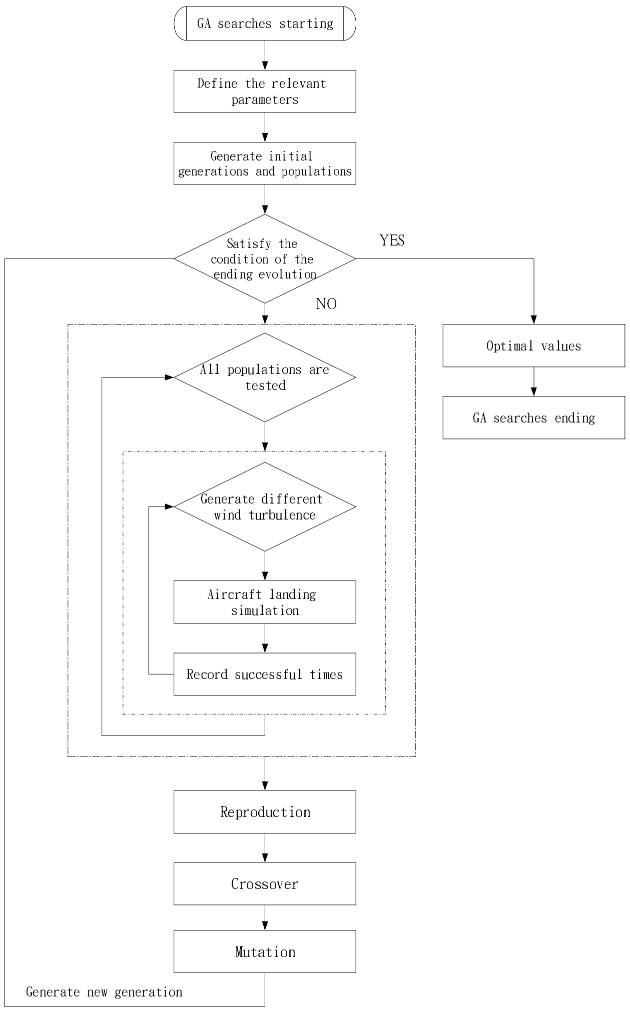

GA searches starting
Define the relevant parameters
Generate initial generations and populations
Satisfy the condition of the ending evolution
YES
NO
Optimal values
GA searches ending
All populations are tested
Generate different wind turbulence
Aircraft landing simulation
Record successful times
Reproduction
Crossover
Mutation
Generate new generation

Fig. 7. Learning process of the RNN-GA scheme

Step2: Calculate new values of these selected genes as follows, where β is a random number and $0 \le \beta \le 1$.

$$p_{new1} = (1-\beta)\cdot p_{m\alpha} + \beta \cdot p_{n\alpha} \tag{18}$$

$$p_{new2} = \beta \cdot p_{m\alpha} + (1-\beta)\cdot p_{n\alpha} \tag{19}$$

Step3: Replace $p_{m\alpha}$ and $p_{n\alpha}$ with p_{new1} and p_{new2}, respectively, and the genes in the right side of the crossover position exchange with each other.

$$Newpattern_1 = \begin{bmatrix} p_{m1} & p_{m2} & \cdots\cdots & p_{new1} & \cdots\cdots & p_{ns} \end{bmatrix}$$

$$Newpattern_2 = \begin{bmatrix} p_{n1} & p_{n2} & \cdots\cdots & p_{new2} & \cdots\cdots & p_{ms} \end{bmatrix}$$

(3) Mutation: The mutation operator creates one new offspring individual for the new population by randomly mutating a randomly chosen gene of the selected individual.

$$x_{new} = x_{old} + \gamma \cdot random\_noise \tag{20}$$

where γ is a real number within [0-1].

The fitness function is:

Fitness = number of successful landing with different turbulence strengths. (21)

4 Simulation Results

Suppose that the aircraft starts the initial states of the ALS as follows: the flight height is 500 ft, the horizontal position before touching the ground is 9240 ft, the flight angle is -3 degrees, and the speed of the aircraft is 234.7 ft/sec. Successful touchdown landing conditions are defined as follows:

(1) $-3 \le \dot{h}_{TD} \le -1$ (ft/sec) (2) $-300 \le x_{TD}(T) \le 1000$ (ft)

(3) $200 \le V_{TD}(T) \le 270$ (ft/sec) (4) $-10 \le \theta_{TD}(T) \le 5$ (degrees)

where T is the time at touchdown, $\dot{h}_{TD}$ is vertical speed, x_{TD} is the horizontal position, V_{TD} is the horizontal speed, and θ_{TD} is the pitch angle. The simulation results are divided into two parts: the RNN with original control gains and the RNN-GA with optimal control gains. These parts are described as follows.

I. RNN with original control gains

For the safe landing of an aircraft using a RNN controller with original control gains from [9] and the wind turbulence speed at 40 ft/sec, the horizontal position at touchdown is 926.1 ft, horizontal velocity is 234.7 ft/sec, vertical speed is -1.9 ft/sec, and pitch angle is 0.2 degrees, as shown in Fig. 8 to Fig. 10. Table 1 shows the results from using different wind turbulence speeds. The RNN controller can successfully guide the aircraft flying through wind speeds of 0 ft/sec to 50 ft/sec.

II. RNN-GA with optimal control gains

By utilizing the combination of RNN and GA, for the safe landing of an aircraft using the RNN controller with optimal control gains and the wind turbulence speed at 70 ft/sec, the horizontal position at touchdown is 973.0 ft, horizontal velocity is 234.7

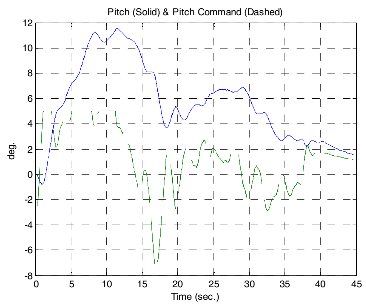


Fig. 8. Aircraft pitch angle and command

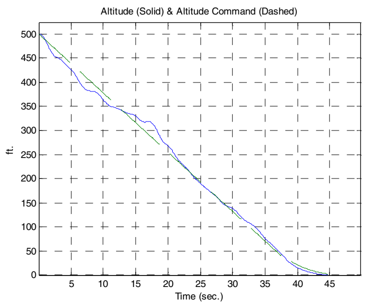


Fig. 9. Aircraft altitude and command

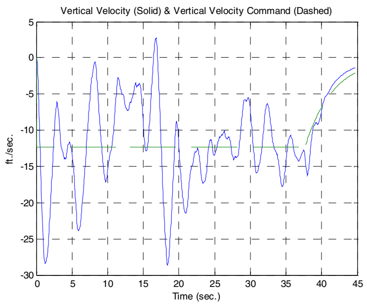


Fig. 10. Aircraft vertical velocity and command

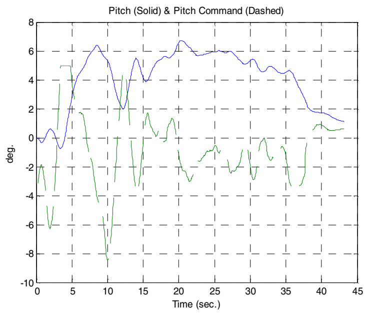


Fig. 11. Aircraft pitch angle and command

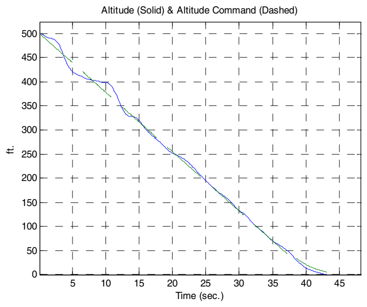


Fig. 12. Aircraft altitude and command

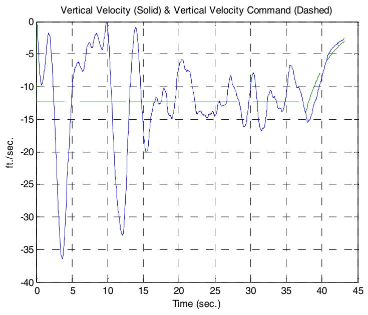


Fig. 13. Aircraft vertical velocity and command

Table 1. The results of RNN from using different turbulence speeds

Wind speed (ft/sec)	10	20	30	40	50
Landing point (ft)	879	867	914	926	949
Aircraft vertical speed (ft/sec)	-2.5	-2.2	-2.1	-1.9	-1.7
Pitch angle(degrees)	-0.9	-0.8	-0.2	0.2	0.6

Table 2. The results of RNN-GA from using different turbulence speeds

Wind speed (ft/sec)	30	50	70	80	90
Landing point (ft)	832	902	973	920	245
Aircraft vertical speed (ft/sec)	-2.3	-2.0	-1.8	-2.0	-1.1
Pitch angle (degrees)	-0.3	0.4	0.9	3.0	3.1

ft/sec, vertical speed is -1.8 ft/sec, and pitch angle is 0.9 degrees, as shown in Fig. 11 to Fig. 13. Table 2 shows the results from using different wind turbulence speeds. The RNN-GA controller can successfully guide the aircraft flying through wind speeds of 0 ft/sec to 90 ft/sec.

5 Conclusion

The automatic landing system of an aircraft is enabled only under limited conditions. If severe wind turbulences are encountered, the pilot must handle the aircraft based on the limits of the automatic landing system. The purpose of this study is to investigate the use of RNN-GA hybrid systems in aircraft automatic landing controls and to make automatic landing systems more intelligent. Current flight control law is adopted in the intelligent design. Tracking performance and adaptive capability are demonstrated through software simulation. For the safe landing of an aircraft using a conventional controller with original control gains [9], the wind turbulence limit is 30 ft/sec. In this study, a RNN controller with original control gains can reach 50 ft/sec, and a well-trained hybrid RNN-GA controller can overcome turbulence to 90 ft/sec. It shows that the proposed controller has better performance than previous studies [6]-[10]. From these simulations, the recurrent neural network controller can be used to successfully replace the conventional controller. This intelligent controller can act as an experienced pilot and guide the aircraft to a safe landing in severe wind turbulence environment.

References

1. Federal Aviation Administration, "Automatic Landing Systems," AC20-57A, Jan. 1997.
2. C.E. Cohen, et al., "Automatic Landing of a 737 Using GNSS Integrity Beacons," *Proc. ISPA*, 1995.
3. DDC-I, "Advanced Auto Landing System from Swiss Federal Aircraft Factory," *Real-Time Journal*, Sprint, 1995.
4. S. Asai, *et al.*, "Development of Flight Control System for Automatic Landing Flight Experiment," *Mitsubishi Heavy Industries Technical Review*, vol. 34, no. 3, 1997.
5. D.N. Kaufmann and B.D. McNally, "Flight Test Evaluation of the Stanford University and United Airlines Differential GPS Category III Automatic Landing System," NASA Technical Memorandum 110355, June 1995.
6. H. Izadi, M. Pakmehr, and N. Sadati, "Optimal Neuro-Controller in Longitudinal Autolanding of a Commercial Jet Transport," *Proc. IEEE International Conference on Control Applications*, CD-000202, pp. 1-6, Istanbul, Turkey, June 2003.

7. D.K.Chaturvedi, R.Chauhan, and P.K. Kalra, “Application of Generalized Neural Network for Aircraft Landing Control System,” *Soft Computing*, vol. 6, pp. 441-118, 2002.
8. Y. Iiguni, H. Akiyoshi, and N. Adachi, “An Intelligent Landing System Based on Human Skill Model,” *IEEE Trans. on Aeros. and Electr. Sys.*, vol. 34, no. 3, pp. 877-882, 1998.
9. C.C.Jorgensen and C.Schley, “A Neural Network Baseline Problem for Control of Aircraft Flare and Touchdown,” *Neural Networks for Control*, pp. 403-425, 1991.
10. M.G. Cooper, “Genetic Design of Rule-Based Fuzzy Controllers,” Ph.D. dissertation, University of California, Los Angeles, 1995.
11. R.L. Haupt and S.E. Haupt, *Practical Genetic Algorithm*, USA, 2004.
12. R.J. Williams and D. Zipser, “A Learning Algorithm for Continually Running Fully Recurrent Neural Network,” *Neural Computing*, vol. 1, pp.270-2801 1989.
13. J.L.Elman, “Finding Structure in Time,” *Cognitive Science*, vol. 14, pp.179-211, 1990.
14. M. Meral and N.S Sengor, “System Identification with Hybrid Elman Network,” *Proc. of IEEE on Signal Processing and Communications Applications Conf.*, pp. 80-83, 2004.
15. A.A. Adewuya, “New Methods in Genetic Search with Real-valued Chromosomes,” M.S. thesis, Dept. of Mechanical Engineering, Massachusetts Institute of Technology, 1996.

An AND-OR Fuzzy Neural Network Ship Controller Design

Jianghua Sui and Guang Ren

Marine Engineering College, Dalian Maritime University, Dalian 116026, P.R. China
suijh@tom.com, reng@dlmu.edu.cn

Abstract. In this paper, an AND-OR fuzzy neural network (AND-OR FNN) and a piecewise optimization approach are proposed. The in-degree of neuron and the connectivity of layer are firstly defined and Zadeh's operators are employed in order to infer the symbolic expression of every layer, the equivalent is proved between the architecture of AND-OR FNN and fuzzy weighted Mamdani inference. The main superiority is shown not only in reducing the input space, but also auto-extracting the rule base. The optimization procedure consists of GA (Genetic Algorithm) and PA (Pruning Algorithm);the AND-OR FNN ship controller system is designed based on input-output data to validate this method. Simulating results demonstrate that the number of rule base is decreased remarkably and the performance is good, illustrate the approach is practicable, simple and effective.

1 Introduction

Fuzzy neural network combines the theories of fuzzy logical and neural network, including learning, association, identification, self-adaptation and fuzzy information process. The logic neurons have received much concern all the time as the important components of neural networks. From model design to algorithm study, there are many achievements. Glorennec[1]proposed a general artificial neuron to realize Lukasiewicz logical operate. Yager[2] employed a group of OWA fuzzy Aggregation operators to form OWA neuron. Pedrycz and Rocha [3] proposed aggregation neurons and referential neurons by integrating fuzzy logic and neural network and discuss the relation about the ultimate network structure and practical problem; Pedrycz et al. [4], [5] constructed a knowledge-based network by AND, OR neurons to solve classified problem and pattern recognition. Bailey et al. [6] extended the single hidden layer to two hidden layers for improve complex modeling problems. Pedrycz and Reformat designed fuzzy neural network constructed by AND, OR neurons to model the house price in Boston [7].

We consider the multi-input-single-output (MISO) fuzzy logic-driven control system based on Pedrycz. Pedrycz[7] transformed T norm and S norm into product and probability operators, formed a continuous and smooth function to be optimized by GA and BP. But there is no exactly symbolic expression for every node, because of the uncertain structure. In this paper, the in-degree of neuron and the connectivity of layer are firstly defined and Zadeh's operators are employed in order to infer the symbolic expression of every layer, and form a continuous and rough function. The equivalence is proved between the architecture of AND-OR FNN and the fuzzy weighted Mamdani inference in order to utilize the AND-OR FNN to optimize fuzzy rules. The piecewise optimization of AND-OR FNN consists of GA and PA. Finally this approach is applied to design the AND-OR FNN ship controller. Simulating results show the performance is good.

2 Fuzzy Neurons and Topology of AND-OR FNN

2.1 Logic-Driven AND, OR Fuzzy Neurons

The AND and OR fuzzy neurons were two fundamental classes of logic-driven fuzzy neurons. The basic formulas governed the functioning of these elements are constructed with the aid of T norm and S norm (see Fig. 1, Fig. 2).Some definitions of the double fuzzy neurons show their inherent capability of reducing the input space.

Definition 1. Let $X = [x_1, x_2......x_n]$ be input variables $W = [w_1, w_2,......w_n]$ be adjustable connections (weights) confined to the unit interval, then the AND fuzzy neuron which completes a T-S norm composition operators is shown in Fig.1, the OR fuzzy neuron which completes an S-T norm composition operators is shown in Fig.2. Both of them are defined by Zadeh's operators as the following expression(1) and (2), respectively.

$$y = \overset{n}{\underset{i=1}{T}}(w_i S x_i)\,,\ y = \overset{n}{\underset{i=1}{\wedge}}(w_i \vee x_i) \tag{1}$$

$$y = \overset{n}{\underset{i=1}{S}}(w_i T x_i)\,,\ y = \overset{n}{\underset{i=1}{\vee}}(w_i \wedge x_i) \tag{2}$$

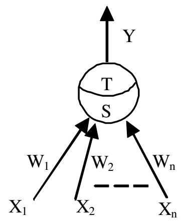


Fig. 1. AND neuron

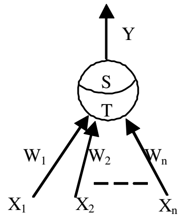


Fig. 2. OR neuron

Owing to the special compositions of neurons, for binary inputs and connections the neurons function is equal to the standard gates in digital circuit. For AND neuron, the closer to 0 the connection w_i is, the more important to the output the corresponding input x_i is. For OR neuron, the closer to 0 connection w_i is, the more important to the output the corresponding input x_i is. Thus the values of connections become the criterion to eliminate the irrelevant input variables to reduce input space.

2.2 Several Notations About AND, OR Neuron

Definition 2. Let w_i be the connection value, x_i be the ith input variable. Then,

$$RD(x_i) = w_i \in [0,1] \tag{3}$$

is the relevant degree between the input x_i and the neuron's output. For AND neuron, if $RD(x_i) = 0$, then x_i is more important feature to the output; if $RD(x_i) = 1$, then x_i is more irrelevant feature to the output, it can be cancelled. For OR neuron, vice versa. Thus the RDV or RDM is the vector or matrix made up of connections, respectively , which becomes the threshold to obstacle some irrelevant input variables, also lead to reduce the input space.

Definition 3. The in-degree of the ith neuron $d^+(neuron_i)$ is the number of input variables, then the in-degree of the ith AND neuron $d^+(AND_i)$ is shown the number of its input variables; the in-degree of the jth OR neuron $d^+(OR_j)$ is shown the number of its input variables.

2.3 The Architecture of AND-OR FNN

This feed-forward AND-OR FNN consists of five layers (Fig.3.), the input layer, fuzzification layer, double hidden layers (one consists of AND neurons, the other consists of OR neurons) and the defuzzification output layer, Here the fuzzy inference and the fuzzy rule base are integrated into the double hidden layers. The inference mechanism behaves as the inference function of the hidden neurons. Thus the rule base can be auto-generated by training AND-OR FNN in virtue of input-output data.

Both W and V are connection matrixes, also imply relevant degree matrix (RDM). Vector H is the membership of consequents. The number of neurons in every layer is n, $n \times t$, m, s and 1, respectively (t is the number of fuzzy partitions.).

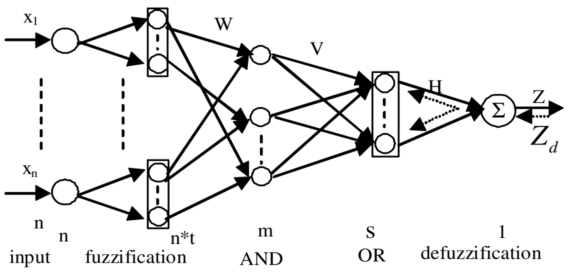


Fig. 3. The architecture of AND-OR FNN

Definition 4. The layer connectivity is the maximum in-degree of every neuron in its layer, including the double hidden layers,

$$Con(AND) = \max(d^+(AND_i)), Con(OR) = \max(d^+(OR_j)) \tag{4}$$

where $d^{+}(AND_i)$ is the in-degree of t he *ith* AND neuron; $d^{+}(OR_j)$ be the in-degree of the *ith* OR neuron.

Remark 1: $Con(AND) \le n$ (n is the number of input variables). $Con(OR) \le m$ (m is the number of AND neurons).

2.4 The Exact Expression of Every Node in AND-OR FNN

According to definitions and physical structure background, the AND-OR FNN model is derived as Fig. 3. The functions of the nodes in each layer are described as follows.

Layer 1: For every node i in this layer, the input and the output are related by

$$O_i^1 = x_i \tag{5}$$

where O_i^1 denotes the output of *ith* neuron in layer 1, $i = 1, 2, \cdots, n$ .Here the signal only transfers to the next layer without processing.

Layer 2: In this layer, each neuron represents the membership function of linguistic variable; Gaussian is adopted as membership function. The linguistic value (small, $\cdots$, very large) A_j are used. The function is shown as.

$$O_{ij}^2 = \mu_{A_j}(x_i) = e^{\frac{-(x_i - m_{ij})^2}{\sigma^2}} \tag{6}$$

where $i = 1, 2, ...n, j = 1, 2, ...t$, m_{ij} and σ is the modal and spread of the *jth* fuzzy partition from the *ith* input variable.

Layer 3: This layer is composed of AND neurons, the function can be expressed as follows based on above.

$$O_k^3 = \underset{p_1=1}{\overset{d^{+}(AND_k)}{T}} (wSO^2) = \underset{p_1=1}{\overset{d^{+}(AND_k)}{\wedge}} (w_{rk} \vee \mu_{A_{ij}}(x_i)) \tag{7}$$

where $k = 1, 2, \cdots, t^n$ $j = 1, 2, \cdots, t$ $i = 1, 2, \cdots, n$ $r = (i-1) \times t + j$, $d^{+}(AND_k)$ is the in-degree of the *kth* AND neuron. t^n is the total of AND neurons in this layer.

Remark 2: when p_1 is fixed and $d^{+}(AND_i) \ge 2$, i must be different. That means the input of the same AND neuron must be from the different x_i .

Layer 4: This layer is composed of OR neurons, the function can be expressed as follows based on above.

$$O_l^4 = \underset{p_2=1}{\overset{d^{+}(OR_l)}{S}} (vTO^3) = \underset{p_2=1}{\overset{d^{+}(OR_i)}{\vee}} (v_{l,k} \wedge O_k^3) \tag{8}$$

where $k=1,2,\cdots,t^n$, $l=1,2,\cdots,s$ $d^+(OR_l)$ is the in-degree of the lth OR neuron. s is the total of OR neurons.

Layer 5: There is only one node in this layer, but includes forward compute and backward training. Center-of-gravity method is adopted for former compute as follows.

$$z=O^5=\frac{\sum O_l^4 h_{l,1}}{\sum O_l^4} \tag{9}$$

where $l=1,2,\cdots,s$, H is the membership function of consequents. The latter is only imported to train data for next optimization. There is no conflict because the double directions are time-sharing. The function is shown as following like (6) above.

$$O^5=\mu_{B_l}(z_d)=e^{\frac{-(z_d-m_j)^2}{\sigma^2}} \tag{10}$$

3 Functional Equivalent Between Fuzzy Weighted Mamdani Inference and AND-OR FNN

3.1 Fuzzy Weighted Mamdani Inference

The fuzzy weighted Mamdani inference system [8] utilizes local weight and global weight to avoid a serious shortcoming in that all propositions in the antecedent part are assumed to have equal importance, and that a number of rules executed in an inference path leading to a specified goal or the same rule employed in various inference paths leading to distinct final goals may have relative degrees of importance.

Assume the fuzzy IF-THEN rules with consequent $y\ is\ B_l$ to be represented as:

$$Rule: if\ \ x_1 \text{is } A_{1j} \text{and} \cdots \text{and } x_i\ \ is\ A_{ij} \text{and} \cdots \text{and } x_n is\ A_{nj} \text{then } y\ is\ B_l \tag{11}$$

where $x_1,\cdots,x_n$ are the input variables, y is the output, A_{ij} and B_l are the fuzzy sets of input and output, respectively. w_{ij} is local weights of the antecedent part; v_i is the class of global weight for every rules. $i=1,2,\cdots,n$, $j=1,2,\cdots,t$, $l=1,2,\cdots,s$, t is the total of antecedent fuzzy sets; s is the total of consequent fuzzy sets. The same B_i is collected to form s complex rules as follows:

$$\begin{aligned}
&Rule1: if\ \ x_1 \text{ is } A_{1i'} \text{ and} \cdots \text{and } \ x_n\ \ is\ A_{ni''} \text{ then } y\ is\ B_1\\
&\qquad or \cdots or\ \ if\ \ x_1 \text{ is } A_{1j'} \text{ and} \cdots \text{and } \ x_n\ \ is\ A_{nj''} \text{ then } y\ is\ B_1\\
&RuleS: if\ \ x_1 \text{ is } A_{1k'} \text{ and} \cdots \text{and } \ x_n\ \ is\ A_{nk''} \text{ then } y\ is\ B_s\\
&\qquad or \cdots or\ \ if\ \ x_1 \text{ is } A_{nl'} \text{ and} \cdots \text{and } \ x_n\ \ is\ A_{nl''} \text{ then } y\ is\ B_s
\end{aligned} \tag{12}$$

where $i',i'',j',j'',k',k'',l',l''\in[1,t]$.

On the view of general fuzzy inference, to a single rule, the firing strength (or weight) of *ith* rule is usually obtained by min or multiplication operator, to the complex rule, the firing strength is the union λ_i, which can be expressed as.

$$\begin{aligned}\lambda_l &= \overset{\Delta_l}{\underset{p_2=1}{\vee}} (\mu_{A_{1i'}\times\cdots\times A_{ni''}}(x_1,\cdots,x_n),\cdots,\mu_{A_{1j'}\times\cdots\times A_{nj''}}(x_1,\cdots,x_n)) \\ &= \overset{\Delta_l}{\underset{p_2=1}{\vee}} \begin{pmatrix} \overset{\Psi_1}{\underset{p_1=1}{\wedge}} \left(\mu_{A_{1i'}}(x_1),\cdots,\mu_{A_{ni''}}(x_n)\right), \\ \cdots, \overset{\Psi_{n\times t}}{\underset{p_1=1}{\wedge}} \left(\mu_{A_{1j'}}(x_1),\cdots,\mu_{A_{nj''}}(x_n)\right) \end{pmatrix}\end{aligned} \tag{13}$$

where Δ is the number of the same consequent B_l, Ψ the number of antecedent parts.

On the view of fuzzy weighted Mamdani inference, local weight and global weight are considered by the mode of $<\vee,\wedge>$ as follows:

$$\lambda_l = \overset{\Delta_l}{\underset{p_2=1}{\vee}} \begin{pmatrix} v_1 \wedge \left(\overset{\Psi_1}{\underset{p_1=1}{\wedge}} \left(w_{1i'} \vee \mu_{A_{1i'}}(x_1),\cdots,w_{ni''} \vee \mu_{A_{ni''}}(x_n)\right)\right), \\ \cdots, v_{\Delta_l} \wedge \left(\overset{\Psi_{n\times t}}{\underset{p_1=1}{\wedge}} \left(w_{1j'} \vee \mu_{A_{1j'}}(x_1),\cdots,w_{nj'} \vee \mu_{A_{nj''}}(x_n)\right)\right) \end{pmatrix} \tag{14}$$

In general, center-of-gravity method is adopted for defuzzification as follows.

$$Y = \frac{\sum_{l=1}^{s} \lambda_l \mu_{B_l}(y)}{\sum_{l=1}^{s} \lambda_l} \tag{15}$$

3.2 Functional Equivalence and Its Implication

From (9) and (15), it is obvious that the functional equivalence between an AND-OR FNN and a fuzzy weighted Mamdani inference can be established if the following is true.

1. The number of AND neurons is equal to the number of fuzzy rules.
2. The number of OR neurons is equal to the number of fuzzy complex rules.
3. The T-norm and S-norm used to compute each rule's firing strength are min and max, respectively.
4. Both the AND-OR FNN and the fuzzy inference system under consideration use the same center-of-gravity method to derive their overall outputs.

Under these conditions, when $RDM(w_{ij})=0$, $RDM(v_{ij})=1$, AND-OR FNN is completely connected, which is shown that every part of antecedent is the same important to the output , $\Psi_k = n$, $\Delta_l = n\times t$ and it is equal to the general fuzzy inference. But it falls short of the practice. When $w_i \in [0,1]$, $v_i \in [0,1]$, every part of antecedent has different important degree to the output and every rules has different important degree to the final output.

4 GA Optimization

In general, gradient-based optimization is often chosen for ANN, Pedrycz [7] has proved that the output of AND neuron and its derivative are heavily affected by the increasing values of “n”. Meanwhile, here the final output of AND-OR FNN is not a smooth function. Obviously gradient-based optimization is not preferred, GA (Genetic Algorithm) is a very effective method to solving the structure optimization problem because of the superiority of robust, random, global and parallel process. Structure optimization is transferred to the species evolution by GA to obtain the optimal solution.

4.1 Code Mode

Parameters encoding is the first phase of GA optimization. Parameters are often transformed to unsigned binary. Here the structure parameters of two hidden layers are considered on focus by double matrixes as follows:

$$RDM(AND) = \left[w_{i,j}^{and} \right]_{m\times(n\times t)}, RDM(OR) = \left[v_{i,j}^{or} \right]_{s\times m} \tag{16}$$

where i, j, m, n, t and s are like before. Lining up the two matrixes to a binary string as initial population represents the structure status. In the initialization, $RDM(w_{ij}) = 0$ and $RDM(v_{ij}) = 1$ are set or the authoritative opinion from experts can also be put into initial population as seeds to reach the best initialization.

4.2 Objective Function

The parameters of the GA used are chosen experimentally. The fitness function maximized by the GA is expressed in the form of the sum of squared errors between target values (data) and outputs of network. This fitness has to be minimized (it stands in contrast with the classic way of using fitness functions in GA whose values are maximized; if necessary a simple transformation of taking the reciprocal of the fitness, $1/fitness$, brings in line with the way of maximizing the fitness)

$$fitness = \frac{1}{(\sum (z_d - z)^2)} \tag{17}$$

5 Pruning Algorithm

The pruning of the network is a process consisting of removing unnecessary parameters and nodes during the training process of the network without losing its generalization capacity [9]. The best architecture can be sought using GA in conjunction with pruning algorithms. There are three situations as follow:

1. For AND neuron, if the connection is equal to zero, this means that the corresponding input impacts the output of this neuron, the connection can be pruned away if

its value is one. For the OR neuron a reverse situation occurs: the connection equal to one implies that the specific input is essential and affects the output of the neuron. The connection can be pruned away if its value is zero.

2. For $RDM(AND)$, the number of zero in every line is shown as the in-degree of this AND neuron, the node can be pruned away if the line is full of one. For $RDM(OR)$, the number of one in every line is shown as the in-degree of this OR neuron, the node can be pruned away if the line is full of zero.
3. For $RDM(AND)$, if there are the same line in the matrix, that means the node is redundant, it can be pruned away. For $RDM(OR)$, that means the node is conflict each other, it need be pruned away.

6 Application to Ship Control

According to the principle of manual steering, it makes full use of the advantages of fuzzy logic and neural network. An AND-OR FNN ship controller is presented to adapt the change of navigation environment and get the information of ship maneuverability automatically. The structure and parameters are learned by GA and PA. Fuzzy rules can be auto-obtained from a group of sample data, which is generated by a fuzzy control system. Fig.4 shows a block diagram of the AND-OR FNN autopilot. Test results show that an effect method has been provided for the improvement of ship steering control.

Double input variable of AND-OR FNN ship controller are heading error $\Delta\psi = \psi(k+1) - \psi(k)$ and yaw rate $\gamma(k) = d\psi / dt$. The control action generated by the autopilot is the command rudder angle $\delta(k)$. The range of values for a given autopilot inputs and output are $(-20^\circ, 20^\circ)$ and $(-2.5^\circ/\sec, 2.5^\circ/\sec)$, It is usually required that the rudder should move from 35° port to 35° starboard within 30s.

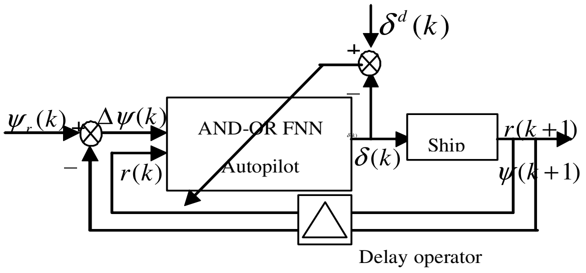


Fig. 4. AND-OR FNN autopilot for ship course-keeping

6.1 The Structure Design and Simulation Result

In this section, an AND-OR FNN ship controller is constructed successfully. Firstly, $\Delta\psi(k)$ and $\gamma(k)$ are double input variable; $\delta(k)$ is the output variable, which are all fuzzied into 3 Gaussian membership functions with 0.5 overlap degree, thus there are 9 pieces of rules. According to analysis before, there are 9 AND neurons and 3 OR neurons in AND-OR FNN. Initialize structure parameters with all connections to form the structure string that is $RDM(w_i)=0$, $RDM(v_i)=1$. The structure and parameters are optimized by GA and PA, some parameters are chosen experimentally, which are including population size=100, crossover rate=0.7, mutation rate=0.01 and selection process is tournament. The ultimate better structure result is transformed into the AND-OR FNN. There are 5 AND neurons and 3 OR neurons left, that is, 5 pieces of fuzzy rules and 3 complex rules. The number of rule base is decreased remarkably The ultimate better performance result is shown as Fig. 5 and Fig. 6. The simulation result has illustrated the effectiveness of proposed method

7 Conclusion

In this paper, we have proposed a novel AND-OR FNN and a piecewise optimization approach, the symbol expressions of every node in the AND-OR FNN are educed in detail, the input space is reduced by special inner structure of AND and OR. The equivalence is proved to the fuzzy weighted Mamdani inference. The fuzzy rule base auto-extracted is equal to optimize the architecture of AND-OR FNN. This novel approach has been validated using AND-OR FNN ship controller design. The simulation results illustrate the approach is practicable, simple and effective and the performance is good.

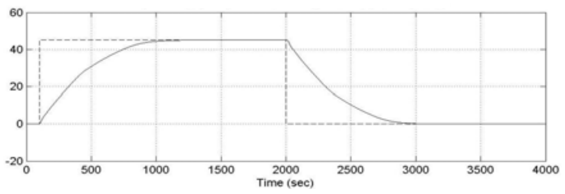


Fig. 5. Ship heading (solid) and desired ship heading (dashed)

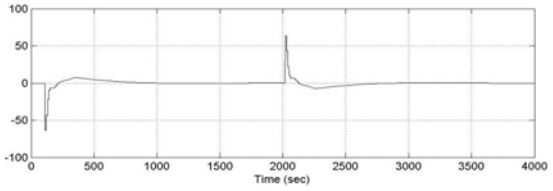


Fig. 6. Rudder angle (δ)

Acknowledgements

This paper is supported by the Doctoral Foundation of Education Committees (2003151005) and Ministry of Communication of P. R. China (200332922505).

References

1. Pierre Yves G.: Neuro-Fuzzy Logic. In Proc. IEEE-FUZZ. New Orleans, (1996)512-518
2. Yager R.: OWA neurons: Anew class of fuzzy neurons. In Proc. IEEE-FUZZ.San Diego.(1992)2316-2340
3. Pedrcy. W. and Rocha. A F.: Fuzzy-set based models of neurons and knowledge-based networks. IEEE.Trans. on Fuzzy System,1(4)(1993)254-266
4. Hirota K, Pedrycz W.: Knowledge-based networks in classification problems. Journal, Fuzzy Sets and Systems, 59(3) (1993)271-279
5. Pedrycz W and Succi.G.: Genectic granular classifiers in modeling software quality. The journal of Systems and software, 75(2005)277-285
6. Bailey S A, Chen Ye hwa.: A Two Layer Network Using the OR/AND Neuron. In: proc. IEEE-FUZZ .Anchorage,(1998)1566-1571
7. Pedrycz W. Reformat M.: Genetically optimized logic models Fuzzy Sets and Systems 150(2) (2005)351-371
8. Yeung D. S. and Tsang E. C. C.:Weighted fuzzy production rules, Fuzzy sets and systems, 88(1997)299-313
9. Rosa Maria Garcia-Gimeno, Cesar Hervas-Martinez and Maria Isabel de Siloniz.: Improving artificial neural networks with a pruning methodology and genetic algorithms for their application in microbial growth prediction in food. International Journal of food Microbiology, 72(2002)19-30

RBF ANN Nonlinear Prediction Model Based Adaptive PID Control of Switched Reluctance Motor Drive

Chang-Liang Xia and Jie Xiu

School of Electrical Engineering and Automation, Tianjin University,
300072 Tianjin, China
clxia@tju.edu.cn

Abstract. The inherent nonlinear of switched reluctance motor (SRM) makes it hard to get a good performance by using the conventional PID controller to the speed control of SRM. This paper develops a radial basis function (RBF) artificial neural network (ANN) nonlinear prediction model based adaptive PID controller for SRM. ANN, under certain condition, can approximate any nonlinear function with arbitrary precision. It also has a strong ability of adaptive, self-learning and self-organization. So, combining it with the conventional PID controller, a neural network based adaptive PID controller can be developed. Appling it to the speed control of SRM, a good control performance can be gotten. At the same time, the nonlinear mapping property and high parallel operation ability of ANN make it suitable to be applied to establish nonlinear prediction model performing parameter prediction. In this paper, two ANN - NNC and NNI are employed. The former is a back propagation (BP) ANN with sigmoid activation function. The later is an ANN using RBF as activation function. The former is used to adaptively adjust the parameters of the PID controller on line. The later is used to establish nonlinear prediction model performing parameter prediction. Compared with BP ANN with sigmoid activation function, the RBF ANN has a more fast convergence speed and can avoid getting stuck in a local optimum. Through parameter prediction, response speed of the system can be improved. To increase the convergence speed of ANN, an adaptive learning algorithm is adopted in this paper that is to adjust the learning rate according to the error. This can increase the convergence speed of ANN and make the system response quick. The experimental results demonstrate that a high control performance is achieved. The system responds quickly with little overshoot. The steady state error is zero. The system shows robust performance to the load torque disturbance.

1 Introduction

The SRM drives have been introduced for the past two decades. Some advantages of this motor are simplicity and robustness of construction; potentially, it is somewhat cheaper than other classes of machines. It also has a high efficiency and large torque output over a wide speed range. It has a wide application in industrial and in home appliances.

In order to get a large output power, the magnetic circuit of SRM is often designed saturation. Together with double salient structure and eddy current effect, all of this leads to the highly nonlinear characteristics of SRM. The conventional PID controller is widely applied to the control of electric machines. But it is hard to get a good performance when applied to control the speed of highly nonlinear SRM.

There have been several methods to tune conventional PID parameters. Among those methods, the well known method is the rules of Ziegler and Nichols for tuning the control gains. But once the parameters of the PID controller tuned according to a certain condition, it will not be tuned again. So, when the characteristic of the nonlinear plant changed, the parameters tuned formerly will not do. In order to get a high performance, the adaptive PID control strategy should be adopted. ANN is formed by the interconnection of artificial neurons. It has a strong ability of adaptive, self-learning, and self-organization. It also has a strong ability of nonlinear mapping and fault tolerance. So, by combining it with the conventional PID controller, an adaptive PID controller based on neural network can be constructed.

In the application of ANN, the parameters of the plant should be identified. The performance of the controller can be improved though the prediction of parameter of the system. Compared with sigmoid activation function BP ANN, the RBF ANN has a more fast convergence speed and can avoid getting stuck in a local optimum. By using the RBF ANN to identify the system parameters, the system can respond quickly.

To increase the convergence speed of ANN, an adaptive learning algorithm is adopted in this paper that is to adjust the learning rate according to the error. This can increase the convergence speed of ANN and make the system response much quickly.

Many researches have been done to implement the control of electrical drives based on ANN. ANN is used to produce the phase current needed for trailing speed [1]. ANN is used to the modeling and control of SRM [2]. The application of ANN to identification and control of SRM and other electric machine is reported [3]-[7]. In summary, the application of ANN to the control of SRM can improve the performance of SRM. In order to get a high performance, an adaptive PID controller based on RBF ANN nonlinear prediction model for SRM is developed in this paper.

2 The PWM Control Method of SRM

The SRM is a doubly salient construction machine. The rotor is driven by the reluctance torque. Under certain condition, the instantaneous torque can be expressed as

$$T = K\frac{U_s^2}{\omega^2}\cdot \tag{1}$$

where U_s is the phase voltage; ω is the rotor speed; K is a constant.

The torque is proportional to the voltage supplied to the winding. So, by adjusting the duty ratio of the voltage PWM, the average voltage will be adjusted, the torque will also be adjusted, then the speed will be changed.

3 The Adaptive PID Controller Based on RBF ANN Nonlinear Prediction Model

The structure of the adaptive PID controller based on RBF ANN nonlinear prediction model for SRM is shown in Fig. 1.

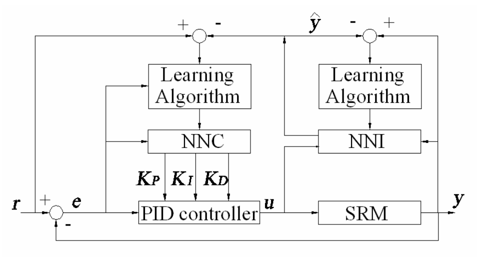


Fig. 1. The structure of the RBF ANN nonlinear prediction model based adaptive PID controller

In the system, the NNC is a sigmoid activation function BP ANN, which adaptively adjusts the parameters of the PID controller in order to get a set of optimal parameters. The PID controller performs the closed loop control of SRM and its parameters are tuned on-line. The NNI is a RBF ANN which acts as the nonlinear prediction model of SRM.

3.1 Adaptive Tuning of Parameters by BP ANN

The equation of conventional increment PID controller can be expressed as

$$u(k) = u(k-1) + K_p\left[e(k) - e(k-1)\right] + \\ K_I e(k) + K_D\left[e(k) - 2e(k-1) + e(k-2)\right] \cdot \tag{2}$$

where K_P□K_I□and K_D is the proportional factor, integration factor, and derivative factor, respectively.

When K_P□K_I□and K_D is considered as in function of the system operation condition, the Eq. (2) can be expressed as

$$u(k) = f\left[u(k-1), K_P, K_I, K_D, e(K), e(k-1), e(k-2)\right] \ . \tag{3}$$

In Eq. (3), $f[\cdot]$ is in function of $K_{P,}$ K_I and K_D. Through the training of NNC, an optimal control strategy can be searched.

In this paper, the neural network NNC is a three-layer sigmoid activation function BP-type ANN. The structure of NNC is shown in Fig. 2. There are four neurons in the input layer, six neurons in the hidden layer and three neurons in the output layer. The input neurons are corresponding to variables that reflect the system operation condition. The output neurons are corresponding to the parameters ($K_{P,}$ $K_{I,}$ and K_D) of the PID controller. For $K_{P,}$ $K_{I,}$ and K_D can't be negative, so the activation function of the neurons in the output layer is a sigmoid activation function. The activation function of the neurons in the hidden layer is a hyperbolic tan activation function.

In this paper, the input vector of the NNC is x=[$r(k)$□$y(k)$□$e(k)$□1].

From Fig. 2, it can be seen that the outputs of these neurons in the input layer are

$$O_j^{(1)} = x(j) \qquad j = 1, 2, \cdots, M \ . \tag{4}$$

where $O_j^{(1)}$ is the output of the neurons in the input layer, M is the number of neurons in the input layer.

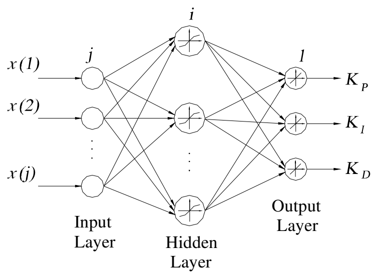


Fig. 2. The structure of the neural network NNC

The inputs and outputs of these neurons in the hidden layer of the NNC are

$$\left.\begin{aligned} net_i^{(2)} &= \sum_{j=0}^{M} w_{ij}^{(2)} O_j^{(1)}(k) \\ O_i^{(2)}(k) &= f\left[net_i^{(2)}(k)\right] \qquad i = 1,2,\cdots,Q \end{aligned}\right\} . \tag{5}$$

where $w_{ij}^{(2)}$ is the weight matrix between the input layer and the hidden layer; f[▪] is the activation function, f[▪]=tanh(x); label (1), (2), and (3) corresponding to the input layer, hidden layer and output layer, respectively; i is the number of neurons in the current layer; j is the number of neurons in the input layer.

The input and output of these neurons in the output layer of the NNC is

$$\left.\begin{aligned} net_l^{(3)}(k) &= \sum_{i=0}^{Q} w_{li}^{(3)} O_i^{(2)}(k) \\ O_l^{(3)}(k) &= g\left[net_l^{(3)}(k)\right] \qquad l = 1,2,3 \\ O_1^{(3)}(k) &= K_P \\ O_2^{(3)}(k) &= K_I \\ O_3^{(3)}(k) &= K_D \end{aligned}\right\} . \tag{6}$$

where $w^{(3)}{}_{li}$ is the weight matrix between the hidden layer and the output layer; g[•] is the activation function, $g[\cdot] = \frac{1}{2}[1 + \tanh(x)]$; l is the number of neurons in the current layer.

The objective function to be minimized is defined as

$$J = \frac{1}{2}\left[r(k) - y(k)\right]^2 . \tag{7}$$

where $r(k)$ is the reference input; $y(k)$ is the output of plant.

The weights adjustment for reduction of error is done by the standard gradient descent method. A momentum term is added to speed up the convergence speed to a globe optimum, which is given as

$$\Delta w_{li}^{(3)}(k) = -\eta(k)\frac{\partial J}{\partial w_{li}^{(3)}} + \alpha\Delta w_{li}^{(3)}(k-1) \cdot \tag{8}$$

where $\eta(k)$ is learning rate; α is momentum factor.

$$\frac{\partial J}{\partial w_{li}^{(3)}} = \frac{\partial J}{\partial y(k)} \cdot \frac{\partial y(k)}{\partial u(k)} \cdot \frac{\partial u(k)}{\partial O_l^{(3)}(k)} \cdot \frac{\partial O_l^{(3)}(k)}{\partial net_l^{(3)}(k)} \cdot \frac{\partial net_l^{(3)}(k)}{\partial w_{li}^{(3)}} \cdot \tag{9}$$

For $\partial y(k)/\partial u(k)$ is unknown, in order to get a good performance, the value of $\partial y(k)/\partial u(k)$ should be replaced by the prediction value of $\partial \hat{y}(k)/\partial u(k)$.

From Eq. (2), it can be derived

$$\left.\begin{array}{l} \dfrac{\partial u(k)}{\partial O_1^{(3)}(k)} = e(k) - e(k-1) \\ \dfrac{\partial u(k)}{\partial O_2^{(3)}(k)} = e(k) \\ \dfrac{\partial u(k)}{\partial O_3^{(3)}(k)} = e(k) - 2e(k-1) + e(k-2) \end{array}\right\} \cdot \tag{10}$$

So, the weights update equation of the output layer of neural network NNC can be derived as

$$\left.\begin{array}{c} \Delta w_{li}^{(3)}(k) = \eta\delta_l^{(3)} O_i^{(2)}(k) + \alpha\Delta w_{li}^{(3)}(k-1) \\ \delta_l^{(3)} = e(k)\dfrac{\partial \hat{y}(k)}{\partial u(k)} \cdot \dfrac{\partial u(k)}{\partial O_l^{(3)}(k)} \cdot g'\left[net_l^{(3)}(k)\right] \\ l = 1,2,3 \end{array}\right\} \cdot \tag{11}$$

In the same way, according to the differential method, the weights update equation of the hidden layer of the NNC can be derived as

$$\left.\begin{array}{c} \Delta w_{ij}^{(2)}(k) = \eta(k)\delta_i^{(2)} O_j^{(1)}(k) + \alpha\Delta w_{ij}^{(2)}(k-1) \\ \delta_i^{(2)} = f'\left[net_i^{(2)}(k)\right]\sum_{l=1}^{3}\delta_l^{(3)} w_{li}^{(3)}(k) \\ i = 1,2,\cdots,Q \end{array}\right\} \cdot \tag{12}$$

where $g'[\cdot] = g(x)[1-g(x)]$, $f' = [1-f^2(x)]/2$.

To increase the convergence speed of ANN, an adaptive learning algorithm is adopted in this paper that is to adjust the learning rate $\eta(k)$ according to the error.

3.2 The Nonlinear Prediction Model Based on RBF ANN

In order to get a high performance, the value of $\partial y(k)/\partial u(k)$ in Eq. (9) should be identified and predicted on line. For SRM is a severely nonlinear plant, a nonlinear prediction model is used to perform the parameter prediction.

In this paper, the SRM can be expressed by a MISO mathematical equation as

$$y(k) = f\left[u(k-1), y(k-1), y(k-2)\right] . \tag{13}$$

In order to predict the value of $\partial \hat{y}(k)/\partial u(k)$, in this paper, a three-layer RBF ANN with three neurons in the input layer, five neurons in the hidden layer and one neuron in the output layer called NNI is employed to perform parameter prediction. The activation function of the neuron in the output layer is a linear activation function. The activation function of the neurons in the hidden layer is a Gaussian type activation function.

In this paper, the input vector of the neurons in the input layer of the neural network NNI is $X=[\ u(k\text{-}1),\ y(k\text{-}1),\ y(k\text{-}2)]^{T}$. The vector used to construct the neural network NNI is $H=[h_1, h_2,\cdots, h_j, \cdots, h_m]^{T}$, in the vector, where h_j is Gaussian type function

$$h_j(x)=\exp\left(-\frac{\left\|X-C_j\right\|^2}{2b_j^2}\right) \qquad j=1,2,\cdots m \ . \tag{14}$$

where C_j is the center of the receptive field of the j^{th} neuron, j=1,2,⋯,m; b_j is the width of the receptive field.

The output of neural network NNI is

$$\hat{y}(k)=y_m(k)=w_1h_1+w_2h_2+\cdots+w_mh_m \ . \tag{15}$$

where w_j is the weight between the hidden layer and output layer.

The objective function of NNI to be minimized is defined as

$$J_1=\frac{1}{2}(y(k)-y_m(k))^2 \ . \tag{16}$$

where y is the output of plant; y_m *i*s the output of NNI.

According to the gradient descent method, the update equation of the weight between the hidden layer and the output layer, the center of the receptive field and the width of the receptive field can be expressed as

$$w_j(k)=w_j(k-1)+\eta'(k)(y(k)-y_m(k))h_j+\alpha'(w_j(k-1)-w_j(k-2)) \ . \tag{17}$$

$$\Delta b_j=(y(k)-y_m(k))w_jh_j\frac{(X-C_j)^2}{b_j^3} \ . \tag{18}$$

$$b_j(k)=b_j(k-1)+\eta'(k)\Delta b_j+\alpha'(b_j(k-1)-b_j(k-2)) \ . \tag{19}$$

$$\Delta c_{ji}=(y(k)-y_m(k))w_j\frac{x_j-c_{ji}}{b_j^2} \ . \tag{20}$$

$$c_{ji}(k)=c_{ji}(k-1)+\eta'(k)\Delta c_{ji}+\alpha'(c_{ji}(k-1)-c_{ji}(k-2)) \ . \tag{21}$$

where $\eta'(K)$ is the learning rate; α' is momentum factor.

The partial derivative of $y(k)$ with respect to $u(k)$ called Jacobian matrix can be expressed as

$$\frac{\partial y(k)}{\partial u(k)} \approx \frac{\partial y_m(k)}{\partial u(k)} = \sum_{j=1}^{m} w_j h_j \frac{c_{ji} - x_1}{b_j^2} \ . \tag{22}$$

where $x_1 = u(k)$.

3.3 Adaptive Tuning of Learning Rate

The reason of the low convergence speed of BP algorithm is that, in order to keep the convergence of the learning algorithm, the learning rate is small. To increase the convergence speed of ANN, this paper adopts an adaptive learning algorithm that is to adjust the learning rate according to the error. When the current error is large than the former, the learning rate will not be changed, otherwise, the learning rate will be increased.

3.4 On-Line Training and Off-Line Training

To reduce computational complexity, improve initial response performance and determine the parameters of ANN, off-line training is done. After off-line training, the control system is applied to on-line training and on-line control. Through on-line training, effect on the performance of the system caused by the change of system parameters can be eliminated. This helps to increase the robustness of the system. By measuring the error, the proposed controller performs on-line control to achieve a high performance. By this method, the initial parameter of the controller is optimized and on-line computational complexity is reduced and the controller can perform a real-time control on-line.

4 Experimental Results

The experimental results are got under the control of the adaptive PID controller based on RBF ANN nonlinear prediction model. The block diagram of the hardware control system for the SRM which based on the TMS320LF2407 type DSP is shown in Fig. 3.

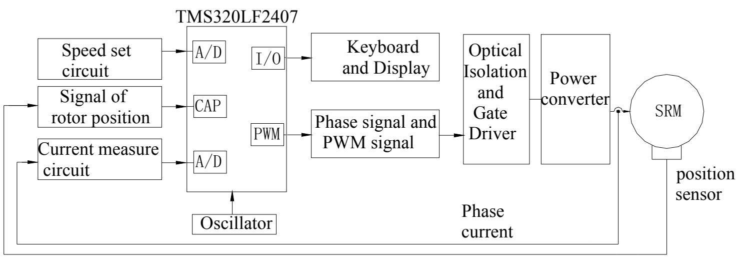


Fig. 3. The block diagram of hardware control system for SRM

With no load, the step response of the SRM under the control of the proposed controller is shown as curve 1 in Fig. 5. When under the control of the proposed controller, the system responds quickly with little overshoot. The steady state error is zero. In Fig. 4, curve 2 stands for the step response under the control of the conventional PID controller. The parameters of the conventional PID controller are tuned as K_P=10, K_I=0.3, K_D=4.2.

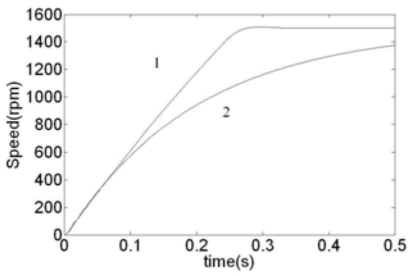


Fig. 4. Step response of SRM

In the dynamic response, the adjustment of the parameters of the adaptive PID controller is shown in Fig. 5.

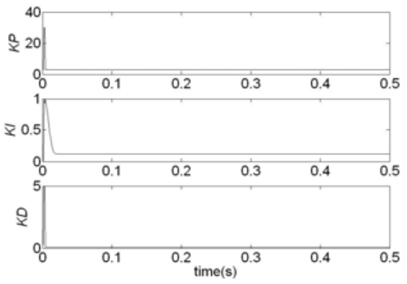


Fig. 5. Adjustment of parameters of the adaptive PID controller in dynamic response

The step response of the system without identification is shown as curve 2 in Fig. 6. The step response of the system with identification is shown as curve 1 in Fig. 6. From the Fig. 6, it can be seen that after the nonlinear prediction model established to predict the parameter of the system, this increases the convergence speed of ANN, the dynamic response of the system is much quick.

When the system is disturbed, the response of the system is shown in Fig. 7. Curve 1 is the response of the system under the control of the proposed controller. Curve 2 is the response of the system under the control of the conventional PID controller. From Fig. 7, it can be seen that under the control of the proposed controller, the speed drop is little and the recover time is short. Under the control of the proposed controller, the system shows a robust performance to the torque disturbance.

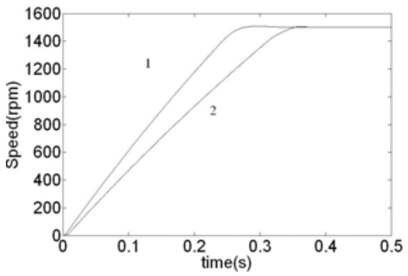


Fig. 6. Step response of SRM with and without identification

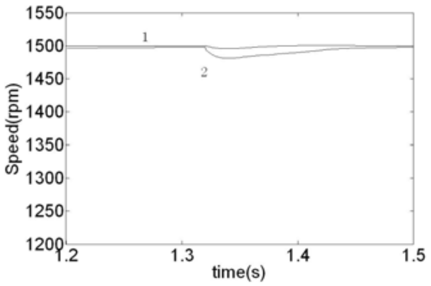


Fig. 7. Response of SRM under torque disturbance

While running at steady state, the phase current is shown in Fig. 8.

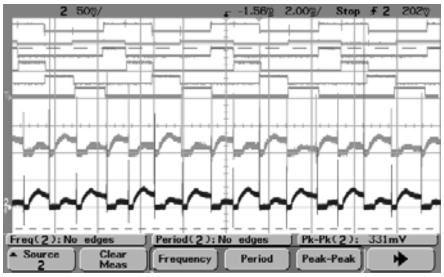


Fig. 8. Phase current of SRM at steady state

5 Conclusions

By using the ANN's strong ability of adaptive, self-learning, and self-organization and combining it with the PID control algorithm, an adaptive PID controller based on RBF ANN nonlinear prediction model is developed in this paper. The parameters of the proposed controller can be adaptively adjusted on-line. This overcome the draw-back of conventional PID controller which's parameters are fixed. Appling it to the speed control of SRM, a high response performance of SRM can be achieved. By using the RBF ANN to construct the nonlinear prediction model of the nonlinear SRM to perform parameter prediction, the response of the SRM can be improved. To increase the convergence speed of ANN, an adaptive learning algorithm is adopted in this paper. This makes the system response much quick. Experimental results prove that a high performance is achieved under the control of the proposed controller. The system responds quickly with little overshoot. The Steady state error is zero. The system shows robust performance to the load torque disturbance.

References

1. Garside J. J., Brown R. H., Arkadan A. A.: Switched reluctance motor control with artificial neural networks. IEEE International Electric Machines and Drives Conference Record, (1997) TB1/2.1–TB1/2.3
2. Trifa V., Gaura E., Moldovan L.: Neuro-control approach of switched reluctance motor drives. MELECON '96, 8th Mediterranean Electrotechnical Conference, Vol. 3, (1996) 1461–1464
3. Liu Jian,Wang Hui: Neural networks indentification for switched reluctance motor drive system . Journal of Shenyang Arch. and Civ. Eng. Inst, Vol. 16, No. 2, (2000) 146–149
4. Rahman, K.M., Rajarathnam, A.V., Ehsani, M.: Optimized instantaneous torque control of switched reluctance motor by neural network. Industry Applications Conference, 1997. Thirty-Second IAS Annual Meeting, IAS '97., Conference Record of the 1997 IEEE, Vol. 1, (1997) 556–563
5. Yilmaz, K., Mese, E., Cengiz, A.: Minimum inductance estimation in switched reluctance motors by using artificial neural networks. 11th Mediterranean Electrotechnical Conference, 2002, MELECON 2002, (2002) 152–156
6. Fahimi, B., Suresh, G., Johnson, J.P., Ehsani, M., Arefeen, M., Panahi, I.: Self-tuning control of switched reluctance motors for optimized torque per ampere at all operating points. Applied Power Electronics Conference and Exposition, 1998. Conference Proceedings 1998, Thirteenth Annual, Vol. 2, (1998) 778–783
7. Senjyu T, Shingaki T, Omoda A et a1: High efficiency drives for synchronous reluctance motors using neural network. IECON 2000, Vol. 2, (2000) 777–782

Hierarchical Multiple Models Neural Network Decoupling Controller for a Nonlinear System*

Xin Wang[1,2] and Hui Yang[2]

[1] Center of Electrical & Electronic Technology, Shanghai Jiao Tong University, Shanghai, P.R. China, 200030
wangxin26@sjtu.edu.cn
[2] School of Electrical & Electronic Engineering, East China Jiao Tong University, 330013

Abstract. For a nonlinear discrete-time Multi-Input Multi-Output (MIMO) system, a Hierarchical Multiple Models Neural Network Decoupling Controller (HMMNNDC) is designed in this paper. Firstly, the nonlinear system's working area is partitioned into several sub-regions by use of a Self-Organizing Map (SOM) Neural Network (NN). In each sub-region, around every equilibrium point, the nonlinear system can be expanded into a linear term and a nonlinear term. Therefore the linear term is identified by a BP NN trained offline while the nonlinear term by a BP NN trained online. So these two BP NNs compose one system model. At each instant, the best sub-region is selected out by the use of the SOM NN and the corresponding multiple models set is derived. According to the switching index, the best model in the above model set is chosen as the system model. To realize decoupling control, the nonlinear term and the interaction of the system are viewed as measurable disturbance and eliminated using feedforward strategy. The simulation example shows that the better system response can be got comparing with the conventional NN decoupling control method.

1 Introduction

In recent years, for linear multivariable systems, researches on adaptive decoupling controller have made much success [1]. As for nonlinear Multi-Input Multi-Output (MIMO) systems, few works have been observed [2,3]. Ansari *et al.* simplified a nonlinear system into a linear system by using Taylor's expansion at the equilibrium point and controlled it using linear adaptive decoupling controller accordingly [4]. However, for a system with strong nonlinearity and high requirement, it can not get good performance [5]. In [6,7], an exact linear system can be produced utilizing a feedback linearization input-output decoupling approach and high dynamic performance was achieved. But accurate information, such as the parameters of the system, must be known precisely. Furthermore, a variable structure controller with sliding model was proposed [8] and industrial experiment in binary distillation columns was presented in [9], which required the system an affine system. Although the design methods above can realize nonlinear decoupling control, there were too many

* This work is supported in part by the National Natural Science Foundation of China 60504010, 50474020 and Shanghai Jiao Tong University Research Foundation.

assumptions required on the system so that they can not be used in the industrial process directly. To solve this problem, Neural Network (NN) decoupling controller was proposed [10]. In [11], NN was used to identify the structure and the parameters of the nonlinear system. In [12], at the origin, the system was expanded into the linear term and the nonlinear term and two NNs were adopted to identify these two terms. Unfortunately, when the equilibrium point was far from the origin, the system lost its stability.

In this paper, for a nonlinear discrete-time MIMO system, a Hierarchical Multiple Models Neural Network Decoupling Controller (HMMNNDC) is designed. The nonlinear system's working area is partitioned into some sub-regions, which is described using a Self-Organizing Map (SOM) NN. In each sub-region, at every equilibrium point, the system is expanded into a linear term with a nonlinear term. Both terms are identified using BP NNs, which compose one system model. All models, which are got from all equilibrium points in this sub-region, compose a multiple models set. At each instant, the best sub-region is chosen out using the SOM NN. Then in the corresponding multiple models set, according to the switching index, the best model is selected out as the system model. To control the system, the nonlinear term and the interactions of the above model is viewed as measurable disturbance and eliminated by the use of the feedforward strategy. The simulations illustrate the effectiveness of the method.

2 Description of the System

The system is a nonlinear discrete-time MIMO system of the form

$$\boldsymbol{y}(t+1)=\boldsymbol{f}[\boldsymbol{y}(t),\cdots,\boldsymbol{u}(t),\cdots], \tag{1}$$

where $\boldsymbol{u}(t)$, $\boldsymbol{y}(t)$ are $n\times 1$ input, output vectors respectively and $\boldsymbol{f}[\cdot]$ is a vector-based nonlinear function which is continuously differentiable and Lipshitz.

Suppose that $(\boldsymbol{u}_1,\boldsymbol{y}_1),\cdots(\boldsymbol{u}_l,\boldsymbol{y}_l),\cdots(\boldsymbol{u}_m,\boldsymbol{y}_m)$ are all equilibrium points. At each equilibrium point $(\boldsymbol{u}_l,\boldsymbol{y}_l)$, using Taylor's formula, it obtains

$$\begin{gathered}\boldsymbol{y}(t+1)=\boldsymbol{y}_l+\sum_{n_1=1}^{n_a}\boldsymbol{f}'_{n_1}\Big|_{\substack{u=u_l\\ y=y_l}}\cdot[\boldsymbol{y}(t-n_a+n_1)-\boldsymbol{y}_l]+\sum_{n_2=0}^{n_b}\boldsymbol{f}'_{n_2}\Big|_{\substack{u=u_l\\ y=y_l}}\cdot[\boldsymbol{u}(t-n_b+n_2)-\boldsymbol{u}_l]\\ +\boldsymbol{o}[\boldsymbol{x}(t)],\end{gathered} \tag{2}$$

where $\boldsymbol{f}'_{n_1}=\dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{y}(t-n_a+n_1)}$, $n_1=1,\cdots,n_a$, $\boldsymbol{f}'_{n_2}=\dfrac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}(t-n_b+n_2)}$, $n_2=0,\cdots,n_b$ and $\boldsymbol{x}(t)=[\boldsymbol{y}(t)-\boldsymbol{y}_l,\cdots;\boldsymbol{u}(t)-\boldsymbol{u}_l,\cdots]$ respectively. The nonlinear term $\boldsymbol{o}[\boldsymbol{x}(t)]$ satisfies $\lim\limits_{\|\boldsymbol{x}(t)\|\to 0}\dfrac{\|\boldsymbol{o}[\boldsymbol{x}(t)]\|}{\|\boldsymbol{x}(t)\|}=0$, where $\|\cdot\|$ is the Euclidean norm operator.

Define

$$\bar{\boldsymbol{y}}(t)=\boldsymbol{y}(t)-\boldsymbol{y}_l, \tag{3}$$

$$\bar{\boldsymbol{u}}(t) = \boldsymbol{u}(t) - \boldsymbol{u}_l , \tag{4}$$

$$\boldsymbol{v}(t) = \boldsymbol{o}[\boldsymbol{x}(t)] , \tag{5}$$

$$\boldsymbol{A}_{n_1}^l = (-1) \cdot \boldsymbol{f}'_{n_1} \Big|_{\substack{u=u_l \\ y=y_l}} , n_1 = 1, \cdots, n_a , \tag{6}$$

$$\boldsymbol{B}_{n_2}^l = \boldsymbol{f}'_{n_2} \Big|_{\substack{u=u_l \\ y=y_l}} , n_2 = 0, \cdots, n_b , \tag{7}$$

$$\boldsymbol{A}^l(z^{-1}) = \boldsymbol{I} + \boldsymbol{A}_1^l z^{-1} + \cdots + \boldsymbol{A}_{n_a}^l z^{-na} , \tag{8}$$

$$\boldsymbol{B}^l(z^{-1}) = \boldsymbol{B}_0^l + \boldsymbol{B}_1^l z^{-1} + \cdots + \boldsymbol{B}_{n_b}^l z^{-nb} . \tag{9}$$

Then system (2) can be rewritten as

$$\boldsymbol{A}^l(z^{-1})\bar{\boldsymbol{y}}(t+1) = \boldsymbol{B}^l(z^{-1})\bar{\boldsymbol{u}}(t) + \boldsymbol{v}(t) . \tag{10}$$

Remark 1. Although the representation of the system (10) is linear, the term $\boldsymbol{v}(t)$ is a nonlinear term. Then it is viewed as measurable disturbance and eliminated by using feedforward method.

3 Design of HMMNNDC

In the industrial process, on the one hand, the environment, where the system runs, is too diverse to satisfy the strict requirement which the nonlinear controller needs. On the other hand, the engineers are willing to employ easy linear control theory because of less mathematical knowledge. So a nonlinear system would always be expanded around the equilibrium point. If better performance is required, more equilibrium points should be needed. However, too many equilibrium points means too many models and too many computations. To solve this problem, a hierarchical structure is designed here.

3.1 Hierarchical Structure

For a nonlinear system, according to the prior information, the whole working area can be partitioned into many sub-regions, which is distinguished by a SOM NN. In each sub-region, at every equilibrium point, two BP NNs are employed to identify the linear and nonlinear term of the system (10). These two NNs compose one system model. All NNs of this sub-region compose the sub-region. At each instant, the best sub-region is selected first, and then, according to the switching index, the models in this sub-region are focused and the best model is chosen out from this sub-region (see Fig.1). To design the corresponding controller, the nonlinear term and the interactions is viewed as measurable disturbance and eliminated using the feedforward strategy.

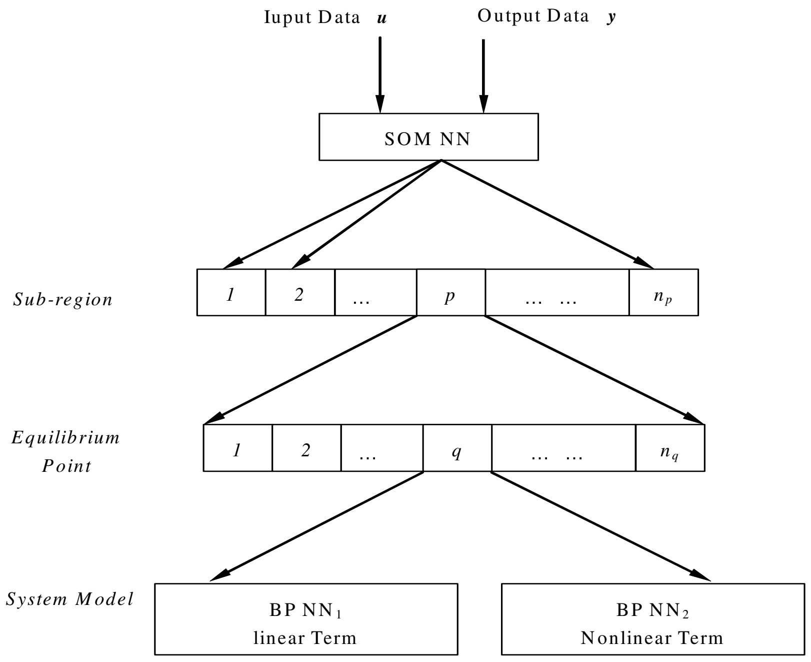


Fig. 1. Hierarchical structure of HMMNNDC

3.2 SOM NN

SOM NN is a NN which can transform an arbitrary dimensional continuous input signal into a lower dimensional discrete output signal preserving topological neighborhoods [13]. Here a SOM NN is employed as a first level of the hierarchical structure to represent the plant dynamics and map the different dynamic regimes into different sub-regions. It is designed as follows [13]

$$\boldsymbol{w}_i(t+1) = \begin{cases} \boldsymbol{w}_i(t) + \eta(t) h_i(t)\left[\boldsymbol{x}(t) - \boldsymbol{w}_i(t)\right], & i \text{ is selected} \\ \boldsymbol{w}_i(t), & otherwise \end{cases}, \tag{11}$$

where $\boldsymbol{x}(t)$ is the input signal consisting of $\boldsymbol{u}(t)$ and $\boldsymbol{y}(t)$, $\boldsymbol{w}(t)$ is the weighting vector, $\eta(t)$ is a learning rate and $h_i(k)$ is a typical neighborhood function, respectively.

3.3 Foundation of System Model

At each equilibrium point $(\boldsymbol{u}_l, \boldsymbol{y}_l)$ of the best sub-region, the system (10) is excited using white noise. One BP network NN_1 is trained off-line to approximate the system's input-output mapping. So $\hat{\boldsymbol{A}}^l(z^{-1})$ and $\hat{\boldsymbol{B}}^l(z^{-1})$, the estimation of the $\boldsymbol{A}^l(z^{-1})$ and $\boldsymbol{B}^l(z^{-1})$, are obtained [14]. Then another BP network, NN_2, is employed to approximate $\boldsymbol{v}(t)$ online, *i.e.*

$$\hat{\boldsymbol{v}}(t) = NN[\boldsymbol{W}, \boldsymbol{x}(t)] \ , \tag{12}$$

where $NN[\cdot]$ means the structure of the neural network and $\boldsymbol{W}$ is the weighting value. So the model at the equilibrium point $(\boldsymbol{u}_l, \boldsymbol{y}_l)$ is obtained. Similarly, the models at all equilibrium points in this sub-region can be set up.

3.4 The Switching Index

At each instant, to the models in the best sub-region, only one model is chosen as the system model according to the switching index, which has the form

$$J_l = \left\| \boldsymbol{e}^l(t) \right\|^2 = \left\| \boldsymbol{y}(t) - \boldsymbol{y}^l(t) \right\|^2 , \tag{13}$$

where $\boldsymbol{e}^l(t)$ is the output error between the real system and the model *l*. $\boldsymbol{y}^l(t)$ is the output of the model *l*. Let $j = \arg\min(J_l)$ correspond to the model whose output error is minimum, then it is chosen to be the best model.

3.5 Multiple Models Neural Network Decoupling Controller Design

For the best model, to realize the decoupling control, the interaction between the input $\bar{u}_j(t)$ and the output $\bar{y}_i(t)$,($j \neq i$) is viewed as measurable disturbance. Then (10) can be rewritten as

$$\boldsymbol{A}^l(z^{-1})\bar{\boldsymbol{y}}(t+1) = \bar{\boldsymbol{B}}^l(z^{-1})\bar{\boldsymbol{u}}(t) + \bar{\bar{\boldsymbol{B}}}^l(z^{-1})\bar{\boldsymbol{u}}(t) + \boldsymbol{v}(t) \ , \tag{14}$$

where $\bar{\boldsymbol{B}}^l(z^{-1}) = \mathrm{diag}\left[\boldsymbol{B}^l_{ii}(z^{-1})\right]$ is a diagonal polynomial matrix with a known nonsingular matrix $\bar{\boldsymbol{B}}^l_0$, $\bar{\bar{\boldsymbol{B}}}^l(z^{-1}) = \boldsymbol{B}^l(z^{-1}) - \bar{\boldsymbol{B}}^l(z^{-1})$. For a simple case, $\boldsymbol{A}^l(z^{-1})$ is assumed to be a diagonal matrix.

Because the nonlinear system is rewritten as a linear equation in (14), the linear adaptive decoupling controller can be designed to control the system, in which the nonlinear term is viewed as measurable disturbance and eliminated with the interaction by the choice of the polynomial matrices. Like the conventional optimal controller design, for the model j , the cost function is of the form

$$J_c = \left\| \boldsymbol{P}(z^{-1})\bar{\boldsymbol{y}}(t+k) - \boldsymbol{R}(z^{-1})\boldsymbol{w}(t) + \boldsymbol{Q}(z^{-1})\bar{\boldsymbol{u}}(t) + \boldsymbol{S}_1(z^{-1})\bar{\boldsymbol{u}}(t) + \boldsymbol{S}_2(z^{-1})\boldsymbol{v}(t) \right\|^2 , \tag{15}$$

where $\boldsymbol{w}(t)$ is the known reference signal, $\boldsymbol{P}(z^{-1}), \boldsymbol{Q}(z^{-1}), \boldsymbol{R}(z^{-1}),\ \boldsymbol{S}_1(z^{-1}),\ \boldsymbol{S}_2(z^{-1})$ are weighting polynomial matrices respectively. Introduce the identity as

$$\boldsymbol{P}(z^{-1}) = \boldsymbol{F}(z^{-1})\boldsymbol{A}(z^{-1}) + z^{-1}\boldsymbol{G}(z^{-1}) \ . \tag{16}$$

Multiplying (14) by $\boldsymbol{F}(z^{-1})$ from left and using (16), the optimal control law can be derived as follows

$$\begin{aligned} &\boldsymbol{G}(z^{-1})\bar{\boldsymbol{y}}(t) + [\boldsymbol{F}(z^{-1})\bar{\boldsymbol{B}}(z^{-1}) + \boldsymbol{Q}(z^{-1})]\bar{\boldsymbol{u}}(t) + [\boldsymbol{F}(z^{-1})\bar{\bar{\boldsymbol{B}}}(z^{-1}) + \boldsymbol{S}_1(z^{-1})]\bar{\boldsymbol{u}}(t) \\ &\qquad + [\boldsymbol{F}(z^{-1}) + \boldsymbol{S}_2(z^{-1})]\boldsymbol{v}(t) = \boldsymbol{R}\boldsymbol{w}(t) \ , \end{aligned} \tag{17}$$

combing (17) with (14), the closed loop system equation is obtained as follows

$$\begin{aligned}\left[\boldsymbol{P}(z^{-1})+\boldsymbol{Q}(z^{-1})\overline{\boldsymbol{B}}^{-1}(z^{-1})\boldsymbol{A}(z^{-1})\right]\boldsymbol{y}(t+1) &= \left[\boldsymbol{Q}(z^{-1})\overline{\boldsymbol{B}}(z^{-1})^{-1}\overline{\overline{\boldsymbol{B}}}(z^{-1})-\boldsymbol{S}_1(z^{-1})\right]\overline{\boldsymbol{u}}(t)\\ &\left[\boldsymbol{Q}(z^{-1})\overline{\boldsymbol{B}}(z^{-1})^{-1}-\boldsymbol{S}_2(z^{-1})\right]\boldsymbol{v}(t)+\boldsymbol{R}(z^{-1})\boldsymbol{w}(t)\,.\end{aligned} \tag{18}$$

To eliminate the nonlinear form and the interactions of the system exactly, let

$$\boldsymbol{Q}(z^{-1})=\boldsymbol{R}_1\overline{\boldsymbol{B}}(1)\,, \tag{19}$$

$$\boldsymbol{S}_1(z^{-1})=\boldsymbol{R}_1\overline{\overline{\boldsymbol{B}}}(1)\,, \tag{20}$$

$$\boldsymbol{S}_2(z^{-1})=\boldsymbol{R}_1\,, \tag{21}$$

$$\boldsymbol{R}(z^{-1})=\boldsymbol{P}(1)+\boldsymbol{R}_1\boldsymbol{A}(1)\,, \tag{22}$$

where $\boldsymbol{R}_1$ is a constant matrix and decided by the designer to guarantee the stability of the closed loop system. So $\boldsymbol{P}(z^{-1}),\boldsymbol{R}_1$ are selected off-line to satisfy

$$\left|\boldsymbol{B}(z^{-1})\boldsymbol{B}^{-1}(1)\boldsymbol{R}_1^{-1}\boldsymbol{P}(z^{-1})+\boldsymbol{A}(z^{-1})\right|\neq 0 \quad |z|>1\,. \tag{23}$$

Although the second $\overline{\boldsymbol{u}}(t)$ is the interaction of the system and viewed as measurable disturbance, to obtain the control input, it must be included. So the control law is rewritten from (17) as

$$\begin{aligned}&[\boldsymbol{F}(z^{-1})\boldsymbol{B}(z^{-1})+\boldsymbol{Q}(z^{-1})+\boldsymbol{S}_1(z^{-1})]\overline{\boldsymbol{u}}(t)+\\ &\boldsymbol{G}(z^{-1})\boldsymbol{y}(t)+[\boldsymbol{F}(z^{-1})+\boldsymbol{S}_2(z^{-1})]\boldsymbol{v}(t)=\boldsymbol{R}\boldsymbol{w}(t)\,.\end{aligned} \tag{24}$$

Remark 2. Equation (24) is a nonlinear equation because $\overline{\boldsymbol{u}}(t)$ is include into the nonlinear term $\boldsymbol{v}(t)$. Considering $\overline{\boldsymbol{u}}(t)$ will converge to a constant vector in steady state, then substitute $\overline{\boldsymbol{u}}(t)$ in the nonlinear term $\boldsymbol{v}(t)$ with $\overline{\boldsymbol{u}}(t-1)$ and solve (24).

4 Simulation Studies

A discrete-time nonlinear multivariable system is described as follows

$$\begin{cases} y_1(t+1)=\dfrac{-0.2y_1(t)}{1+y_1^2(t)}+\sin[u_1(t)]-0.5\sin[u_1(t-1)]+1.5u_2(t)+0.2u_2(t-1)\\ y_2(t+1)=0.6y_2(t)+0.2u_1(t)+1.3u_1(t-1)+u_2(t)+u_2^2(t)+\dfrac{1.5u_2(t-1)}{1+u_2^2(t-1)}\end{cases} \tag{25}$$

which is the same as the simulation example in [12]. The known reference signal $\boldsymbol{w}$ is set to be a time-varying signal. When $t=0$, w_1 equals to 0 and when t is 40, 80, 120, 160, 200, it changed into 0.05, 0.15, 0.25, 0.35, 0.45 respectively, while w_2 equals to 0 all the time.

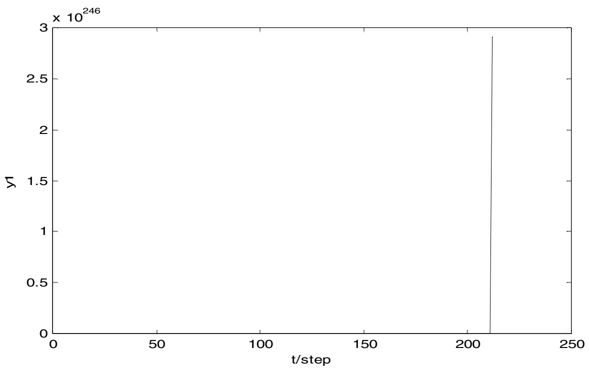


Fig. 2. The output $y_1(t)$ of NNDC

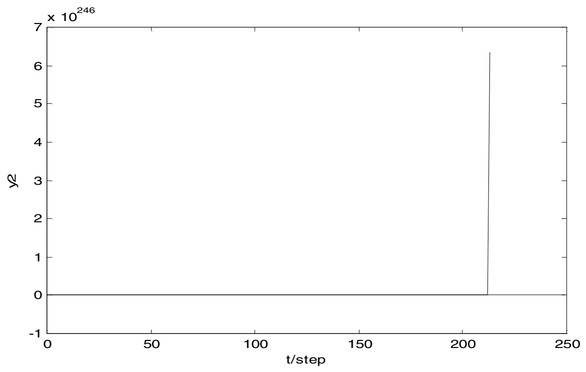


Fig. 3. The output $y_2(t)$ of NNDC

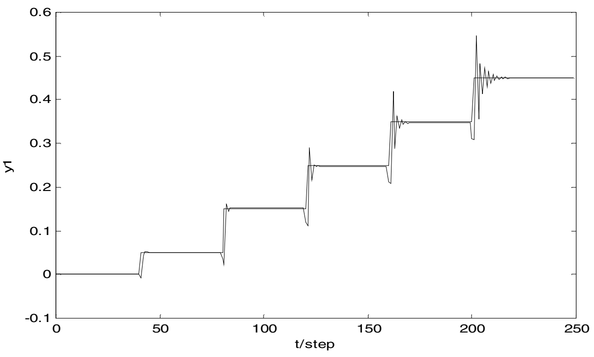


Fig. 4. The output $y_1(t)$ of HMMNNDC

In Fig.2 and 3, the system (25) is expanded only at the original point (0,0) and a Neural Network Decoupling Controller (NNDC) is used. In Fig.4 and 5, the system is

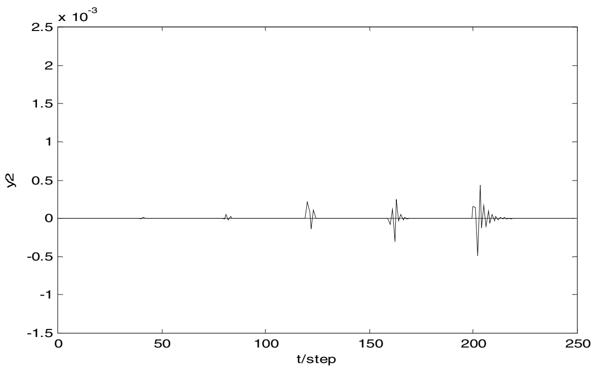


Fig. 5. The output $y_2(t)$ of HMMNNDC

expanded at six equilibrium points, *i.e.* $[0,0]^T$, $[0.1,0]^T$, $[0.2,0]^T$, $[0.3,0]^T$, $[0.4,0]^T$ and. Note that the equilibrium points are far away from the set points. The results show that although the same NNDC method is adopted, the system using NNDC loses its stability (see Fig.2 and 3), while the system using HMMNNDC not only gets the good performance but also has good decoupling result (see Fig.4 and 5).

5 Conclusion

A HMMNNDC is designed to control the discrete-time nonlinear multivariable system. A SOM NN is employed to partition the whole working area into several sub-regions. In each sub-region, around each equilibrium point, one NN is trained offline to identify the linear term of the nonlinear system and the other NN is trained online to identify the nonlinear one. The multiple models set is composed of all models, which are got from all equilibrium points. According to the switching index, the best model is chosen as the system model. The nonlinear term and the interaction of the system are viewed as measurable disturbance and eliminated using feedforward strategy.

References

1. Wang X., Li S.Y., *et al.*: Multi-model Direct Adaptive Decoupling Control with Application to the Wind Tunnel. ISA Transactions. 44 (2005) 131–143
2. Lin Z.L.: Almost Disturbance Decoupling with Global Asymptotic Stability for Nonlinear Systems with Disturbance-affected Unstable Zero Dynamics. Systems & Control Letters. 33 (1998) 163–169
3. Lin Z.L., Bao X.Y., Chen B.M.: Further Results on Almost Disturbance Decoupling with Global Asymptotic Stability for Nonlinear Systems. Automatica. 35 (1999) 709–717
4. Ansari R.M., Tade M.O.: Nonlinear Model-based Process Control: Applications in Petroleum Refining. Springer. London (2000)
5. Khail H.K.: Nonlinear Systems. Prentice Hall Inc. New Jersey (2002)

6. Germani A., Manes C., Pepe P.: Linearization and Decoupling of Nonlinear Delay Systems. Proceedings of the American Control Conference. (1998) 1948 –1952
7. Wang W.J., Wang C.C.: Composite Adaptive Position Controller for Induction Motor Using Feedback Linearization. IEE Proceedings D Control Theory and Applications. 45 (1998) 25–32
8. Wai R.J., Liu W.K.: Nonlinear Decoupled Control for Linear Induction Motor Servo-Drive Using The Sliding-Mode Technique. IEE Proceedings D Control Theory and Applications. 148 (2001) 217–231
9. Balchen J.G., Sandrib B.: Elementary Nonlinear Decoupling Control of Composition in Binary Distillation Columns. Journal of Process Control. 5 (1995) 241–247
10. Haykin S.S.: Neural Networks: A Comprehensive Foundations. Prentice Hall Inc. New Jersey (1999)
11. Ho D.W.C., Ma Z.: Multivariable Nnternal Model Adaptive Decoupling Controller with Neural Network for Nonlinear Plants. Proceedings of the American Control Conference. (1998) 532–536
12. Yue H., Chai T.Y.: Adaptive Decoupling Control of Multivariable Nonlinear Non-Minimum Phase Systems Using Neural Networks. Proceedings of the American Control Conference. (1998) 513–514
13. Kohonen T.: Self-Organizing Feature Maps. Springer-Verlag. New York (1995)
14. Hornik K., Stinchcombe M., White H.: Universal Approximation of an Unknown Mapping and Its Derivatives using Multilayer Feedforward Networks. Neural Networks. 3 (1990) 551–560.

Sensorless Control of Switched Reluctance Motor Based on ANFIS*

Chang-Liang Xia and Jie Xiu

School of Electrical Engineering and Automation, Tianjin University,
300072 Tianjin, China
clxia@tju.edu.cn

Abstract. The accurate rotor position information is very important for high performance operation of switched reluctance motor (SRM). Traditionally, there is a mechanical rotor position sensor. But this reduces the reliability and increases cost and size of SRM. In order to overcome the disadvantage of mechanical rotor position sensor, a sensorless operation method of SRM based on adaptive network based fuzzy inference system (ANFIS) is developed in this paper. The rotor position can be estimated by using the unique relationship between rotor position, flux linkage and phase current. In this paper, the ANFIS is used to map this relationship. Among the sensorless position estimation method, approach based on fuzzy neural network (FNN) is one of the promising methods. By combining the benefits of artificial neural network (ANN) and fuzzy inference system (FIS) in a single model, the ANFIS shows characteristics of fast and accurate learning, the ability of using both linguistic information and data information and good generalization capability. For its antecedents are fuzzy sets, the noise in the input signals can be restrained. This approach shows a characteristic of robustness. For its consequent is in linear function of input variables, it has a simple structure and low computation complexity. So, it is well suited to be applied on-line. Applying it to the rotor position estimation, a high accuracy and robust sensorless rotor position estimator is presented. The experimental results proved the effectiveness of the proposed method.

1 Introduction

Switched reluctance motor drive is a novel variable speed drive. It has a wide application in industrial and in home appliances. The SRM is characterized by simplicity and low cost. It has high efficiency and large torque output over a wide speed range. It also presents smaller size and lower weight comparatively to the induction machine[1].

The rotor position information is very important for high performance operation of SRM. For the energize of the SRM phases needs to be synchronized with rotor position to obtain desired control of speed and torque. In a conventional SRM, there has a mechanical position sensor. This mechanical position sensor reduces the reliability and increases the complexity of electrical connection, size and cost of SRM. In order to overcome the disadvantage of mechanical position sensor, position sensorless operation of SRM has become an attractive research region. Various position sensorless

* This work is supported by Natural Science Foundation of Tianjin.

control methods have been developed. Such as method using the impressed voltage pulse technique [2] and improved voltage pulse technique sensorless control [3], method based on fuzzy logic [4], method based on artificial neural network [5]-[8]. In summary, those methods can be classified as tow groups. One is hardware detecting approach. Another is computational approach based on relationship between rotor position, current and flux linkage of SRM. The former may produce negative torque and has a restrain to the speed range. The later is a purely electrical measurement approach which needs only current and voltage signals and does not need extra mechanical measurement equipment. For ANFIS having capability of using both linguistic information and numerical information and the advantage of simple structure, fuzzy antecedents, linear consequent, good generalization ability and fast and accuracy learning capability, the rotor position can be estimated by using ANFIS to map the relationship between rotor position, flux linkage and current.

2 Structure and Rotor Position Estimation Method of SRM

The SRM is a double salient motor. It only has concentrated winding on the stator pole. There is no winding or magnet on the rotor pole. The structure of a typical SRM with eight stator and six rotor poles is shown in Fig. 1. The motor is driven by the reluctance torque produced by the electromagnetic field excited by the current flow in the stator winding. By consecutive energization of successive phases, continuous rotation in either direction is possible. The idealized inductance profile for one phase of a SRM is shown in Fig. 2. Meanwhile different current profile corresponding to different firing angle is also shown in Fig. 2. From Fig. 2, it can be seen that the firing angle has a great influence on the amplitude of phase current. For amplitude of current corresponding to the amplitude of torque, the tuning of firing angle can adjust the torque and speed of SRM. In the direction of increasing flux linkage, the positive torque or the motor torque is created. While in the direction of decreasing flux linkage, the negative torque or generating torque is created. The phase windings are switched on and off at appropriate instances to ensure the desired direction of rotation with desired torque and speed. Meanwhile the rotor position information is used to generate correct commutation sequence and instants. So, the accurate information of rotor position is very important for operation of SRM.

The traditional measurement method of rotor position is by using a mechanical position sensor. This increases cost, size and electrical connection complexity of SRM. Especially for some application, elimination of mechanical position sensor can improve reliability and reduce cost. For the double salient structure of SRM, the flux linkage is in function of both phase current and rotor position which is shown in Fig. 3. From the unique relationship between flux linkage, phase current and rotor position, the rotor position can be estimated. In this paper, the ANFIS is used to map the relationship between rotor position, flux linkage and phase current. After the model based on ANFIS established, the measured phase current and calculated flux linkage are input to the ANFIS to estimate the rotor position. An advantage of this method is that error will not be accumulated for the flux linkage returns to zero at each cycle allowing the integrator to be reset. Among the sensorless position estimation method, FNN based approach is one of the promising method. One of FNN – the ANFIS has the advantage of high accuracy and fast computational speed. It also shows a robust characteristic. The application of ANFIS to estimate the rotor position of SRM can present a good performance. The speed control range will be extended further. It has the ability to be tolerant to input signal noise and error. This increases the reliability and robustness of this approach.

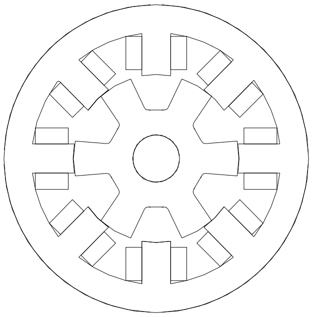


Fig. 1. Structure of SRM with 8/6 poles

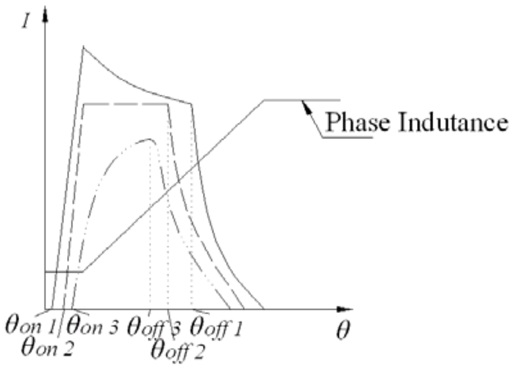


Fig. 2. Different turn-on and turn-off angle for different current

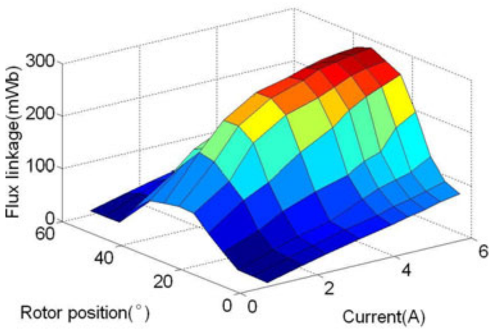


Fig. 3. Mapping surface of flux linkage with respect to rotor position and phase current

3 The Mathematical Model of SRM

The basic mathematical model equations to describe the dynamics of a SRM are given as follows.

1) Terminal Voltage Equations: The terminal voltage equations can be easily written according to the circuit configuration of the system, the k^{th} phase equations can be written as

$$U_k = R_k i_k + \frac{d\psi_k}{dt} \cdot \tag{1}$$

where U_k is terminal voltage of k^{th} phase; R_k is the resistance of k^{th} phase winding; i_k is the current flow in the k^{th} phase winding; Ψ_k is the flux linkage of k^{th} phase winding.

For the mutual flux linkage of SRM is very small, so the mutual flux linkage can be neglected for the convenient of analysis. When the mutual flux linkage and saturation are neglected, the flux linkage of one phase of SRM can be calculated from the phase current and voltage of an energized phase of the SRM which can be expressed as

$$\psi_k = \int (U_k - R_k i_k) dt \cdot \tag{2}$$

2) Mechanical Equation: The balance equation between the electromagnetic torque and the load torque can be expressed as

$$T_e = J\frac{d^2\theta}{dt^2} + D\frac{d\theta}{dt} + T_L \cdot \tag{3}$$

where T_e is the electromagnetic torque; J is the moment of inertia of the rotor; D is the coefficient of viscous friction; θ is rotor position; T_L is the load torque.

3) Torque Production: Due to the saturation of magnetic circuit, the torque is also in nonlinear function of rotor position and phase current. It is also hard to be analytically calculated. The instantaneous torque at any rotor position can be obtained by derivative of co-energy with respect to rotor position θ. This can be expressed as

$$T_e = \left.\frac{\partial W'}{\partial \theta}\right|_{i=CONST} \cdot \tag{4}$$

where $W' = \int_0^i \psi di$ is the co-energy of one phase.

4 Adaptive Network Based Fuzzy Inference System

By combining fuzzy inference system and neural network (NN) in a single model and taking advantage of the ability of using the expert knowledge and the strong learning ability of NN, the ANFIS shows high approaching precision, fast convergence speed and simple structure [9]. It has a wide application in modeling, control and function approaching.

The typical structure of an ANFIS is shown in Fig. 4. Where a circle indicates a fixed node, a square indicates an adaptive node. For simplicity, there are only two inputs *x*, *y* and one output *z*. A first order Takagi and Sugeno's (T-S) type fuzzy rules are used in this paper, which can be expressed as

Rule 1: If x is A_1 and y is B_1, then $z_1=p_1x+q_1y+r_1$
Rule 2: If x is A_2 and y is B_2, then $z_2=p_2x+q_2y+r_2$

where A_i and B_i are the fuzzy sets in the antecedent; p_i, q_i, r_i are the consequent parameters.

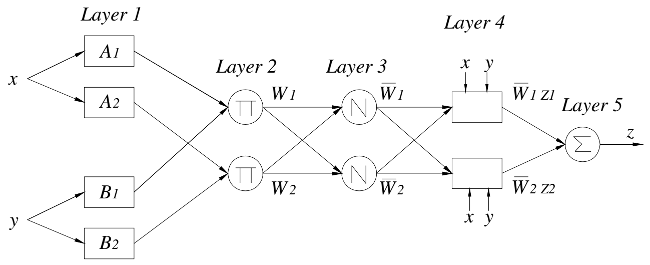


Fig. 4. Structure of ANFIS

As shown in Fig. 4, the ANFIS has five layers:

Layer 1: Every node *i* in this layer transform the crisp values to a fuzzy one

$$\begin{aligned} O_i^1 &= \mu_{Ai}(x), \qquad i=1,2 \cdot \\ O_i^1 &= \mu_{Bi}(x), \qquad i=3,4 \cdot \end{aligned} \tag{5}$$

where $\mu_{Ai}(x)$ and $\mu_{Bi}(x)$ are the membership function (MF). In this paper, the Gaussian function is used as the MF

$$\mu(x)=e^{-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)^2} . \tag{6}$$

where c and σ are the parameters which determine the shape of the MF. The parameters in this layer are referred to as the antecedent parameters.

Layer 2: Every node in this layer calculates the firing strength of a rule by multiplying the degree of MF of input signals as

$$w_i = \mu_{Ai}(x)\mu_{Bi}(y), \qquad i=1,2 \cdot \tag{7}$$

Layer 3: The i^{th} node in this layer calculates the ratio of the i^{th} rule's firing strength to the sum of all rules' firing strengths:

$$\overline{w_i} = \frac{w_i}{w_1+w_2}, \qquad i=1,2 \cdot \tag{8}$$

where w_i are called the normalized firing strengths.

Layer 4: In this layer, every node has the function

$$O_i^4 = \overline{w}_i z_i = \overline{w}_i \left(p_i x + q_i y + r_i \right), \qquad i = 1, 2 . \tag{9}$$

where w_i is the output of layer3, and $\{p_\text{i}, q_\text{i}, r_\text{i}\}$ is the parameter set.

Layer 5: The single node in this layer computes the overall output as the summation of all incoming signals, which can be expressed as

$$O_i^5 = \sum_{i=1}^{2} \overline{w}_i z_i = \frac{w_1 z_1 + w_2 z_2}{w_1 + w_2} . \tag{10}$$

To minimize the error between the output of the ANFIS and the desired output, the hybrid learning algorithm is employed in this paper which combines the least square method (LSM) and the back propagation (BP) algorithm to train rapidly and adapt the ANFIS. The consequent parameters are identified using the LSM when the values of the premise parameters are fixed. Whereas the antecedent parameters are updated by the stand BP algorithm while by holding the consequent parameters fixed and the error propagated from the output end to the input end.

5 Experimental Results

The characteristics of SRM can be described by a two input (flux linkage and current) – one output (rotor position angle) function. This function can be modeled by analytical mathematical equation. But this is a hard and complex task. Also, the precision of this method may be not very high. However, instead of using complex mathematical equations to describe this function, ANFIS provide a simple way of modeling which can take into consideration the static and dynamic effects of the motor. The training data is defined as a two input one output data pairs. The total number of data for training is 105. Each point of measured data presented to the ANFIS are given as

$$\left(\psi^{(n)}, i^{(n)}, \theta^{(n)} \right)$$

where n is the n^{th} data pair; ψ is flux linkage; i is current; θ is rotor position angle. The model used for calculating the rotor position is shown in Fig. 5.

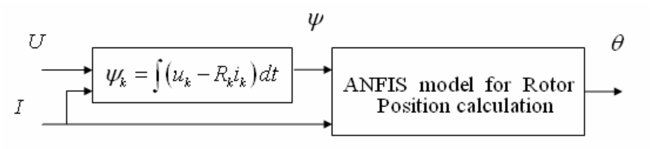
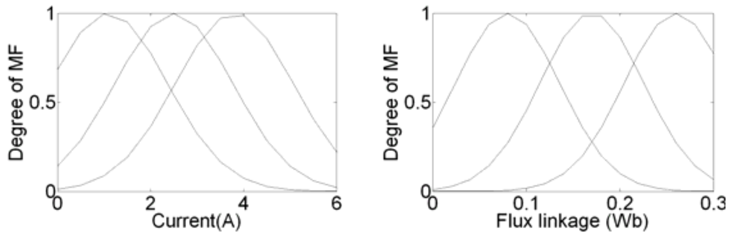

U
I
$\psi_k = \int (u_k - R_k i_k) dt$
ψ
ANFIS model for Rotor Position calculation
θ

Fig. 5. ANFIS model for rotor position calculation

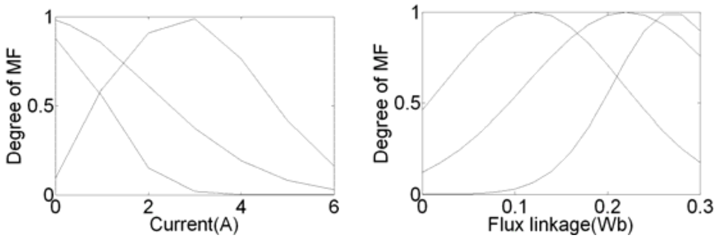
To establish the model for rotor position estimation, in this paper, for each input three fuzzy sets are used for fuzzification. The number of rules is then 9. The MF of which is a Gaussian type function as Eq. (6). It can be seen that the Gaussian MF is specified by two parameters. Therefore, the number of parameters to be optimized is 39, of which 12 are the antecedent parameters and 27 are the consequent parameters.

The initial values of MF are predetermined by analyzing the measured data. The distribution of the initial MF covering the whole universe is shown in Fig. 6(a).

After training, the degrees of MF are shown in Fig. 6(b). From Fig. 6, it can be seen that by using ANFIS, the MF can be adjusted automatically in the training process. This makes the distribution of the MF more reasonable. So, the precision of the model can be improved.



(a) Initial membership functions of input variables



(b) Membership function of input variables after training

Fig. 6. Membership function of input variables

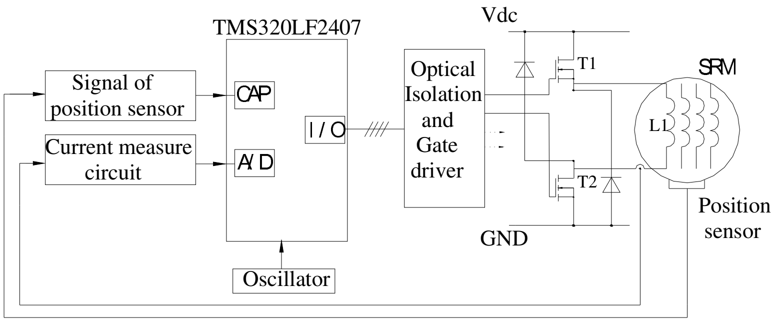
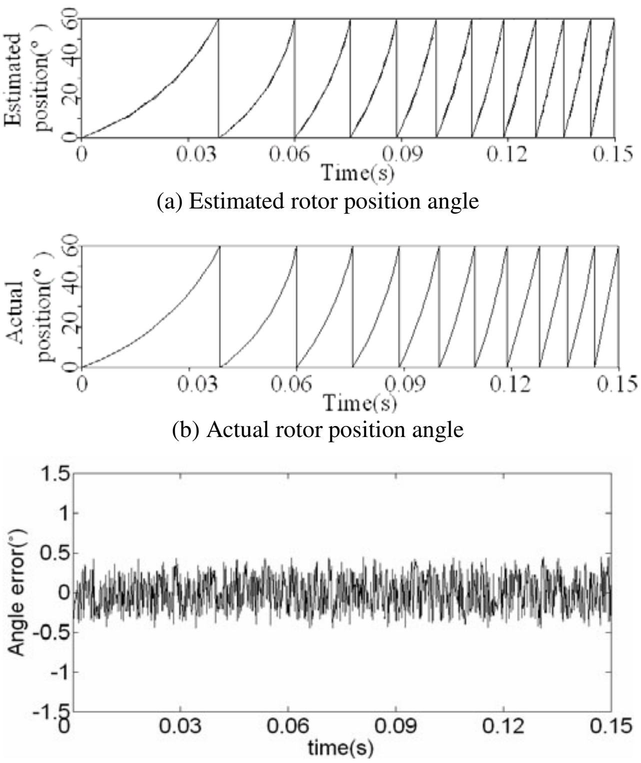


Fig. 7. Hardware configuration

After the model established, it is applied on-line to estimate the rotor position. The proposed approach is implemented on a TMS320LF2407 type DSP. The hardware configuration is shown in Fig. 7. It consists of several distinct sub system: the DSP, the power converter, the 8/6 SRM and the optical isolation gate driver. Firstly, the initially phase is selected to be energized. Then during running condition, the flux linkage calculated according to Eq. (2) and the phase current are fed to the ANFIS to calculate the rotor position angle. The estimated rotor position angle is shown in Fig. 8 (a). While the actual measured rotor position angle is shown in Fig. 8 (b). The error between actual and estimated rotor position angle is shown in Fig. 8 (c). From Fig. 8, it can be seen that the maximum error is less than 0.5°. The proposed approach has a high precision.



(c) Error between estimated and actual rotor position angle

Fig. 8. Rotor position angle

Currents in one of the four phase can be seen for this transient start up case in Fig. 9.

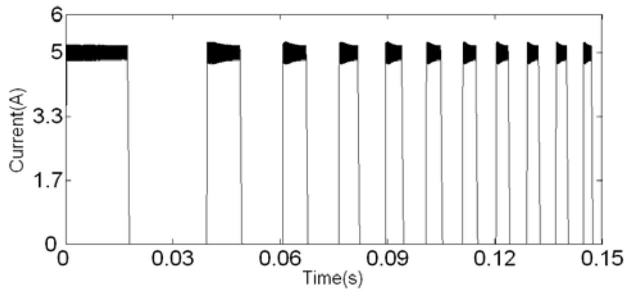


Fig. 9. Phase current at transient start up case

6 Conclusion

In this paper, an ANFIS based high precision and robust sensorless rotor position estimation method is developed. By taking advantage of benefits of T-S type FIS and ANN, the ANFIS presents a superior performance when applied to estimate the rotor position of SRM. The ANFIS can approximate any nonlinear function with arbitrary precision. It has the ability to be tolerant to input signal noise and error. This increases the reliability and robustness of this approach. For the consequent of the ANFIS is in a linear function of input variables, the structure of the ANFIS is simple and the computational complexity is low. This makes it suitable to be applied on line. Compared with rotor position estimation method based on FIS and ANN, the rotor position estimation method based on ANFIS developed in this paper has the advantage of high precision and robustness. In order to estimate the rotor position, firstly a model based on ANFIS for mapping the nonlinear relationship between rotor position, flux linkage and phase current is established off-line. Then it is applied on-line to estimate the rotor position. Experimental results show that a high accuracy and robust rotor position estimator is achieved.

References

1. Lawrenson P T ed.: Variable-speed switched reluctance motors. IEE Proc. Vol. 127(B), No. 4, (1980) 253–265
2. Li Jingnan, Wang Xudong, Zhou Yongqin: Sensorless rotor position detection of SRM based on voltage pulses to two phases. Electric Machines and Control, Vol. 6, No. 1, (2002) 6–9
3. Guo, H.J.; Takahashi, M.; Watanabe, T.; Ichinokura, O.: A new sensorless drive method of Switched Reluctance Motors based on motor's magnetic characteristics. IEEE Transactions on Magnetics, Vol. 37, No. 4, (2001) 2831–2833
4. Zhongfang Wang; Cheok, A.D.; Lim Khiang Wee: Sensorless rotor position estimation algorithm for switched reluctance motors using fuzzy logic. IEEE 32nd Annual Power Electronics Specialists Conference, Vol. 3, (2001) 1701–1706
5. Reay, D.S.; Dessouky, Y.; Williams, B.W.: The use of neural networks to enhance sensorless position detection in switched reluctance motors. 1998 IEEE International Conference on Systems, Man, and Cybernetics, Vol. 2, (1998) 1774–1778
6. Hudson, C.; Lobo, N.S.; Krishnan, R.: Sensorless control of single switch based switched reluctance motor drive using neural network. IEEE Conference of Industrial Electronics Society, Vol. 3, (2004) 2349–2354
7. Won-Sik Baik; Min-Huei Kim; Nam-Hun Kim et al: Position sensorless control system of SRM using neural network. IEEE 35th Annual Power Electronics Specialists Conference, Vol. 5, (2004) 3471–3475
8. Bellini A., Filippetti F., Franceschini G. et al: Position sensorless control of a SRM drive using ANN-techniques. The IEEE Industry Applications Conference, Vol.1, (1998) 709–714
9. Jang, J.-S.R.: ANFIS: adaptive-network-based fuzzy inference system. IEEE Transactions on Systems, Man and Cybernetics, Vol. 23, No. 3, (1993) 665–685

Hybrid Intelligent PID Control for MIMO System

Jih-Gau Juang[1], Kai-Ti Tu, and Wen-Kai Liu

Department of Communications and Guidance Engineering
National Taiwan Ocean University, Keelung, Taiwan, ROC
[1] jgjuang@mail.ntou.edu.tw

Abstract. This paper presents a new approach using switching grey prediction PID controller to an experimental propeller setup which is called the twin rotor multi-input multi-output system (TRMS). The goal of this study is to stabilize the TRMS in significant cross coupling condition and to experiment with set-point control and trajectory tracking. The proposed scheme enhances the grey prediction method of difference equation, which is a single variable second order grey model (DGM(2,1) model). It is performed by real-value genetic algorithm (RGA) with system performance index as fitness function. We apply the integral of time multiplied by the square error criterion (ITSE) to form a suitable fitness function in RGA. Simulation results show that the proposed design can successfully adapt system nonlinearity and complex coupling condition.

1 Introduction

PID is the control algorithm most often used in industrial control. It is implemented in industrial single loop controllers, distributed control systems, and programmable logic controllers. There are two reasons why it is the majority in industrial process. The first reason is its simple structure and the well-known Ziegler and Nichols tuning algorithms which have been developed [1-2] and successfully used for years. The second reason is that the controlled processes in industrial plant can almost be controlled through the PID controller [3-4]. The drawback is that the parameters of the PID controller are partially tuned by trial and error process, which makes it less intelligent. To overcome this problem, some intelligent computation techniques such as genetic algorithm or grey system may provide a solution for it.

The concept of grey system was firstly proposed by Deng from the control problem of unknown systems in 1981 [5]. Next year, he proposed the paper of grey control system which made the grey system research to be studied widely later [6]. Grey theory mainly integrates key concepts of system theory, information theory, and control theory. Grey system theory has been widely utilized in the system modeling, information analysis, and prediction fields. Grey prediction controller was proposed by Cheng in 1986 [7]. The large prediction step using in grey prediction controller usually causes worse transient response, but small prediction step yields fast response. Therefore, the forecasting step-size in the grey prediction controller can be switched according to the error of system during different periods of the response. It differs from the traditional techniques, in which its prediction still can be generated by grey model construction stage even facing unknown systems. Therefore, many approaches of grey prediction controllers have been developed in recent years [8-9].

In this paper, a switching grey prediction PID controller is utilized, which is combined with real-value genetic algorithms (RGA) [10]. The parameters of the controller are tuned by RGA which fitness function is formed by specific index such as System Performance Index [11]. The System Performance Index is an optimization for parameters tuning of control system. A modified system performance index of the known integral of time multiplied by squared error criterion (ITSE) is used in this paper. The strict model of mathematics is needless with respect to the searching parameters of PID controller which is obtained by RGA, or the structure of grey prediction controller which is obtained by switching mechanism. For this reason the design process is more concise and the controller is more robust to the traditional controller. According to simulation results, the new approach improves the performances in set-point control and trajectory tracking.

2 System Description

The TRMS, as shown in Fig. 1, is characterized by complex, highly nonlinear and inaccessibility of some states and outputs for measurements, and hence can be considered as a challenging engineering problem [12]. The control objective is to make the beam of the TRMS move quickly and accurately to the desired attitudes, both the pitch angle and the azimuth angle in the condition of decoupling between two axes. The TRMS is a laboratory set-up for control experiment and is driven by two DC motors. Its two propellers are perpendicular to each other and joined by a beam pivoted on its base that can rotate freely in the horizontal and vertical plane. The joined beam can be moved by changing the input voltage to control the rotational speed of these two propellers. There is a Pendulum Counter-Weight hanging on the joined beam which is used for balancing the angular momentum in steady state or with load. In certain aspects its behavior resembles that of a helicopter. It is difficult to design a suitable controller because of the influence between two axes and nonlinear movement. From the control point of view it exemplifies a high order nonlinear system with significant cross coupling. For easy demonstration in the vertical and horizontal separately, the TRMS is decoupled by the main rotor and tail rotor.

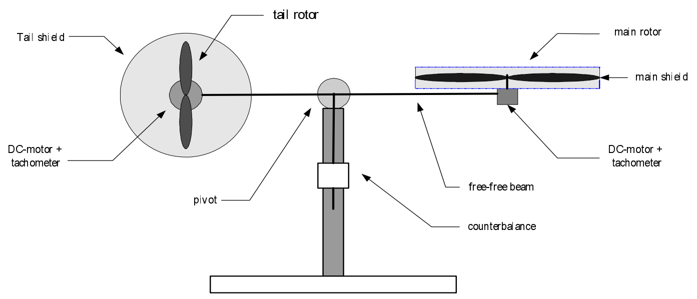


Fig. 1. The laboratory set-up TRMS

A block diagram of the TRMS model is shown in Fig. 2, where M_v is the vertical tuning moment, J_v is the moment of inertia with respect to horizontal axis, α_v is the vertical position (pitch position) of TRMS beam, l_m is the arm of aerodynamic force from main rotor, l_t is the effective arm of aerodynamic force from tail rotor, g is the acceleration of gravity, ω_m is the rotational speed of main rotor, $F_v(w_m)$ is the nonlinear function of aerodynamic force from main rotor, k_v is the moment of friction force in horizontal axis, Ω_v is the angular velocity (pitch velocity) of TRMS beam, Ω_h is the angular velocity (azimuth velocity) of TRMS beam, α_h is the horizontal position (azimuth velocity) of TRMS beam, M_h is the horizontal turning torque, J_h is the nonlinear function of moment of inertia with respect to vertical axis, ω_t is the rotational speed of tail speed, $F_h(\omega_t)$ is the nonlinear function of aerodynamic force from tail rotor, k_h is the moment of friction force in horizontal axis, J_{tr} is the vertical angular momentum from tail rotor, J_{mr} is the vertical angular momentum from main rotor, S_v is the vertical turning moment, S_h is the horizontal turning moment, S_f is the balance factor, U_v and U_h are the DC-motor control inputs. In order to control TRMS in the vertical and horizontal separately, the main rotor and tail rotor are decoupled. Detail mathematical model can be found in [12].

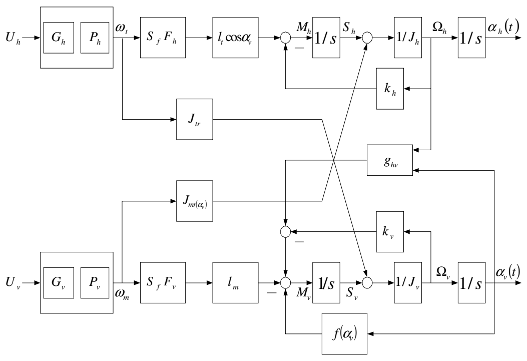


Fig. 2. Block diagram of TRMS model

3 Control System Design

The proposed control scheme utilizes three techniques to control the TRMS, which are grey prediction with second order difference equation of DGM(2,1) model, switching grey prediction, and RGA. The structure is shown in Fig. 3. The steps of grey prediction modeling can be summarized as follows [13].

Let $y^{(0)}$ be a non-negative original data sequence, then

$$y^{(0)} = \{y^{(0)}(1), y^{(0)}(2), \ldots, y^{(0)}(n)\}, \ n \geq 4. \tag{1}$$

Take the accumulated generating operation (AGO) on $y^{(0)}$, then the AGO sequence $y^{(1)}$ can be described by

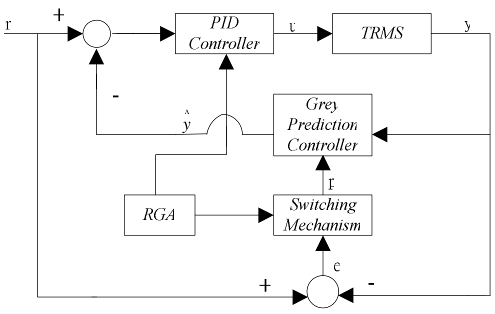


Fig. 3. The block diagram of the switching grey prediction PID control

$$y^{(1)}(k) = AGO \circ y^{(0)} = \sum_{m=1}^{k} y^{(0)}(m), k = 1,2,..., n. \tag{2}$$

The DGM(2,1) model is expressed as

$$y^{(1)}(k+2) + ay^{(1)}(k+1) + by^{(1)}(k) = 0\ , \tag{3}$$

where the *a* and *b* are undecided coefficients of second order difference equation. The parameters *a* and *b* can be solved by means of the least square method as follows

$$\begin{bmatrix} a \\ b \end{bmatrix} = (B^T B)^{-1} B^T Y \tag{4}$$

where

$$Y = \begin{bmatrix} y^{(1)}(3) \\ y^{(1)}(4) \\ \vdots \\ y^{(1)}(n) \end{bmatrix} \quad B = \begin{bmatrix} -y^{(1)}(2) & -y^{(1)}(1) \\ -y^{(1)}(3) & -y^{(1)}(2) \\ \vdots & \vdots \\ -y^{(1)}(n-1) & -y^{(1)}(n-2) \end{bmatrix}$$

Take the Z transform for (3); we obtain the following characteristic equation

$$\left(z^2 + az + b\right) = 0 \tag{5}$$

and the roots of (5) are represented as

$$z_1 = \frac{-a+\sqrt{a^2-4b}}{2}\ ,\ z_2 = \frac{-a-\sqrt{a^2-4b}}{2}\ . \tag{6}$$

Therefore, there are three cases for (6): Case I. $z_1 \neq z_2$; Case II. $z_1 = z_2$; Case III. z_1 and z_2 are a conjugate pair.

Case I: If $z_1 \neq z_2$, the solution of the prediction model of second order difference equation by using inverse Z transform is

$$\hat{y}^{(1)}(k+p) = c_1 z_1^{k+p} + c_2 z_2^{k+p} \tag{7}$$

The constants c_1 and c_2 are obtained by solving the initial condition of (7) from $k=1$ and $k=2$:

$$c_1 = \frac{z_1 y^{(0)}(1) - y^{(0)}(1) - y^{(0)}(2)}{z_1 z_2 - z_2^2},$$
$$c_2 = \frac{z_2 y^{(0)}(1) - y^{(0)}(1) - y^{(0)}(2)}{z_1 z_2 - z_1^2}. \quad (8)$$

Case II: If $z_1 = z_2$, the second difference equations becomes

$$\hat{y}^{(1)}(k+p) = c_1 z_1^{k+p} + c_2 (k+p) z_2^{k+p}. \quad (9)$$

The constants c_1 and c_2 are

$$c_1 = \frac{(2z_1 - 1) y^{(0)}(1) - y^{(0)}(2)}{z_1^2},$$
$$c_2 = \frac{(1 - z_1) z_1 y^{(0)}(1) + y^{(0)}(2)}{z_1^2}. \quad (10)$$

Case III: If z_1 and z_2 are a conjugate pair, the second order difference equation becomes

$$\hat{y}^{(1)}(k+p) = c_1 \rho^{k+p} \sin\varphi(k+p) + c_2 \rho^{k+p} \cos\varphi(k+p), \quad (11)$$

$$\rho = \sqrt{\left(-\frac{a}{2}\right)^2} + \sqrt{\left(\frac{\sqrt{4b - a^2}}{2}\right)^2} = \sqrt{b}, \quad \varphi = \tan^{-1}\left(-\sqrt{\frac{4b - a^2}{a}}\right),$$

$$c_1 = \frac{y^{(0)}(1)\rho^2 \cos 2\varphi - y^{(0)}(1)\rho \cos\varphi - y^{(0)}(2)\rho \cos\varphi}{\rho^3 (\sin\varphi \cos 2\varphi - \cos\varphi \sin 2\varphi)},$$
$$c_2 = \frac{y^{(0)}(1)\rho \sin\varphi + y^{(0)}(2)\rho \sin\varphi - y^{(0)}(1)\rho^2 \sin 2\varphi}{\rho^3 (\sin\varphi \cos 2\varphi - \cos\varphi \sin 2\varphi)}. \quad (12)$$

Because the second order difference equation of the prediction model is based on the AGO data, we must take the inverse accumulated generating operator (IAGO) to get the non-negative $y^{(0)}$ data by the following relationship

$$\hat{y}^{(0)}(k+p) = \hat{y}^{(1)}(k+p) - \hat{y}^{(1)}(k+p-1), \quad (13)$$

where p is the forecasting step-size. Based on the above description, the basic grey prediction method can be described as follows

$$\hat{y}^{(0)} = IAGO \circ DGM(2,1) \circ AGO \circ y^{(0)}. \quad (14)$$

However, the response sequence of the system may be positive or negative. Therefore, we have to map the negative sequence to the relative positive sequence by some methods of data mapping. In this paper, we take the inverse mapping generating operator (IMGO) defined as follows.

Let $y^{(0)}$ be an original sequence and $y_m^{(0)}$ be the MGO image sequence of $y^{(0)}$, then

$$y_m^{(0)} = MGO \circ y^{(0)} = bias + y^{(0)}, \quad (15)$$

where the bias is a shift factor, and must be greater than $y^{(0)}$. Then the IMGO can also be obtained as follows

$$y^{(0)} = IMGO \circ y_m^{(0)} = y_m^{(0)} - bias. \quad (16)$$

Therefore, the modified grey prediction can be constructed by

$$\hat{y}^{(0)} = IMGO \circ IAGO \circ DGM(2,1) \circ AGO \circ MGO \circ y^{(0)}. \tag{17}$$

Influence of using different step-size on the system is shown in Table 1. The goal is to find a proper forecasting step-size between the given positive and negative one. The controller not only can reduce the overshoot, but also can cause a shorter rise time than the conventional design methods. The switching mechanism is defined by [14]

$$p = \begin{cases} pys & \text{if} \quad e > \theta_l \\ pym & \text{if} \quad \theta_l \ge e > \theta_s \\ pyl & \text{if} \quad \theta_s \ge e \end{cases}, \tag{18}$$

where p is the forecasting step-size of the system. *pys*, *pym* and *pyl* are the forecasting step-sizes for the large error, the middle error, and the small error, respectively. θ_l and θ_s are the switching time of the large error and the small error, respectively.

Table 1. Different forecasting step-size influence on the system

Forecasting Step-size / System response	Positive forecasting step-size	Negative forecasting step-size
Overshoot	Small	Big
Rising time	Long	Short
Steady state time	Slow	Fast
Adapt region	Close	Leave

A system performance index is used for fitness function in RGA. It is an optimization criterion for parameters tuning of control system. It deals with a modification of the known integral of time multiplied by squared error criterion (ISTE). In order to influence a characteristic value of a signal it is not necessary to add a special term to the ITSE which will increase the selection pressure in RGA. An evident possibility is to divide the integral criterion in special error section for each characteristic value. The addition of integral sections in the time horizon is straight forward and always the same unit. For example, the aim of control optimization is to determine the control parameters in order to minimize the error signal e as shown in Fig. 4. The general form and an often used performance index are shown below.

$$I = \int_0^T f\big(r(t)\ ,e(t)\ ,u(t)\ ,y(t)\ ,t\big)\ dt\ , \tag{19}$$

$$f = f_{ITSE}(e(t),u(t),t) = t^n(e^2(t) + \alpha u^2(t)), n > 1,\ \alpha > 0 \tag{20}$$

The following equation represents system performance index:

$$I_{PITSE} = t^n * \begin{bmatrix} 1 * overshoot\ I \\ + 2 * overshoot\ II \\ + risetime \\ + steady\ state\ time \end{bmatrix} + control\,. \tag{21}$$

Fig. 5 shows the definition of different sections. The *Rise time* is evaluated as the signal difference area between $r(t)$ and $y(t)$ from $t = 0$ to T, where the gain of $y(t)$ is $0.9r(t)$. The section may also include the undershoot area. *Overshoot I* and *Overshoot II* are the areas between the overshoot or maximum peak and $r(t)$. The *Overshoot II* section is double evaluated in the sense of more selection pressure and smooth fitness landscape. It leads to faster convergence by raising the section pressure or weighting value. The *ssTime* is the steady state error section. The *control* is control force to the TRMS. The following equation represents the system performance index in Fig. 5, which meets the requirements above. Flow chart of the switching prediction PID control with RGA is shown in Fig. 6.

$$I_{PITSE} = t^n \left(\begin{array}{c} 2 * \int (y(t) - Mp)\ dt \Big|_{y(t) \ge Mp} \\ + \int (y(t) - r(t))\ dt \Big|_{Mp > y(t) \ge r(t)} \\ + \int 0.9 * r(t)\ dt \Big|_{0.9 \ge y(t)} \\ + \int (y(t) - r(t))\ dt \Big|_{|e(t)| \le 0.05 * r(t)} \end{array} \right). \tag{22}$$

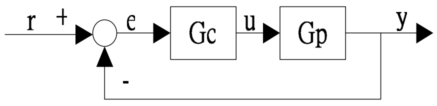


Fig. 4. Example of control system

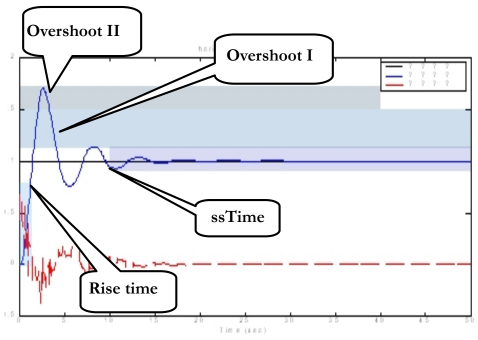


Fig. 5. Piecewise Integral of Time weighted Squared Error – PITSE

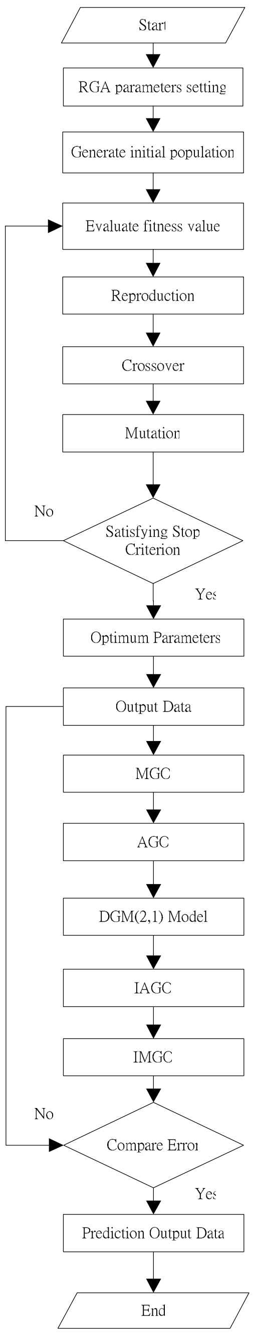


Fig. 6. The flow chart of the switching prediction PID control with RGA

4 Simulations

In the process of RGA, population size is 10. The first step is to evaluate each fitness value of individuals (chromosomes) and subsequently grade them by fitness function. The individuals are selected probabilistically by their fitness values. Using these selected individuals the next population is generated through a process of Adewuya crossover law [15]. Mutation is applied with a very low probability 0.025. The target parameters of the control system are the forecasting step-size, switching times, and the PID control gains, which are *pylh*, *pysh*, *pylv*, *pysv*, θ_{lh} , θ_{sv} , K_{Phv}, K_{Ihv}, K_{Dhv}, K_{Pvh}, K_{Ivh}, K_{Dvh}, K_{Pvv}, K_{Ivv}, K_{Dvv}, K_{Phh}, K_{Ihh}, *and* K_{Dhh}. During the simulation in Simulink, reference input of horizontal is 1 rad and initial condition is 0 rad. Reference input in vertical is 0.2 rad and initial condition is -0.5 rad. Simulation time is from 0 to 50 seconds. System performance requirements are: maximum overshoot less than 1% ; rising time, delay time and steady state time as short as possible. System Performance Index is

$$I_{PITSE} = t^2 \begin{pmatrix} 2\int (y(t) - Mp)\, dt \,\Big|_{y(t) \geq 1.01} \\ + \int (y(t) - r(t))\, dt \,\Big|_{1.01 > y(t) \geq 1.0} \\ + \int 0.9\, r(t)\, dt \,\Big|_{0.9 \geq y(t)} \\ + \int (y(t) - r(t))\, dt \,\Big|_{|e(t)| \leq 0.05\, r(t)} \end{pmatrix}. \tag{23}$$

Fitness function in RGA is

$$Fitness\_Function = \frac{10000}{I_{pitse\_h}} + \frac{10000}{I_{pitse\_v}}. \tag{24}$$

Simulation results are shown in Fig. 7 to Fig. 9. Compare to previous work [16], as shown in Table 2, the TRMS output performance has been improved a lot.

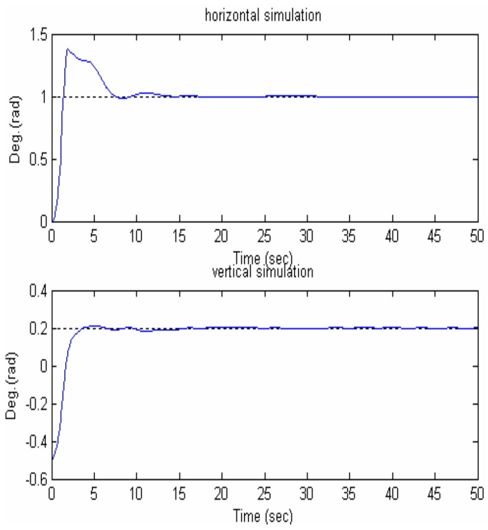


Fig. 7. Step response in cross-coupling system

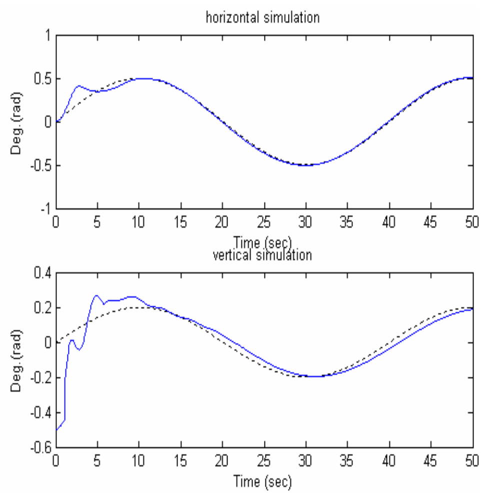


Fig. 8. Sine wave response in cross-coupling system

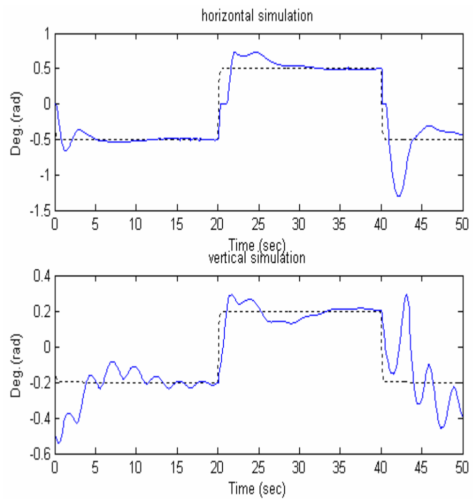


Fig. 9. Square wave response in cross-coupling system

Table 2. Comparison of total error in cross-coupling system

reference	V or H	[16]	This paper	improvement
set-point	Horizontal	50.0221	28.7228	42.6%
	Vertical	32.8766	29.4717	10.4%
sine wave	Horizontal	14.0243	13.6415	2.7%
	Vertical	72.2283	46.1231	36.1%
square wave	Horizontal	125.3977	103.6016	17.4%
	Vertical	99.3753	74.6588	24.9%

5 Conclusion

This paper proposes a design of a grey prediction controller which is simple and easy to be realized. Grey theory has successfully implemented in TRMS control. In here, utilization of switching grey prediction and RGA provides good adaptive predictions and pre-compensations for the PID controller. This approach has successfully overcome the influence of cross-coupling between two axes. In set-point control, it reduces maximum overshoot, rising time, steady state time and delay time. It is also more suitable for tracking a desired trajectory in horizontal and vertical planes simultaneously. Compare to previous work, the simulation results show that the new approach reduces the total error and improves the system performance.

References

1. B.C. Kuo. *Automatic Control Systems*, Englewood Cliffs, NJ:Prentice-Hall, 1995.
2. J.G. Ziegler and N. B. Nichols, "Optimum Settings for Automatic Controllers," *Trans. ASME*, pp. 759-768, 1942.
3. R. A. Krohling, H. Jaschek, and J. P. Rey, "Designing PI/PID Controllers for a Motion Control System Based on Genetic Algorithms," *Proceedings of the 12th IEEE International Symposium on Intelligent Control*, pp. 125-130, 1997.
4. R. A. Krohling and J. P. Rey, "Design of Optimal Disturbance Rejection PID Controllers Using Genetic Algorithms," *IEEE Transactions on Evolutionary*, 2001.
5. J. L. Deng, "Control Problems of Unknown Systems," *Proc. of the Cilateral Meeting on Control Systems*, 156-171, 1981.
6. J. L. Deng, "Control Problems of Grey Systems," *System and Control Letters*, vol. 1, no. 5, 288-294, 1982.
7. B. Cheng, "The Grey Control on Industrial Process," *Journal of Huangshi College*, vol. 1, pp. 11-23, 1986. (in Chinese)
8. C. Y. Li and T. L. Huang, "Optimal Design of the Grey Prediction PID Controller for Power System Stabilizers by Evolutionary Programming," *Proceedings of the IEEE International Conference on Networking, Sensing & Control*, pp. 1370-1375, 2004.
9. H. C. Lu, "Grey Prediction Approach for Designing Grey Sliding Mode Controller," *Proceedings of the IEEE Conference on Systems, man and Cybernetics*, pp. 403-408, 2004.
10. R.L. Haupt and S.E. Haupt, *Practical Genetic Algorithm*, USA, 2004.
11. Dipl.-lng Ivo Boblan, "A New Integral Criterion Parameter Optimization of Dynamic System with Evolution Strategy," VDI Berichte Nr.1526, *Computational Intelligence*, pp.143-150, 2000.
12. Feedback Co, *Twin Rotor MIMO System 33-220 User Manual,* 3-000M5, 1998.
13. K. L. Wen, F. H. Chen and M. L. Huang, "Existence of Grey Model GM(2,1)," *The Journal of Grey System*, pp.209-214, 1998.
14. C.C. Wong and C.C. Chen, "Switching Grey Prediction PID Controller Design," *The Journal of Grey System*, vol.9, no.4, pp.323-334, Dec. 1997.
15. A. A. Adewuya, *New Methods in Genetic Search with Real-valued Chromosomes*, Master's Thesis, Dept. of Mechanical Engineering, Massachusetts Institute of Technology, 1996.
16. W. K. Liu, J. H. Fan and J. G.. Juang, "Application of Genetic Algorithm Based on System Performance Index on PID Controller," *Proc. of Conference **on Artificial** Intelligence and Application,* FP4-3, 2004.

H^{∞} Neural Networks Control for Uncertain Nonlinear Switched Impulsive Systems

Fei Long[1,2], Shumin Fei[2], Zhumu Fu[2], and Shiyou Zheng[2]

[1] School of Informational Engineering, Guizhou University
550003 Guiyang, China
feilong73@56.com,
flong1973@yahoo.com.cn
[2] Department of Automatic Control, Southeast University
210096 Nanjing, China

Abstract. Based on RBF (radial basis function) neural network, an adaptive neural network feedback control scheme and an impulsive controller for output tracking error disturbance attenuation of nonlinear switched impulsive systems are given under all admissible switched strategy in this paper. Impulsive controller is designed to attenuate effect of switching impulse. The RBF neural network is used to compensate adaptively for the unknown nonlinear part of switched impulsive systems, and the approximation error of RBF neural network is introduced to the adaptive law in order to improve the tracking attenuation quality of the switched impulsive systems. Under all admissible switching law, impulsive controller and adaptive neural network feedback controller can guarantee asymptotic stability of tracking error and improve disturbance attenuation level of tracking error for the overall switched impulsive system.

1 Introduction

A switched nonlinear system is a hybrid system that comprises a collection of nonlinear subsystems together with a switching signal that specifies the switching among the subsystems. Switched dynamical systems have been attracting much attention because the study for these systems is not only academically challenging, but also of practice importance. The study for switched systems is well motivated from several aspects. Firstly, from the practical point of view, switching among different system structures is an essential feature of many engineering systems including power systems and disk drivers. Secondly, from the modeling point of view, as complex (intelligent) systems are very hard to model (analyze) globally at the whole range of operation, multiple-model provides a convenient and efficient way to model these systems. Thirdly, from the control point of view, multi-controller switching provides an effective mechanism to cope with highly complex systems and/or systems with large uncertainties.

During the last decades, applications of neural networks in system identification and control have been extensively studied. The study for non-switched nonlinear systems using universal function approximation has received much attention and many methods have been proposed (see [1], [2], [3], [4], [5], [6] for therein references). Typically, these methods use neural networks as approximation models for the unknown part of non-switched systems. It has been shown that successful identification and control may be possible using neural networks for complex nonlinear dynamic systems whose mathematical models are not available from first principles. In [7] and [8], it was shown that for stable and efficient online control using the back propagation learning algorithm, the identification must be sufficiently accurate before the control action is initiated. In practical applications, it is desirable to have a systematic method of ensuring stability, robustness, controllability, observability, H disturbance attenuation quality and other performance properties of system. Recently, several good neural network control approaches have been proposed based on Lyapunov stability theory ([9], [10], [11], [12] and [13]). One key advantage of these schemes is that the adaptive laws were derived based on Lyapunov synthesis and, therefore, guaranteed the stability of non-switched systems without the requirement for offline training.

However, most of these good results are restricted in non-switched systems. Due to the difference between non-switched systems and switched systems, a stable controller designed in non-switched system may become unstable in switched system via unsuitable switching rule, thus we may run into troubles when we implement these networks controllers in switched system in which the data are typically available only at switching time instants. Therefore, the study for switched system based on neural network is necessary and significant. Unfortunately, all these good neural network controllers are designed for non-switched systems while there are a few attentions for switched control systems presently. In [14], the output tracking stabilization is studied for a class of switched nonlinear systems based on RBF (radial basis function) neural network. In [15], [16] and [17], output tracking error disturbance attenuation problem for a class of switched nonlinear systems is investigated in SISO case and MIMO case, respectively.

As we know, many practical systems in physics, biology, engineering, and information science exhibit impulsive dynamical behaviors due to abrupt changes at certain instants during the dynamical processes. But for hybrid and switched systems, as an important model for dealing with complex real systems, there is little work concerning impulsive phenomena ([18], [19], [20], [21], [22], [23] and [24]). In this note, impulsive phenomena are introduced into switched nonlinear systems. Furthermore, we investigate tracking disturbance attenuation capability for such systems that exhibit impulsive behaviors at switching time instant by using RBF neural networks.

This paper is organized as follows. In Section 2, H^{∞} control problem for a class of switched nonlinear systems with impulsive behavior is stated. Section 3 focuses on the design method of adaptive feedback control scheme and impulsive controller based on RBF neural network. Finally the conclusion is drawn in Section 4.

2 System Description and Preliminaries

Consider the following switched nonlinear system with impulsive behavior is given by

$$\begin{cases} y^{(n)}(t) = f_{\sigma(t)}(y(t), y^{(1)}(t), \cdots, y^{(n-1)}(t)) + g_{\sigma(t)}(y(t), y^{(1)}(t), \cdots, y^{(n-1)}(t))u(t) \\ \begin{bmatrix} \Delta y(t) \\ \Delta y^{(1)}(t) \\ \vdots \\ \Delta y^{(n-1)}(t) \end{bmatrix} = \begin{bmatrix} y(t^+) - y(t^-) \\ y^{(1)}(t^+) - y^{(1)}(t^-) \\ \vdots \\ y^{(n-1)}(t^+) - y^{(n-1)}(t^-) \end{bmatrix} = E_{\sigma(t^+),\sigma(t^-)} \begin{bmatrix} y(t) \\ y^{(1)}(t) \\ \vdots \\ y^{(n-1)}(t) \end{bmatrix} + u_I(t) \end{cases} \tag{1}$$

where $y \in \mathbb{R}$ is the measured output of system. $u \in \mathbb{R}$ is the control input signal. $u_I \in \mathbb{R}^n$ is the impulsive control to be designed.

$$y(t^+) \overset{def}{=} \lim_{h\to 0^+} y(t+h), \quad y(t^-) \overset{def}{=} \lim_{h\to 0^+} y(t-h) \ .$$
$$y(t^-) = y(t),\ y^{(1)}(t^-) = y^{(1)}(t),\ \cdots,\ y^{(n-1)}(t^-) = y^{(n-1)}(t).$$

The symbol $(*)^{(k)}$ denotes the k^{th} derivative of $(*)$ and

$$\sigma : [0,+\infty) \mapsto \{1,2,\cdots,N\} \overset{def}{=} \underline{\mathbb{N}}$$

(N is finite natural number, i.e. $N < \infty$) stands for the piecewise constant switching signal. Moreover, $\sigma(t_k) = i$ means that the i^{th} subsystem

$$y^{(n)}(t) = f_i(y(t), y^{(1)}(t), \cdots, y^{(n-1)}(t)) + g_i(y(t), y^{(1)}(t), \cdots, y^{(n-1)}(t))u(t)$$

is active at time instant $t = t_k$. $\sigma(t_m^-) = i$ and $\sigma(t_m^+) = j$ mean that the switched nonlinear system (1) is switched from the i^{th} subsystem to the j^{th} subsystem at time instant $t = t_m$. $E_{i.j}(i, j \in \underline{\mathbb{N}})$ is known $n \times n$ constant matrices. Especially, $E_{i.i} = 0 (i \in \underline{\mathbb{N}})$ means the switching between the same sub-system are smooth and without impulse, in other words, there is no impulse when a subsystem is remaining active. The functions $f_i(\cdot)$ and $g_i(\cdot)(i \in \underline{\mathbb{N}})$ are unknown smooth function.

Now we design RBF neural network to approximate the functions $f_i(x)$ and $g_i(x)(i \in \underline{\mathbb{N}})$, that is,

$$\hat{f}_i(x, \theta_f) = \theta_f^T \varphi_{fi}(x), \quad \hat{g}_i(x, \theta_g) = \theta_g^T \varphi_{gi}(x)$$

where $x = \left(y, y^{(1)}, \cdots, y^{(n-1)}\right)^T$, θ_f and θ_g are vector of adjustable weights.

$\varphi_{fi}(x) = \left(\varphi_{fi1}(x), \cdots, \varphi_{fip}(x)\right)^T$ and $\varphi_{gi}(x) = \left(\varphi_{gi1}(x), \cdots, \varphi_{giq}(x)\right)^T$ denote vectors of Guassian basis function.

$\varphi_{fij}(x) = \exp(-\left\|x - c_{fij}\right\|^2 / \sigma_{fij}^2)$ and $\varphi_{gij}(x) = \exp(-\left\|x - c_{gij}\right\|^2 / \sigma_{gij}^2)$ stand for Guassian basis function, where (c_{fij}, σ_{fij}) and (c_{gij}, σ_{gij}) is, respectively, center vector and width of the j^{th} hidden element for RBF neural network to approximate the functions $f_i(x)$ and $g_i(x)(i \in \underline{\mathbb{N}})$. The radial basis function neural network is shown in fig. 1.

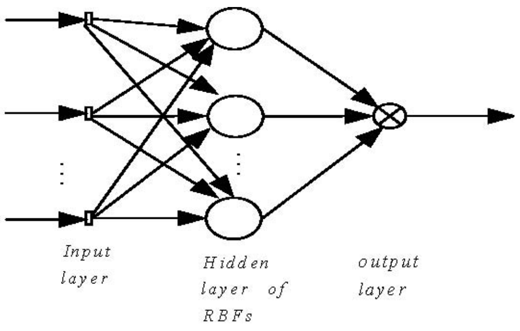


Fig. 1. Schematic of RBF neural network

By calling up the literature [11] and the neural network theory [25], on compact set $X \subset \mathbb{R}^n$, for every $\varepsilon > 0$, there exist two Guassian basis function vectors $\varphi_{fi}(\cdot): X \mapsto \mathbb{R}^p, \varphi_{gi}(\cdot): X \mapsto \mathbb{R}^q$ $(i \in \underline{\mathbb{N}})$, and two weight vectors $\hat{\theta}_f \in \Omega_f$, $\hat{\theta}_g \in \Omega_g$ such that

$$\left|f_i(x) - \hat{\theta}_f^T \varphi_{fi}(x)\right| \le \varepsilon, \quad \left|g_i(x) - \hat{\theta}_g^T \varphi_{gi}(x)\right| \le \varepsilon,$$

where Ω_f and Ω_g is known compact subset of $\mathbb{R}^p$ and $\mathbb{R}^q$, respectively.

Therefore, the optimal weights θ_f, θ_g defined as

$$\begin{cases} \theta_f = \arg \min\limits_{\hat{\theta}_f \in \Omega_f} \min\limits_{1 \le i \le N} \sup\limits_{x \in X} \left|f_i(x) - \hat{\theta}_f^T \varphi_{fi}(x)\right| \\ \theta_g = \arg \min\limits_{\hat{\theta}_g \in \Omega_g} \min\limits_{1 \le i \le N} \sup\limits_{x \in X} \left|g_i(x) - \hat{\theta}_g^T \varphi_{gi}(x)\right| \end{cases}$$

and RBF neural network reconstruction approximate error ω_i for the i^{th} subsystem of switched impulsive system (1) defined as

$$\omega_i = f_i(x) - \hat{f}(x, \theta_f) + (g_i(x) - \hat{g}(x, \theta_g))u$$

are well pose, where $\hat{f}_i(x, \theta_f) = \theta_f^T \varphi_{fi}(x), \quad \hat{g}_i(x, \theta_g) = \theta_g^T \varphi_{gi}(x)$.

Setting $e = y - y_r$ (is any given bounded reference output signal) denotes the output tracking error,

$$\tilde{\theta} = \hat{\theta} - \theta \left(\theta = \begin{bmatrix} \theta_f^T & \theta_g^T \end{bmatrix}^T, \hat{\theta} = \begin{bmatrix} \hat{\theta}_f^T & \hat{\theta}_g^T \end{bmatrix}^T \right)$$

denotes the adaptive parameter error.

The objective in this paper is, for any given $\gamma > 0$, to design adaptive neural network controller $u(t)$ and impulsive controller $u_I(t)$ such that the switched impulsive system (1) satisfies the following two performances for all admissible switching strategy $\sigma(t)$.

i). $\lim\limits_{t\to\infty} e(t)=0$ when the network reconstruction error ω_k admits a upper bounder; and there exists a constant $M>0$ such that $\|\theta\|\le M$.

ii). $\int\limits_0^T \underline{e}^T(t)\underline{e}(t)dt \le \gamma^2 \int\limits_0^T \omega^T(t)\omega(t)dt + S(\underline{e}(0),\tilde{\theta}(0))$, where $\underline{e}=(e,e^{(1)},\cdots,e^{(n-1)})^T$, $\|\omega\|=\max\{\|\omega_1\|,\cdots,\|\omega_N\|\}$, S denotes appropriate positive-definite function.

In this article, we make the following assumptions.

Assumption 1. There exists a positively real number l such that for all $x\in\mathbb{R}^n$ and every $i\in\underline{\mathbb{N}}$, $|g_i(x)|\ge l$.

Assumption 2. The reference signal y_r is n-times continuous in the interval $[0,+\infty)$ and its derivatives up to order n are known and uniformly bounded in the time interval $[0,+\infty)$.

3 Controller Design

Consider the following adaptive neural network feedback controller and impulsive controller

$$\begin{cases} u(t)=\hat{g}^{-1}_{\sigma(t)}(x,\theta_g)(-\hat{f}_{\sigma(t)}(x,\theta_f)+\Lambda) \\ u_I(t)=-E_{\sigma(t^-),\sigma(t^+)}\begin{bmatrix} y_r(t) \\ y_r^{(1)}(t) \\ \vdots \\ y_r^{(n-1)}(t) \end{bmatrix} \end{cases} \tag{2}$$

where $y_r^{(n)}+\lambda_n(y_r^{(n-1)}-y^{(n-1)})+\cdots+\lambda_1(y_r-y)$, $(\lambda_1,\lambda_2,\cdots,\lambda_n)$ is Hurwitz vector, i.e., the matrix

$$A=\begin{pmatrix} 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 \\ -\lambda_1 & -\lambda_2 & \cdots & -\lambda_n \end{pmatrix}$$

is stable.

The output tracking error dynamic equation of switched impulsive system (1) is given by

$$\begin{cases} \dot{\underline{e}}(t)=A\underline{e}(t)+B(\omega_{\sigma(t)}+\tilde{f}_{\sigma(t)}(x,\tilde{\theta}_f)+\tilde{g}_{\sigma(t)}(x,\tilde{\theta}_g)u) \\ \Delta\underline{e}=\underline{e}(t^+)-\underline{e}(t^-)=E_{\sigma(t^+),\sigma(t^-)}\underline{e}(t) \end{cases} \tag{3}$$

Where $\underline{e}=(e,e^{(1)},\cdots,e^{(n-1)})^T$, $\tilde{\theta}_f=\hat{\theta}_f-\theta_f$, $\tilde{\theta}_g=\hat{\theta}_g-\theta_g$, $B=(0,0,\cdots,1)^T$.

By calling up the literature [11] and the neural network theory [25], in view of the assumption 1, it is obvious that $\left|\hat{g}_i^{-1}(x,\theta_f)\right| \le 1/(l-\varepsilon), \forall i \in \underline{\mathbb{N}}$. Again by means of the approximation property of RBF neural network, the neural network feedback controller $u(t)$ is bounded.

Now, we introduce an important concept---switching sequence, which can be use in the later.

Definition 3.1. The sequence

$$\left\{(t_m, r_m) \middle| \ r_m = 1,2,\cdots,N; m=1,2,\cdots\right\}$$

is said to be switching sequence, if

i) $\sigma(t_m^-) \ne \sigma(t_m^+)$,

ii) $\sigma(t) = \sigma(t_m^-) = r_m, t \in [t_{m-1}, t_m)$.

Moreover, the interval $[t_{m-1}, t_m)$ is said to be dwell time interval of the $r_m - th$ subsystem.

According to the above analysis, for the switched impulsive system (3), we have the following result.

Theorem 3.1. Consider the output tracking error dynamic equation of switched impulsive system (1), i.e. the system (3) and suppose that, for any given $\gamma > 0$ and symmetrical positive-definite matrices $Q_1, Q_2, \cdots, Q_N$, there exist N symmetrical positive-definite matrices $P_1, P_2, \cdots, P_N$ such that for $i = 1,2,\cdots,N$

$$\begin{bmatrix} A^T P_i + P_i A + Q_i & \gamma^{-1} P_i B \\ \gamma^{-1} B^T P_i & -I \end{bmatrix} < 0 \tag{4}$$

$$\|\omega_i\|^2 \le \gamma^2 C_1 C_2 \tag{5}$$

and for $i, j = 1,2,\cdots,N$, the following matrix inequality holds.

$$\begin{bmatrix} -P_i & (I+E_{i,j})^T P_j \\ P_j(I+E_{i,j}) & -P_j \end{bmatrix} < 0 \tag{6}$$

Then there exists RBF neural network controller such that the resulting closed-loop system satisfies i) and ii) for all admissible switching rule $\sigma(t)$. In this case RBF neural network controller is taken as (2) and

$$\begin{cases} \dot{\theta}_f = \mu \underline{e}^T P_i B \varphi_{fi}(x); & \theta_f \in \Omega_f, t \in \Omega_i \\ \mu^{-1} \tilde{\theta}_f^T (\dot{\theta}_f - \mu \underline{e}^T P_i B \varphi_{fi}(x)) \ge 0; & \theta_f \in \Omega_{\delta f} \setminus \Omega_f, t \in \Omega_i \end{cases} \tag{7}$$

$$\begin{cases} \dot{\theta}_g = \rho \underline{e}^T P_i B \varphi_{gi}(x) u; & \theta_g \in \Omega_g, t \in \Omega_i \\ \rho^{-1} \tilde{\theta}_g^T (\dot{\theta}_g - \rho \underline{e}^T P_i B \varphi_{gi}(x) u) \ge 0; & \theta_g \in \Omega_{\delta g} \setminus \Omega_g, t \in \Omega_i \end{cases} \tag{8}$$

where $\Omega_i = \{t \mid \text{ the } i^{th} \text{ subsystem is active at time instant } t\}$, $\Omega_f \subset \Omega_{\delta f}$, $\Omega_g \subset \Omega_{\delta g}$ ($\Omega_{\delta f} \subset \mathbb{R}^p$ and $\Omega_{\delta g} \subset \mathbb{R}^q$ are known compact subset), μ and ρ denote, respectively, the learning rate of RBF neural networks for $f_i(x)$ and $g_i(x)$,

$$C_1 = \min\{\lambda_{\min}(Q_1), \lambda_{\min}(Q_2), \cdots, \lambda_{\min}(Q_N)\} ,$$

the set E_0 is any given known compact subset of $\mathbb{R}^n$ and contains $\underline{e}(0)$,

$$E = \left\{\underline{e} \in \mathbb{R}^n \,\middle|\, \|\underline{e}\|^2 \le C_2\right\}, C_2 > \max_{\underline{e} \in E_0} \|\underline{e}\|^2 .$$

Proof: Consider the Lyapunov function candidate

$$V = \sum_{i=1}^{N} \alpha_i(t) \underline{e}^T P_i \underline{e} + \mu^{-1} \tilde{\theta}_f^T \tilde{\theta}_f + \rho^{-1} \tilde{\theta}_g^T \tilde{\theta}_g \tag{9}$$

where

$$\alpha_i(t) = \begin{cases} 1, & t \in \Omega_i \\ 0, & t \notin \Omega_i \end{cases} ,$$

$\Omega_i = \{t \mid \text{ the } i^{th} \text{ subsystem is active at time instant } t\}$.

Using (4) and (7)-(8), , the derivative of V along the trajectories of the system (3) is given by $\forall t \in (t_{m-1}, t_m] \subset \Omega_{r_m}$

$$\begin{aligned} \dot{V} &\le \underline{e}^T (A^T P_{r_m} + P_{r_m} A) \underline{e} + 2\underline{e}^T P_{r_m} B \omega_{r_m} \\ &< -\underline{e}^T Q_{r_m} \underline{e} + 2\underline{e}^T P_{r_m} B \omega_{r_m} - \underline{e}^T P_{r_m} B B^T P_{r_m} \underline{e} \\ &\le -\underline{e}^T Q_{r_m} \underline{e} + \gamma^2 \left\|\omega_{r_m}\right\|^2 \end{aligned} \tag{10}$$

In view of $C_1 = \min\{\lambda_{\min}(Q_1), \lambda_{\min}(Q_2), \cdots, \lambda_{\min}(Q_N)\}$ and $C_2 > \max_{\underline{e} \in E_0} \|\underline{e}\|^2$, for every $\underline{e} \in E$, we have

$$\dot{V} < -C_1 C_2 + \gamma^2 \left\|\omega_{r_m}\right\|^2 \tag{11}$$

It can conclude form (5) that for every $t \in (t_{m-1}, t_m] \subset \Omega_{r_m}$, $\dot{V}(t) < 0$. Without of generally, suppose that $\sigma(t_m^+) = r_{m+1}$, then by (6) and (9)

$$\begin{aligned} V(t_m^+) - V(t_m^-) &= \underline{e}^T(t_m^+) P_{r_{m+1}} \underline{e}(t_m^+) - \underline{e}^T(t_m^-) P_{r_m} \underline{e}(t_m^-) \\ &= \underline{e}^T(t_m)\left[(I + E_{r_m, r_{m+1}})^T \times P_{r_{m+1}} (I + E_{r_m, r_{m+1}}) - P_{r_m}\right] \underline{e}(t_m) < 0 \end{aligned} \tag{12}$$

Therefore by Lyapunov stability theorem, i) holds.

Next, we prove that ii) holds. By means of $\|\bar{\omega}\| = \max\{\|\bar{\omega}_1\|,\cdots,\|\bar{\omega}_N\|\}$, $C_1 = \min\{\lambda_{\min}(Q_1),\lambda_{\min}(Q_2),\cdots,\lambda_{\min}(Q_N)\}$ and (10), we have

$$\int_{t_{m-1}}^{t_m} \underline{e}^T(t)\underline{e}(t)dt \le \frac{\gamma^2}{C_1}\int_{t_{m-1}}^{t_m} \omega^T(t)\omega(t)dt + V(t_{m-1}^+) - V(t_m^-),\ m = 1,2,\cdots \tag{13}$$

Let

$$\{(t_m, r_m) \mid r_m \in \underline{\mathbb{N}}; m = 1,2,\cdots,s; 0 = t_1 < t_2 < \cdots < t_s \le T\}$$

be every switching sequence in the interval [0, that is to say, the switching time instant $t_1, t_2, \cdots, t_s, t_{s+1}, \cdots$ satisfies $0 = t_1 < t_2 < \cdots < t_s \le T \le t_{s+1}$.
Noting that $t_1 = 0$ and $x(0) = 0$, in view of (6), (12) and (13), $\forall \omega \in L_2[0,T]$

$$\begin{aligned}\int_0^T \underline{e}^T(t)\underline{e}(t)dt &= \sum_{m=1}^{s-1}\int_{t_m}^{t_{m+1}} \underline{e}^T(t)\underline{e}(t)dt + \int_{t_s}^T \underline{e}^T(t)\underline{e}(t)dt \\ &\le \frac{\gamma^2}{C_1}\sum_{m=1}^{s-1}\int_{t_m}^{t_{m+1}} \omega^T(t)\omega(t)dt + \int_{t_s}^T \omega^T(t)\omega(t)dt + V(0) - V(T) + \sum_{i=2}^{s}(V(t_i^+) - V(t_i^-)) \\ &< \frac{\gamma^2}{C_1}\int_0^T \omega^T(t)\omega(t)dt + \underline{e}^T(0)P_{r_1}\underline{e}(0) + \mu^{-1}\tilde{\theta}_f^T(0)\tilde{\theta}_f(0) + \rho^{-1}\tilde{\theta}_g^T(0)\tilde{\theta}_g(0)\end{aligned}$$

Hence, ii) holds. This ends the proof. ◊

4 Conclusions

For a class of switched impulsive systems represented by input-output models, the tracking error attenuation problem is investigated by using RBF neural networks. Neural network adaptive feedback controller and impulsive controller are given. Approximation errors of RBF neural networks are introduced to the adaptive law in order to improve the tracking error attenuation quality for overall switched impulsive systems. The adaptive feedback controller and impulsive controller designed can guarantee asymptotical stability and disturbance attenuation performance of tracking error for the whole switched system under all admissible switching rules.

Acknowledgement

The authors would like to thank the anonymous reviewers for their constructive and insightful comments for further improving the quality of this work. This work was partially supported by National Natural Science Foundation of China under Grant No. 60574006, the Higher Education Doctor's Discipline Foundation under Grant No.20030286013 and the Natural Science Foundation of Jiangsu Province under Grant BK 2003405.

References

1. Liu, C., Chen, F.: Adaptive Control of Nonlinear Continuous Systems Using Neural Network---General Relative Degree and MIMO Case. International Journal of Control 58 (1993) 317-335
2. Narendrk, K. S., Mukhopadhyay, S.: Adaptive Control of Nonlinear Multivariable System Using Neural Network. Neural Network 7 (1994) 737-752
3. Liu, G. P., et al: Variable Neural Networks for Adaptive Control of Nonlinear Systems. IEEE Trans Systems, man, Cybermetics-Part C 29 (1999) 34-43
4. Patino, H. D., Liu, D.: Neural Network-Based Model Reference Adaptive Control Systems. IEEE Trans Systems, man, Cybermetics-Part B 30 (2001) 198-204
5. Sanner, R., Slotine, J. J.: Gaussian Networks for Direct Adaptive Control. IEEE Trans on Neural Networks 3 (1992) 837-864
6. Sridhar, S., Hassan, K. K.: Output Feedback Control of Nonlinear Systems Using RBF Neu-ral Networks. IEEE Trans. Neural Network 11 (2000) 69-79
7. Levin, A. U., Narendra, K. S.: Control of Nonlinear Dynamical Systems Using Neural Networks-Part II: Observability, Identification, and Control. IEEE Trans. Neural Networks 7 (1996) 30-42
8. Narendra, K. S., Parthasarathy, K.: Identification and Control of Dynamic Systems Using Neural Networks. IEEE Trans. Neural Networks 1 (1990) 4-27
9. Ge, S. S., et al,: A Direct Method for Robust Adaptive Nonlinear Control with Guaranteed Transient Performance. System Control Letter 37 (1999) 275-284
10. Lewis, F. L., et al,: Multilayer Neural-Net Robot Controller with Guaranteed Tracking Performance. IEEE Trans. Neural Networks 7 (1999) 388-398
11. Polycarpou, M. M.: Stable Adaptive Neural Control Scheme for Nonlinear Systems. IEEE Trans. Automatic Control 41 (1996) 447-450
12. Ge, S. S., et al: Adaptive Neural Networks Control of Robotic Manipulators. Singapore: World Scientific (1998)
13. Ge, S. S., et al: Stable Adaptive Neural Network Control. MA: Kluwer, Norwell (2001)
14. Long, F., Fei, S. M.: State Feedback Control for a Class of Switched Nonlinear Systems Based on RBF Neural Networks. In Proc. 23rd Chinese Control Conference 2 (2004) 1611-1614
15. Long, F., Fei, S. M., Zheng, S. Y.: H-infinity Control for Switched Nonlinear Systems Based on RBF Neural Networks. In: Wang, J., Liao, X, Yi, Z. (eds.): Advance in Neural Networks---ISNN2005. Lecture Notes in Computer Science, Vol. 3498. Springer-Verlag, Berlin Heidelberg New York (2005) 54–59
16. Long F., Fei S. M., Fu Z. M., Zheng S. Y.: Adaptive Neural Network Control for Switched System with Unknown Nonlinear Part by Using Backstepping Approach: SISO Case, In: Wang, J. et al (eds.): Advance in Neural Networks---ISNN2006. Lecture Notes in Computer Science, vol. 3972. Springer-Verlag, Berlin Heidelberg (2006) 842-848
17. Long, F., Fei, S. M., Zheng, S. Y.: Adaptive H^{∞} Control for A Class of Switched Nonlinear Systems Based on RBF Neural Networks. International Journal of Hybrid Systems 4 (2004) 369-380
18. Xu, H. L., Liu, X. Z., Teo, K. L.: Robust H-Infinity Stabilization with Definite Attendance of an Uncertain Impulsive Switched System. ANZIAM Journal 46 (2005) 471-484
19. Wang, Y. J., Xie, G. M., Wang, L.: Reachability and Controllability of Switched Linear Systems with State Jumps. Proceedings of the IEEE international conference on systems, man and cybernetics 1-5 (2003) 672-677

20. Xie, G. M., Wang, L.: Reachability of Switched Linear Impulsive Systems. Proceedings of the IEEE Conference on Decision and Control 6 (2003) 6271-6276
21. Zhang, H. T., Liu, X. Z.: Robust H-Infinity Control on Impulsive Switched Systems with Disturbance. Control Theory and Applications 21 (2004) 261-266
22. Xie, G. M., Wang, L.: Necessary and Sufficient Conditions for Controllability and Observability of Switched Impulsive Control Systems. IEEE Transactions on Automatic Control 49 (2004) 960-966
23. Guan, Z. H., Hill, D. J., Shen, X.: On Hybrid Impulsive and Switching Systems and Application to Nonlinear Control. IEEE Trans. on Automatic Control 50 (2005) 1058-1062
24. Long, F., Fei, S. M.: Tracking Stabilization for A Class of Switched Impulsive Systems Using RBF Neural Networks. Dynamics of Continuous Discrete and Impulsive Systems---Series A: Mathematical Analysis 13 (Part 1, Suppl. S) (2006) 356-363
25. Haykin, S.: Neural Networks: A Comprehensive Foundation (2nd edition). Prentice Hall, New York (1994)

Reliable Robust Controller Design for Nonlinear State-Delayed Systems Based on Neural Networks

Yanjun Shen, Hui Yu, and Jigui Jian

Institute of Nonlinear Complex system, College of Science,
Three Gorges University, Yichang, Huibei 443002, China
{shenyj, Yuhui}@ctgu.edu.cn, jianjigui@sohu.com

Abstract. An approach is investigated for the adaptive guaranteed cost control design for a class of nonlinear state-delayed systems. The nonlinear term is approximated by a linearly parameterized neural networks(LPNN). A linear state feedback H_∞ control law is presented. An adaptive weight adjustment mechanism for the neural networks is developed to ensure H_∞ regulation performance. It is shown that the control gain matrices and be transformed into a standard linear matrix inequality problem and solved via a developed recurrent neural network.

1 Introduction

Neural networks have often demonstrated their usefulness in the control of continuous or discrete nonlinear or unknown systems [1]. Most notably, due to the unknown and highly nonlinear behavior of neural networks during their learning phase, guaranteeing stability becomes a major problem [2,3].

In [3-5] A class of multi-layer neural networks that admits a linear difference inclusion (LDI) state-space representation to approximate a nonlinear systems has been observed and used in the stability analysis via a Lyapunov function. However, regrading H_∞ control by neural networks, to the best of our knowledge, only a few results are published[6-10]. For example, in [11], a class of nonlinear system is approximated by two multilayer perceptrons. In [12], an LDI state-space representation for a class of multilayer neural networks was established. Since time-delays are often present in all actuation and measurement, it is important to study the stability problems for time-delay systems [13-16].

In this paper, we discuss an H_∞ design approach for a class of nonlinear state-delayed systems by using neural networks. Throughout this paper, The Euclidean norm $\parallel x \parallel = \sqrt{x^T x}$; the weight Euclidean norm $\parallel x \parallel_Q = \sqrt{x^T Q x}$, where Q denotes the weighting matrix; $x : [0, \infty) \mapsto \Re^n$ belongs to the space $L_2^n[0, \infty)$ with the norm $\parallel x(t) \parallel_{L_2} = \sqrt{\int_0^\infty x^T(t)x(t)dt}$,if $\parallel x(t) \parallel_{L_2} < \infty$.

2 Neural Net-Based Description

Consider a nonlinear plant model described as follows:

$$\dot{x} = Ax + Bu + A_d x(t-\tau) + f(x) + d \tag{1}$$

where, A, B, A_d are known constant matrices; $x \in \Re^n$ is state vector; $u \in \Re^m$ is input control vector; $y \in \Re^z$ is output vector; $f(\cdot)\Re^n \to \Re^n$ is continuous nonlinear mapping with $f(0) = 0$ but not assumed *a prior* known; $d \in \Re^n$ is $d \in L_2^n[0, \infty]$; $\tau > 0$ is the delay parameter.

The model structure $f(x)$ are to be parameterized by LPNNs. For instance, high-order neural networks(HONNs), RBF networks, adaptive fuzzy systems, wavelet networks, etc., [17], [18],[19] belong to this class of NN. In general, the LPNNs are mathematically described by

$$y = WS(\zeta) \tag{2}$$

In this paper, the function is selected as follows:

$$s(\nu) = \lambda \frac{1 - e^{-\nu/q}}{1 + e^{-\nu/q}},\ \lambda > 0,\ q > 0 \tag{3}$$

More precisely, (1) becomes

$$\dot{x} = Ax + Bu + A_d x(t-\tau) + W^* S(x) + \Delta f(x) + d \tag{4}$$

where, $\Delta f(x) = f(x) - W^* S(x)$. Assume that $\| \Delta f(x) \| \leq \epsilon \| x \|$. Furthermore, (4) can be written as follows:

$$\dot{x} = Ax + Bu + A_d x(t-\tau) - \tilde{W} S(x) + W S(x) + \Delta f(x) + d \tag{5}$$

where, $\tilde{W} = W - W^*$. Next, we introduce some lemmas, which are needed in the proof of the main theorem.

Lemma 1

$$| s(\nu) | \leq \frac{\lambda}{q} | \nu | \tag{6}$$

Proof: We only proof when $\nu > 0$, the inequality holds. Let $h(\nu) = \frac{\lambda \nu}{q} - \lambda \frac{1-e^{-\nu/q_i}}{1+e^{-\nu/q_i}}$. Then, $h'(\nu) = \lambda(\frac{1}{q} - \frac{2e^{-\nu/q}}{q(1+e^{-\nu/q})^2}) > 0$, and $h(0) = 0$. So, we can obtain that $\frac{1-e^{-\nu/q}}{1+e^{-\nu/q}} \leq \frac{\nu}{q}$, i.e., $| s(\nu) | \leq \frac{\lambda}{q} | \nu |$.

Lemma 2: *Let D is a positive defined diagonal matrix, $S_1(x) = [s_1(x_1), s_1(x_2), \cdots, s_1(x_n)]^T$, and $\Sigma = diag[\lambda/q_1, 1/q_2, \cdots, \lambda/q_n]$, then the following inequality holds*

$$S_1^T(x) D S_1(x) \leq x^T \Sigma D \Sigma x \tag{7}$$

Proof: Assume $D = diag[d_1, d_2, \cdots, d_n]$, then,

$$S_1^T(x) D S_1(x) = d_1 s_1^2(x_1) + d_2 s_1^2(x_2) + \cdots + d_n s_1^2(x_n)$$

By lemma 1, we have $s_1^2(x_i) \leq \lambda^2/q_i^2 x_i^2, i = 1, 2, \cdots, n$. Thus, the inequality (7) holds.

Lemma 3[20]: *Given any real matrices Q_1, Q_2, Q_3 with appropriate dimensions such that $0 < Q_3 = Q_3^T$, the following inequality holds:*

$$Q_1^T Q_2 + Q_2^T Q_1 \leq Q_1^T Q_3 Q_1 + Q_2^T Q_3^{-1} Q_2 \tag{8}$$

3 H_∞ Neural Control Design

The state feedback control law is designed using the available plant information

$$u = Kx \tag{9}$$

By substituting into (5) gives the perturbed closed-loop system

$$\dot{x} = (A + BK)x + A_d x(t-\tau) - \tilde{W}S(x) + WS(x) + \Delta f(x) + d \tag{10}$$

Consider now the condition ensuring H_∞ regulation performance and stability of the perturbed system under feedback control.
i)H_∞ regulation with control penalty
The performance index under consideration is defined by [21]

$$\frac{\int_0^{t_f} [\| x(t) \|_Q^2 + \| u(t) \|_R^2] dt}{\int_0^{t_f} \| d(t) \|^2} < \rho^2, \quad \forall d(t) \in L_2^n[0, t_f] \tag{11}$$

where $Q = Q^T > 0$ and $R = R^T > 0$ are the weighting matrices, ρ is the attenuation level, usually a prescribed value.
ii) H_∞ regulation without control penalty

$$\frac{\int_0^{t_f} \| x(t) \|_Q^2 \, dt}{\int_0^{t_f} \| d(t) \|^2} < \rho^2, \quad \forall d(t) \in L_2^n[0, t_f] \tag{12}$$

Theorem 1: *Consider the perturbed system (5) with $x(0) = 0$. The controller (9) such that the closed-loop system achieves the regulation performance described by (11) provided that there exist a common matrix $P = P^T > 0$ satisfying:*

$$\begin{aligned} \Omega \equiv (A + BK)^T P + P(A + BK) + A_d^T A_d + \epsilon^2 I + \Sigma\Sigma \\ + PWW^T P + (2 + 1/\rho^2)PP + Q + K^T RK < 0 \end{aligned} \tag{13}$$

with the weight updating law:

$$\dot{W} = Px(t)S^T(x) \tag{14}$$

Proof: Construct the following Lyapunov-Krasovskii function:

$$V(t) = x^T(t)Px(t) + tr\{\tilde{W}^T \tilde{W}\} + \int_{t-\tau}^{t} x^T(\nu) A_d^T A_d x(\nu) d\nu \tag{15}$$

where $P = P^T > 0$, The time derivative of $V(t)$ is

$$\begin{aligned}\dot{V}(t) = {} & x^T(t)[(A+BK)^T P + P(A+BK)]x(t) - 2S^T(x)\tilde{W}^T Px(t) \\ & +2S^T(x)W^T Px(t) + x^T(t-\tau)A_d^T Px(t) + x^T(t)PA_d x(t-\tau) \\ & +2\varDelta f(x)^T Px(t)) + x^T(t)A_d^T A_d x(t) - x^T(t-\tau)A_d^T A_d x(t-\tau) \\ & +2x^T(t)Pd + 2tr\{\dot{\tilde{W}}^T \tilde{W}\}\end{aligned} \tag{16}$$

Meanwhile, we note that the following equality holds:

$$\begin{aligned}& x^T(t-\tau)A_d^T Px(t) + x^T(t)PA_d x(t-\tau) - x^T(t-\tau)A_d^T A_d x(t-\tau) \\ & = -[A_d x(t-\tau) - Px(t)]^T[A_d x(t-\tau) - Px(t)] + x^T(t)PPx(t)\end{aligned} \tag{17}$$

Therefore, we have

$$\begin{aligned}& x^T(t-\tau)A_d^T Px(t) + x^T(t)PA_d x(t-\tau) - x^T(t-\tau)A_d^T A_d x(t-\tau) \\ & \leq x^T(t)PPx(t)\end{aligned} \tag{18}$$

Thus

$$\begin{aligned}\dot{V}(t) \leq {} & x^T(t)[(A+BK)^T P + P(A+BK) + PP + A_d^T A_d]x(t) \\ & -2S^T(x)\tilde{W}^T Px(t) + 2S^T(x)W^T Px(t) + 2\varDelta f(x)^T Px(t)) \\ & +2x^T(t)Pd + 2tr\{\dot{\tilde{W}}^T \tilde{W}\}\end{aligned} \tag{19}$$

Since

$$\begin{aligned}2S^T(x)W^T Px(t) & \leq S^T(x)S(x) + x^T(t)PWW^T Px(t) \\ & \leq x^T(t)[\varSigma\varSigma + PWW^T P]x(t) \\ 2\varDelta f(x)^T Px(t)) & \leq \varDelta f^T(x)\varDelta f(x) + x^T(t)PPx(t) \\ & \leq x^T(t)[\epsilon^2 I + PP]x(t) \\ 2x^T(t)Pd & \leq 1/\rho^2 x^T(t)PPx(t) + \rho^2 d^T d\end{aligned}$$

Meanwhile, using the weight updating law described by (14), we can obtain

$$\begin{aligned}\dot{V}(t) \leq {} & x^T(t)[(A+BK)^T P + P(A+BK) + A_d^T A_d + \epsilon^2 I + \varSigma\varSigma \\ & +PWW^T P + (2 + 1/\rho^2)PP]x(t) + \rho d^T d\end{aligned} \tag{20}$$

Clearly, if there exists a common matrix $P = P^T > 0$ such that the following algebraic Riccati matrix inequalities

$$\begin{aligned}\varOmega \equiv {} & (A+BK)^T P + P(A+BK) + A_d^T A_d + \epsilon^2 I + \varSigma\varSigma \\ & +PWW^T P + (2 + 1/\rho^2)PP + Q + K^T RK < 0\end{aligned} \tag{21}$$

is satisfied, then (20) becomes

$$\dot{V}(t) \leq x^T(t)\varOmega x(t) - x^T(t)Qx(t) - u^T(t)Ru(t) + \rho^2 d^T d \tag{22}$$

Next, integrating both sides of the inequality from $t = 0$ to $t_f(> 0)$ yields

$$\begin{aligned}& \int_0^{t_f} x^T(t)Qx(t)dt + \int_0^{t_f} u^T(t)Ru(t)dt \\ & \leq \int_0^{t_f} x^T(t)\varOmega x(t)dt + \rho^2 \int_0^{t_f} d^T d dt + V(0) - V(t_f)\end{aligned} \tag{23}$$

Since $V(t) \geq 0$ for all $t \geq 0$, then

$$\int_0^{t_f} x^T(t)Qx(t)dt + \int_0^{t_f} u^T(t)Ru(t)dt \leq x^T(0)Px(0) + \rho^2 \int_0^{t_f} d^T d dt \tag{24}$$

This clearly shows that, as the dynamics starts with $x(0) = 0$, the H_∞ performance index defined by (11) is guaranteed with the prescribed level ρ.

It is impractical to analytically solve a common solution $P = P^T > 0$. Introduce the variables $X = P^{-1}$ and $Y = KP^{-1}$ transforms (21) into

$$\begin{aligned} XA^T + AX + Y^TB^T + BY + X[A_d^TA_d + \epsilon^2 I + Q + \Sigma\Sigma]X \\ +[WW^T + (2+\rho^2)I] + Y^TRY < 0 \end{aligned} \tag{25}$$

Using Schur's theorem[21], (26) can be equivalently expressed in the form of LMI's :

$$\begin{bmatrix} \begin{matrix} XA^T + AX + Y^TB^T \\ +BY + [WW^T + (2+\rho^2)I] \end{matrix} & X & Y^T \\ X & -[A_d^TA_d + \epsilon^2 I + Q + \Sigma\Sigma]^{-1} & 0 \\ Y & 0 & -R^{-1} \end{bmatrix} < 0 \tag{26}$$

or equivalently

$$(\bar{A}X + \bar{B}Y)C + C^T(X\bar{A}^T + Y^T\bar{B}^T) + \bar{Q} < 0 \ \ , X > 0 \tag{27}$$

where $\bar{A} = \begin{bmatrix} A \\ I \\ 0 \end{bmatrix}$, $\bar{B} = \begin{bmatrix} B \\ 0 \\ I \end{bmatrix}$, $C = [I \ \ 0 \ \ 0]$

$$\bar{Q} = \begin{bmatrix} WW^T + (2+\rho^2)I & 0 & 0 \\ 0 & -[A_dA_d + \epsilon^2 I + Q + \Sigma\Sigma]^{-1} & 0 \\ 0 & 0 & -R^{-1} \end{bmatrix}$$

The corresponding control gain is determined by

$$K = YX^{-1} \tag{28}$$

4 Construction of Neuro-solvers for LMI

As it can see from (9) and (27) that the control gain is dependent on matrix $\bar{Q}$, which is determined by the result of neural networks identification of W, solutions to the LMI (27) should therefore be solved adaptively. To solve the problem, several slack matrices are imposed which convert the problem of (27) into two matrix equalities:

$$G_\sigma(X, Y, R_\sigma) = (\bar{A}X + \bar{B}Y)C + C^T(X\bar{A}^T + Y^T\bar{B}^T) + \bar{Q} + \tilde{R}_\sigma\tilde{R}_\sigma \tag{30a}$$

$$G_1(X, R_1) = X - \tilde{R}_1\tilde{R}_1^T \tag{30b}$$

where X and Y are the solution matrices, and G_σ and G_1 are the objective matrices; the slack matrices $\tilde{R}_\sigma$ and $\tilde{R}_1$ are restricted to be nonsingular, positive definite as

$$\tilde{R}_\sigma = H_\sigma(R_\sigma)$$

$$= \begin{bmatrix} h_{\sigma 1}(r_{\sigma,11}) & 0 & 0 & \cdots & 0 \\ r_{\sigma,21} & h_{\sigma 2}(r_{\sigma,22}) & 0 & \cdots & 0 \\ r_{\sigma,31} & r_{\sigma,32} & h_{\sigma 3}(r_{\sigma,33}) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ r_{\sigma,(2n+m)1} & r_{\sigma,(2n+m)2} & r_{\sigma,(2n+m)3} & \cdots & h_{\sigma(2n+m)}(r_{\sigma,(2n+m)(2n+m)}) \end{bmatrix} \quad (31a)$$

$$\tilde{R}_1 = H_1(R_1) = \begin{bmatrix} h_{11}(r_{1,11}) & 0 & 0 & \cdots & 0 \\ r_{1,21} & h_{12}(r_{1,22}) & 0 & \cdots & 0 \\ r_{1,31} & r_{1,32} & h_{13}(r_{1,33}) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 0 \\ r_{1,n1} & r_{1,n2} & r_{1,n3} & \cdots & h_{1n}(r_{\sigma,nn}) \end{bmatrix} \quad (31b)$$

with $R_\sigma \in \Re^{(2n+m)\times(2n+m)}$,$R_1 \in \Re^{n\times n}$, $\tilde{r}_{s,jj} = h_s(r_{s,jj}) > 0$,$s = \sigma, 1, \forall j$. The existence of decomposition $\tilde{R}_s\tilde{R}_s^T$ for a positive definite matrix is based on the Cholesky decomposition [22].

The second step is to established a convex computation energy function:

$$E[G(X, Y, R_\sigma, R_1)]$$

$$= \sum_\sigma \sum_{i=1}^{2n+m} \sum_{j=1}^{2n+m} e_{ij}[g_{\sigma,ij}(X, Y, R_\sigma)] + \sum_{i=1}^{2n+m} \sum_{j=1}^{2n+m} e_{i,ij}[g_{1,ij}(X, R_1)] \quad (32)$$

where $e_{s,ij}$ are the objective function. The neural dynamics for solving the linear matrix equation can be described as follows:

$$\frac{dX(t)}{dt} = -\eta_X \frac{\partial E}{\partial X(t)} = -\eta_X \Phi_X \quad (33a)$$

$$\frac{dY(t)}{dt} = -\eta_Y \frac{\partial E}{\partial Y(t)} = -\eta_Y \Phi_Y \quad (33b)$$

$$\frac{dR_\sigma(t)}{dt} = -\eta_{r\sigma} \frac{\partial E}{\partial \tilde{R}_\sigma(t)} = -\eta_{r\sigma} \Xi_\sigma \quad R_\sigma(0) \neq 0 \quad (33c)$$

$$\frac{dR_1(t)}{dt} = -\eta_{r1} \frac{\partial E}{\partial \tilde{R}_1(t)} = -\eta_{r1} \Xi_r \quad R_1(0) \neq 0 \quad (33d)$$

$$\tilde{R}_s = H_s(R_s), s = \sigma, 1 \quad (33e)$$

where the matrix derivative

$$\frac{\partial E}{\partial X} = \begin{bmatrix} \frac{\partial E}{\partial x_{11}} & \cdots & \frac{\partial E}{\partial x_{1n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial E}{\partial x_{n1}} & \cdots & \frac{\partial E}{\partial x_{nn}} \end{bmatrix}, X \in \Re^{n\times n}$$

The same derivatives apply for $\partial E/\partial Y$ and $\partial E/\partial \tilde{R}_s, s = \sigma, 1$. In the above, $X(t), Y(t), R_s(t), s = \sigma, 1$ are activation state matrices of the recurrent neural networks, $\eta_X, \eta_Y, \eta_{rs} > 0$ are learning rates and

$$[\Phi_X]_{ij} = \sum_{\sigma} \sum_{k=1}^{2n+m} \sum_{l=1}^{2n+m} \frac{\partial g_{\sigma,kl}(X,Y,R_\sigma)}{\partial x_{ij}} f_{\sigma,kl}(g_{\sigma,kl}(X,Y,R_\sigma))$$
$$+ \sum_{k=1}^{n} \sum_{l=1}^{n} \frac{\partial g_{1,kl}(X,R_1)}{\partial x_{ij}} f_{1,kl}(g(X,R_1))$$
$$[\Phi_Y]_{ij} = \sum_{\sigma} \sum_{k=1}^{2n+m} \sum_{l=1}^{2n+m} \frac{\partial g_{\sigma,kl}(X,Y,R_\sigma)}{\partial y_{ij}} f_{\sigma,kl}(g_{\sigma,kl}(X,Y,R_\sigma))$$
$$+ \sum_{k=1}^{n} \sum_{l=1}^{n} \frac{\partial g_{1,kl}(X,R_1)}{\partial y_{ij}} f_{1,kl}(g(X,R_1))$$
$$[\Xi_\sigma]_{ij} = \sum_{k=1}^{2n+m} \sum_{l=1}^{2n+m} \frac{\partial g_{\sigma,kl}(X,Y,R_\sigma)}{\partial \tilde{r}_{\sigma,ij}} f_{\sigma,kl}(g_{\sigma,kl}(X,Y,R_\sigma))$$
$$[\Xi_1]_{ij} = \sum_{k=1}^{n} \sum_{l=1}^{n} \frac{\partial g_{1,kl}(X,R_1)}{\partial \tilde{r}_{1,ij}} f_{1,kl}(g_{1,kl}(X,R_1))$$
$$i,j = 1,2,\cdots,n$$

and $f_{s,kl}(g_{s,kl}) = \partial e_{s,kl}/\partial g_{s,kl}$ is the activation function. Note that $e_{s,kl}(g_{s,kl})$ and $f_{s,kl}(g_{s,kl})$ are function of $g_{s,kl}$ only. The constraints $\tilde{r}_{s,jj} > 0, \forall s, i$ imposing on $R(t)$ can be fulfilled by employed a limiting integrator with a nonlinear transformation

$$h_{si}(r_{s,jj}) = \begin{cases} r_{s,jj}, & r_{s,jj} > \epsilon, \\ \epsilon, & r_{s,jj} < \epsilon \end{cases} \forall s, j \tag{34}$$

where ϵ is an arbitrarily small positive constant. The proof for the networks stability follows the derivative proposed elsewhere [12][23]. Applying the derivative presented in (33) and lemma 3 in [23], the derivative neural dynamics for solving the LMI of (30) is now given by

$$\frac{dX(t)}{dt} = -\eta_X\{\sum_{\sigma}[\bar{A}^T F_\sigma(X,Y,\tilde{R}_\sigma)C + C^T F_\sigma(X,Y,\tilde{R}_\sigma)\bar{A}] + F_1(X,\tilde{R}_1)\} \tag{35a}$$
$$\frac{dY(t)}{dt} = -\eta_Y\{\sum_{\sigma}[\bar{B}^T F_\sigma(X,Y,\tilde{R}_\sigma)C + C^T F_\sigma(X,Y,\tilde{R}_\sigma)\bar{B}]\} \tag{35b}$$
$$\frac{dR_\sigma(t)}{dt} = -\eta_{r\sigma} F_\sigma(X,Y,\tilde{R}_\sigma)\tilde{R}_\sigma, R_\sigma(0) \neq 0, \forall \sigma \tag{35c}$$
$$\frac{dR_1(t)}{dt} = -\eta_{r1} F_1(X,Y,\tilde{R}_1)\tilde{R}_1, R_1(0) \neq 0, \tag{35d}$$

where the activation matrices are

$$F_\sigma[X,Y,\tilde{R}_\sigma) = F_\sigma[(\bar{A}X + \bar{B}Y)C + C^T(X\bar{A}^T + Y^T\bar{B}^T) + \tilde{Q} + \tilde{R}_\sigma\tilde{R}_\sigma^T]$$
$$F_1(X,\tilde{R}_1) = F_1(X - \tilde{R}_1\tilde{R}_1^T)$$

Note that to ensure the steady state of X is positive definite, we require that $R_\sigma(t)$ and $R_1(t)$ converge faster than X and Y. Therefore, we choose $\eta_{r\sigma}, \eta_{r1} >> \eta_X, \eta_Y$.

5 Example

Consider the following nonlinear system:

$$\dot{x}(t) = \begin{bmatrix} -0.5 & 0 \\ 0.5 & -3 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

$$+ \begin{bmatrix} 0 & 0.5 \\ 0.7 & -0.1 \end{bmatrix} x(t-2) + \begin{bmatrix} 0 \\ \exp(-(x_1+x_2))\cos(x_1+x_2) - 1 \end{bmatrix} + d$$

where, $x_i(t) = 0$ for $t \in [-2, 0]$ and d denotes the vector with random entries, chose from a normal distribution mean zero and variance one. First, specify the prescribed attenuation level $\rho = 0.5$, and the approximation errors of LPNNs $\epsilon = 0.02$. In addition, choose the weighting matrix for the performance index $Q = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.3 \end{bmatrix}$, $R = [0.33]$. A LPNNs was used to approximate the nonlinear term $\exp(-(x_1+x_2))\cos(x_1+x_2)-1$. Select $S(x) = [6(\frac{1-e^{-x_1/2}}{1+e^{-x_1/2}}) \quad 8(\frac{1-e^{-x_1}}{1+e^{-x_1}})]^T$. The initial values of W, Y, R_σ, R_1 are placed by the uniformly distributed random numbers in [-1,1]. Select the $\eta_X = 5$, $\eta_Y = 5$, $\eta_{r\sigma} = 80$, $\eta_{r1} = 80$. Use the weight update law (14) to adaptively train the LPNNs. Propagate the recurrent neural dynamics to (33) find the solution of LMIs (30), where

$$\bar{A} = \begin{bmatrix} -1 & 0 \\ 0.5 & -3 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad \bar{B} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} \bar{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Fig.1 and Fig.2 illustrate the convergence behavior of X and Y, respectively.

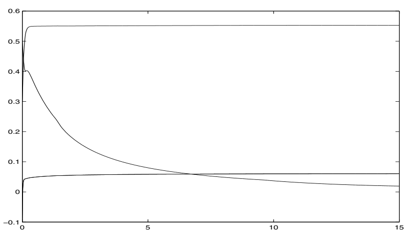


Fig. 1. Transient of X

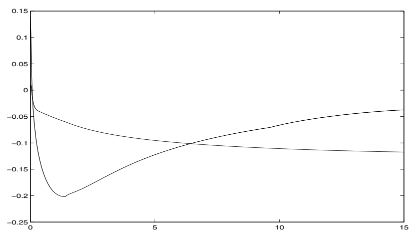


Fig. 2. Transient of Y

The steady state of the matrix P was obtained as

$$P(\infty) = \begin{bmatrix} 2.6185 & -7.4131 \\ -7.4131 & 67.9346 \end{bmatrix}$$

Clearly, P is symmetric and positive definite. The control gain $k(\infty) = [13.8327 - 127.7577]^T$. Fig. 3 and Fig.4 show, respectively, the state response for the system without control and with neuro control.

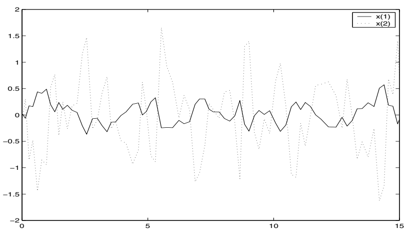


Fig. 3. State response of system without control

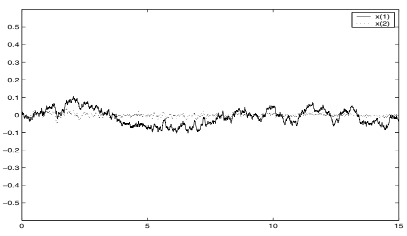


Fig. 4. State response of system with control

6 Conclusions

In this paper, we have addressed the design and analysis an adaptive neural net-based H_∞ controller for for a class of nonlinear state-delayed systems . The non-linear term is approximated by a linearly parameterized neural networks(LPNN). A linear state feedback guaranteed cost control law is presented. An adaptive weight adjustment mechanism for the neural networks is developed to ensure H_∞ regulation performance.

Acknowledgments

This work was supported by the Science Foundation of Education Commission of Hubei Province D200613002, the Doctoral Pre-research Foundation of Three Gorges University.

References

1. Suykens, J.A.K., Vandewalle, J., and De Moor, B.: Artificial neural networks for modelling and control of nonlinear systems. Kluwer, Boston, Massachussetts, 1996.
2. Bass . E., and Lee, K.Y.: Robust control of nonlinear system using norm-bounded neural networks, IEEE Int. Conf. on Neural Networks , (4)1994 2524-2529.
3. Lin, C.L.: Control of perturbed systems using neural networks, IEEE trans. Neural Netw., (9)1998 1046-1050.
4. J.A.K. Suykens, J.Vandewalle, and B. De Moor: Artificial Neural Networks for Modelling and Control of nonlinear system. Norwell, MA:Kluwer, 1996.
5. J.A.K. Suykens, J.Vandewalle, and B. De Moor: Lur's systems with multilayer perceptron and recurrent neural networks: Absolute stability and dissipativity. IEEE Trans.Automat, Contr., (44)1999 770-774.
6. Tanaka, K.: An approach to stability criteria of neural-network control systems. IEEE Trans. Neural Netw,. (7)1996 629-642.
7. Leu, Y.G., Lee, T. T., and Wang, W.Y.: Online tuning of fuzzy-neural network for adaptive control of nonlinear dynamical systems, IEEE Trans. Syst., Man. Cybern. Part B, (27)1997 1034-1043.
8. Limanond. S. and Si, J.: Neural-network-based control design: an LMI approach, IEEE Trans. Neural Netw., (9)1998 1442-1429.

9. J.A.k. Suykens, J.Vandewalle, and B. De Moor: Artificial Neural Networks for Modelling and Control of Nonlinear Systems. Norwell, MA: Kluwer, 1996.
10. Chun-Liang Lin and Tsai-Yuan Lin: An Design Approach for Neural Net-based Control Schemes, IEEE Trans. On. Automatic control, (46)2001 1599-1605.
11. C.-L. Lin and T.-Y. L.: Approach to adaptive neural net-based H_∞ control design, IEE Proc.-Control Theory Appl. (4)2002 331-342.
12. Dugard, L. and Verriest, E.I.: Stability and control of time-delay systems, Prentice Hall, Springer-Verlag, Berlin, 1997.
13. Lee, J.H., Kim, S.W. and Kwon, W.H.: Memoryless H_∞ controllers for state delayed systems, IEEE Trans. Autom. Control, (39)1994 159-162.
14. Niculescu, S.I.: H_∞ memoryless control with an a-stability constraint for time-delay systems: An LMI approach, IEEE Trans. Autom. Control, (43)1998 739-748.
15. Choi, H.H., and Chong, M.J.: An LMI approach to H_∞ controller design for linear time-delays, Automatical, (3)1997 737-739.
16. E.B.Kosmatopoulos, M.M.Polycarpou, M.A.Christodoulou, and P.A.Ioan: High-order neural networks structures for identification of dynamical systems, IEEE Trans. Neural Networks, 6(1995) 422-431.
17. R.M.Sanner and J.J.E.Slotine: Structurally dynamic wavelet networks for the adaptive control of uncertain robotic systems, in Proc. IEEE 34th Conf. Decision and Control , New Orleans, LA, DEC. 1995 2460-2467.
18. M.M.Polycarpou and P.A.Ioannou: Identification and control of nonlinear systems using neural network models: Design and stability analysis, Dept. Eng. Sys., Southern Calif., Los Angeles, Tech. Rep. 1991.
19. E.N.Sanchez and J.P.Perez: Input-to-state stability (ISS) analysis for dynamic NN, IEEE Trans. Circuits Syst. I, (46)1999 1395-1398.
20. BOYD,S., GHAOUI,L.E., FERON,E., and BALAKRISHNAN, V.: Linear matrix inequality in system and control theory, (SIAM, Philadelphia, 1994)
21. Stewart, G.W.: Introduction to matrix computation, (Academic Press, Orlando, Florida, 1973)
22. Chun-Liang L., Chi-Chih L. and Teng-Hsien H., A neural networks for linear matrix inequality problems, IEEE Trans. On neural networks, (11)2000 1078-1092.
23. A.Stoorvogel: The H_∞ Control Problem: A state Space Approach. Upper Saddle River, NJ:Prentice-Hall, 1992.
24. C.M.Marcus and R.M.Westervelt: Stability of analog neural networks with time delay, Phys. Rev., A 39(1989) 347-359.
25. S.A.Campbell and P.Driessche. Frustration: stability and delay induced oscillations in a neural network model, SIAM J.Appl. Math., 56(1996) 245-255.
26. P.Baldi and A. F. Atiya: How delays affect neural dynamics and learning, IEEE Trans. Neural Networks, (5)1994 612-621.
27. A. Stoorvogel: The H_∞ Control Problem: A State Space Approach . Upper Saddle River, NJ: Prentice-Hall, 1992.

Neural Network Applications in Advanced Aircraft Flight Control System, a Hybrid System, a Flight Test Demonstration

Fola Soares[1], John Burken[2], and Tshilidzi Marwala[3]

[1] Contek Research, El Segundo, California, U.S.A
fola@contekresearch.com
[2] NASA Dryden Flight Research Center, Edwards, CA, U.S.A
john.burken@mail.dfrc.nasa.gov
[3] University of the Witwatersrand, Johannesburg, South Africa
t.marwala@ee.wits.ac.za

Abstract. Modern exploration missions require modern control systems that can handle catastrophic changes in behavior, compensate for slow deterioration in sustained operations, and support fast system identification. The dynamics and control of new vehicles remains a significant technical challenge. Neural network based adaptive controllers have these capabilities, but they can only be used safely if proper Verification and Validation can be done. Due to the nonlinear and dynamic nature of an adaptive control system, traditional Verification and Validation (V&V) and certification techniques are not sufficient for adaptive controllers, which is a big barrier in their deployment in the safety-critical applications. Moreover, traditional methods of V&V involve testing under various conditions which is costly to run and requires scheduling a long time in advance. We have developed specific techniques, tools, and processes to perform design time analysis, verification and validation, and dynamic monitoring of such controllers. Combined with advanced modelling tools, an integrated development or deployment methodology for addressing complex control needs in a safety- and reliability-critical mission environment can be provided.

1 Introduction

The performance of aircraft systems is highly dependent on the capabilities of the guidance, navigation and control systems. This necessitates the need to have sophisticated and reliable control systems. In order to maximize the performance, the control system needs to be adaptive in nature. However, due to the nonlinear and dynamic nature of adaptive control systems, traditional Verification and Validation (V&V) and certification techniques are not sufficient for adaptive controllers, which is a big barrier in their deployment in the safety-critical applications [1]. Moreover, traditional methods of V&V involve testing under various conditions which is costly to run and requires scheduling a long time in advance. This paper introduces an advanced V&V tool that enables a rapid, high-confidence, and cost efficient design of revolutionary systems which is validated by the IFCS flight test. To demonstrate the capabilities of the proposed V&V tools, the results from the laboratory shall be compared to the results obtained from Matlab/Simulink for the IFCS flight.

2 Research Overview and Background

Developing flight control systems for today's aerospace vehicles is time consuming. While the theory is understood, its application is lengthened as a result of three factors. First, flight control is an interdisciplinary subject that integrates mathematical models, control theory, computers, hydraulic and electrical systems, specifications, and pilots. Combining these pieces into a unified framework is a challenge. Second, imperfect knowledge of a contributing component results in costly flight-test iterations. Manufacturers have previously allotted 25% of the total flight test development time for flight control evaluation and iteration. Third, after a control system design is finalized, it can take two weeks to just evaluate the control system against the myriad of often conflicting design specifications. The use of adaptive control system has been growing in flight control. In the past few years, there has been an increasing interest within the control community in exploring the promise of biologically motivated algorithms, like fuzzy sets, neural networks as well as genetic algorithms to solve difficult optimization and control problems [2],[3]. Recently, an on-line adaptive architecture that employs a Sigma-Pi neural network has been applied to augment the attitude control system [4] and it has been shown that the on-line neural network in adaptive control architecture is very effective in dealing with the performance degradation problem of the trajectory tracking control. Melin and Castillo [5] have also used adaptive intelligent control of aircraft systems with a hybrid approach combining neural networks, fuzzy logic and fractal theory. Extensive research has also been conducted to investigate the certification of an adaptive flight control [6]. Highly reliable adaptive control systems are needed to fulfill the present and future aerospace needs. Adaptive control technologies that incorporate learning algorithms can enable a wide range of expanded capability, including reconfigurable avionics systems, automatic vehicle recovery and safing, real-time load, performance, and health monitoring. Adaptive control systems can be used to improve or maintain vehicle control in unknown, changing, or poorly defined operating environments. The verification and validation of adaptive neural flight control systems that can provide adaptive control to an aircraft without using extensive gain-scheduling or explicit system identification [7] are of great interest. However, adaptive systems that can reconfigure themselves "on the fly have been proposed for both commercial and military aircraft, as well as a host of NASA spacecraft. As capable and desirable as these new control technologies appear, adaptive control systems can only become part of the future vision if performance guarantees can be established. Rigorous methods for adaptive software verification and validation must be developed by NASA and others to ensure that disabling control system software failures will not occur, and to demonstrate certification requirements can be satisfied. To help bridge this gap, NASA is conducting Intelligent Flight Control System (IFCS) flight tests research aimed at developing usable procedures and methods that can verify the reliability of adaptive flight control system software. The IFCS is designed to incorporate self learning neural network concepts into flight control software to enable a pilot to maintain control and safely land an aircraft that has suffered a failure to a control surface or damage to the airframe. The IFCS GEN II project goal is the development and flight evaluation of a direct adaptive neural network based flight control system. The direct adaptive approach incorporates neural networks that are applied directly to the flight control system feedback errors to provide adjustments to aircraft performance in both normal flight and with system failures. The IFCS project is a collaborative effort among NASA centers namely Dryden Flight Research Center (DFRC) and Ames Research Center (ARC), the universities and the industry partners. The technical approach includes: integrating the GEN II online learning neural networks with an advanced flight controller; testing the software on the NASA 837 flight control ground test bench; and installing the system in the aircraft and conducting a flight test evaluation in a limited flight envelope at DFRC. Flight testing includes simulated failures to the control system. The results of the tests demonstrate the performance and the test process for advanced online learning neural network technology. The IFCS GEN II project research objectives as defined in the F-15 IFCS GEN II Project Plan are to: 1) Implement and fly neural network software for flight controls (in support of Information Technology Base, Intelligent Systems, and Vehicle Systems programs) 2) Develop Verification and Validation procedures for flight critical checkout of non-deterministic neural net software (in support of Information Technology Base, Intelligent Systems, and Vehicle Systems programs) 3) Demonstrate safe in-flight simulated failure recovery using learning neural network software with adaptive flight controls (in support of the Design for Safety program, Information Technology Base program, Intelligent Systems program, and Vehicle Systems programs)

3 Proposed Approach

In order to fulfill the requirements for complex systems and missions for aerospace systems, a unified approach for Verification and Validation for design, analysis, implementation, and monitoring of the system is necessary. Our approach uses a unique combination of mathematically rigorous analysis (e.g., Lyapunov based methods, Bayesian) with intelligent testing and dynamic performance monitoring. Due to the adaptive nature, the performance of an adaptive controller needs to be monitored during the actual mission for safe operation in unknown and possibly changing environments. Only a dynamic measure can tell for sure if the controller is still within the safety margin, even under unanticipated or un-modeled changes in the environment. A signal of low control performance can be used for early fault detection and analysis, potentially reducing the effort. The dynamics and control of new vehicles remains a significant technical challenge. The V&V tool was used to abate these challenges in a cost-effective manner to achieve credible flight control designs.

3.1 Conventional (Non-adaptive) Feedback Control System

Fig. 1 illustrates the basic anatomy of a simple conventional control system. In the case of an aviation example, sensors measure the aircraft state parameters to be controlled (e.g., pitch angle). After some signal conditioning, the measured state is compared to the desired state to generate a measure of the difference or error. The controllers function is to produce a control inputs to be sent to the aircraft control surfaces to reduce the measured error.

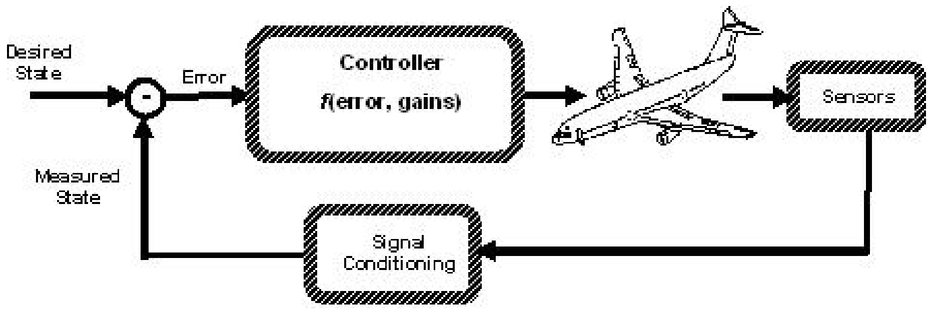


Fig. 1. Conventional (non-adaptive) feedback control system

The most widely used traditional non-adaptive design is the PID (Proportional Integral- Differential) controller, due to its simplicity, performance and robustness. Tuning the controller is a matter of finding the right gain settings. If the gains are selected too large, the system may exhibit instability; yet if selected too low, the system response may become sluggish. Although fairly simple to implement, PID and other types of conventional controllers unfortunately have the limitation that once the controller is put into operation, the gains cannot be changed. If the performance of the controller degrades after start-up, the only remedy is to stop the controller, re-tune the gains, and then restart the controller. This process continues until a combination of parameters is found that produces the desired results [8]. If the aircraft or spacecraft to be controlled or its operating environment should change significantly, new gains will be needed to optimize performance. The challenge is that re-tuning the gains may not always be practical if the behavior of the process changes too frequently, too rapidly, or too much. The tuning process can be excessively time consuming. Moreover, changes to the plant that favor increased gain settings may also cause the control system to become unstable.

3.2 Adaptive Control System Architecture

Learning or system identification is one of the primary objectives of adaptive control. One approach is to identify a mathematical model of a plant by using historical process data to predict future behavior. Adaptive model-based controllers generate real time updates automatically while the controller is on-line and can then use the adapted model to select future control actions that will drive the process accurately even if its behavior changes over time. Fig. 2 provides a notional diagram of an adaptive control system to illustrate the role of learning. The controller gains are not fixed, but rather learned by a system identification method or neural network. For example, the control law can be an LQG or H2 regulator concept which calculates the control gains based in part on adaptive system identification. The system identification may be done in real-time (on-line) by a number of means (e.g., Kalman filter, LMS algorithm) and may either directly identify control derivatives or transfer matrix model elements. The control signals are thus not only generated as a function of the performance error only, but also of the identified system parameters. It is also possible to formulate the controller to contain both minimum variance and PID control elements. For such cases, the single controller block in Fig. 2 can be envisioned to contain sub-blocks for the PID and minimum variance control components without loss of generality.

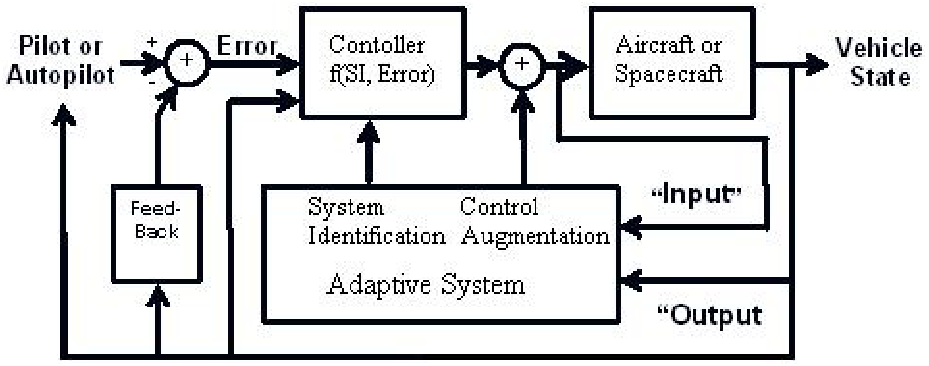


Fig. 2. Generic adaptive control system

In addition to system identification, some learning systems may also be designed to produce direct control augmentation commands. Such systems typically use novel identification methods such as adaptive inverse identification or neural networks to directly synthesize the adaptive controls needed to maintain aircraft or spacecraft performance. This approach is typically used to achieve adaptation for cases in which the controller block contains only fixed-gain, PID elements.

4 Research Challenges

4.1 Software Verification and Validation

A serious challenge hindering the deployment of advanced, flight-critical software is the requirement to show that it can operate as intended and with very high reliability. Adaptive controllers that can make rapid and automatic adjustments to enable self-healing in the event of vehicle damage, might also act to make a healthy aircraft un-flyable or a safety hazard to other vehicles. How can it be assured that safety-critical malfunctions never occur? The software implementation must be thoroughly analyzed and checked to provide sufficient assurance of its intended functionality, safety, and the absence of aberrant functionality. This process of analyzing and checking the correctness of software is termed verification and validation.

4.2 Special V&V Challenges of Adaptive Control Software

The additional verification complexity posed by adaptive control systems primarily stems from the use of the learning algorithm. It is fairly easy to realize (from Figure 2) that if the controller gains are based on numerical values passed from the learning algorithm to the controller, then malfunctioning of the learning algorithm can lead to the calculation of controller gains that are too low or too high, and hence produce suboptimal controller performance. In this section, the special challenges of adaptive control systems will be examined. Generally, these special challenges relate to maintaining stable and convergent learning. Adaptive control systems derive their adaptive utility by using learning algorithms to identify transfer matrix models or determine the coefficients of neural networks. A difficulty is finding suitable values for the learning gains that provide stable adaptation, while allowing convergence to the desired solution in a sufficiently short time.

5 Experimentation

The Generation II concept is based on a dynamic inversion controller with a model-following command path. The feedback errors are regulated with a proportional plus integral (PI) controller. This basic system is augmented with an Adaptive Neural Network that operates directly on the feedback errors. The Adaptive Neural Network adjusts the system for un-predicted behavior, or changes in behavior resulting from damage. Demonstration of this direct adaptive neural network is the primary objective of the IFCS Generation II flight project as mentioned in section 2 above.

5.1 Flight Test Description - NASA 837

The test aircraft used for the IFCS programs is a pre-production NF-15B (USAF S/N 71-0290, NASA 837), which has been extensively modified to serve as a flexible platform for flight controls research. This aircraft is equipped with a digital fly-bywire flight control system, canard control surfaces, and thrust vectoring nozzles. Thrust vectoring was not used for this program.

5.2 Envelope Clearance Approach

A systematic flight test followed a build-up approach, with prior checkout of all test maneuvers in the simulator. Initial flights will verify basic functionality of the installed GEN II systems. Engage/disengage checks are performed, and the auto-disengage envelope are verified. A standard sequence of maneuver groups are flown, consisting of 1g clearance maneuvers, followed by a series of basic maneuvering points, and finally a handling qualities research assessment, consisting of 1g formation flight and 3g tracking maneuvers. This sequence is repeated progressing through the available IFCS configurations: conventional mode, default enhanced mode, Neural Net ON, default enhanced mode with simulated failures introduced, and Neural Net ON with simulated failures introduced. Two types of simulated control system failures are available: locking one stabilator at a selected angle from trim, and changing the programmed canard response gain factor (canard gain multipliers). The failures are introduced separately in a build up sequence, and aircraft response evaluated using the standard maneuver sequence of 1g clearance, basic maneuvering, and handling qualities tests.

6 Results

The following section will show results from simulations of [A] matrix and [B] matrix failures. One type of simulated failure, which represents an aerodynamic type of failure, inserts a multiplier onto the canard surface command (change in Cmalpha). The second type of simulated failure, which represents a surface failure, inserts a jammed stabilator failure [B matrix]. Results from the simulation are presented to illustrate the flight test flown under FC1 and FC2 that highlight the benefits provided by the Gen 2 control system. For simulation, all of the pilot inputs to the simulation time histories are from canned piloted stick inputs and no attempt to correct for the aircraft attitudes are added to the piloted inputs. This canned pilot input method was used only for comparison purposes and not intended for flight test. The controller is a rate command system, therefore the attitudes such as bank angle (phi) are for comparison purposes only; and as such are used for disturbance rejection trade-off studies. For instance, when a failure is imparted on the aircraft and the resulting attitudes change minimally, the control system is said to have good robustness properties.

7 Discussion and Conclusions

This paper demonstrates the application of our V&V tools for adaptive control system as demonstrated by actual flight tests flown at NASA DFRC. It presented simulation results of a neural network adaptive controller compensating for errors resulting from aerodynamic and control surface failures. The controller is a hybrid controller that uses a simplified dynamic inverse control for the pitch and roll axes, while using a classical β-dot controller for the yaw axis. The neural network is an on-line direct adaptive algorithm that attempts to drive the error between the reference model and the commanded state to zero. The failures demonstrated are an aerodynamic failure type ([A] matrix) and a jammed control surface failure ([B] matrix). In both failure cases, benefits where shown to be obtained using neural networks compared to the non-adaptive controller as predicted by the V&V tool. The [A] matrix failure showed improved damping and better tracking with the neural networks active compared to the no adaptation case. And for the [B] matrix failure with a jammed surface, the neural networks reduce the cross coupling by 40 percent, implying the pilot can fly to the desired trajectory/path easier with less tracking errors.

References

1. Cukic, B.: The need for verification and validation techniques for adaptive control system, Proceedings of the Fifth International Symposium on Autonomous Decentralized Systems, (2001) 297–298.
2. Mehrabian, A. R., Lucas, C., Roshanian, J.: Aerospace launch vehicle control: an intelligent adaptive approach, Aerospace Science and Technology, Vol. 10, (2006) 149-155.
3. Kim B. S., Calise A. J.: A Nonlinear flight control using neural networks, Journal of Guidance, Control Dynamics, Vol. 20 (1997) 26-33.
4. Lee, S, Ha, C., Kim B.S.: Adaptive nonlinear control system design for helicopter robust command augmentation, Aerospace Science and Technology, Vol. 9, (2005) 241-251.
5. Melin, P, Castillo, O.: Adaptive intelligent control of aircraft systems with a hybrid approach combining neural networks, fuzzy logic and fractal theory, Applied Soft Computing, Vol. 3 (2003) 353-362.
6. Gobbo D. D, Mili, A.: An application of relational algebra: specification of a fault tolerant flight control system, Electronic Notes in Theoretical Computer Science Vol. 44 (2003) 1–18.
7. Peterson, B. B., Narendra, K. S., Bounded error adaptive control, IEEE Transactions on Automatic Control, Vol. 27, (1982) 1161–1168.
8. Van de Vegte, J.: Feedback control systems, Prentice Hall International Editions, Third edition, (1994) 134–148

Vague Neural Network Based Reinforcement Learning Control System for Inverted Pendulum

Yibiao Zhao[1], Siwei Luo[2], Liang Wang[3], Aidong Ma[3], and Rui Fang[2]

[1] Beijing Jiaotong University, School of Traffic and Transportation
[2] Beijing Jiaotong University, School of Computer and Information Technology
[3] Beijing Jiaotong University, School of Electronics and Information Engineering
100044 Beijing, P.R. China
Yibiao_zhao@ieee.org

Abstract. Reinforcement learning is a class of model-free learning control method that can solve Markov decision problems. But it has some problems in applications, especially in MDPs of continuous state spaces. In this paper, based on the vague neural networks, we propose a Q-learning algorithm which is comprehensively considering the reward and punishment of the environment. Simulation results in cart-pole balancing problem illustrate the effectiveness of the proposed method.

1 Introduction

Reinforcement learning (RL) has been an active research area not only in machine learning but also in control engineering, operations research and robotics in recent years. It is a model-free learning control method that can solve Markov decision problems.

In general, the machine learning is separated into the two categories of supervised and unsupervised learning by whether the teaching signal is needed or not. The supervised learning needs teaching signal from the exact modeling of the environment, but the reinforcement learning is generally unsupervised learning algorithm. It is the process by which the response of a system to a stimulus is strengthened by reward and weakened by punishment.

But there are still many difficulties for the application of reinforcement learning control, especially in MDPs with continuous state space, such as slow convergence. In this paper, we discussed some problems with Q-learning algorithm in cart-pole balance problem and proposed Vague Neural Networks to solve it.

Neural networks have extensively been used as associative memory and optimization. They have many well known advantages such as error tolerant and self-learning capacity. The base idea of the fuzzy neural network is to realize the process of fuzzy reasoning by the structure of neural network and to make the parameters of fuzzy reasoning be expressed by the connection weights of neural network. But Fuzzy theory has it's shortages as Daniel J.Buehrer and Wen-Lung Gun pointed out fuzzy neural network has its shortages: fuzzy membership function has only one single value, it cannot get more reasonable classified and cognizable results [2]. Vague set theory is a new extension form fuzzy set. Vague sets' distinguishing feature is having a truth-membership function and a false-membership function. It presents both of the opposite factors to deal with nonlinearities and uncertain of system in the research fields. It overcomes the disadvantage of membership function in fuzzy set which cannot characterize both the similarity and dissimilarity between pairs of objects. Therefore, VNN has been conceived as a new effective tool to deal with ambiguous data and applied successfully in different fields.

In this paper, we use vague neural network (VNN) to memorize and optimize the positive and negative Q values of Q-learning algorithm. Compared with the former method and simulation results, we present an optimal scheme for the design of a vague neural network as a controller. And the simulation results on the cart-pole balancing problem illustrate the effectiveness and the advantages of the proposed method.

2 Cart-Pole Balancing Problem

A common benchmark problem for nonlinear controllers is the inverted pendulum problem. The earliest application of neural networks to the inverted pendulum problem is accomplished by Widrow and Smith (1964) and Widrow (1987). Other researchers' work (Barto, 1983; Anderson, 1987; Berenji, 1991, 1992, 1993) also involves an inverted pendulum control by neural networks.

It is one of the simplest inherently unstable systems and has a broad base for comparison throughout the literatures. The goal is to apply a fixed magnitude force to one side of the cart to accelerate it at each time step, so as to keep the pole always upright and within a fixed angle from the vertical, and to keep the cart away from the ends of the track (Fig.1).

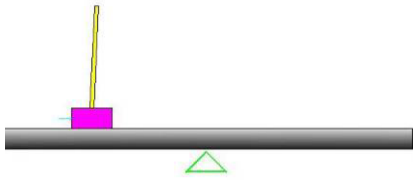


Fig. 1. Inverted pendulum model

2.1 Inverted Pendulum Modeling

The mass of the cart plus the pole m=1.1kg, the gravity acceleration constant g=9.8m/s2, half the length of the pole l =0.5m, the mass of the pole m_p =0.1kg, the output of the agent F_t =+10N or –10N, the sampling period of Δt =0.02s. The pole must be kept within $\pm 0.21 rad$ from vertical. Equations of Motion can refer to many other sources.

In this problem, the action size is just two: push left and push right. And we define the cart-pole system into 3*3*6*3=162 states shown as follow:

Table 1. State space

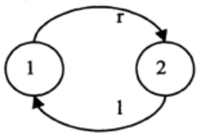
Cart position	-2.4 ~ -0.8		-0.8 ~ 0.8		0.8 ~ 2.4	
Velocity of cart	<-0.5		-0.5 ~ 0.5		>0.5	
Pole angle	<-6	-6 ~ -1	-1 ~ 0	0 ~ 1	1 ~ 6	>6
Angle velocity of the pole	<-50		-50 ~ 50		>50	

2.2 The Problems of Q-Learning in Inverted Pendulum

Inverted pendulum is an inherently unstable system. In real control applications, we find there are some problems which have a bad impact on the learning results.

2.2.1 Limit Cycle Problem

As Yu Zheng pointed out, in [5], that Q-learning will get into the limit cycles problem in inverted pendulum applications. The high degree exploration can not solve this problem, but rather intensify it. In inverted pendulum controlling, Agent receives a negative signal as a penalty, and learns an optimal policy to keep the inverted pendulum stand as long as possible. The requirement will lead the limit cycle bring terrible effect to the convergence of Q-learning. The negative Q values of each pair of state-action in limit cycle will update iteratively, which make negative Q values of all pairs of state-actions converge to zero.



$$Q(1,r) \leftarrow (1-\alpha)Q(1,r) + \alpha\gamma Q(2,l)$$
$$Q(2,l) \leftarrow (1-\alpha)Q(2,l) + \alpha\gamma Q(1,r)$$

Fig. 2. A simple limit cycle situation

In this way, we can not choose the optimal control action from comparing negative Q values of every action in each state. The optimal control policy can not keep it stability, and it will be breakdown ultimately.

A penalty only makes the Q-learning get a policy to avoid failure. But in the control of inverted pendulum, we require Q-learning get a control policy which keeps the inverted pendulum swing around the balance position. The optimal policy of avoid failure is not equal to the optimal policy of successful control. We must give agent a reward when the inverted pendulum is in the balance position.

2.2.2 Convergence Problem

On the other hand, the definition of reinforcement learning process is based on the MDPs with discrete state spaces. Apparently, the state spaces of inverted pendulum do not meet the need of it. Although we can solve the problem by subdivision, it leads new problem, curse of dimensionality.

In MDPs of continuous state spaces, the result of one state-action pair is not the same all the time. Different results lead to distinct responds of environment. Once a good state take an action, then get a bad response or even failed, the penalty will cause the Q value fluctuate or vary irregularly. As a result, it makes the Q value can not converge continuously. In this way, learning algorithm can not get the optimal control policy, namely, the dimensions of state spaces directly impact on the converge result. Many reference ascribe the problem to an uncertain reward and action, it's an inaccurate explanation. Actually, continuous state spaces have a certain reward.

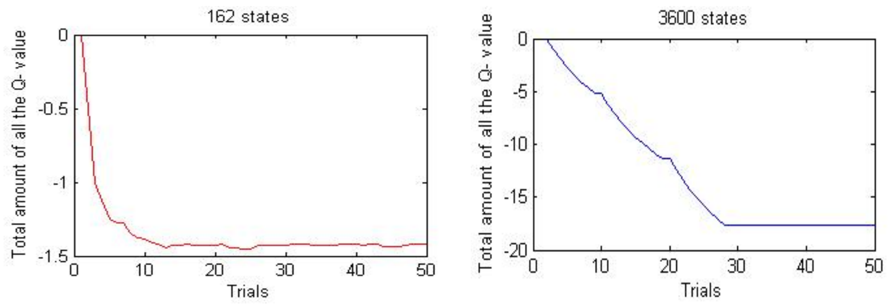


Fig. 3. Dimensions of state spaces impact on the convergence

To solve these problems, we have to get a positive Q value and a negative Q value from reinforcement signal. At the beginning, penalties will make the Q-learning get a policy to avoid failure. With the learning progressing, the influence of Q+ value must be increasing, or even replace it ultimately. It can effectively avoid the limit cycle and convergence problems. One should point out is simply add the Q+ to Q- is not only useless but also makes the convergence effect worse. Besides, only a reward and a Q + are unfeasible, the pole is hard to get in target states, especially with random restart situation. And the convergence is very slow, or even can't converge. Thus, how to build the relationship between Q+ and Q- is worth to study. In this paper, we just add Q+ to Q- in the time of choosing action in order to take comparison of former method, but it obviously impact convergence.

With discussion above, in this paper, we will propose a new method of Q-learning with VNN to solve these problems.

3 Vague Neural Networks

The fuzzy neural networks are achieved by adding a fuzzification layer to a conventional feed forward neural network. The difference between fuzzy neural networks and conventional neural networks lies on how they estimate sampled input-output relationships. They differ in the kind of samples used, how they represent and store those samples, and how they associatively "inference" or map inputs to outputs. Here, VNN generalizes and estimates fuzzy functions with vague set samples, it can handle real inputs as well as fuzzy inputs.

3.1 Vague Set

Let, t_v and, f_v be the truth-membership function, and false-membership function of the Vague set V. $t_v(x)$ is a lower bound on the grade of membership of x derived from the evidence for x, and $f_v(x)$ is a lower bound on the negation of x derived from the evidence against x. $t_v(x)$ and $f_v(x)$ both associate a real number in the interval [0,1] with each x in U, where $t_v(x)+f_v(x)\le 1$. A vague set in U is illustrated in Fig.4

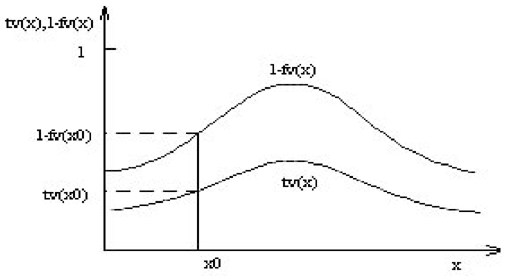


Fig. 4. Vague set

3.2 The Structure of Vague Neuron

In Fig.5, $x_1(t)$, $x_1(f)$, $x_2(t)$, $x_2(f)$, … , $x_n(t)$ $x_n(f)$ express other neural elements' axon output, $x_i(t)$ $x_i(f)$ shows the prompting and restraint of axon. W1 …Wn are the synapse connection between, each artificial neural element need to accord with following equation

$$S_i=\sum_{i=1}^{n}W_i[x_i(t)-\theta_t][x_i(f)-\theta_f] \tag{1}$$

$$\mu_i=g(S_i) \tag{2}$$

$$y_i(t)=t(\mu_i),y_i(f)=f(\mu_i) \tag{3}$$

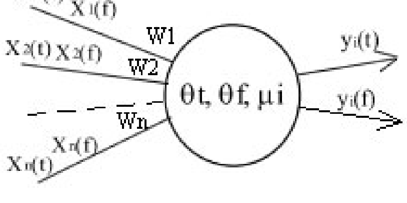


Fig. 5. The structure of vague neuron

The equation express neural element i electric potential summation after synapse, θt, θf are thresholds, in formulas, μi is the state of neural element i, yi is the output of neural element i. $t(\mu_i)$ is a monotonous increase function, while $f(\mu_i)$ monotonous decrease, when μi increase, $y_i(t)$ increase, $y_i(f)$ decrease.

3.3 Vague IF-THEN Control Rules

In VNN, The fuzzy if-then rules are extended to vague if-then rules described as:

IF x_t is $X_t(1)$ and x_f is $X_f(1)$ THEN y_t is $Y_t(1)$ and y_f is $Y_f(1)$;

IF x_t is $X_t(2)$ and x_f is $X_f(2)$ THEN y_t is $Y_t(2)$ and y_f is $Y_f(2)$;

……

IF x_t is $X_t(n)$ and x_f is $X_f(n)$ THEN y_t is $Y_t(n)$ and y_f is $Y_f(n)$.

There are n rules in all. Vague logical is represented as rule relation matrix R:

$$R = \bigcup_{i=1}^{n} [(X_t(i) \times X_f(i)) \rightarrow (Y_t(i) \times Y_f(i))] \tag{4}$$

After structure analysis and approximate process, above-mentioned dual-input and dual-output vague controller could be represented as:

$$Y = \bigcap_{i=1}^{n} Y_i \tag{5}$$

$$Y_k = Y_t(k) \cap Y_f(k) = \left[X_t(k) \times X_f(k) \right] \circ R_k \tag{6}$$

Where $x_t(k)$, $x_f(k)$ are input variables in the kth second, "and" is logic operator"$\wedge$". $X_t(i)$, $X_f(i)$, $Y_t(i)$ and $Y_f(i)$ are linguistic terms characterized by two vague sets on domain of x_i and y respectively.

The character of VNN is to build the vague relationship between input and output. It can have some deferent structure and algorithm in real applications. VNN is generalized form FNN, actually, it can have the same structure as a specified FNN, it can also adopt FNN's learning algorithm for training. For some real applications, FNN can be easily extended to VNN by replacing the single membership function and fuzzy if-then rule by ones of vague set.

4 Reinforcement Learning with VNN

Reinforcement learning is the process by which the response of a system to a stimulus is strengthened by reward and weakened by punishment. Rewards and punishments represent favorable and unfavorable environmental reactions to a response (Fig.6). Such reactions are evaluated by its effort to achieve the goal. A reinforcement learning system seeks its goals by strengthening some responses and suppressing others.

Reinforcement learning control is based on psychological learning theories. A controller receives evaluation of an action based on the definition of a utility function or reinforcement variable U. if the evaluation is positive then the probability associated with that action is increased, and the probabilities associated with all other actions are decreased. Conversely, if the evaluation is negative, then the probability of the given action is decreased and the probabilities associated with all other actions are increased. Loosely speaking, the reward signal positively reinforces those states of the controller that contribute to improvement, while the punishment signal negatively reinforces the states that produced improper behavior. Thus, learning from both of reward and punishment and getting the Q+ and Q- are essential.

The main part of learning agent is VNN which can produce outputs from its inputs immediately. The network is trained to output Q+ and Q-values for the current input and the current internal state. The output units of the network have a linear activation function to produce Q+ values and Q- values of arbitrary magnitude. For the hidden units, the sigmoid function is used.

The action is selected by the equation:

$$action = \max\left(Q(i,a)\right) \tag{7}$$

If $Q(i,left) > Q(i,right)$, push left. Else push right. The action selected is also fed back into the neural network through a one step time delay. The agent learning structure, learning procedure, VNN layout is shown in Fig.6, 7, and 8 respectively.

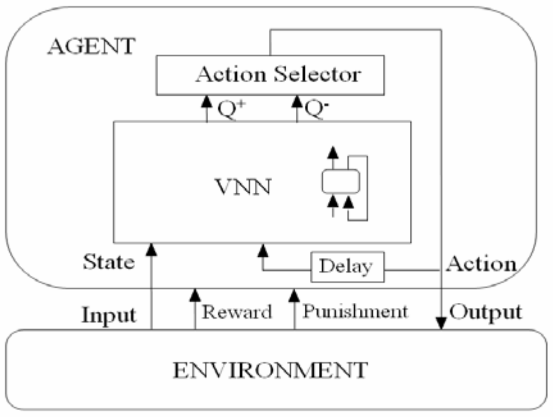


Fig. 6. Structure of the learning agent with VNN

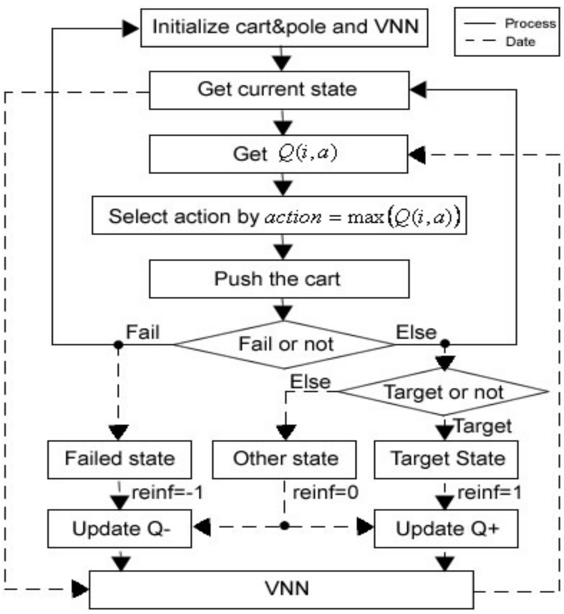


Fig. 7. Learning procedure of VNN control system

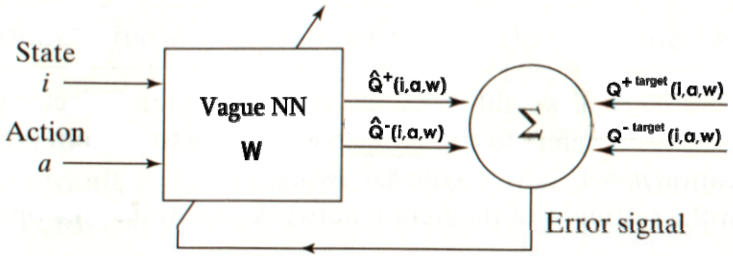


Fig. 8. Neural network layout for approximating the target Q+ and Q- factor

5 Simulation Results

Based on VNN, we utilize the Q-learning control system to simulate the cart-pole balance control problem in MATLAB. In order to illustrate performance of this system, we simply use the parameters in literature [6], and compared with two methods. The running performance of proposed method is distinguished well.

The main deference between two methods is the use of positive Q+ values. We can see distribution of Q values after training in Fig.9. The up state spaces show the state of left pushing action, and the bottom state spaces are related to the right pushing action. The left figure shows the Q- values of former method, and the right one shows both the Q+ and Q- values of proposed method. Obviously, the upward ones are the Q+ values.

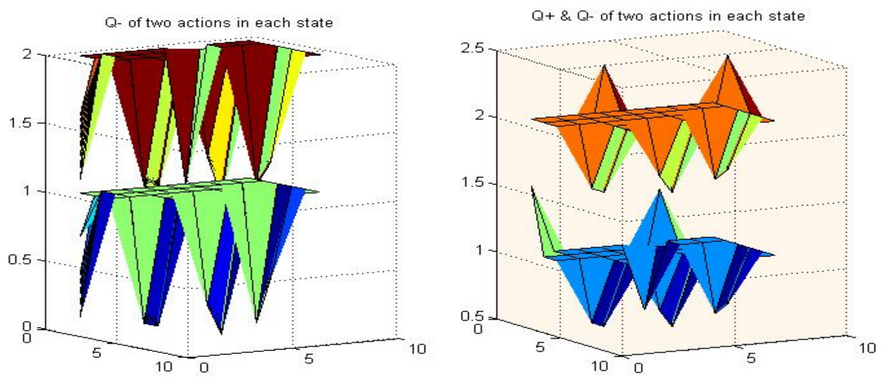


Fig. 9. The distribution of Q values after training in former and proposed method

In Fig.10, we can get the performance of VNN based Q-learning control system compared with the method in the literature and the method using Q+ only. The result of VNN based Q-learning is distinguished well which is almost between 2000 and 10000 successful balance iteration. And the former method still has some good performance of thousands of successful balance iteration, but it is much lesser than the proposed one. With the curve of Q+ values learning, the successful balance iteration is less than 100, which is almost the time of pole from initial state to fall down, so the learning results are bad, without supporting with Q- value, it is hard to get into the target position. But the convergence of proposed method is not so remarkable. It results from simply calculating Q+ and Q- together without consideration of convergence. After considering the effect of Q+ and Q- in different time epochs, it has a better result.

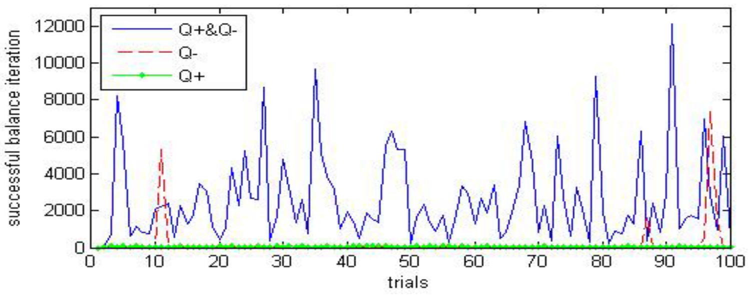


Fig. 10. Success balance iterations of each trial in use of Q+&Q-, Q-, and Q+

Another advantage of VNN based Q-learning is the trajectory which is restricted in a small area, it is clearly illustrated in Fig.11. The trajectory of Pole angle and Pole angle velocity of VNN based method is good, while it is just chosen from 10 trails at the beginning. Pay attention to the scale of the latter figure, it is 10 times lesser than the former method. In this way, we can get the conclusion that receiving reward, getting Q+, and combining it with Q- in VNN is a resultful method.

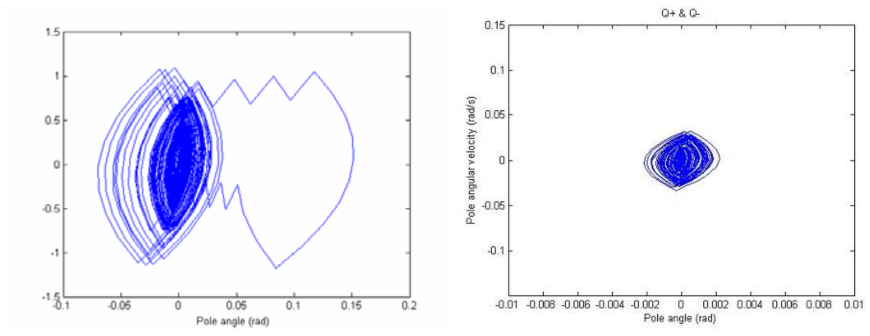


Fig. 11. Trajectory of Pole angle and Pole angle velocity in former and proposed method

6 Conclusions

In this paper, a VNN based control system is implemented to balance a cart-pole system. Based on VNN, we combine two Q values of state-action pairs in Q-learning. A new reinforcement learning algorithm of neural network is proposed. Simulation results of inverted pendulum show that the two output neurons play different roles in reinforcement learning, the combination of them has an excellent effect on the Q-learning result.

Actually, VNN can be considered as a specific FNN, it can be utilized in the FNN's application field by simply replaced the single membership function and the fuzzy if-then rule. We have already tried it in fault diagnosis, pattern recognition and control system field. All the experiment proves that it's correct and feasible. So we can believe it is a promising scheme dealing with ambiguous data in the future.

References

[1] HORIKAWA,S. FURUHASHI,T, and UCHIKAWA, Y: On fuzzy modeling using fuzzy neural network with the back-propagation algorithm, IEEE Trans., 1992, NN-3,(5),pp.801-806;

[2] Gau W L, Buehrer D J. Vague sets [J].IEEE Transactions on Systems,Man,and Cybernetics,1993,23(2):610-614;

[3] Bustince H, Burillo P. Vague sets are intuitionistic fuzzy sets. Fuzzy Sets and Systems 1996, 79(1): 403~405;

[4] Chen,S.M. Fuzzy system reliability analysis based on vague set theory. 1997 IEEE International Conference on Computational Cybernetics and Simulation.1997,2:1650-1655;

[5] Yu Zheng, Siwei Luo, ziang Lv, The negative effect on the control of inverted pendulum caused by the limit cycle in reinforcement learning. ICNN&B '05, pp 772-775;

[6] Ahmet Onat, "Q-learning with recurrent Neural Networks as a Controller for the Inverted Pendulum Problem", The Fifth International Conference on Neural Information Processing, pp 837-840, October 21-23, 1998;

[7] Anderson.C.W, Learning to control an inverted pendulum using neural networks, IEEE Control System Magazine, Vol9, No.3, 1989, pp 31-37.

Neural-Network Inverse Dynamic Online Learning Control on Physical Exoskeleton

Heng Cao[1], Yuhai Yin[1], Ding Du[1], Lizong Lin[1], Wenjin Gu[2], and Zhiyong Yang[2]

[1] East China University of Science and Technology, School of Mechanical and Power Engineering, Postbox: 401. Postcode: 200237. Shanghai, China
hengcao@ecust.edu.cn
[2] Navy Aeronautical Engineering College, Department of Control Engineering. Yantai, Shandong province, China

Abstract. Exoskeleton system which is to assist the motion of physically weak persons such as disabled, injured and elderly persons is discussed in this paper. The proposed exoskeletons are controlled basically based on the electromoyogram (EMG) signals. And a mind model is constructed to identify person's mind for predicting or estimating person's behavior. The proposed mind model is installed in an exoskeleton power assistive system named IAE for walking aid. The neural-network is also be used in this system to help learning. The on-line learning adjustment algorithm based on multi-sensor that are fixed on the robot is designed which makes the locomotion stable and adaptable.

1 Introduction

Recent progress in robotics and mechatronics technology brings a lot of benefits not only in the field of industries, but also in the fields of welfare and medicine. The purpose of this paper is to invent the exoskeletons to help those people with walking disabilities or enhance people's strength, endurance, and speed in many activities. These exoskeletons can assist the motion of physically weak persons such as elderly persons, make user hike further, jump higher, and run faster. Solders will be able to walk fast with heavy weapons.

In order to be useful and accepted by many people who is really need it, these exoskeletons should need some performance characteristics, including comfortable, safe, long life.

The electromyogram (EMG) signals of human muscles are important signal to understand how the patient intends to move. The EMG signals can be used as input information for the control of robotic systems. The proposed exoskeleton system automatically assists the patient's motion for daily activity based on the skin surface EMG signals.

One of the expected tasks of a robot is to assist a person's work. In order that the robot assists a person, it is necessary to estimate what action the person is going to do. The behavior of the person is decided by the mind controlled by emotion whether the person does care or worry about something or not in usual activity. To carry out a cooperative work, the person should take care of a robot because behavior of the robot is altered by his/her mind. We want to install a function of predicting a robot's behavior for cooperating with human behavior. Though the EMG contains lots of important information, but it is still difficult to predict the motion of the patient. Because of many muscles are involved in the motion. And it is difficult to gain the same EMG signals for the same motion from the same patient since the EMG signal is affected by many factors. Further more, the EMG signals are always different among patients. So in order to resolve this problem, fuzzy-neural control has been applied to realize the sophisticated real-time control of the exoskeleton system for motion assist of the patient. The fuzzy-neural controller is supposed to control the joints of the exoskeleton system based on lots of the skin surface EMG signals of the leg muscles. In this paper, we also propose an efficient adaptation evaluation method for the fuzzy-neural controller. Experiment has been performed to evaluate the proposed exoskeleton and its control system.

2 System of Neural-Fuzzy Controller

The proposed exoskeleton system is supposed to be attached to the lateral side of a patient directly. The architecture of the exoskeleton system is shown in Fig.1. The exoskeleton system consists of two kinds of main links (four links are rotation joints and the other four links are pitching joints).

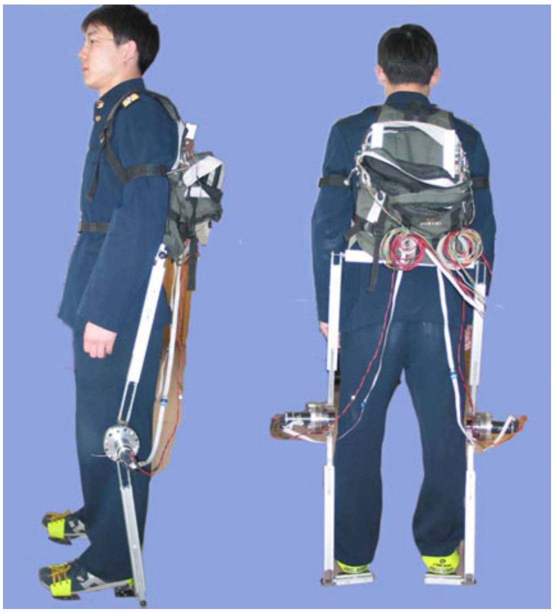


Fig. 1. Architecture of the Exoskeleton System

A fuzzy-neural controller, a combination of a flexible fuzzy controller and an adaptive neural controller, has been applied as a controller for the proposed exoskeleton system in this study. The initial fuzzy IF-THEN control rules are designed based on the analyzed human leg motion patterns in the pre-experiment, and then transferred to the neural network form. The EMG characteristics of human leg muscles studied in another researches [1], [2], are also taken into account.

In the proposed control method, the definition of the antecedent part of the fuzzy IF-THEN control rules for leg motion is adjusted based on the activation level of leg muscles, since the amount of the EMG signals of biceps is affected by leg motion. The effect of the leg posture change is also taken into account in the controller [3], since the leg posture change affects the amount of the EMG signals generated for the joint motion. In the proposed fuzzy-neural controller, there are 16 rules (3patterns) for leg motion, 32rules for leg motion, and 2rules for controller switching between the EMG based control and the ankle force sensor based control. Here $\sum$ means sum of the inputs, $\prod$ means multiplication of the inputs. Two kinds of nonlinear functions are applied to express the membership function of the fuzzy-neural controller.

In order to extract the features from the raw EMG signals, the MAV (Mean Absolute Value) is calculated and used as input signals to the fuzzy-neural controller. The equation of the MAV is written as:

$$MAV = \frac{1}{N}\sum_{k=1}^{N}\left|x_k\right| \tag{1}$$

Where x_k is the voltage value at k^{th} sampling, N is the number of samples in a segment. The number of samples is set to be 100 and the sampling time is set to be 1ms in this study.

The input variables of the fuzzy-neural controller are the MAV of EMG of eleven kinds of muscles, leg angles (vertical and horizontal angles), and force signals from the ankle force sensor. Four kinds of fuzzy linguistic variables (ZO: zero, PS: positive small, PM: positive medium, and PB: positive big) are prepared for the MAV of EMG. Three kinds of fuzzy linguistic variables (EA: Extended, FA: Flexed, and IA: Intermediate angle) for leg angles.

The outputs of the fuzzy-neural controller are the torque command for shoulder motion, and the desired impedance parameters and the desired angle for leg motion of the exoskeleton. The torque command for the leg joint of the exoskeleton is then transferred to the force command for each driving wire. The relation between the torque command for the leg joint of the exoskeleton and the force command for driving wires is written as the following equation:

$$\tau_s = J_s^T f_{sd} \tag{2}$$

Where τs is the torque command vector for the leg joint of the exoskeleton system, f_{sd} is the force command vector for the driving wires, and J_s is the Jacobian which relates the exoskeleton's joint velocity to the driving wire velocity. Force control carried out to realize the desired force (f_{sd}) in driving wires by the driving motors for leg motion of the exoskeleton system.

Impedance control is performed with the derived impedance parameters and the derived desired angle for the knee joint control of the exoskeleton system. The equation of impedance control is written as:

$$\tau_e = M_e(\ddot{q}_d - \ddot{q}) + B_e(\dot{q}_d - \dot{q}) + K_e(q_d - q) \tag{3}$$

Where τe denotes torque command for the knee joint of the exoskeleton system, M_e is the moment of inertia of the leg link and human subject's foreleg, B_e is the viscous coefficient generated by the fuzzy-neural controller, K_e is the spring coefficient generated by the fuzzy-neural controller, K_e is the spring coefficient generated by the fuzzy-neural controller, q_d is the desired joint angle generated by the fuzzy-neural controller, and q is the measured knee joint angle of the torque command for the driving motor for the knee motion of the exoskeleton system. There is also a mind model installed in the exoskeleton power assistive system named IAE (Intelligent Assistive Exoskeleton) for walking aid. The Environment Disturbance Index (ED) is defined as a ratio of the feasible velocity to the desired walking velocity. Then the ED ranges from normalized value 0.0 for most hazardous road, to value 1.0 for the smoothest road. And ED is:

$$ED = b - a \cdot \tan(\theta) \tag{4}$$

Where θ is the angle of hip joint for every step, a is a coefficient and b is an offset which is chosen so that it may be set to ED equals 1 at the time of flat road.

IAE needs to identify operator's mind to assist operator cooperatively, and it is realized through identifying operator's Emotion Index. In this section, we describe a method of identifying Emotion Index (*M*) and estimating Desired Input (*d*) of third layer of the mind model simultaneously using the information from the sensors of IAE when the operator walks along a path which consist of a flat passage, gentle stairs, and steep stairs.

Operator's mind model installed in IAE for identification and estimation is written as follows.

$$\hat{v} = \hat{M} \cdot \hat{d} + (1 - \hat{M})d \cdot ED \tag{5}$$

Where $\hat{M}$ is the estimated Emotion Index, and $\hat{d}$ is the estimated Desired Input, and $\hat{v}$ is the estimated walking velocity. Walking velocity, which is a reference input for identification and estimation can be obtained from a step width (*S*) and a time-interval (*T*) spent on every step as depicted in preceding chapter. An error e between the calculated walking velocity and the estimated one by operator's model is written as follows:

$$e = v - \hat{v} \tag{6}$$

Adjustment of $\hat{M}$ for identification is written as equation using steepest descend method.

$$\Delta\hat{M} = -k_m \frac{\partial e^2}{\partial \hat{M}} = k_m \cdot 2 \cdot e \cdot (1 - RC)\hat{d} \tag{7}$$

Likewise, correction of $\hat{d}$ is written as:

$$\Delta\hat{d} = -k_d \frac{\partial e^2}{\partial \hat{d}} = k_d \cdot 2 \cdot e \cdot \hat{M} \tag{8}$$

Since the gain k_m for identification and the gain k_d for estimation are defined as positive, M and d are estimated so that error converges in zero.

This method for identification and estimation is illustrated in Fig.2.

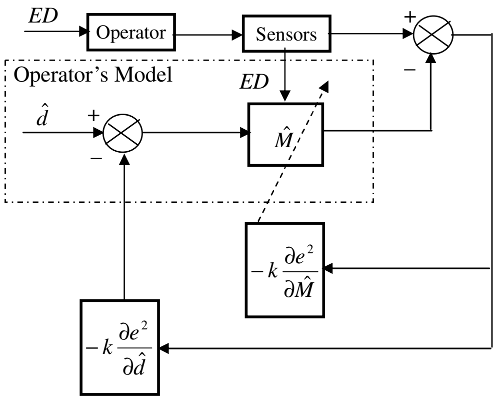

ED
Operator
Sensors
+
−
Operator's Model
ED
$\hat{d}$
+
−
$\hat{M}$
$-k\frac{\partial e^2}{\partial \hat{M}}$
$-k\frac{\partial e^2}{\partial \hat{d}}$

Fig. 2. Architecture of Identifying Emotion Index and estimating Desired Input

3 Control of Neural-Network Inverse Dynamic Offline Learning

Narrow inverse learning is used widely. It not only can determine the sample concourse easily, but also can construct a special inverse model. The schematic of inverse learning is depicted as Fig.3. Narrow inverse learning frame is usually used BP algorithm; the effect of Neural Network model is affording an error Propagation communication. The convergence of BP algorithm is slow. Compare to BP algorithm, Recursive Least Square (RLS) don't need to guess the learning rate and coefficients, it converge fast and has high precision. But the problem of algorithm stability exists. If the RLS algorithm inverse learning project based on U-D disassemble, then the algorithm stability is ensured and the efficiency is increased.

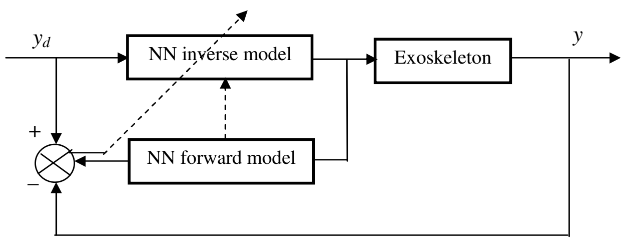


Fig. 3. Specialized Inverse Learning on Neural Network

In the realization of algorithm, the inverse dynamic model of single hidden layer forward network approaching robot is used. Taking exoskeleton joint position signal

q, velocity signal $\dot{q}$ and accelerate signal $\ddot{q}$ as network input signal, the network output joint constrained control U. Form the viewpoint of system identification, in order to learn system inverse dynamic better, the inspire system input signal should include ample frequency response and converge network quickly.

1. Initialize weigh value W(0), U(0) and D(0), selecting the random value in [-1, 1], and determining study rate η=0.6 and error range ε=0.005;
2. Collecting robot input signal X(t) = τ(t), input signal d_p(t)=[q(t) $\dot{q}$ (t) $\ddot{q}$ (t)] as sample;
3. Input(i =1, 2) of computing network: $z_1^{(i)} = X^T W_1^{(i)}$,

$$y_1^{(1)} = f(z_1^{(1)}) = \frac{1-e^{-z_1^{(1)}}}{1+e^{-z_1^{(1)}}}, z_1^{(2)} = (y_1^{(1)})^T W_1^{(2)}, y_1^{(2)} = f(z_1^{(2)}) = \frac{1-e^{-z_1^{(2)}}}{1+e^{-z_1^{(2)}}}$$

4. Obtain F(t), g(t) and β (t) by G(t)=U(t)D(t)U^T(t), F(t)=U^T(t-1)X(t), g(t)=D(t-1)F(t), β (t)= λ +F^T(t)g(t);
5. Computing Kalman filter gain equation of all layers: $K^t(t) = \dfrac{U(t-1)g(t)}{\beta_p(t)}$;
6. computing signal of back propagation: the output layer: $\delta^{(2)} = (d_p - y_p^{(2)})$, the hidden layer: $\delta^{(1)} = f'(z_1^{(1)})\sum_j \delta_j^{(2)} W_j^{(2)}, j = 1,\dots,p$, $f'(z^{(1)}) = (1/2)(1-f^2(z^{(1)}))$;
7. Correction weigh value: the output layer: $W_t^{(2)}(t) = W_t^{(2)}(t-1) + K_t^{(2)}(t)(d_t - y_i^{(2)})$, the other layer: $W_t^{(t)}(t) = W_t^{(1)}(t-1) + \eta K_t^{(1)}(t)\delta_i^{(1)}(t)$;
8. Computing $\hat{d}_t(t)$ and $v_y(t)$ from

$$\hat{d}_t(t+1) = \frac{\beta_{i-1}}{\lambda\beta_i}\bar{d}_t(t), \qquad 1 \le i \le p,$$

$$v_{1i}(t) = u_{il}(t-1)g_i(t), \qquad 1 \le i = j \le p,$$

$$v_{1j}(t) = u_{i(j-1)}(t) + u_{1j}(t-1)g_i(t-1), \qquad 1 \le i < j \le p,$$

$$u_{1j}(t) = u_{ij}(t-1) - \frac{f_j(t)}{\beta_{j-1}(t)} v_{1(j-1)}(t), \qquad 1 \le i < j \le p,$$

9. If $E = \frac{1}{2}\sum_{P=1}^{M}(d_p - y^L)^2 > \varepsilon$, turn to step 2, otherwise stop learning

4 Control of Neural-Network Inverse Dynamic Online Learning and Simulation Comparison

There is a limitation of offline inverse dynamic control, because when robot is interrupted by environment, the exoskeleton need to change walking track to ensure system stability and continuation. The offline training scheme do not study online, so the change of dynamic property can't reflect correctly. It is necessary to introduce an online training method, then the response of neural network nonlinear compensation control to inverse dynamic property can be ensured, and the Robust stability and control precision is also ensured.

In order to ensure network can study while system running, a proper online training algorithm need to choose. The RLS algorithm based on U-D disassemble can realize online recursive, and ensure algorithm stability and algorithm efficiency. In the actual control system, the exoskeleton can be recognized as slow time-changing system. then the online learning progress can be placed outside servo-loop, that is learning can choose slow sampling frequency, which can be realized by coordinated management. Thus, the sampling frequency of servo-system can be increased, the whole control system can realize multi-velocity sampling configuration, the weight of servo-loop computing can be reduced, and the performance of system is ensured.

The step of RLS algorithm based on U-D disassemble is shown as follow:

The difference between online and offline learning is that, the former can study while control, but the latter can't study while learning. Thus, when object parameter change or system disturbed, the learning configuration can adjust network weigh value timely and achieve better control effect. The offline and online learning result curve are shown as Fig.4.and Fig.5.while the operator weight increase 8%. We can find that the online learning can reflect the change of inverse dynamic characteristic timely. So the control error is small. The simulation experiment proved that online learning control has high Robustness and better effect.

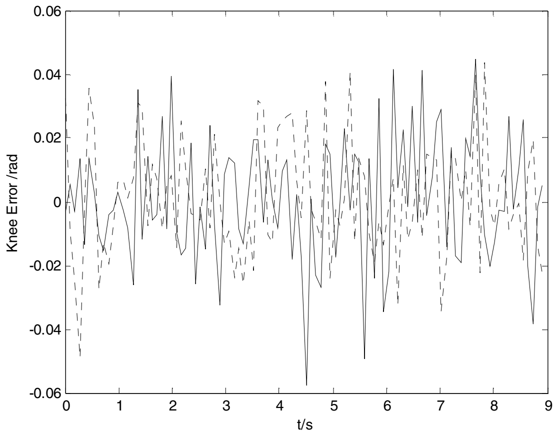


Fig. 4. Error Response Curve of Knee Joint by Offline Learning. Dashed—normal; Real line--add by 8%.

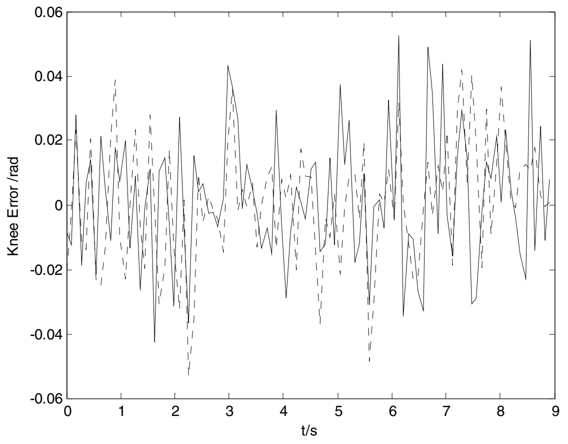


Fig. 5. Error Response Curve of Knee Joint by Online Learning. Dashed—normal; Real line--add by 8%.

5 Conclusion

Exoskeletons that enhance human strength, endurance, and speed are feasible and will someday be popular. In this paper, a fundamental mind model is developed for a robot to carry out a cooperative work with a person by identifying person's mind condition which affects his behavior. The assist level of the system can be adjusted based on his/her physical and physiological condition. The robot is a complex non-linear system which utilizes neural-network for both offline and online learning. On-line learning based on the inverse dynamics that are learned off-line, is proposed in this paper. Simulation demonstrated that the tracking precision is higher, adaptive ability is better, and the locomotion is more stable and reliable.

References

1. M.Ohta and A.Sato.:On a Robot Which Walks with a Person by Keeping a Mind Contact. Conf.ROBOTIC2000, Berlin, Germany. (2000) 225-238
2. Blaya,J.:Force Controlleble Ankel Foot Orthosis to Assist Drop-foot Gait. Mech.Eng.MS Thesis, MIT.(2002)
3. Brown, P.,D.Jones, S.K.Singh, and J. M. Rosen.:The Exoskeleton Glove for Control of Paralyzed Hands. Proceedings IEEE International Conference on Robotics and Automation. （1993） 642-647.
4. W.J.Lee, C.S.Ouyang, S.J.Lee.: Contructing Neuro-Fuzzy Systems with TSK Fuzzy Rules and Hybrid SVD-Based Learning. Poc.of IEEE International Conf. on Fuzzy Systems.(2002)
5. B.Hudgins, P.Parker, and R.N.Scott.: A New Strategy for Multifunction Myoeletric Control. IEEE Trans. On Biomedical Engineering, vol.40, no.l. (1993)82-94,
6. Y.Diao,K,M.Passino.: Adaptive Neuro-Fuzzy Control for Interpolated Nonlinear Systems. IEEE Trans. On Fuzzy Systems, vol.10, no.5. (2002)583-595

7. A.Ishii, A.TAkanishi.: Emotion Expression of a Biped Personal Robot, IEEE International Conference on Intelligent Robots and Systems, Takamatsu.(2000)191-196
8. A.Takanishi, H.Takanobu.: Robot Personalization Based on the Mental Dynamics. IEEE/RSJ International Conference on Intelligent Robots and Systems, Takamatsu.(2000)8-14
9. A.Kun, W.Miller.:Adaptive Dynamic Balance of a Biped Robot using Neural NetWorks. IEEE Int. Conf. on Robotic and Automation, Minneapolis.(1966)240-245
10. Kawamoto, H. and Y. Sankai.: EMG-based Hybrid Assistive Leg for Walking Aid Using Feedforward Controller. ICCAS 2002. International Conference on Control, Automation and Systems.(2002)190-193
11. Canales, L. and M.M. Stanisic.: Prelliminary Design of an Exoskeleton Shoulder Joint without Dead Positions. IEEE International Conference on Systems Engineering .(1998)94-96
12. Kosso, E.V.: A minimum Energy Exoskeleton. Camahan Conference on Electronic Prosthetics.(1973)86-89

Fuzzy Adaptive Particle Filter Algorithm for Mobile Robot Fault Diagnosis

Zhuohua Duan [1,2], Zixing Cai[1], and Jinxia Yu[1,3]

[1] School of Information Engineering, Shaoguan University, 512003,Shaoguan, Guangdong, China
duanzhuohua@163.com
[2] College of Information Science and Engineering, Central South University, 410083,Changsha, Hunan, China
[3] Department of Computer Science & Technology, Henan Polytechnic University, 454003, Jiaozuo, Henan , China

Abstract. A fuzzy adaptive particle filter for fault diagnosis of dead-reckoning sensors of wheeled mobile robots was presented. The key idea was to constrain sampling space to a fuzzy subset of discrete fault space according to domain knowledge. Domain knowledge was employed to describe 5 kinds of planar movement modes of wheeled mobile robots. The uncertainties of domain knowledge (due to imprecise driving system and inaccurate locomotion system) were represented with fuzzy sets. Five subjection functions were defined and aggregated to determine discrete transitional probability. Two typical advantages of this method are: (1) most particles will be drawn from the most hopeful area of the state space; (2) logical inference abilities can be integrated into particle filter by domain constraints. The method is testified in the problem of fault diagnosis for wheeled mobile robots.

1 Introduction

Fault detection and diagnosis (FDD) is increasingly important for wheeled mobile robots (WMRs), especially those under unknown environments such as planetary exploration [1]. Particle filter (also known as sequential Monte Carlo) is a promising approach for mobile robot fault diagnosis, and has received many attentions [2-7]. The merits of particle filters includes the abilities of handling nonlinear non-Gaussian state space, estimating discrete and continuous states simultaneously, and adjusting the computational complexity by the number of particles.

Two challengeable problems for fruitfully using particle filters are: (1) the degeneracy problem, i.e. after a few iterations, all but one particle will have negligible weight; (2) the problem of sample impoverishment, i.e. many particles will collapse to a few particles [8]. The most promising approach to handle these problems of general particle filter is adaptive particle filter, such as KLD-sampling method proposed by [9], and the Rao-Blackwellised particle filter (RBPF) proposed by [10]. The key idea of the KLD-sampling method is to bound the approximation error by adapting the size of sample sets by the Kullback-Leibler distance. RBPF decreases the size of sample sets dramatically by only sampling the discrete states, and the trade-off is that each discrete state of operation is described with one different linear-Gaussian state space model.

Researchers have already noticed that reducing the dimensionality of the state space is a feasible approach to approximate the true distribution accurately and efficiently. In this paper, a fuzzy adaptive particle filter, which adjusts state space according domain knowledge, is presented. The key idea is that most particles are drawn from the most hopeful regions of the state space. Firstly, a general framework for particle filter, which integrates domain knowledge, is put forward. The, domain knowledge is exploited to constrain sampling space to a fuzzy subset of the 'universal' state space. In mobile robot diagnosis problem, domain knowledge is used to describe movement modes of the robot, including at rest, straight-line movement and rotation etc. The uncertainties of domain knowledge (due to imprecise driving system and inaccurate locomotion system) were represented with fuzzy sets. Five subjection functions were defined and aggregated to determine discrete transitional probability. Two typical advantages of this method are: (1) most particles will be drawn from the most hopeful area of the state space; (2) logical inference abilities can be integrated into particle filter by domain constraints.

2 Particle Filter Based Fault Diagnosis

Particle filter is a Monte Carlo (i.e. choosing randomly) method to monitor dynamic systems, which non-parametrically approximates probabilistic distribution using weighted samples (i.e. particles). PF gives a computationally feasible method for state estimation of hybrid systems. Furthermore, a single particle filter can represent discrete and continuous states simultaneously and can represent any distribution (including non-Gaussian).

The main idea for using PF as a fault diagnosis method is described as following.

Let $\mathbb{S}$ represent the finite set of discrete fault and operational modes of the system, $s_t \in \mathbb{S}$ the state of the system to be diagnosed at time t and $\{s_t\}$ the discrete, first order Markov chain representing the evolution of the state over time. The problem of state monitoring and fault diagnosing of the system consists of providing a belief (a distribution over the state set $\mathbb{S}$) at each time step as it evolves based on the following transition model:

$$p(s_t=j|s_{t-1}=i), i,j \in \mathbb{S} \tag{1}$$

Each of the discrete fault and operational modes changes the dynamics of the system. Let $\mathbf{x}_t$ denote multivariate continuous state of the system at time t. The non-linear conditional state transition models are denoted by $p(\mathbf{x}_t|\mathbf{x}_{t-1},s_t)$. The state of the system is observed through a sequence of measurements, $\{\mathbf{z}_t\}$, based on the measurement model $p(\mathbf{z}_t|\mathbf{x}_t,s_t)$.

The problem of state monitoring and fault diagnosing consists of estimating the marginal distribution $p(s_t|\mathbf{z}_{1..t})$ of the posterior distribution $p(\mathbf{x}_t,s_t|\mathbf{z}_{1..t})$. A recursive estimate of this posterior distribution may be obtained using the Bayes filter:

$$p(s_t,\mathbf{x}_t|\mathbf{z}_{1..t})=\eta_t p(\mathbf{z}_t|s_t,\mathbf{x}_t)\int\sum_{s_{t-1}} p(s_{t-1},\mathbf{x}_{t-1}|\mathbf{z}_{1..t-1})p(s_t,\mathbf{x}_t|s_{t-1},\mathbf{x}_{t-1})d\mathbf{x}_{t-1} \tag{2}$$

There is no closed form solution to this recursion. PFs appropriate the posterior with a set of *N* fully instantiated state samples or particles $\{(s_t^{[1]}, \boldsymbol{x}_t^{[1]}),\ldots,(s_t^{[N]}, x_t^{[N]})\}$ and importance weights $\{w_t^{[i]}\}$:

$$\hat{P}_N(\boldsymbol{x}_t, s_t \mid \boldsymbol{z}_{1..t}) = \sum_{i=1}^{N} w_t^{[i]} \delta_{\boldsymbol{x}_t^{[i]}, s_t^{[i]}}(\boldsymbol{x}_t, s_t) \tag{3}$$

where δ(.) denotes the Dirac delta function. The appropriation in equation (3) approaches the true posterior density as N→∞. Because it is difficult to draw samples from the true posterior, samples are drawn from a more tractable distribution *q(.)*, called the proposal (or importance) distribution. The importance weights are used to account for the discrepancy between the proposal distribution *q(.)* and the true distribution $p(x_t,s_t|z_{1..t})$. The importance weight of sample $(s_t^{[i]}, x_t^{[i]})$ is

$$w_t^{[i]} = p(x_t^{[i]}, s_t^{[i]} \mid x_{t-1}^{[i]}, s_{t-1}^{[i]}, z_t) / q(x_t^{[i]}, s_t^{[i]} \mid x_{t-1}^{[i]}, s_{t-1}^{[i]}, z_t) \tag{4}$$

The general particle filter algorithm is expressed as following.

```
1) Initialization:
   for $i$=1,…,N, sample $s_0^i \sim p(s_0)$, $\boldsymbol{x}_0^i \sim p(\boldsymbol{x}_0)$, $w_0^i$=1/N;
2) Sequential Importance Sampling:
   for $i$=1,…,N,
      $s_t^i \sim p(s_t \mid s_{t-1}^i)$ ;sampling discrete states
      $\boldsymbol{x}_t^i \sim p(\boldsymbol{x}_t \mid s_t^i, \boldsymbol{x}_{t-1}^i)$; sampling continuous state;
      $w_t^i \sim p(z_t \mid s_t^i, \boldsymbol{x}_t^i)$; computing weights;
   end
   $\overline{w}_t^i = \frac{w_t^i}{\sum_{j=1}^{N} w_t^i}$ ; normalization;
3) Resampling: Generate a new set $\{\tilde{\boldsymbol{x}}_t^i, \tilde{s}_t^i\}_{i=1}^N$ from $\{\boldsymbol{x}_t^i, s_t^i\}_{i=1}^N$ such that
   Pr(( $\tilde{\boldsymbol{x}}_t^i, \tilde{s}_t^i$ ) =( $\boldsymbol{x}_t^j, s_t^j$ ))= $\overline{w}_k^j$ .
   $w_t^i$ =1/N;
```

3 Kinematics Models and Fault Modes

3.1 Kinematics Models

The velocity kinematics model of mobile robots is shown in (5),

$$\begin{cases} v = (v_L + v_R)/2 \\ \dot{\varphi} = (v_R - v_L)/D \end{cases} \tag{5}$$

where v denotes linear speed (mm/s), $\dot{\varphi}$ denote yaw rate (rad/s). v_L, v_R denote linear speed of left and right wheels respectively. D denotes the axis length. The state vector is $\mathbf{x}=(v, \dot{\varphi})^T$.

3.2 Measurement Models

MORCS-1(1st MObile Robot of Central South University) is a 6-wheels/5-wheels mobile robot, which is capable of moving in complex 3D terrain [13]. The four front wheels are driving wheels and each front wheel is equipped with a step motor and a wheel encoder. The rotational speeds of wheels are measured with 4 encoders. The yaw rate is measured with a gyroscope. The measurement vector is $\mathbf{z}=(z_{LF}, z_{LR}, z_{RF}, z_{RR}, z_G)^T$, where z_{LF}, z_{LR}, z_{RF}, z_{RR}, z_G denote measurements of left front, left rear, right front, right rear encoder and gyroscope respectively.

The measurement model of MORCS-1 is shown in (6),

$$\mathbf{z}_t=\mathbf{h}_t(\mathbf{x}_t)+\mathbf{v}_t \tag{6}$$

where $\mathbf{z}_t$ denotes measurements at *t*. $\mathbf{v}_t$ denotes measurement noises at *t*, which is assumed to be zero mean white Gaussian sequences with the covariance matrix **R**.

In normal state, $\mathbf{h}=((v-\dot{\varphi}.D/2)/r, (v-\dot{\varphi}.D/2)/r, (v+\dot{\varphi}.D/2)/r, (v+\dot{\varphi}.D/2)/r, \dot{\varphi})^T$.

3.3 Fault Models

Each sensor can be normal or fault. A system of 5 sensors has $2^5=32$ kinds of modes: including normal mode and 31 kinds of fault modes. All these modes form the discrete state space, $S=\{ i \mid 1 \leq i \leq 32 \}$, in which '1' denotes normal state, '2' denotes the left front wheel encoder fails, '3' denotes the left rear wheel encoder fails, …, '31' denotes all wheel encoders fail (only the gyroscope is ok), and '32' denotes all sensors fail.

Let h^i denote the measurement model of state 'i'. For example,

$\mathbf{h}^2=[0,(v-\dot{\varphi}.D/2)/r,(v+\dot{\varphi}.D/2)/r,(v+\dot{\varphi}.D/2)/r,\dot{\varphi}]^T$;

$\mathbf{h}^6=[(v-\dot{\varphi}.D/2)/r,(v-\dot{\varphi}.D/2)/r,(v+\dot{\varphi}.D/2)/r,(v+\dot{\varphi}.D/2)/r,0]^T$.

4 Domain Constraints and Representation

4.1 Movement Modes of Mobile Robots

The movement modes are determined by the driving speeds of left and right sides of the robot. Let u_L and u_R denote the driving speeds of left and right wheels respectively. The constraints of the five movement modes are:

(a) At Rest (M_1), $u_L= u_R=0$;
(b) Straight Line (M_2), $u_L= u_R \neq 0$;
(c) Rotation 1(M_3), $u_L=0$, $u_R \neq 0$;
(d) Rotation 2(M_4), $u_L \neq 0$, $u_R=0$;
(e) Rotation 3 (M_5), $u_L \neq 0$, $u_R \neq 0$, $u_L \neq u_R$;

These movement modes determine 5 sets. To handle the errors of the driving system and the locomotion system, fuzzy sets are used to represent the 5 movement modes. Let M_1, M_2, M_3, M_4, M_5 denote the fuzzy sets of corresponding movement modes. The subjection functions are:

$$\mu_{M_1}(u_L,u_R)=1/(1+u_L^2+u_R^2) \tag{7a}$$

$$\mu_{M_2}(u_L,u_R)=1-(u_L-u_R)^2/[(u_L-u_R)^2+1] \tag{7b}$$

$$\mu_{M_3}(u_L,u_R)=1000u_R^2/[(1+u_L^2)(1+1000{u_R}^2)] \tag{7c}$$

$$\mu_{M_4}(u_L,u_R)=1000u_L^2/[(1+u_R^2)(1+1000u_L^2)] \tag{7d}$$

$$\mu_{M_5}(u_L,u_R)=1000u_R^2u_L^2(u_R-u_L)^2/[(1+10u_R^2)(1+10u_L^2)(1+10(u_R-u_L)^2)] \tag{7e}$$

4.2 Domain Constraints

Each movement mode defines a set of detectable fault modes (a subset of $\mathbb{S}$), as shown in Table I.

Table 1. Movement Modes and Detectable Fault Sets

Conditions	Movement Modes	Fault Subsets
$u_L=u_R=0$	At Rest (M_1)	$\mathbb{S}_1$
$u_L=u_R\neq0$	Straight Line (M_2)	$\mathbb{S}_2$
$u_L=0$, $u_R\neq0$	Rotation 1 (M_3)	$\mathbb{S}_3$
$u_L\neq0$, $u_R=0$	Rotation 2 (M_4)	$\mathbb{S}_4$
$u_L\neq u_R$, $u_L\neq0$, $u_R\neq0$	Rotation 3 (M_5)	$\mathbb{S}_5$

where $\mathbb{S}_1$={1}, $\mathbb{S}_2$= {1, 2, 3, 4, 5, 7, 8, 9, 11, 12, 14, 17, 18, 20, 23, 27}; $\mathbb{S}_3$={1, 4, 5, 6, 14, 15, 16, 26}; $\mathbb{S}_4$={1, 2, 3, 6, 7, 10, 13, 19}; $\mathbb{S}_5=\mathbb{S}$. The characteristic functions are $C_k:\mathbb{S}\to\{0, 1\}$ (k=1,2,3,4,5),

$$C_k(i)=\begin{cases}1, & i\in\mathbb{S}_k\\ 0, & otherwise\end{cases} \quad \text{(k=1,2,3,4,5)} \tag{8}$$

4.3 Conditional Probabilities Based on Fuzzy Aggregation

Integrating (7) and (8), the following aggregation operator is used to calculate $p_a(s|u_L,u_R)$. $p_a(s|u_L,u_R)$ denotes the detectability of fault mode 's' when the left and right driving speeds are u_L,u_R.

$$p_a(s|u_L,u_R)=\eta\sum_{k=1}^{5}C_{\mathbb{S}_k}(s)\mu_{M_k}(u_L,u_R) \tag{9}$$

$\eta=(\sum_{s\in\mathbb{S}}\sum_{k=1}^{5}C_{S_k}(s)\mu_{M_k}(u_L,u_R))^{-1}$, is normalization factor.

5 Fuzzy Adaptive Particle Filter

5.1 Transition Probability $p_b(s_{t+1}=j|s_t=i)$

Generally, the probability of multiple components fail simultaneously is larger the single component fails. For the fault diagnosis problem mentioned in Section II, 6 kinds of transitional probability are, satisfying that,

$$\sum_{k=0}^{5} C_5^k p_k = 1\text{, } p_0 \geq p_1 \geq p_2 \geq p_3 \geq p_4 \geq p_5 \tag{10}$$

where k denotes the number of fault components.

For example, p_0, p_1, p_2, p_3, p_4 and p_5 are set as 0.0425, 0.0380, 0.0335, 0.0290, 0.0245 and 0.02. $p_b(s_{t+1}=1| s_t=1) = p_0 = 0.0425$, $p_b(s_{t+1}=32| s_t=1) = p_5=0.02$.

5.2 Conditional Transition Probability $p(s_{t+1}=j|s_t=i, u_L, u_R)$

The conditional transition probability, $p(s_{t+1}=j|s_t=i, u_L, u_R)$, can be derived directly according to $p_a(s|u_L,u_R)$ and $p_b(s_j| s_i)$ as follows,

$$p(s_{t+1}=j|s_t=i, u_L, u_R) = \mu_3\ p_a(s=j\ |u_L,u_R)p_b(s_{t+1}=j|s_t=i) \tag{11}$$

where, $\mu_3 = (\sum_{s_i \in S} p_a(s_i)p_b(s_j | s_i))^{-1}$ is normalization factor.

Fig. 1 shows several typical conditional transition probability distributions. These distributions take into account the information of domain knowledge and provide more information with respect to the uniform distribution.

5.3 Fuzzy Adaptive Particle Filter

The fuzzy adaptive particle filter is shown as follows.

```
Algorithm Fuzzy Adaptive Particle Filter (FAPF)
1.  for i=1,…,N, sample $s_0^i \sim p(s_0)$, $x_0^i \sim p(x_0)$, $w_0^i$=1/N;
2.  for each time step t do
3.  compute conditional transition probability, $p(s_t| s_{t-1},u_L,u_R)$, according (11)
4.    for i=1 to N
5.      $s_t^i \sim p(s_t | s_{t-1}^i, u_L, u_R)$;
6.      $x_t^i \sim p(x_t | s_t^i, x_{t-1}^i)$;
7.      $w_t^i \sim p(z_t | s_t^i, x_t^i)$;
8.  end for
9.  State estimation: $\hat{s}_t = \arg\max_{s_t} \hat{P}_N(s_t | z_{1..t})$
10. $\bar{w}_t^i = \frac{w_t^i}{\sum_{j=1}^N w_t^i}$;
11. Resampling;
12. end for
```

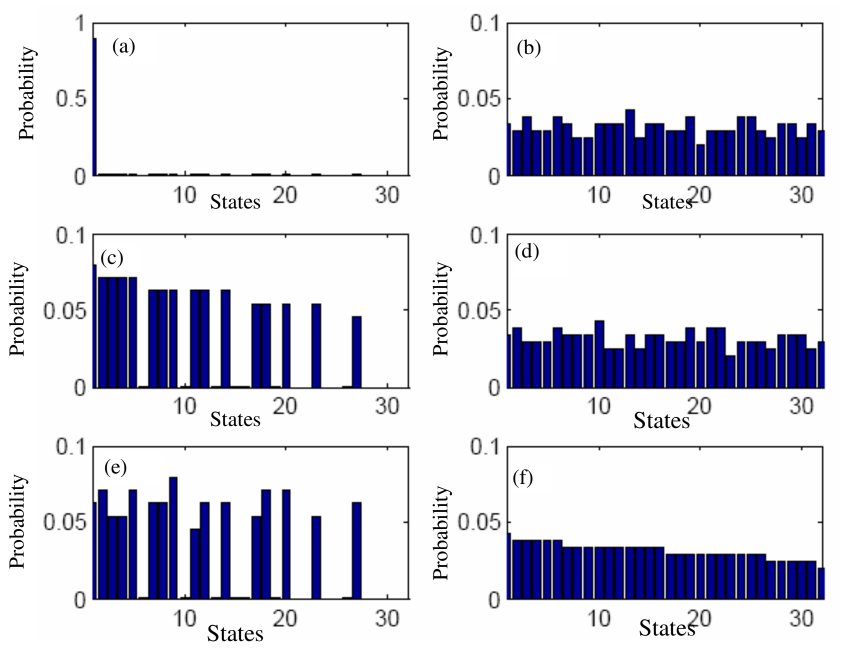


Fig. 1. Several typical conditional discrete transition probability $p(s_{t+1}|s_t, u_L, u_R)$ (a)$p(s_{t+1} | s_t=1, u_L=0, u_R=0)$; (b)$p(s_{t+1} | s_t =13, u_L=9.9mm/s, u_R=-9.9mm/s)$; (c)$p(s_{t+1} | s_t=1, u_L=9.9mm/s, u_R=9.9mm/s)$; (d)$p(s_{t+1} | s_t=10, u_L=9.9mm/s, u_R=-9.9mm/s)$; (e)$p(s_{t+1} | s_t =9, u_L=9.9mm/s, u_R=9.9mm/s)$; (f)$p(s_{t+1} | s_t =1, u_L=9.9mm/s, u_R=-9.9mm/s)$;

6 Experiment Analysis

The driving speed of left and right side are shown in Fig. 2(a) and (b). The movement state of the robot is shown in Fig. 2(c). Measurements of sensors are shown in Fig. 2(d)-(h). When the robot is rotating during the period of 7.65 -13.65s, 5 kinds of faults (‘2’, ‘7’, ‘17’, ‘27’, ‘32’) occurred in sequence. When the robot is moving straight forward during the period of 14.7-37.3s, 4 kind of fault (‘2’, ‘7’, ‘17’, ‘27’) took place. Two kinds of methods, general particle filter (GPF for short) and fuzzy adaptive particle filter (FAPF for short) are employed to diagnose these faults. Two typical estimation results of FAPF and GPF are shown in Fig. 3. It shows that FAPF is much more accurate than GPF. It is noticed that when the robot is moving straight line, GPF classifies the mode of gyroscope as ‘normal’ and ‘fault’ stochastically. This is the main source of misdiagnoses in GPF.

The efficiency and accuracy of both methods are compared and reported in Table II. The ‘error rate’ denotes the percentage of mismatches between the estimation of particle filters and the true states. Two methods, FAPF and GPF, are tested with different particle numbers. Table II shows that FAPF is superior to GPF in accuracy, and has almost the same efficiency with GPF.

Table 2. Efficiency and accuracy of FAPF and GPF

	GPF		FAPF	
Particle Number	Total Time(s)	Error rate(%)	Total Time(s)	Error rate (%)
60	16.0	27.5	18.8	4.50
80	19.5	26.4	24.1	3.43
100	22.7	26.0	28.7	3.00
120	25.7	24.6	33.9	3.00
140	29.1	24.0	39.7	3.00
160	32.6	23.1	43.8	2.78

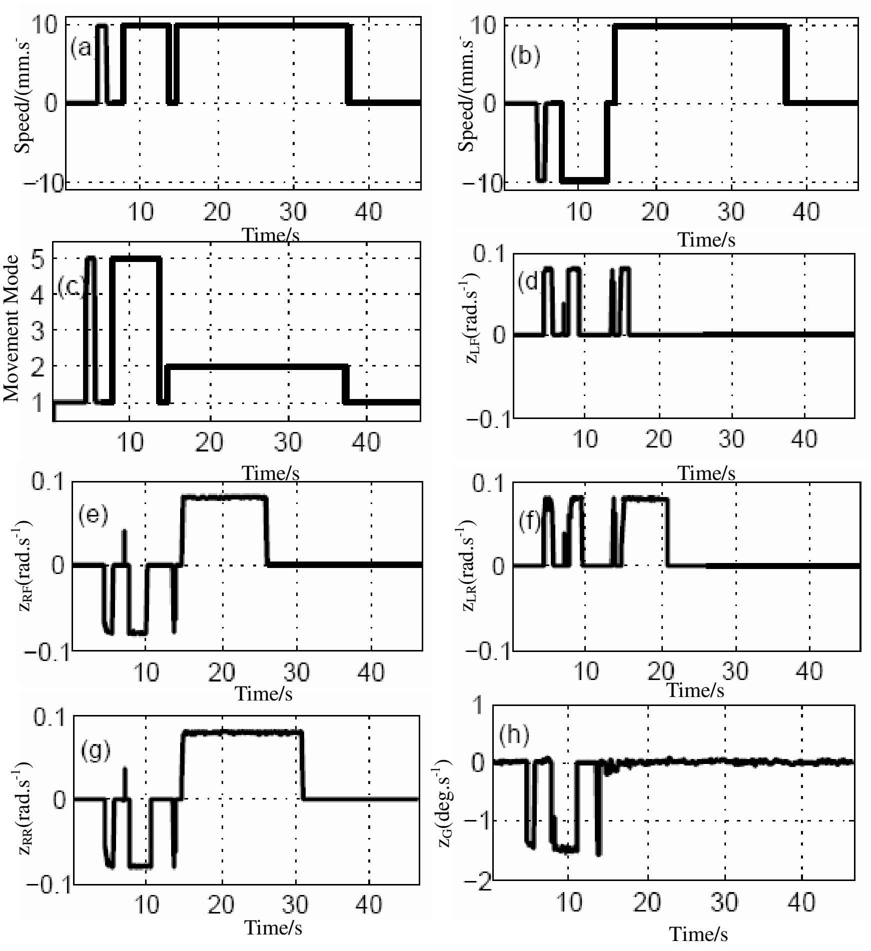


Fig. 2. Driving speeds and measurements (a)Left driving speed; (b)Right driving speed; (c)Movement modes (d)Left front encoder measurement (e)Right front encoder measurement;(f) Left rear encoder measurement (g)Right rear encoder measurement (h)Gyroscope measurement

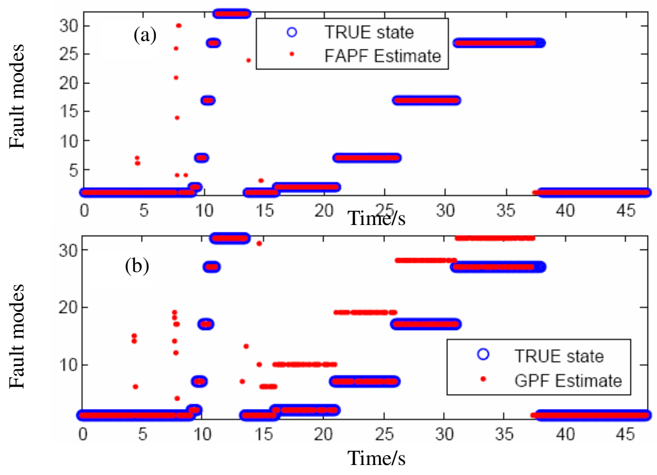


Fig. 3. State estimation of fuzzy adaptive particle filter (FAPF) and general particle filter (GPF) (a) FAPF estimation vs. true states; (b) GPF estimation vs. true states;

7 Conclusions

In this paper, a fuzzy adaptive particle filter (FAPF) was presented and its application in the field of fault diagnosis for mobile robots was discussed. FAPF integrated domain knowledge, which was represented with fuzzy sets, into particle filters. Domain knowledge was employed to constrain sampling space to a fuzzy subset of the universal state space. Main advantages of this scheme include: (1) most particles will be drawn from most hopeful regions of the state spaces, (2) logical inference abilities can be integrated into particle filter by domain constraints.

The approach is testified in the dead-reckoning system fault diagnosis problem of a real mobile robot, MORCS-1. The experimental results show that FAPF is much more accurate the GPF, and is slightly slower than GPF.

Acknowledgment

This work is partially supported by the National Natural Science Foundation of China Grant #60234030 and National Basic Research Project of China (A1420060159).

References

1. Cai, Z. X., He, H. G., Chen, H.: Some issues for mobile robot navigation under unknown environments. Control and Decision, vol. 17, no. 4, pp.385-390, 2002. (in Chinese)
2. Roumeliotis, S. I., Sukhatme, G. S., Bekey, G. A.: Sensor fault detection and identification in a mobile robot. IEEE/RSJ Int’l Conf. on Intelligent Robots and Systems. pp.1383-1388, 1998..

3. Goel, P., Dedeoglu, G., Roumeliotis, S. I., Sukhatme, G. S.: Fault detection and identification in a mobile robot using multiple model estimation and neural network. IEEE Int'l Conf. on Robotics & Automation, pp.2302-2309, 2000.
4. Hashimoto, M., Kawashima, H., Nakagami, T., Oba, F.: Sensor fault detection and identification in dead-reckoning system of mobile robot: interacting multiple model approach. Int'l Conf. on Intelligent Robots and Systems, pp.1321-1326, 2001
5. Hashimoto, M., Kawashima, H., Nakagami, T., Oba, F.: Sensor fault detection and identification in dead-reckoning system of mobile robot: interacting multiple model approach. Int'l Conf. on Intelligent Robots and Systems, pp.1321-1326, 2001
6. Verma, V., Langford, J., Simmons, R.: Non-Parametric Fault Identification for Space Rovers. International Symposium on Artificial Intelligence and Robotics in Space (iSAIRAS), 2001
7. Verma, V., Gordon, G., Simmons, R.: Efficient Monitoring for Planetary Rovers. International Symposium on Artificial Intelligence and Robotics in Space, 2003.
8. Arulampalam, M. S., Maskell, S., Gordon, N., Clapp, T.: A Tutorial on Particle Filters for On-line Nonlinear/Non-Gaussian Bayesian Tracking. IEEE Transactions on Signal Processing, v 50, n 2, pp.174-188, 2002
9. Kwok, C., Fox, D., Meila, M.: KLD-Sampling: Adaptive Particle Filters. Advances in Neural Information Processing Systems (NIPS), 2001
10. de Freitas, N.: Rao-Blackwellised particle filtering for fault diagnosis. Proceedings of IEEE Aerospace Conference. 2002
11. Mo, Y. Xiao, D.: Hybrid system monitoring and diagnosing based on particle filter algorithm. Acta Automatica Sinica, vol. 29, no. 5, pp. 641-648, 2003
12. Verma, V., Thrun, S., Simmons, R.: Variable Resolution Particle Filter. Proceedings of the International Joint Conference on Artificial intelligence, 2003.

Tracking Control of a Mobile Robot with Kinematic Uncertainty Using Neural Networks

An-Min Zou[1], Zeng-Guang Hou[1], Min Tan[1],
Xi-Jun Chen[2], and Yun-Chu Zhang[1,3]

[1] Laboratory of Complex Systems and Intelligence Science, Institute of Automation, The Chinese Academy of Sciences, P.O. Box 2728, Beijing 100080, China
{anmin.zou, zengguang.hou, min.tan, yunchu.zhang}@ia.ac.cn
[2] P.O. Box 2651, Beijing 100084, China
chenxijun@compsys.ia.ac.cn
[3] School of Information and Electric Engineering, Shandong University of Architecture and Engineering, Jinan 250101, China

Abstract. In this paper, a kinematic controller based on input-output linearization plus neural network (NN) controller is presented for tracking control of a mobile robot with kinematic uncertainty. The NN controller, whose parameters are tuned on-line, can deal with the uncertainty imposed on the kinematics model of mobile robots. The stability of the proposed approach is guaranteed by the Lyapunov theory. Simulation results show the efficiency of the proposed approach.

1 Introduction

In recent years, tracking control of mobile robots has received wide attention. There are some methods have been proposed to solve the tracking problem of a mobile robot, such as backstepping [1], sliding mode [2], pure pursuit [3], neural networks [4] [5], fuzzy logic systems [6] [7], linearization [8]. However, in these papers, the wheels of the robot are assumed to be pure rolling and nonslipping and the uncertainty imposed on the kinematics model of mobile robots has not been considered. In practical situations, any set of sensor data used to estimate the robot pose (position and orientation) may be incomplete and distorted by noises and errors, at the same time, the wheels of the robot may be skidding and slipping. In this case, the nonholonomic constraints are violated. The existing methods for dealing with uncertainties include adaptive control and robust control. Adaptive control schemes reduce uncertainties by learning them. Most of the adaptive controllers involve certain types of function approximators such as neural networks in their learning mechanism. On the other hand, robust control scheme focuses on compensating the uncertainties by employing high gain feedback. In [5] [9], the adaptive NN controllers have been proposed due to the universal approximation property of NN [10], where the NN weights are tuned on-line and the stability is guaranteed by Lyapunov theory.

This paper presents a kinematic tracking controller for mobile robots with kinematic uncertainty which combines input-output linearization method and

NN control with adaptive and robust control techniques. The NN controller, whose parameters are tuned on-line, can deal with the uncertainty imposed on the kinematics model of mobile robots. The stability of the proposed approach is guaranteed by the Lyapunov theory.

This paper is organized as follows. Some basis of neural networks are described in Section 2. Section 3 discusses the design of kinematic controller based on the input-output linearlization method and NN. Stability of the proposed approach is proven by the Lyapunov theory. Simulation results are presented in Section 4 and conclusions are given in Section 5.

2 Feedforward Neural Networks

A two-layer feedforward NN has output

$$y_i = \sum_{j=1}^{N_h} [\omega_{ij} \sigma(\sum_{k=1}^{n} (\upsilon_{jk} x_k + \theta_{\upsilon k})) + \theta_{\omega i}] \quad i = 1, 2, \cdots, m, \tag{1}$$

where $x = (x_1, x_2, \cdots, x_n)^T$ and $y = (y_1, y_2, \cdots, y_m)^T$ are the input and output vectors of the NN, respectively; N_h is the number of hidden-layer neurons; $\theta_{\upsilon k}$ and $\theta_{\omega i}$ are the thresholds; ω_{ij} and υ_{jk} are the interconnection weights of the input-to-hidden-layer and hidden-to-output-layer, respectively, and $\sigma(\cdot)$ are activation functions. In this paper, we shall use the sigmoid activation function

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{2}$$

Equation (1) can be rewritten in matrix form as follows

$$y = W^T \sigma(V^T x), \tag{3}$$

where the vector activation function is defined by $\sigma(z) = (\sigma(z_1), \sigma(z_2), \cdots, \sigma(z_n))^T$ for a vector $z \in R^n$; the thresholds are included as the first columns of weight matrices.

Let $f(x)$ be a smooth function from R^n to R^m. Then, it can be approximated by

$$f(x) = W^T \sigma(V^T x) + \varepsilon_n, \tag{4}$$

where ε_n is the NN functional approximation error. An estimate of $f(x)$ can be given by

$$\hat{f}(x) = \hat{W}^T \sigma(\hat{V}^T x). \tag{5}$$

3 Kinematic Controller Design

This paper we consider of a class of mobile robots with the following kinematics model

$$\dot{q} = J(q)u + d, \tag{6}$$

where $q = (x, y, \theta)^T$ is the pose of the robot; $u = (v, \omega)^T$, $0 \leq v \leq v_{max}$, $|\omega| \leq \omega_{max}$ are the linear and angular velocities of the robot, respectively; d denotes unknown bounded disturbance, which violates the pure rolling and nonslipping constraints, and $J(q)$ is given by

$$J(q) = \begin{pmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ 0 & 1 \end{pmatrix}. \tag{7}$$

The trajectory tracking problem for a mobile robot is described as follows: given the reference pose $q_r = (x_r, y_r, \theta_r)^T$, the reference velocities $u_r = (v_r, \omega_r)^T$ with $v_r > 0$ for all time and $\dot{q}_r = J(q_r)u_r$, find a smooth velocity u_c such that $\lim_{t\to\infty}(q_r - q) = 0$ and $\lim_{t\to\infty}(u_r - u_c) = 0$.

The output of the robot is defined as

$$Y = \begin{pmatrix} x + D\cos(\theta) \\ y + D\sin(\theta) \end{pmatrix} \tag{8}$$

with D a positive constant.

The time derivative of Y is

$$\dot{Y} = \begin{pmatrix} \cos(\theta) & -D\sin(\theta) \\ \sin(\theta) & D\cos(\theta) \end{pmatrix} u + \bar{d} \equiv Eu + \bar{d}, \tag{9}$$

where $\bar{d}$ is

$$\bar{d} = \begin{pmatrix} 1 & 0 & -D\sin(\theta) \\ 0 & 1 & D\cos(\theta) \end{pmatrix} d, \tag{10}$$

it is obvious that $\bar{d}$ is bounded such that $\|\bar{d}\| \leq \bar{d}_M$ and we assume that $\bar{d}$ is a function of the robot output Y.

Let the control u be

$$u = E^{-1}(v + u_n + u_s), \tag{11}$$

where v is the new input vector, and u_n is the adaptive NN control and u_s is the sliding mode component to be determined below, then we can obtain input-output linearlization as follows

$$\dot{Y} = v + u_n + u_s + \bar{d}. \tag{12}$$

The new input v is defined as

$$v = \dot{Y}_r + Ke_\eta, \tag{13}$$

where K is a positive constant matrix; $Y_r = (x_r + D\cos(\theta_r), y_r + D\sin(\theta_r))^T$ is the reference output and $e_\eta = e - \eta sat(\frac{e}{\eta})$ with $e = Y_r - Y$ the tracking error, η a small positive constant and $sat(\cdot)$ a saturation function. The vector saturation is defined by $sat(z) = (sat(z_1), sat(z_2), \cdots, sat(z_n))^T$ for a vector $z \in R^n$. The constant η describes the width of a boundary layer, which is applied to prevent discontinuous control transitions. Then we have

$$\dot{e} = -Ke_\eta - u_n - u_s - \bar{d}. \tag{14}$$

Assume that the robot reference outputs are constrained in the compact subset A_d of the state space. In this paper, the compact subset A_d is reconstructed

$$A_d = \{Y \| \|Y - Y_0\|_{p,\pi} \leq 1\}, \tag{15}$$

where Y_0 is a fixed vector in the state space of the plant, and $\|Y\|_{p,\pi}$ is a weighted p norm of the form

$$\|Y\|_{p,\pi} = \left\{ \sum_{i=1}^{2} \left(\frac{|Y_i|}{\pi_i} \right)^p \right\}^{1/p}, \tag{16}$$

with π_i strictly positive weights. Then, in the subset A_d, $\bar{d}$ can be approximated by neural networks as follows

$$\bar{d} = W^T \sigma(V^T z) + \varepsilon_n, \tag{17}$$

where z with $z_i = \frac{Y_i - Y_{0i}}{\pi_i^{1-p}}$ is the input vector of the NN, then the set A_d is equivalent to $A_d = \{z \| \|z\|_p \leq 1\}$; ε_n is the NN functional approximation error. Since it is impossible to guarantee *a prior* that the robot trajectory will remain in the set A_d, the following reasonable assumption is given

Assumption 1:

$$||\varepsilon_n(z)|| \leq \varepsilon_M + \alpha(z), \quad \alpha(z) = 0 \quad \text{if} \quad z \in A_d, \tag{18}$$

where $\alpha(z)$ represents the approximation error outside the compact subset A_d. An estimate of $\bar{d}$ can be given by

$$\hat{\bar{d}} = \hat{W}^T \sigma(\hat{V}^T z). \tag{19}$$

Remark: The input vectors of the NN are normalized, which is motivated by the facts that better convergent performance of NN can be obtained by using the normalized data as inputs to train the two-layer feedforward NN.

By defining $\varepsilon_d = \bar{d} - \hat{W}^T \sigma(\hat{V}^T z)$, then we have

$$\begin{aligned}
\varepsilon_d &= \bar{d} - \hat{W}^T \sigma(\hat{V}^T z) = \bar{d} - W^T \sigma(V^T z) + W^T \sigma(V^T z) - \hat{W}^T \sigma(\hat{V}^T z) \\
&= \varepsilon_n + W^T \sigma(V^T z) - W^T \sigma(\hat{V}^T z) + W^T \sigma(\hat{V}^T z) - \hat{W}^T \sigma(\hat{V}^T z) \\
&= \varepsilon_n + W^T \tilde{\sigma} + \tilde{W}^T \sigma(\hat{V}^T z) \\
&= \varepsilon_n + W^T \tilde{\sigma} - \hat{W}^T \tilde{\sigma} + \hat{W}^T \tilde{\sigma} + \tilde{W}^T \sigma(\hat{V}^T z) \\
&= \varepsilon_n + \tilde{W}^T \tilde{\sigma} + \hat{W}^T \tilde{\sigma} + \tilde{W}^T \sigma(\hat{V}^T z)
\end{aligned} \tag{20}$$

with $\tilde{\sigma} = \sigma(V^T z) - \sigma(\hat{V}^T z)$, $\tilde{W} = W - \hat{W}$, and $\tilde{V} = V - \hat{V}$.

The Taylor series expansion of $\sigma(z)$ for a given z may be written as

$$\sigma(V^T z) = \sigma(\hat{V}^T z) + \sigma'(\hat{V}^T z)\tilde{V}^T z + O(\tilde{V}^T z) \tag{21}$$

with

$$\sigma'(\hat{z}) = \frac{\partial \sigma(z)}{\partial z}|_{z=\hat{z}} \tag{22}$$

the Jacobian matrix and $O(\tilde{V}^T z)$ denoting the high-order terms in the Taylor series. Then we have

$$\tilde{\sigma} = \sigma'(\hat{V}^T z)\tilde{V}^T z + O(\tilde{V}^T z). \tag{23}$$

Substituting (23) into (20) results in

$$\varepsilon_d = \tilde{W}^T \sigma(\hat{V}^T z) - \tilde{W}^T \sigma'(\hat{V}^T z)\hat{V}^T z + \hat{W}^T \sigma'(\hat{V}^T z)\tilde{V}^T z + \phi \tag{24}$$

with $\phi = \varepsilon_n + \tilde{W}^T \sigma'(\hat{V}^T z) V^T z + W^T O(\tilde{V}^T z)$ denoting a residual term.

Some assumptions are given as follows:

Assumption 2: The ideal NN weights are bounded so that $\|W\|_F \leq W_M$, $\|V\|_F \leq V_M$ with $\|\cdot\|_F$ the Frobenius norm.

Assumption 3: The desired reference trajectory is continuous and bounded so that $\|q_r\| \leq q_M$ with q_M a known scalar bound.

Assumption 4: The reference linear velocity v_r is bounded, and $v_r > 0$ for all $t \geq 0$, and the angular velocity ω_r is bounded.

Lemma 1 (Bounds on the Robot Reference Output Y_r): The reference output of the robot is bounded by

$$\|Y_r\| \leq q_M + D \equiv Y_M. \tag{25}$$

Lemma 2 (Bounds on NN Input z): For each time t, the NN input z is bounded by

$$\|z\| \leq c_0 + c_1\|e\| \tag{26}$$

with c_i positive constants.

Lemma 3 (Bounds on Taylor Series High-Order Terms): For sigmoid activation functions, the high-order terms in the Taylor series are bounded by

$$\|O(\tilde{V}^T z)\| = \|\tilde{\sigma} - \sigma'(\hat{V}^T z)\tilde{V}^T z\| \leq c_2 + c_3\|\tilde{V}\|_F + c_4\|\tilde{V}\|_F\|e\| \tag{27}$$

with c_i positive constants.

Lemma 4 (Bounds on the Residual Term ϕ): For $\forall z \in A_d$, ϕ is bounded by

$$\begin{aligned}
\|\phi\| &\leq \varepsilon_M + c_2\|W\|_F + c_3\|W\|_F\|\tilde{V}\|_F + c_4\|W\|_F\|\tilde{V}\|_F\|e\| \\
&\quad + c_3\|V\|_F\|\tilde{W}\|_F + c_4\|V\|_F\|\tilde{W}\|_F\|e\| \\
&\leq \varepsilon_M + c_2 W_M + c_3 W_M\|\tilde{V}\|_F + c_4 W_M\|\tilde{V}\|_F\|e\| \\
&\quad + c_3 V_M\|\tilde{W}\|_F + c_4 V_M\|\tilde{W}\|_F\|e\| \\
&\leq C_0 + C_1\|\tilde{Z}\|_F + C_2\|\tilde{Z}\|_F\|e\| \\
&\leq C_0 + C_1 Z_M + C_1\|\hat{Z}\|_F + C_2 Z_M\|e\| + C_2\|\hat{Z}\|_F\|e\| \\
&= (C_0 + C_1 Z_M, C_1, C_2 Z_M, C_2) \cdot (1, \|\hat{Z}\|_F, \|e\|, \|\hat{Z}\|_F\|e\|)^T \\
&\equiv \beta^T \cdot Y_f
\end{aligned} \tag{28}$$

with $Z = diag\{W, V\}$ and $\|Z\|_F \leq Z_M$, and C_i are positive constants.

Remark: Lemma 4 demonstrates that the residual term ϕ is bounded by a linear expression with a known function vector. Thus, adaptive control techniques can be employed to deal with this residual term.

The NN adaptive law is now described below

$$u_n = (1 - m(t))(-\hat{W}^T\sigma(\hat{V}^T z) + \hat{\beta}^T Y_f sat(\frac{e}{\eta})) \tag{29}$$

$$u_s = k_s m(t) sat(\frac{e}{\eta}) \tag{30}$$

$$\dot{\hat{W}} = (1 - m(t))(-F\sigma(\hat{V}^T z)e_\eta^T + F\sigma'(\hat{V}^T z)\hat{V}^T z e_\eta^T) \tag{31}$$

$$\dot{\hat{V}} = -(1 - m(t))G z e_\eta^T \hat{W}^T \sigma'(\hat{V}^T z) \tag{32}$$

$$\dot{\hat{\beta}} = (1 - m(t))H\|e_\eta\|Y_f, \tag{33}$$

where $\hat{\beta}$ is the estimate of β; k_s is a positive constant satisfying $k_s \geq \bar{d}_M$ and F, G, and H are symmetric positive definite matrices. The modulation function $m(t)$ is chosen as

$$m(t) = \begin{cases} 0 & \text{if} \quad z \in A_d \\ \frac{\|z\|_p - 1}{\Psi} & \text{if} \quad z \in A_\Psi - A_d \\ 1 & \text{if} \quad z \in A_\Psi^c \end{cases}, \tag{34}$$

where A_Ψ is chosen as $A_\Psi = \{z | \|z\|_p \leq 1 + \Psi\}$, and Ψ is a small positive constant, denoting the width of the transition region. The modulation function $m(t)$ generates a smooth switching between the adaptive NN and sliding modes. If the robot output state in the set A_d, the pure NN control performs, while outside the set A_Ψ, the pure sliding mode control behaves and forces the robot output state back into A_d. In between the region $A_\Psi - A_d$, the two modes are effectively blended using a continues modulation function. Then (14) becomes

$$\begin{aligned}\dot{e} = &-Ke_\eta - (1-m(t))(\tilde{W}^T\sigma(\hat{V}^Tz) - \tilde{W}^T\sigma'(\hat{V}^Tz)\hat{V}^Tz)\\ &-(1-m(t))\hat{W}^T\sigma'(\hat{V}^Tz)\tilde{V}^Tz - (1-m(t))\hat{\beta}^TY_f sat(\frac{e}{\eta}) - (1-m(t))\phi\\ &+m(t)(-k_s sat(\frac{e}{\eta}) - \bar{d}). \qquad (35)\end{aligned}$$

Theorem 1. *Consider the mobile robot system (6) with control (11), where the adaptive NN control u_n is given by (29) and the sliding mode component u_s is given by (30). Let the parameters in (29) be adjusted by the adaptive laws (31)-(33), respectively, and suppose that Assumptions 1-4 are satisfied and define $\tilde{\beta} = \beta - \hat{\beta}$. Then*

1. *$\tilde{W}$, $\tilde{V}$, $\tilde{\beta}$, and e_η are uniformly ultimately bounded.*
2. *The tracking error $\|e\| \le \eta$ is obtained asymptotically.*

Proof. (1) Consider the following Lyapunov candidate

$$L = \frac{1}{2}(e_\eta^T e_\eta + tr\{\tilde{W}^TF^{-1}\tilde{W}\} + tr\{\tilde{V}^TG^{-1}\tilde{V}\} + \tilde{\beta}^TH^{-1}\tilde{\beta}) \qquad (36)$$

with $tr\{\cdot\}$ the trace. Note that though $\dot{e}_{\eta i}$ is not defined when $|e_{\eta i}| = \eta$, $(d/dt)e_{\eta i}^2$ is well-defined and continuous everywhere and can be written as $(d/dt)e_{\eta i}^2 = 2e_{\eta i}\dot{e}_i$.

The time derivative of L is

$$\dot{L} = e_\eta^T\dot{e} - tr\{\tilde{W}^TF^{-1}\dot{\hat{W}}\} - tr\{\tilde{V}^TG^{-1}\dot{\hat{V}}\} - \tilde{\beta}^TH^{-1}\dot{\hat{\beta}}. \qquad (37)$$

Substituting (29)-(33) and (35) into (37) yields

$$\begin{aligned}\dot{L} = &-e_\eta^TKe_\eta - (1-m(t))(e_\eta^T\tilde{W}^T\sigma(\hat{V}^Tz) - e_\eta^T\tilde{W}^T\sigma'(\hat{V}^Tz))\\ \\ &-(1-m(t))e_\eta^T\hat{W}^T\sigma'(\hat{V}^Tz)\tilde{V}^Tz - (1-m(t))e_\eta^T\hat{\beta}^TY_f sat(\frac{e}{\eta})\\ &-(1-m(t))e_\eta^T\phi + m(t)e_\eta^T(-k_s sat(\frac{e}{\eta}) - \bar{d})\\ &-tr\{\tilde{W}^TF^{-1}\dot{\hat{W}}\} - tr\{\tilde{V}^TG^{-1}\dot{\hat{V}}\} - \tilde{\beta}^TH^{-1}\dot{\hat{\beta}}\\ = &-e_\eta^TKe_\eta - (1-m(t))\|e_\eta\|\hat{\beta}^TY_f - (1-m(t))e_\eta^T\phi\\ &-(1-m(t))\|e_\eta\|\tilde{\beta}^TY_f + m(t)(-k_s\|e_\eta\| - e_\eta^T\bar{d})\\ \le &-e_\eta^TKe_\eta - (1-m(t))e_\eta^T\phi - (1-m(t))\|e_\eta\|\beta^TY_f\\ \le &-e_\eta^TKe_\eta\\ \le &-\lambda_{min}(K)e_\eta^Te_\eta, \qquad (38)\end{aligned}$$

where the facts for $\forall z \in A_d$, $\|\phi\| \le \beta^TY_f$ and $e_\eta sat(\frac{e}{\eta}) = \|e_\eta\|$ have been used, and $\lambda_{min}(K)$ is the minimum eigenvalue of matrix K. According to Lyapunov theory, this demonstrates that $\tilde{W}$, $\tilde{V}$, $\tilde{\beta}$, and e_η are uniformly ultimately bounded.

(2) Integration of (38) from $t = 0$ to ∞ results in

$$\int_0^\infty \|e_\eta(t)\|^2 dt \leq \frac{L(0) - L(\infty)}{\lambda_{min}(K)}. \tag{39}$$

Noting that $L(t)$ is a non-increasing function of time and low bounded, this implies that $L(0) - L(\infty) < \infty$, then we have $e_\eta \in L_2$. The boundedness of e_η implies $e_\eta \in L_\infty$. Because we have proven that all the variables on the right-hand side of (35) are bounded, we have $\dot{e}_\eta \in L_\infty$. Using Barbalat's lemma [11] (if $e_\eta \in L_2 \bigcap L_\infty$ and $\dot{e}_\eta \in L_\infty$, then $\lim_{t\to\infty} e_\eta(t) = 0$), we have $\lim_{t\to\infty} e_\eta(t) = 0$, this means that the inequality $\|e\| \leq \eta$ is obtained asymptotically.

Remark: Theorem 1 demonstrates that the tracking errors depend on the choice of η. If $\eta \to 0$, then $e \to 0$. In this case, $sat(\frac{e}{\eta})$ becomes $sgn(e)$. Using the saturation switch function instead of the signum function, chatter is overcome.

4 Simulation Studies

In this section, we shall provide some simulation results to show the effectiveness of our proposed methods. The parameters used in the controller are: $D = 0.5$, $K = diag\{10, 15\}$, $k_s = 0.6$, $\eta = 0.01$, $\Psi = 0.2$, $v_{max} = 2$m/s, $\omega_{max} = 5$rad/s. For the NN controller, we selected the sigmoid activation function with $N_h = 10$ hidden-layer neurons, $F = 0.005 \cdot diag\{1, 2, \cdots, 10\}$, $G = 0.005 \cdot diag\{5, 10\}$ and $H = 0.0005 \cdot diag\{1, 2, 3, 4\}$. The sampling time is 0.01s. Note that these parameters are the same in the two cases of the simulations.

Case 1: The tracking path is a circle, which is defined by $x^2 + y^2 = 1$, as shown in Fig. 1. The reference linear and angular velocities are $v_r = 1$m/s and $\omega_r = 1$rad/s, respectively. The initial pose of the desired virtual cart is $(0, -1, 0)^T$. The actual robot starts at $(-0.5, -1.5, 0)^T$. The compact set A_d is chosen as $A_d = \{Y|(\frac{Y_1^2+Y_2^2}{2})^{1/2} \leq 1\}$. The elements of the external disturbance are randomly selected in interval [-0.2, 0.2].

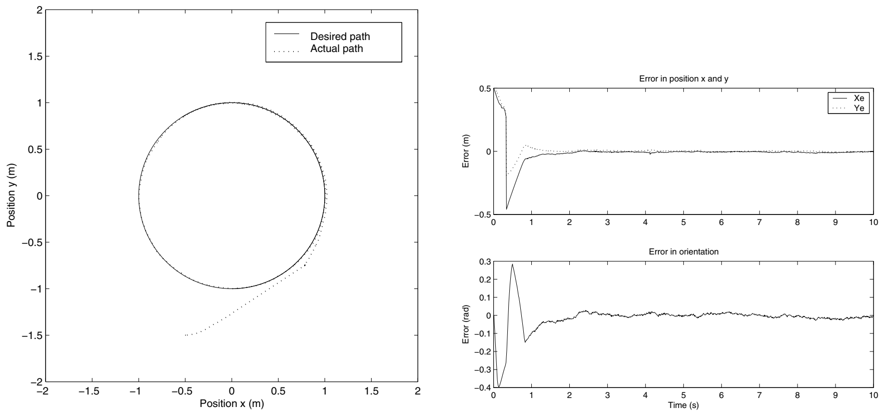
Case 2: The initial pose of the desired virtual cart is $(0, -1, 0)^T$. The actual robot starts at $(-0.5, -1.5, 0)^T$. The reference linear and angular velocities are given as follows

$$v_r = \begin{cases} 0.5 + 0.5\sin t, & 0 \leq t < 4 \text{ s} \\ 1.5, & 4 \leq t < 6 \text{ s} \\ 1 + \cos t, & 6 \leq t < 8 \text{ s} \\ 1, & 8 \leq t \leq 10 \text{ s} \end{cases} \tag{40}$$

and

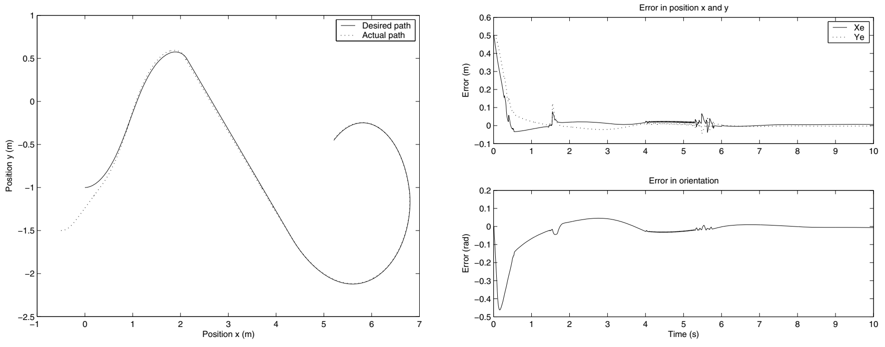
$$\omega_r = \begin{cases} \cos t, & 0 \leq t < 4 \text{ s} \\ 0, & 4 \leq t < 6 \text{ s} \\ 1 + 0.5\sin t, & 6 \leq t < 8 \text{ s} \\ 1, & 8 \leq t \leq 10 \text{ s} \end{cases}. \tag{41}$$

It is assumed that the external disturbance is $d = (0.1 \sin t, 0.1 \cos t, 0.1 \sin t)^T$. The compact set A_d is chosen as $A_d = \{Y | (\frac{(Y_1-3.5)^2}{24.1} + \frac{(Y_2+1.4)^2}{3.92})^{1/2} \leq 1\}$. The simulation results are shown in Fig. 2. The results show that the proposed NN controller greatly improves the performances of robot tracking with model uncertainty.



(a) (b)

Fig. 1. Tracking using NN control with uncertainty : (a) mobile robot trajectory; (b) position and orientation errors



(a) (b)

Fig. 2. Tracking using NN control with uncertainty: (a) mobile robot trajectory; (b) position and orientation errors

5 Conclusions

A stable control algorithm based on input-output linearization method and NN control capable of dealing with the uncertainty imposed on the kinematic model of mobile robots is proposed in this paper. In the proposed scheme, the NN controller with a set of "on-line" tunable parameters with no "off-line learning phase" needed is employed to approximate the uncertainty of the robot due to the universal approximation property of the NN. Stability of the proposed method is proven by Lyapunov theory. Some simulations are provided in order to illustrate the feasibility of the proposed method.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (Grants 60205004, 50475179 and 60334020), the National Basic Research Program (973) of China (Grant 2002CB312200), and the Hi-Tech R&D Program (863) of China (Grants 2002AA423160 and 2005AA420040).

References

1. Kanayama, Y., Kimura, Y., Miyazaki, F., Noguchi, T.: A Stable Tracking Control Method for an Autonomous Mobile Robot. Proceedings of IEEE International Conference on Robotics and Automation, **1** (1990) 384-389
2. Yang, J. M., Kim, J. H.: Sliding Mode Control for Trajectory Tracking of Nonholonomic Wheeled Mobile Robots. IEEE Transactions on Robotics and Automation, **15**(3)(1999) 578-587
3. Anfbal, O., Guillermo, H.: Stability Analysis of Mobile Robot Tracking. Proceedings of 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems, **3**(1995) 461-466
4. Yuan, G., Yang, S. X., Mittal, G. S.: Tracking Control of a Mobile Robot Using a Neural Dynamics based Approach. Proceedings of IEEE International Conference on Robotics and Automation, **1** (2001) 163-168
5. Fierro, R., Lewis, F. L.: Control of a Nonholonomic Mobile Robot Using Neural Networks. IEEE Transactions on Neural Networks, **9** (4) (1998) 589-600
6. Zou, A., Hou, Z., Tan, M., Zhao, Z.: Tracking Control of a Nonholonomic Mobile Robot Using a Fuzzy-Based Approach. ICNC06-FSKD06, (2006) (Accepted)
7. Zou, A., Hou, Z., Wang, H., Tan, M.: Stabilizing Control of a Nonholonomic Mobile Robot: A Fuzzy-Based Approach. Proceeding of the International Conference on Sensing, Computing, and Automation, (2006) 3219-3224
8. Kim, D. H., Oh, J. H.: Tracking Control of a Two-wheeled Mobile Robot Using Input-output Linearization. Control Engineering Practice, **7**(3) (1999) 369-373
9. Lewis, F. L., Yesidirek, A., Liu, K.: Multilayer Neural-Net Robot Controller with Guaranteed Tracking Performance. IEEE Transactions on Neural Networks, **7** (2) (1996) 388-399
10. Homik, K., Stinchcombe, M., White, H.: Multilayer Feedforward Networks are Universal Approximators. Neural Networks, **2** (1989) 359-366
11. Sastry, S., Bodson, M.: Adaptive Control: Stability, Convergence, and Robustness. Englewood Cliffs, NJ: Prentice-Hall, 1989

Movement Control of a Mobile Manipulator Based on Cost Optimization

Kwan-Houng Lee[1] and Tae-jun Cho[2]

[1] School of Electronics & Information Engineering, Cheongju University, Naedok-Dong 36, Sangdang-Gu Cheongju-City, Chungbuk, 360-764, Republic of Korea
khlee368@cju.ac.kr
[2] School of Civil and Env. Engineering, Cheongju University, Naedok-Dong36, Sangdang-Gu, Cheongju-City, Chungbuk, 360-764, Republic of Korea
taejun@cju.ac.kr

Abstract. In this paper, using the pre-determined specific tasks, a solid and complete solution for the optimal control of the mobile manipulator is proposed based on a divide and conquer scheme. In the scheme, a mobile manipulator is virtually divided into a mobile robot and a task robot. All the tasks are also divided into task segments that can be performed by only the task robot. An optimal configuration of the task robot is defined by the task oriented manipulability measure for given task segment. And using a cost function for optimality defined as a combination of the square errors of the desired and actual configurations of the mobile robot and of the task robot, the job which the mobile manipulator performs is optimized. We figured out the solution for the optimal configuration of a mobile manipulator with a series of tasks.

1 Introduction

Mobile manipulators have been used in various fields, such as factory automation, underwater exploration, robotic surgery, space exploration, nuclear power plant maintenance, and etc[1]. They are supposed to be used more widely in the future. The two main characteristics of mobile manipulators are high kinematic redundancy and dynamic nonhomogeneity[2]. The former results from the addition of the platform DOF(Degree Of Freedom) to the manipulator[3]. A mobile manipulator generates the redundant DOF in kinematics different from the other redundant robot with the fixed base structure and with at least six DOF. Pin and Culioli[4] proposed the concept of commutation configuration, and formulated the path planning problem as multi-objective optimization, which was settled by the minmax approach. Yamamoto and Yun[5] demonstrated the coordinated motion of mobile manipulators using the concept of preferred operation region, which maximized its manipulability measure. Seraji[6] regarded the whole mobile manipulator as a redundant manipulators. Carriker formulated the coordination of manipulation and locomotion as a nonlinear optimization problem in order to solve the problem of kinematic redundancy[7].

Fig. 1 shows the mobile manipulator, the robot, which is implemented for the experiment. Fig. 2 shows the block diagram of mobile manipulator configuration, where the mobile robot is wheel-driven and is capable of moving its platform up and down on the basis of gravity center. The task robot is mounted on the center of the platform. We used 80C196KC microprocessor as the motor controller to control the three motors concurrently. The motor-drive point with an IGBT was configured in H-bridge form. We mounted the ROB3 with 5 joints as the task robot Then, we installed a gripper at the end-effector to grip things. In addition to that, we mounted the portable pc, which was used as the host computer to monitor the controller of the mobile manipulator and to monitor the stats of the robot.

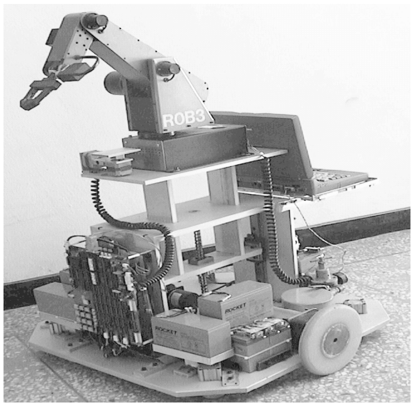


Fig. 1. Mobile Manipulator PURL-II

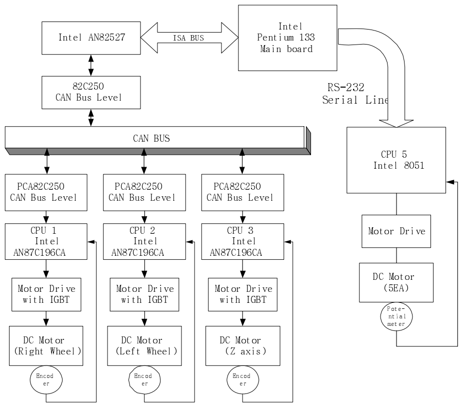


Fig. 2. Block diagram of PURL-II

2 System Operation Algorithm

2.1 Definition of Cost Function

When moving an object from some points to a desired point for the mobile manipulator, the base frame position of task robot varies according to the movement of mobile robot. Therefore through inverse kinematics, the task planning has many solutions with respect to the robot movement. For the robot to perform the task efficiently after defining the constraint condition, we must find the accurate solution to satisfy both the optimal accomplishment of the task and the efficient completion of the task. In this paper, we have the objective of movement-minimization of the whole robot when performing the task. Therefore we expressed the vector for mobile manipulator states as:

$$\boldsymbol{s} = \begin{bmatrix} \boldsymbol{p} \\ \boldsymbol{q}_t \end{bmatrix} \tag{1}$$

where $\boldsymbol{p} = [x \quad y \quad z \quad \theta]^T$ and $\boldsymbol{q}_t = [q_{t1} \quad q_{t2} \quad \cdots \quad q_{tn}]^T$. Here, $\boldsymbol{s}$ is the vector for the robot and consists of $\boldsymbol{p}$ representing the position and direction of mobile robot in Cartesian space and $\boldsymbol{q}_t$ representing the n joint variables of the task robot. Now planning the task to minimize the whole movement of mobile manipulators, a cost function, L, is defined as

$$\begin{aligned} L = \Delta \boldsymbol{s}^T \Delta \boldsymbol{s} &= (\boldsymbol{s}_f - \boldsymbol{s}_i)^{\mathrm{T}} (\boldsymbol{s}_f - \boldsymbol{s}_i) \\ &= (\boldsymbol{p}_f - \boldsymbol{p}_i)^{\mathrm{T}} (\boldsymbol{p}_f - \boldsymbol{p}_i) + (\boldsymbol{q}_{tf} - \boldsymbol{q}_{ti})^{\mathrm{T}} (\boldsymbol{q}_{tf} - \boldsymbol{q}_{ti}) \end{aligned} \tag{2}$$

where $s_i = [\boldsymbol{p}_i \quad \boldsymbol{q}_{ti}]^T$ represents the initial states of the mobile manipulator, and $s_f = [\boldsymbol{p}_f \quad \boldsymbol{q}_{tf}]^T$ represents the final states after having accomplished the task. In the final states, the end-effector of the task robot must be placed at the desired position $\boldsymbol{x}_d$. For that purpose the equation (2) must be satisfied. In (2), we denoted as $R(\boldsymbol{q}_m)$ and $T(\boldsymbol{q}_{tf})$ respectively, as the rotational transformation to the base of the task robot and the translation vector of task robot.

$$\boldsymbol{x}_d = R(\boldsymbol{q}_m) T(\boldsymbol{q}_{tf}) + \boldsymbol{x}_f \tag{3}$$

where $\boldsymbol{x}_d$ represents the desired position of task robot, $\boldsymbol{x}_f$ is the final position of mobile manipulator, and $\boldsymbol{q}_m = [q_r \quad q_l \quad z]^T$. We can express the final position of the mobile manipulator $\boldsymbol{x}_f$ as the function of the desired coordinate $\boldsymbol{x}_d$, joint variables $\boldsymbol{q}_m$ and $\boldsymbol{q}_{tf}$, then the cost function that represents the robot movement is expressed as the $n \times 1$ space function of $\boldsymbol{q}_m$ and $\boldsymbol{q}_{tf}$ as

$$L = \{\boldsymbol{x}_d - R(\boldsymbol{q}_m) f(\boldsymbol{q}_{tf}) - \boldsymbol{x}_i\}^T \{\boldsymbol{x}_d - R(\boldsymbol{q}_m) f(\boldsymbol{q}_{tf}) - \boldsymbol{x}_i\} + \{\boldsymbol{q}_{tf} - \boldsymbol{q}_{ti}\}^{\mathrm{T}} \{\boldsymbol{q}_{tf} - \boldsymbol{q}_{ti}\} \tag{4}$$

In (4), $\boldsymbol{q}_m$ and $\boldsymbol{q}_{tf}$ which minimize the cost function L must satisfy the condition,

$$\nabla L = \begin{bmatrix} \dfrac{\partial L}{\partial \boldsymbol{q}_m} \\ \dfrac{\partial L}{\partial \boldsymbol{q}_{tf}} \end{bmatrix} = 0 \tag{5}$$

Because the cost function is nonlinear, it is difficult to find analytically the optimum solution that satisfies (5). So in this paper, we find the solution numerically using the gradient method described as

$$\begin{bmatrix} \boldsymbol{q}_{m(k+1)} \\ \boldsymbol{q}_{tf(k+1)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{q}_{m(k)} \\ \boldsymbol{q}_{tf(k)} \end{bmatrix} - \eta \nabla L \Big|_{q_{m(k)}, q_{tf(k)}} \tag{6}$$

This recursive process will stop, when $\|\nabla L\| < \varepsilon \approx 0$: when $\boldsymbol{q}_{m(k)}$ and $\boldsymbol{q}_{tf(k)}$ are optimums. Through the optimum solutions of $\boldsymbol{q}_m$ and $\boldsymbol{q}_{tf}$, the final robot state s_f can be calculated as

$$s_f = \begin{bmatrix} \boldsymbol{p}_f \\ \boldsymbol{q}_{tf} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_d - R(\boldsymbol{q}_m) f(\boldsymbol{q}_{tf}) \\ \boldsymbol{q}_{tf} \end{bmatrix} \tag{7}$$

2.2 Cartesian Coordinate Path Planning of the Mobile Robot

For the robot to move and to take a posture from the initial position of the mobile robot to the workspace, we established the coordinate system as the system shown in Fig. 3. When the mobile robot starts at the current position $X_i = (x_i, y_i)$ and moves to position $X_p = (x_p, y_p)$ to include position $X_d = (x_d, y_d)$, $X_d \in X_{traj}$ the final position of the end-effector, within a workspace, we represent the current robot direction, direction error from the current position to the desired position, distance error to the desired point, the mobile robot direction at the desired position as ϕ, α, e, θ respectively, as shown in Fig 3 . When the mobile robot moves from (x_i, y_i) to (x_d, y_d), we make α and e to be minimized, θ to be the direction of the mobile robot at the desired position. Kinematics Eq $\dot{x} = -v \cdot \cos\alpha$, $\dot{y} = v \cdot \sin\alpha$, $\dot{\theta} = \omega$ e > 0, Lyapunov Candidate Function is denoted as the following equation (8)[8][9][10].

$$V = V_1 + V_2 = \frac{1}{2} \cdot \lambda \cdot e^2 + \frac{1}{2}\left(\alpha^2 + h\theta^2\right) \tag{8}$$

where λ and h are arbitrary positive real number, respectively.

Differentiating equations (8), we obtained the following equation (9).

$$\dot{V} = -\lambda \cdot e \cdot v \cdot \cos\alpha + \alpha\left(-\omega + \frac{v \cdot \sin\alpha}{\alpha} \cdot \frac{\alpha + h\theta}{e}\right) \tag{9}$$

In order to be converged stably, $\dot{V}$ has to be less than 0. Hence, nonlinear mobile robot controller is designed as follows:

$$\begin{aligned} v &= \gamma\ (e\cos\alpha), \quad (\gamma > 0) \\ \omega &= k \cdot \alpha + \gamma\ \frac{\cos\alpha \cdot \sin\alpha}{\alpha}(\alpha + h\theta), \quad (k, h > 0) \end{aligned} \tag{10}$$

Therefore, $\dot{V}$ satisfies the convergence condition that $\phi \cong \theta$, when $t \to \infty$ as the following equation(11), $e, \alpha \cong 0$ by mobile robot controller.

$$\dot{V} = -\lambda(\gamma\cos^2\alpha)e^2 - k\alpha^2 \le 0 \tag{11}$$

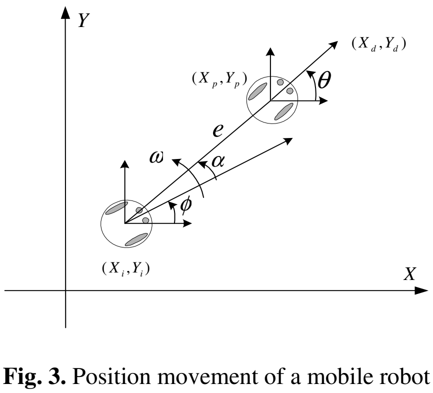


Fig. 3. Position movement of a mobile robot

3 Simulation

Using the algorithm proposed in the paper, we simulated drawing a big rectangle on the wall. Fig. 4 is the result of simulation, which shows the motion of the mobile robot to carry the task robot to the initial position for the rectangular drawing. The simulation conditions are as follows: Initial position: $(x_i, y_i, z_i) = (-10, -10, 1)$

Initial angle between mobile robot and x-axis: $\pi/8$ *rad*

Goal position: $(x_i, y_i, z_i) = (0,0,1)$

Goal angle between mobile robot and x-axis: 0 *rad.* 1. In this simulation, v_x, v_y, and v_z are considered as task constrained components; ω_x, ω_y, and ω_z are considered as task free components. The lengths of link and the allowable range of joint variable are as follows: d1=0.3, l1=0.2, l2=0.33,

$-80^\circ < q_{t1} < 80^\circ, -35^\circ < q_{t2} < 65^\circ, -100^\circ < q_{t3} < 0^\circ$.

2. The given task is composed of four task segments which are defined as the basic units of task with the same task requirements:
Task segment 1: Move from (0.16, 0.45, 0.7) to (1.16, 0.45, 0.7)
Task segment 2: Move from (1.16, 0.45, 0.7) to (1.16, 0.45, 1.0)
Task segment 3: Move from (1.16, 0.45, 1.0) to (0.16, 0.45, 1.0)
Task segment 4: Move from (0.16, 0.45, 1.0) to (0.16, 0.45, 0.7).

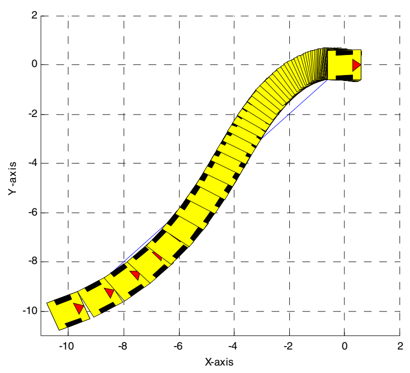


Fig. 4. Simulation results to show the motion of mobile robot

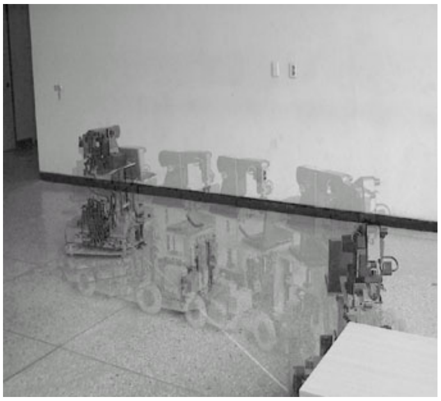


Fig. 5. Result of the experiment

4 Experiments

Fig. 5 is the result of the experiment. The desired point is the top point of the box in the right side of picture. The environment of this experiment is conditioned as follows:

1. Mobile robot moves from the origin of world frame to the initial position to perform the task: Original position: $(x_i, y_i, z_i) = (0, 0, 0.8)$, Original angle between

mobile robot and x-axis: $\pi/8$ *rad* , Initial position: $(x_i, y_i, z_i) = (2, 2, 0.8)$, Initial angle between mobile robot and x-axis: 0*rad.*

2. In this experiment, v_x, v_y, and v_z are considered as task constrained components; ω_x, ω_y, and ω_z are considered as task free components; the length of link and the range of joint variable are set as follows: d1=0.3, l1=0.2, l2=0.33,

$-80^\circ < q_{t1} < 80^\circ$, $-35^\circ < q_{t2} < 65^\circ$, $-100^\circ < q_{t3} < 0^\circ$ ·

3. The given task is composed of four task segments which are defined as the basic units of task with the same task requirements:
Task segment 1: Move from (2.16, 0.48, 0.7) to (3.16, 0.48, 0.7)
Task segment 2: Move from (3.16, 0.48, 0.7) to (3.16, 0.48, 1.0)
Task segment 3: Move from (3.16, 0.48, 1.0) to (2.16, 0.48, 1.0)
Task segment 4: Move from (2.16, 0.48, 1.0) to (2.16, 0.48, 0.7).

The scenario for this experiment is illustrated in Fig. 6. The mobile robot is initially at the origin of the world frame. The task robot is configured as the joint angles of $(70.4^\circ, 38.6^\circ, -51.7^\circ)$. The coordinate of the end-effector is set up as (0.16, 0.45, 0.9).

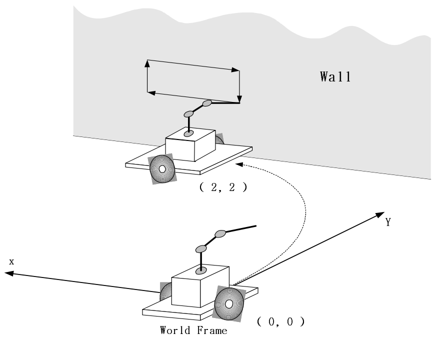


Fig. 6. Scenario for the experiment

5 Conclusions

The effective control methodology of the two serially connected robots – a mobile manipulator – is proposed. A task can be divided into several task segments according to the required motion components. For each task segment, a desired configuration for the task robot is specified considering the task execution efficiency. The actual configuration is controlled to be close to the desired one. To verify the idea experimentally, a mobile manipulator is implemented by the serial connection of a task robot and a mobile robot, which are controlled by a host PC governing the cooperation between them. When we controlled mobile robot to change workspace, we controlled the mobile robot to decrease the distance and direction errors by Lyapunov function. Task robot took an appropriate posture to execute a task using Manipulability Ellipsoid in the workspace, executing a task by moving a mobile robot in that posture so that the end-effector of a task robot can reach the desired position. Therefore, the mobile manipulator executed the task beyond current workspace with an appropriate posture to accomplish the given task in cooperation of mobile robot and task robot. In the further study, exact position control is needed by using a sensor to calibrate position error for velocity control. Then we can execute cooperation control of the two local robots by making a cost function, weighting values for the mobility of a mobile robot and the manipulability ellipsoid of a task robot. Moreover, the dynamic control of the mobile manipulator is left assuming that the precise control of the mobile robot is possible by adding absolute position sensors, such as gyro sensors and a CCD camera. Searching an optimal configuration for ${}^{d}GME$, and defining ${}^{d}GME$ for a task segment are interesting subjects to be investigated.

References

1. Krzysztof Tchon and Janusz Jakubiak.: An Extended Jacobian Inverse Kinematics Algorithm for Doubly Nonholonomic Mobile Manipulators. International Conference on Robotics and Automation Barcelona, Spain, April (2005) pp.1548-1553
2. Tamaka.H, Minami.M,Mae.Y.: Evaluation of Obstacle Avoidance Ability for Redundant Mobile Manipulators. Industrial Electronics Society, IECON2005.32nd Annual Conference of IEEE 6-10. No.1 Aug (2005) pp.1768-1773
3. Xin-Chun Li, Dong-bin Zhao, Jian-Qiang Yi, Xiong-Huilu.:A Coordinated and Hierarchical path Planning Approach for Mobile Manipulators. Proceedings of the Fourth International Conference on Machine Learning and Cybernetics, Guangzhou, 18-21 Aug (2005) pp.3013-3018
4. Francois G. Pin, Jean-Christophe Culioli, David B.Reister.: Using Minmax Approaches to Plan Optimal Task Commutation Configurations for Combined Mobile Platform-Manipulators Systems. IEEE Trans on Robotics and Automation. Vol.1 (1994) pp.44-54
5. Yamanoto Y, Yun X P.: Coordinating Locomotion and Manipulation of a Mobile Manipulator. Proceedings of the 31st Conference on Decision and Control (1992) pp.2643-2648
6. Seraji H.: Motion Control of Mobile Manipulators. Proceedings of IEEE/RSJ International Conference of Intelligent Robots and Systems (1993) pp.2056-2063
7. Tchon,K. Muszynski.R.: Instantaneous Kinematics and Dexterity of Mobile Manipulators. International Conference on Robotics & Automation San Francisco, April (2000) pp.2493-2498
8. Aicardi,M.: Closed Loop Steering of Unicycle-Like Vehicles via Lyapunov Techniques. IEEE Robotics and Automation Magazine (1995) pp.27-35
9. Mark W, Spong.: Robot Dynamics and Control. John Wiley & Sons (1989)
10. Akira Mohri.: Sub-Optimal Trajectory Planning of Mobile Manipulator. International Conference Robotics & Automation. May (2001) pp.21-26

Synthesis of Desired Binary Cellular Automata Through the Genetic Algorithm

Satoshi Suzuki and Toshimichi Saito

Hosei University, Koganei, Tokyo, 184-8584, Japan

Abstract. This paper presents a GA-based synthesis algorithm of a cellular automaton (CA) that can generate a desired spatio-temporal pattern. Time evolution of CA is determined by a rule table the number of which is enormous even for relatively small size CAs: the brute-force search is almost impossible. In our GA-based synthesis algorithm, a gene corresponds to a rule and a masking technique is used to preserve gene(s) with good fitness. Performing basic numerical experiments we have confirmed that the masking works effectively and the algorithm can generate a desired rule table. We have also considered an application to reduction of noise inserted randomly to a spatio-temporal pattern.

1 Introduction

Cellular automata (CA) are nonlinear dynamical systems in which time, space and state variables are all discrete [1] [2]. CA dynamics is governed be a rule table that is equivalent to a function of integers and the CA can generate rich spatio-temporal patterns and the patterns relate to engineering applications including self-replication [3] [4], simulation of natural phenomena [5], block cipher [6] and signal processing [7] [8]. Classification rule tables and finding a desired rule table is basic to analyze the rich spatiotemporal dynamics and to realize engineering applications. However, as size of CA increases, search space of the possible rule tables is to be incredibly large [4] hence finding a desired rule table is to be very hard, at least direct search is impossible. In order to search a desired rule table, several approaches have been studied. A method based on genetic algorithm (GA) may be strong candidate to realize effective search [4] [9] [10]. In the method a rule table corresponds to a chromosome and GA runs to optimize some given fitness function. However, existing algorithms are rather complicated and search tends to be hard for complicated dynamics and/or fitness.

This paper studies a simple search algorithm to synthesize desired binary cellular automata (BCA) that is a simple version of CA. The BCA has binary state variable. its rule table is equivalent to a set of Boolean functions and can generates a variety of binary spatiotemporal patterns. In our algorithm the chromosome is a binary sequence corresponding to outputs of the Boolean functions, the fitness measures error between a desired pattern and a pattern by the chromosome, and GA runs to minimize the error. Major difference from existing algorithms is masking to preserve suitable genes. Performing basic experiments

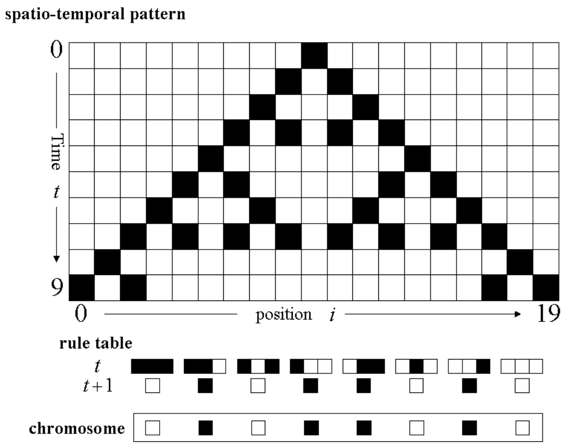


Fig. 1. 3-neighbor binary CA: spatio-temporal pattern and rule table that is equivalent to the binary vector (-1, -1, -1, 1, -1, -1, 1, -1)

for relatively complicated patterns, we can confirm that the masking is very effective to increase success rate of finding desired rule tables. We then apply this algorithm to noise reduction problem of desired patterns with random noise and have confirmed that the masking is also effective in this problem. These results provide basic information to establish systematic synthesis method of desired CA and its application to signal/image processing.

2 Binary Cellular Automata

CA are typical digital dynamical systems that exhibit huge variety of spatio-temporal patterns [1], [2] and BCA is a simple version of CA. The BCA has binary state variable and we consider the of L cells arranged on a ring site. Let $a_i(t) \in \{-1, 1\}$ denote the state variable of position i at discrete time t and let i be a cell index mod L: $i \in \{1, \cdots, L\}$ and $i + L = i$. The dynamics is described by

$$\begin{aligned} a_i(t+1) = F_C(a_{i-k}(t), a_{i-k+1}(t), \cdots, a_{i+k}(t)) \\ F_C : \boldsymbol{B}^r \to \boldsymbol{B}, \quad r \equiv 2k+1, \end{aligned} \tag{1}$$

where k is a positive integer at most $L/2$ and r is referred to as the number of neighbors of a cell. The Boolean function F_C corresponds to a rule table. Fig. 1 shows a rule table of 3-neighbor BCA ($r = 3$) consisting of 2^3 rules. This rule table is equivalent to a binary vector with length 2^3. As an initial condition is given at $t = 0$, this BCA can exhibit an interesting spatio-temporal pattern like the Sierpinski triangle as suggested in the figure. Since possible number of this rule table is 2^{2^3}, the 3-neighbor BCA has variation of 2^{2^3}. Detailed analysis of the 2^{2^3} patterns can be found in [2]. In general, an r-neighbor BCA is governed by a rule table with length 2^r (F_C in (1) is an r-dimensional binary function) and

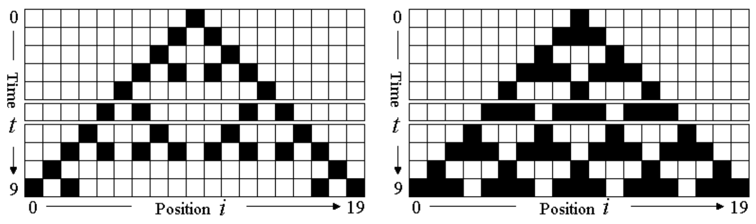


Fig. 2. Two example patterns where fitness at time 5 is 14/19

the number of possible rule tables is 2^{2^r}. That is, as size of neighbors increases the variation of rule tables is to be incredibly large and huge variety of spatio-temporal patterns can be generated.

3 Learning Algorithm

In order to describe the algorithm we give some preparations. First, in our GA, a rule corresponds to a gene and a rule table corresponds to a chromosome. As shown in Fig. 2, output of a rule is -1 (white) or 1 (black) and the rule table is described by a binary sequence. Length of chromosome is 2^m and is equal to that of rule table. For example, the length is 32 for 5-neighbor rule table.

Second, our fitness function measures error between a desired image and an image generated by a chromosome (rule table). The fitness at time t is given by

$$f(t) = \frac{1}{N}\sum_{i=0}^{N-1} f_i(t), \quad f_i(t) = \begin{cases} 1 & \text{if } a_i^t = A_i^t \\ 0 & \text{if } a_i^t \neq A_i^t \end{cases} \tag{2}$$

where i is position of a cell, N is the number of cells, $A_i(t)$ is spatio-temporal pattern A_i^t and $a_i(t)$ is state of a cell. Integrating $f(t)$ we obtain the total fitness:

$$F_c(T) = \frac{1}{T}\sum_{t=0}^{T} f(t) \tag{3}$$

For example spatio-temporal patterns in Fig. 2, 17 cells satisfy $a_i^t = A_i^t$ at $t = 5$ over 19 cells hence $f(5) = \frac{14}{19}$. Repeating similar calculations and integrating them we obtain the total fitness. We then define the main algorithm.

Algorithm

STEP 0:
Some spatio-temporal pattern is given as an input data.

STEP 1: Initialization
Initial chromosome group with the population size K is generated randomly. Let initial generation be $g = 0$.

STEP 2: Masking
For each gena, calculate coincidence rate of state by a gene and corresponding state in a desired pattern. For example, if rule h_5 is applied M times in spatio-temporal pattern by a chromosome and N outputs corresponds to desired pattern then we obtain coincidence rate $h_5 = N/M$. If a gene has coincidence rate over T_m then the gene is masked, where T_m is a threshold parameter. The masked gene can change only by mutation: can not change by any other operation.

STEP 3: Evaluation
Calculate fitness $F_c(t)$ of each chromosome.

STEP 4: Elite Strategy
Leave chromosome having the best fitness for the next generation.

STEP 5: GA Operation
Generate other offspring. Parents are chosen by roulette wheel selection, and applied one-point crossover and mutation. Note again that a masked gene can not be changed by crossover but by mutation.

STEP 6: Termination
If generation is $g = G$, the algorithm is terminated. Otherwise, $g = g + 1$ and go to STEP 2.

4 Numerical Experiments

In order to investigate efficiency of our learning algorithm we have performed fundamental numerical experiments. For the experiments we give several preparations. First, we adopt the λ parameter [1] in order to characterize spatio-temporal patterns of BCA.

$$\lambda = (2^N - n)/2^N = (2^5 - n)/2^5$$

where n is the number of rules that output -1 and $N = 5$ is the number of neighbors. Fig. 3 shows typical spatio-temporal patterns for some values of λ parameter. Roughly speaking, a pattern tends to be complex as λ approaches 0.5 [1]. For simplicity we focus on the following case in the experiments

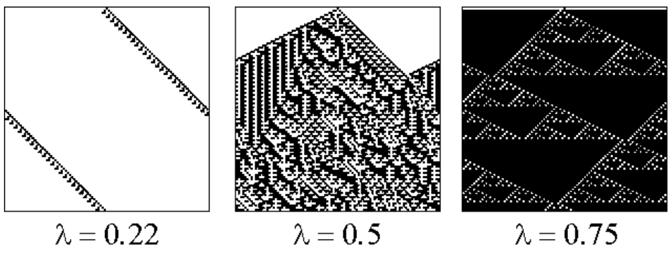


Fig. 3. Typical spatio-temporal patterns and λ parameter

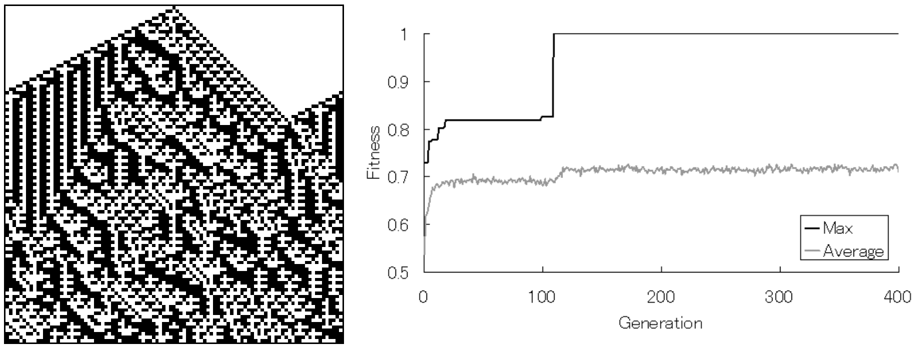


Fig. 4. Example of spatio-temporal pattern synthesized in the experiment. left: spatio-temporal pattern. right: Process of synthesis.

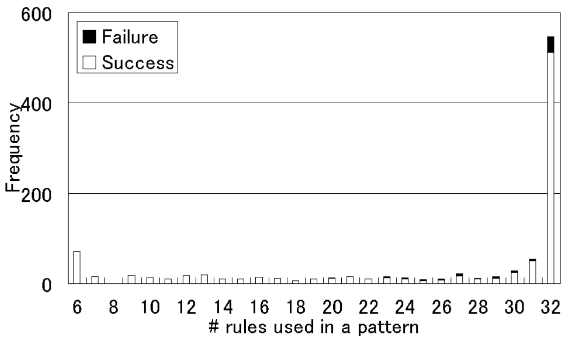


Fig. 5. Results of synthesis that uses this algorithm(trial 1000 times)

$$\lambda = 0.5 \text{ for 5-neighbor CA}$$

Also, let the number of cells be 99 and let time steps be 99. As the initial state, let a cell at $i = 49$ be black and all the other cells be white (-1). After trial-and-errors we have fixed parameters of GA: population size $K = 400$ probability of crossover $P_c = 0.9$, probability of mutation $P_m = 0.1$, threshold of masking $F_{th} = 0.7$ and maximum generation $G = 1000$. We then show results of two basic experiments.

4.1 Experiment 1

GA Fig. 4 left shows a spatio-temporal pattern generated artificially by a rule table generated randomly. We use it as a desired pattern virtually. Fig. 4 right shows the maximum and average fitness for a set of chromosome for each generation. At $g = 109$, the maximum fitness reaches 1.0 and we can obtain the desired pattern. Repeating similar experiments 1000 times we have summarized the results as shown in Fig. 5. Fig. 6 shows corresponding results from experiments without masking of genes. These results show that the masking improves

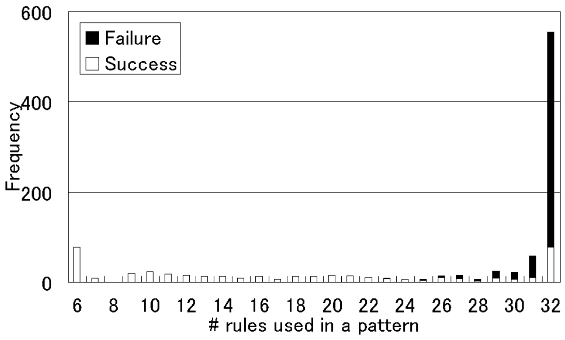


Fig. 6. Results of synthesis without masking(trial 1000 times)

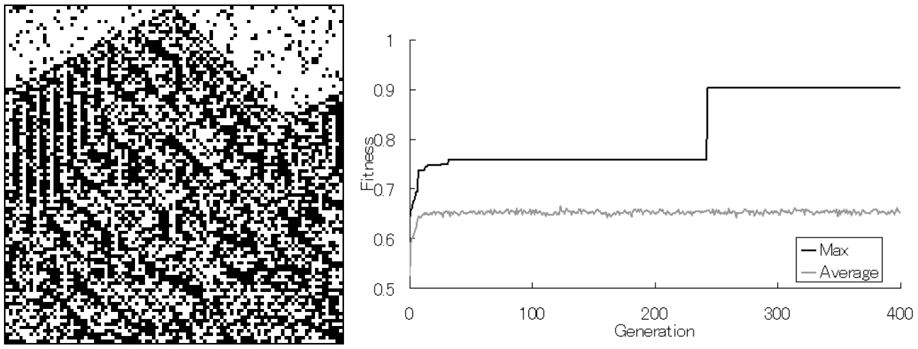


Fig. 7. Example of noise reduction. left: 10% noisy pattern. right: Process of reduction.

success rate dramatically for large number of rule used in a spatio-temporal pattern. The masking is very effective to increase the success rate and our algorithm have achieved over 95% success rate.

4.2 Experiment 2: Application to Noise Reduction

We consider noise reduction problem from a desired spatio-temporal pattern with noise. In the experiment, we invert pixels randomly in a pattern at the probability of 10%. For example, Fig. 7 left is given by adding noise to the spatio-temporal pattern Fig. 4. Applying our algorithm, the fitness increased for 1 and converged some constant value as shown in Fig. 7 right. The constant fitness $\neq 1$ is not consistent with the noisy pattern but original pattern: the original pattern can be reproduced. Repeating similar experiments 1000 times we have summarized the results as shown in Fig. 8. Even when the noise was added, we obtain an excellent results in about 90% success rate. It goes without saying that lower noise gives higher success rate.

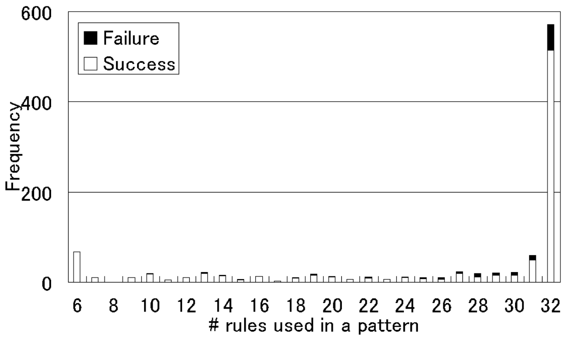


Fig. 8. Results of noise reduction(trial 1000 times)

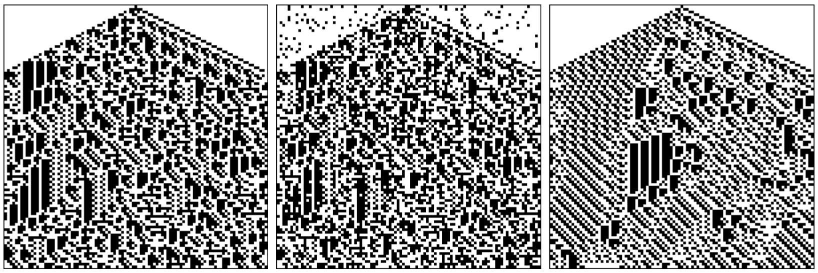


Fig. 9. An example of direct search. Left: Desired pattern. Center: Noisy pattern. Right: Result of direct search.

4.3 Comparison with Direct Search

The preceding two experiments suggest that our algorithm is effective to synthesize desired CAs, however, the algorithm should be compared with other methods. The simplest method is direct search: if we have a spatiotemporal pattern in which the number of neighbors is known and all the rules are used then we can construct a rule table by direct extraction from the pattern. However this direct search has problems for noisy patterns. When we extract rules from the noisy pattern, there must exist contradiction rules. In such a case it is natural to fix the rule based on majority principle. However, if one pixel is inverted as a noise in a pattern, the inverted pixel affects not only as an output but also as a cell of input. For example, 5-neighbor CA, one inverted pixel make one contradiction as output and at most 5 contradictions as a input cell: total 6 contradictions. Therefore it is hard to construct desired rule-tables from a noisy patters as suggested in an example in Fig. 9. As compared with this, the fitness of our GA-based algorithm measures difference between patterns and one noisy pixel affects only one pixel. This is the reason why our algorithm can construct a rule table for a desired pattern.

5 Conclusions

We have studied a GA-based search algorithm to synthesize desired BCA. Introducing masking to preserve suitable genes, our algorithm can find desired rule table for relatively complicated spatiotemporal patterns. The algorithm is effective also for noise reduction problem of desired patterns with random noise. Future problems include detailed analysis of learning process, application to larger problems, and FPGA-based hardware implementation for practical problems.

References

1. S. Wolfram, "Universality and Complexity in Cellular Automata, " Physica D, 10 pp. 1-35, 1984.
2. L. O. Chua, S. Yoon and R. Dogaru, " A nonlinear dynamics perspective of Wolfram's new kind of science. Part I: Threshold of complexity," Int. J. Bifurcation and Chaos, 12, pp. 2655-2766, 2002.
3. J. Y. Perrier, M. Sipper and J. Zahnd, " Toward a viable, self-reproducing universal computer," Physica D, 87, pp. 335-352, 1996
4. J. D. Lohn and J. A. Reggia, "Automatic discovery of self-replicating structures in cellular automata," IEEE Trans. Evolutionary Computation, 1, 3, pp. 165–178, 1997.
5. L. Bull, I. Lawson, A. Adamatzky and B. DeLacyCostello, B. (2005) " Towards predicting spatial complexity: a learning classifier system approach to cellular automata identification," Proc. of CEC, pp. 136-141, 2005.
6. M. Seredynski and P. Bouvry, "Block Cipher based on Reversible Cellular Automa, " Proc. of CEC, pp. 2138-2143, 2004
7. M. Wada, J. Kuroiwa and S. Nara, "Completely reproducible description of digital sound data with cellular automata," Phys. Lett. A 306, pp. 110–1158, 2002.
8. M. Wada, J. Kuroiwa and S. Nara, "Errorless reproduction of given pattern dynamics by means of cellular automata," Phys. Rev. E 68,036707, pp. 1–8, 2003.
9. H. Kajisha and T. Saito, "Synthesis of self-replication cellular automata using genetic algorithms," Proc. of IJCNN, V, pp. 173–177, 2000.
10. T. Yamamichi, T. Saito, K. Taguchi and H. Torikai, Synthesis of binary cellular automata based on binary neural networks, Proc. of IJCNN, pp. 1361 - 1364, 2005.

On Properties of Genetic Operators from a Network Analytical Viewpoint

Hiroyuki Funaya and Kazushi Ikeda

Department of Systems, Kyoto University
Kyoto 606-8501 Japan
funaya@sys.i.kyoto-u.ac.jp, kazushi@i.kyoto-u.ac.jp

Abstract. In recent years, network analysis has revealed that some real networks have the properties of small-world and/or scale-free networks. In this paper, a simple Genetic Algorithm (GA) is regarded as a network where each node and each edge respectively represent a population and the possibility of the transition between two nodes. The characteristic path length, which is one of the most popular criterion in small-world networks, is derived analytically. The results show how the crossover operation works in GAs to shorten the path length between two populations, compared to the length of the network with the mutation operation.

1 Introduction

There have been several theoretical results on the properties of genetic algorithms (GAs). One is the schemata theorem, which shows that a set of individuals (a schema) with a higher fitness on average is more likely to survive than a set with a lower fitness [1,2]. Another result, based on the information-theoretic method, regards a GA as a finite-state ergodic Markov chain and discusses the convergence to the optimal solution in asymptotics [3,4,5].

In this paper, we take another approach to the problem why GAs are good quasi-optimizers: We regard a GA as a network, where a node is a possible set of individuals, and investigate the connectivity of the network from a network analytical point of view. Network analysis has recently attracted much attention as a new method to analyze complex phenomena in the world, where the following two properties have been found in many real networks [6,7,8,9,10,11]: One is referred to as a small-world network, which means that a network simultaneously has dense local connections and short pairwise distances. The other is a scale-free network, which means that the distribution of the orders of nodes in a network has a long tail obeying the power law.

We shed light on the former property. That is, we analytically derive the characteristic path length (CPL) ν, defined as the shortest path length (SPL) between two nodes averaged over all possible pairs. Since it is expected that a GA with a smaller CPL takes a shorter time to find a solution, we see how the two basic genetic operations in GAs, crossover and mutation, affect the CPL.

In the following, we show how to construct a network from a GA and derive its CPL analytically, which is confirmed by numerical analyses.

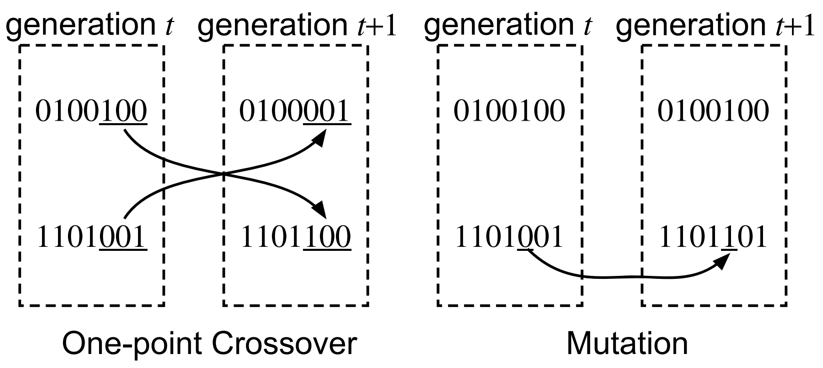


Fig. 1. Genetic operations: one-point crossover and mutation. An example with $L = 7$.

2 Simple Genetic Algorithms

Since the analytical derivation of CPL is difficult in general, we select the simplest GA formulated below from many variants of GAs proposed since the original GA was born [1,2].

Each individual consists of a binary sequence of length L. That is, we have 2^L kinds of individuals. A set of individuals defines a population. In this paper, we assume that each population has only two individuals. Then, the cardinality of the different populations becomes

$$N \equiv 2^{L-1}(2^L - 1). \tag{1}$$

When the generation proceeds, a population changes by one of the two basic genetic operations, one-point crossover or mutation. The former randomly chooses one crossover-point from $L-1$ candidates and exchanges the bits rightward from the point, while the latter randomly chooses one of $2L$ loci (or gene-positions) in the two individuals and inverts its bit from 0 to 1 or vice versa (Fig. 1).

Note that we do not treat any fitness function because we are only considering the possibility of population-transition from one in a generation to another in the next generation.

3 Network of a Simple GA

We regard a population as a node of a GA network. Hence, the network has N nodes. Two nodes are linked by an edge if and only if one of the two nodes can change to the other in a one-point crossover or mutation operation (Fig. 2). Obviously, the edge is undirected because the reverse change is also possible. Note that a node has $2L$ edges from the mutation operation and $L-1$ edges resulting from the crossover operation since an individual is a binary sequence of length L and a population consists of two individuals.

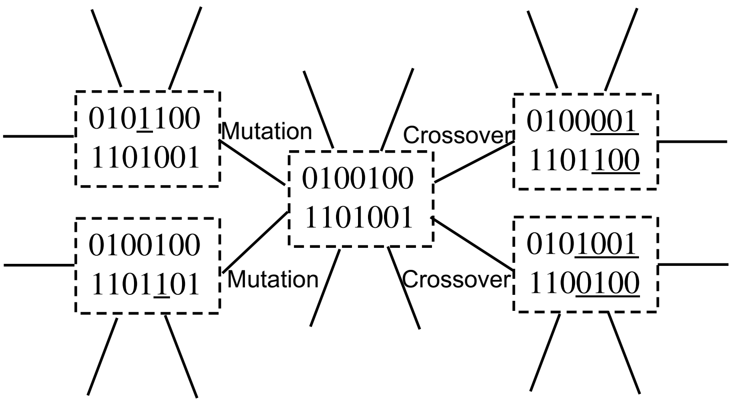


Fig. 2. A part of the network of a genetic algorithm

If a network consists of only the edges from mutation, it is a lattice of the $2L$-dimensional hypercube because an individual is an L-bit sequence and a population consists of two individuals. Therefore, the path-length of any two distinct nodes is the same as the Manhattan distance, that is, L on average. This means that we need L generations to reach a quasi-optimal population from an initial one.

On the contrary, the edges from crossover are shortcuts in the network where plural bits can change at once. It is likely that these shortcuts enable GAs to find a quasi-optimal solution in a short time. The purpose of this study is to evaluate quantitatively how these shortcuts work to shorten the CPL and to clarify their effects in GAs.

4 Four Types of Loci

In many cases of network analysis, the CPL is numerically calculated from the empirical data collected. However, the CPL of the GA network treated here can be derived analytically due to its simplicity, as shown below. Before derivation, we classify L loci into four types according to how two populations can be matched by genetic operations since neither crossover nor mutation change the locus of a gene (Fig. 3).

The first type is referred to as Type 1, where all four genes have the same alphabet, 0 or 1. That is, the genes at the locus are 0000 or 1111 and hence the cardinality is two. This means that the path length resulting from this locus is zero.

The second is Type 2, where the two genes of a population are the same but the two genes of the other population are different. That is, one of the genes at the locus is different from the others such as 0001, 0010, and so on, and hence the cardinality is eight. Because the crossover operation cannot change the alphabet of the former, mutation is necessary and the path length resulting from this locus is one.

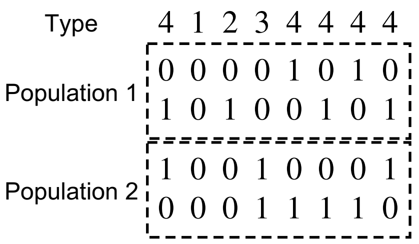


Fig. 3. An example of the types of loci in two nodes

The third is Type 3, where the two genes of each population are the same but the two populations have different alphabets. That is, the genes at the locus are 0011 or 1100, and hence the cardinality is two. Because the crossover operation cannot change the alphabet, mutation is necessary and the path length resulting from this locus is two, if the genes at the other loci are identical.

The last is Type 4, where each population has two different genes. That is, the genes at the locus are 0110, 1001, 0101 or 1010. The former two are termed Type 4-1 while the others Type 4-2. The path length in Type 4-2 is zero and that in Type 4-1 is two when either of crossover and mutation is applied only to match the bits at the locus. However, since the crossover operation can change plural bits at once, the shortest path of the network consisting of only the edges from mutation never exceeds that from crossover.

Note that the path length is calculated in Type 4 as if the individuals in a population are ordered. In the next subsection we discuss how this affects the CPL before deriving it.

5 Ordered Individuals in a Population

Suppose we add the left of the left-most locus to the crossover-points. Obviously, this does not affect the CPL ν because the edges corresponding to the crossover operation at the point make no sense to match the bits of two populations.

Let g_1 and g_2 be two individuals of a population (g_1, g_2). Then (g_1, g_2) and (g_2, g_1) represent the same node since individuals in a population are not ordered. We denote the CPL of the network by $\tilde{\nu}$ where individuals in a population are ordered. Then, the cardinality of the different populations becomes

$$\tilde{N} \equiv 2^{2L} \tag{2}$$

and $\tilde{\nu}$ is expressed as

$$\tilde{\nu} = \frac{2N\nu + L}{\tilde{N}} \tag{3}$$

since $N\nu$ represents the sum of the shortest path lengths of all distinct-node pairs and L that of all identical-node pairs. Hence

$$\tilde{\nu} = \nu\left(1 - 2^{-L}\right) + L2^{-2L} \tag{4}$$

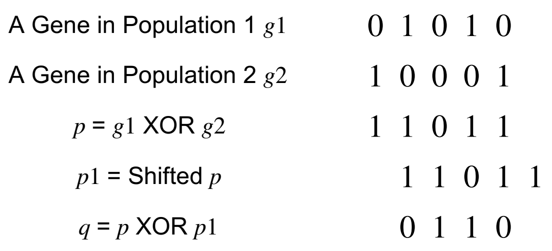

A Gene in Population 1 g1 0 1 0 1 0
A Gene in Population 2 g2 1 0 0 0 1
p = g1 XOR g2 1 1 0 1 1
p1 = Shifted p 1 1 0 1 1
q = p XOR p1 0 1 1 0

Fig. 4. How to calculate the SPL of two nodes consisting of the loci of Type 4

stands and $\tilde{\nu}$ and ν are one-to-one. Moreover, the difference is negligible when L is large. We therefore derive $\tilde{\nu}$ instead of ν in the following.

6 Shortest Path Length for Crossover Operations

The crossover operation cannot change the type of a locus and the mutation operation works bitwise. Therefore, the loci belonging to Types 1, 2 and 3 respectively contribute zero, one and two for the SPL no matter where they are located.

Hence, the SPL of two nodes is the sum of the above and the SPL of the two shorter nodes consisting of only the loci of Type 4. The latter for l-bit individuals can be calculated using the following procedure, as shown in Fig. 4:

1. Take l-bit XOR bitwise between an individual in a population and one in the other population and denote it by p.
2. Take $(l-1)$-bit XOR bitwise between p and 1-bit shifted p and denote it by q.
3. Count the number of 1s in q.

It is easily shown that the above procedure does not depend on which individual is chosen from a population because one individual is the bitwise-inverted binary sequence of the other in a population when all loci belong to Type 4.

7 Characteristic Path Length

The main idea for derivation is to count the number of node-pairs with the SPL M, instead of evaluating the SPL of each node-pair directly.

Let the numbers of Type 1, 2, 3 and 4 in L loci be denoted by l_1, l_2, l_3 and l_4, and the numbers of links in M by m_1, m_2, m_3 and m_4, respectively. Here,

$$L = l_1 + l_2 + l_3 + l_4, \tag{5}$$
$$M = m_1 + m_2 + m_3 + m_4, \tag{6}$$
$$m_1 = 0, \tag{7}$$
$$m_2 = l_2, \tag{8}$$
$$m_3 = 2l_3 \tag{9}$$

stand by definition.

First, we consider the case of $m_4 = 0$. Taking into account the number of combinations of positions and the cardinality of each type, the number of node-pairs satisfying (6) is written as

$$\frac{L! 2^{l_1} 8^{l_2} 2^{l_3}}{l_1! l_2! l_3!} \tag{10}$$

for fixed l_1, l_2 and l_3. Since they must satisfy

$$l_2 + 2l_3 = 0 \tag{11}$$
$$l_1 + l_2 + l_3 = L \tag{12}$$
$$0 \leq l_3 \leq \lfloor L/2 \rfloor \tag{13}$$

from (5) to (9), the ratio of the number of node-pairs to the possible pairs for $m_4 = 0$ is

$$\frac{1}{2^{4L}} \sum_{M=1}^{2L} \sum_{l_3=0}^{\lfloor L/2 \rfloor} \frac{M L! 2^{L+2M-4l_3}}{(L - M + l_3)!(M - 2l_3)! l_3!}, \tag{14}$$

which is denoted by $\tilde{\nu}_1$.

Next, we consider the case of $m_4 > 0$. For fixed l_1, l_2, l_3 and l_4, the number of combinations of positions is equal to

$$\frac{L!}{l_1! l_2! l_3! l_4!} \tag{15}$$

and the number of combinations of places where crossover occurs is ${}_{l_4-1}\mathrm{C}_{m_4}$. Taking into account the cardinality of Type 4 and the possibility that the left-most in the Type 4 loci belongs to Type 4-1 or Type 4-2, the total number of node-pairs is written as

$$\frac{L! 2^{l_1} 8^{l_2} 2^{l_3} 2^{l_4+1} {}_{l_4-1}\mathrm{C}_{m_4}}{l_1! l_2! l_3! l_4!}. \tag{16}$$

Summing up for all possible combinations of l_1, l_2, l_3, l_4 and m_4 under the condition

$$l_1 + l_2 + l_3 + l_4 = L, \tag{17}$$
$$m_1 + m_2 + m_3 + m_4 = M \tag{18}$$

where $m_1 = 0$, $m_2 = l_2$, $m_3 = 2l_3$ and $m_4 \leq l_4 - 1$, the ratio of the number of node-pairs to the possible pairs is written as

$$\frac{1}{2^{4L}} \sum_{M=1}^{2L} \sum_{l_1,l_2,l_3,l_4,m_4} \frac{M L! 2^{l_1} 8^{l_2} 2^{l_3} 2^{l_4+1} {}_{l_4-1}\mathrm{C}_{m_4}}{l_1! l_2! l_3! l_4!}, \tag{19}$$

which is denoted by $\tilde{\nu}_2$.

The complete expression for the CPL is

$$\tilde{\nu} = \tilde{\nu}_1 + \tilde{\nu}_2. \tag{20}$$

8 Numerical Analyses

Table 1 shows $\hat{\nu}$ for some L's derived from (20). These values coincide with those numerically calculated from actual networks. Note that since the CPL of the network consisting of only the edges from mutation is L, the ratio shows how the crossover operation shortens the CPL.

Table 1. The CPL $\hat{\nu}$ for some L's

L	$\hat{\nu}$	ratio
3	2.3359	0.7786
8	6.5501	0.8188
13	10.887	0.8375
18	15.253	0.8474
23	19.626	0.8533
28	24.000	0.8571
33	28.375	0.8598
38	32.750	0.8618
43	37.125	0.8634
48	41.500	0.8646
53	45.875	0.8656
58	50.250	0.8664
63	54.625	0.8671
68	59.000	0.8676

9 Conclusions and Discussions

We regarded a simple GA as a network and derived its CPL analytically. The result shows how the crossover operation works to shorten the CPL of a network

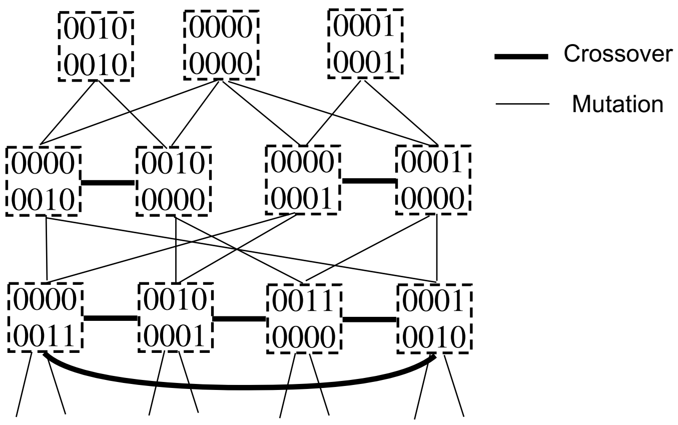


Fig. 5. A schematic view of a GA network. Only the edges from mutation link the nodes belonging to different rows. The nodes in the first and second rows are not connected by the edges from crossover.

consisting of only the edges from mutation. In short, even when the crossover operation is applied, the CPL is not so small, and the same order $O(L)$ as in the case when only the mutation operation is employed.

We see that the CPL is rather large even when the edges from crossover are added. This may be because the network is foliated into slices between which the mutation operation must be applied, as the crossover operation cannot change the type of a locus as shown in Fig. 5. Moreover, some of such slices are divided into small parts which are not linked by the edges from crossover as seen in the first and second rows in Fig. 5.

As everyone knows, the result in this analysis is preliminary for the following two reasons: One is that the population used here has only two individuals. It seems possible to extend the results to more general cases where a population has more than two individuals in a similar way. The other is that the selection pressures of the GA were neglected. These can be introduced by cutting links with a small transition probability, although this operation will make it difficult to calculate the CPL analytically. When these matters are investigated in the future, the results will give some insight into designing effective GAs.

References

1. Holland, J.H.: Adaptation in Natural and Artificial Systems. Univ. Michigan Press (1975)
2. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Pub. (1989)
3. Suzuki, J.: A markov chain analysis on simple genetic algorithms. IEEE Trans. on Systems, Man and Cybernetics **25**(4) (1995) 655–659
4. Suzuki, J.: A further result on the markov chain model of genetic algorithms and its application to a simulated annealing-like strategy. IEEE Trans. on Systems, Man and Cybernetics, Part B **28**(1) (1998) 95–102
5. Mühlenbein, H., Höns, R.: The estimation of distributions and the minimum relative entropy principle. Evolutionary Computation **13**(1) (2005) 1–27
6. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature **393** (1998) 440–442
7. Watts, D.J.: Small Worlds: The Dynamics of Networks between Order and Randomness. Princeton Univ. Press, Princeton, NJ (1999)
8. Barabasi, A.L., Albert, R.: Emergence of scaling in random networks. Science **286** (1999) 509–512
9. Callaway, D.S., Newmann, M.E.J., Strogatz, S.H., Watts, D.J.: Network robustness and fragility: Percolation on random graphs. Physical Review Letters **85** (2000) 5468–5471
10. Strogatz, S.H.: Exploring complex networks. Nature **410** (2001) 268–276
11. Albert, R., Barabasi, A.L.: Statistical mechanics of complex networks. Review of Modern Physics **74** (2002) 47–97

SDMOGA: A New Multi-objective Genetic Algorithm Based on Objective Space Divided

Wangshu Yao[1], Chen Shifu[2], and Chen Zhaoqian[2]

[1] School of Computer Science & Technology, Soochow University
Suzhou Jiangsu 215006, China
wshyao@suda.edu.cn
[2] National Laboratory for Novel Software Technology Nanjing University
Nanjing 210093, China

Abstract. Most contemporary multi-objective evolutionary algorithms (MOEAs) have high computational demand. In this paper, a new MOEA based on objective space divided named SDMOGA is proposed. SDMOGA transforms the Pareto ranking into the sum of interval index ranking among individuals in objective space divided, and uses a method of individual crowding operator similar to adaptive grid to keep population diversity. Experimental results on four nicely balance functions show that SDMOGA has high efficiency, low run-time complexity and good convergence.

1 Introduction

Most real-world engineering optimization problems are multi-objective in nature, since they normally have several (usually conflicting) objectives that must be satisfied at the same time. These problems are known as multi-objective optimization problems(MOP)[1,2] in contrast with single-objective optimization problems(SOP).Because there are several objectives functions to be optimized at the same time in MOP, there is no unique solution instead of all of the good trade-off solutions available(the called Pareto optimal set) to MOP.

Evolutionary Algorithms(EAs) seem also particularly desirable for solving MOP because they deal simultaneously with a set of possible solutions(the called population) which allows us to find several members of the Pareto optimal set in a single run of the algorithm, instead of having to perform a series of separate runs as in the case of the traditional mathematical programming techniques. Additionally, evolutionary algorithms are less susceptible to the shape or continuity of the Pareto front (e. g., they can easily deal with discontinuous and concave Pareto fronts).

In this paper, an efficient and simple multi-objective genetic algorithm based on objective space divided (named as SDMOGA for short) is proposed. The remainder of this paper is organized as follows. Section 2 introduces the background of SDMOGA. Section 3 introduces the SDOMGA in details. Section 4 and section 5 respectively presents the experimental results and the concluding remarks.

2 Background

After the first implementation of a multi-objective evolutionary algorithm in the mid-1980s[3], a considerable amount of multi-objective evolutionary algorithms (MOEAs) have been developed[5,7]. Coello[8] reviews some of the most popular MOEAs reported in the literatures, indicating some of their main application, their advantages and disadvantages. The common defect of these algorithms is high runtime complexity. This is partly due to the fact that MOEAs based on Pareto ranking requires that each solution be compared to a large number of other solutions and time-consuming methods to keep population diversity. Another explanation lies on the fact that often MOEA research has disregarded run-time complexity. However, reducing the run-time complexity of MOEAs is very important for MOEAs' application. Coello[9] thinks that parallelism MOEAs and use of more efficient data structures are two future challenges. Both tasks reduce the run-time complexity of MOEAs.

Most contemporary MOEAs are based on two kinds of Pareto ranking[2] methods. One proposed by Goldberg assigns Pareto optimal solutions the same rank and other solutions some less desirable rank. Another proposed by Fonseca and Fleming ranks a solution according to the number of solutions dominating it. Both Pareto ranking methods require that each solution be compared to a large number of other solutions. Their common feature is that all Pareto optimal solutions have a higher probability to be selected as individuals of next generation than dominated ones.

3 The Multi-objective Genetic Algorithm Based on Objective Space Divided

The main work of general MOEAs contains two parts, identifying the Pareto solutions from the population and convergence to the Pareto optimal front, and then distributing the solution on the Pareto optimal frontier uniformly. The multi-objective genetic algorithm based on objective space divided (SDMOGA) also focuses on the two tasks, but it reduces run-time complexity of MOEA.

First, SDMOGA divides the current objective space so that every individual locates in someone subspace. Second, it deletes some individuals in order that the number of individual in every subspace is not more than one. Third, it assigns fitness value to every individual according the sum of subspace index. Last, it selects next population according the index and run evolution operator for next population. The main operators of SDMOGA are described in detail as follows.

3.1 Dividing Objective Space Operator

In order to divide objective space, some concepts are defined as follows:

Definition 1 (Maximum Objective Vector,MAXOV). For a given population,the maximum objective vector(MAXOV) is defined as:

$$MAXOV = (maxobj_1, maxobj_2......maxobj_k) \tag{1}$$

$$maxobj_j = \max_{ind_i \in Pop} \{ind_i^j\} \qquad j = 1, 2, \ldots, k \tag{2}$$

Where the *Pop* is the current population, ind_i^j is the jth objective value of the ith individual in the current population.

Definition 2 (Minimum Objective Vector, MINOV). For a given population, the Minimum objective vector (MINOV) is defined as:

$$MINOV = (minobj_1, minobj_2......minobj_k) \tag{3}$$

$$minobj_j = \min_{ind_i \in Pop} \{ind_i^j\} \qquad j = 1, 2, \ldots, k \tag{4}$$

Where the *Pop* is the current population, ind_i^j is the jth objective value of the ith individual in the current population.

Definition 3 (Current Objective Space). The current objective space is the area that is bounded by MAXOV and MINOV.

Definition 4 (Dividing Interval,DIVI). The dividing interval of every objective in objective space is defines as:

$$N = \alpha + \lceil 1.01^{\beta} \rceil \tag{5}$$

$$DIVI = (MAXOV - MINOV)/N \tag{6}$$

Where α of equation (5) is a parameter that is generally set a small number between three and seven. because the population is far away from the Pareto front of multi-objective optimization problem. The β of equation (5) is a parameter that is the times of dividing objective space. The symbol $\lceil x \rceil$ is the minimum integer that isn't more than x.

Definition 5 (Dividing Objective Space, DIVOS). The dividing objective space is the current objective space divided according to equation (5) and (6).

To reduce the run time, SDMOGA only runs the dividing objective space operator when the number of individuals locating at the outside of the current objective space is more than a parameter *max_out_num* set by users.If someone objective value of an individual is less than the same objective value of MINOV or more than the same objective value of MAXOV,SDMOGA considers this individual locates at the outside of the current objective space.

3.2 Crowding Operator

After dividing the current objective space, all the individuals of the population locate at different subspaces. However, the dividing objective space operator does not ensure that there is only an individual in a subspace. SDMOGA requires that each subspace only has an individual and the redundant individuals must be deleted. the crowding operator is designed to complete this task.

The crowding operator contains two steps. The first step calculates the distance between all individuals and the origin in the subspace. The origin of a subspace is a boundary point that all objective value is minimal in all boundary points.The second step selects the individual that the distance is minimal and deletes other ones.

Figure 1(a) shows the distribution of individuals at someone generation in MOEAs. MOEAs based on Pareto ranking select all individuals in the first area and one individual in the second area. However, the individuals in the first area are very crowd. In this paper, SDMOGA thinks the crowding nondominated individuals are worse than the dominated ones, which are diversity and dominated by a few individuals, to keep population diversity. For example some individuals of the first area are worse than those of the second area in figure 1(a). SDMOGA deletes some crowding nondominated individuals by dividing objective space and crowding operator in order to select some dominated ones. For example, in figure 1(b) some dominated ones in the second area are selected as individuals of next generation after some nondominated individuals in the first area are deleted.

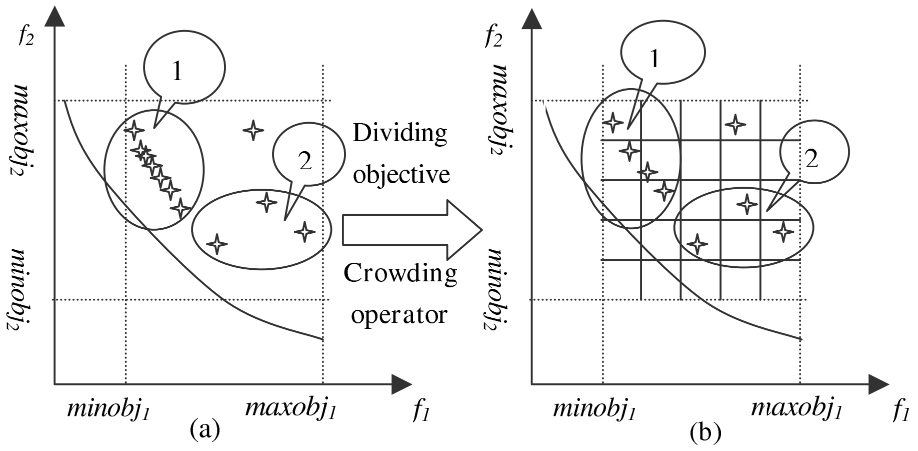


Fig. 1. Dividing objective space and crowding operator

3.3 Selection Operator Based on the Sum of Subspace Index

After dividing objective space and crowding operator, SDMOGA computes the subspace index of individual according to equation (7) and (8).

$$index_i^j = \lceil (ind_i^j - MINOV^j)/DIVI^j \rceil \tag{7}$$

$$SumIndex_i = \sum_{i=1}^{k} index_i^j \tag{8}$$

Where the ind_i^j is the jth objective value of the ith individual. The $MINOV^j$ is the jth value of the MINOV. The $index_i^j$ is the jth objective index of the ith individual. The k is the number of objective. The $SumIndex_i$ is the sum index of the ith individual.

General MOEAs based on Pareto ranking always first select the nondominated individuals as individual of next generation. These MOEAs give a higher probability to the nondominated individuals than dominated ones. However, some crowding individuals are deleted during the step of crowding operator in SDMOGA. After dividing objective space and crowding operator, the population is diversity. Now the nondominated individuals must first be selected to the population of next generation. We design a selection operator based on the sum of subspace index instead of Pareto ranking for SDMOGA. The method also ensures that the nondominated individuals are first selected.

Figure 2 shows the distribution of individuals of a two objective optimization problem after dividing objective space and crowding operator. In figure 2, the subspace index of the seven individuals is respectively (2,1),(1,2),(3,2),(2,3), (4,3),(3,4) and (3,1). The sum of their subspace index is respectively 3, 3, 5, 5, 7, 7 and 4. The order of the sum is 1, 2, 7, 4, 3, 5 and 6. However, according to the number of individuals dominating them, the result of sorting the seven individuals is also 1,2,7,4,3,5 and 6. This indicates that the selection operator based on the sum of their subspace index is as the same as the selection operator based on Pareto ranking.

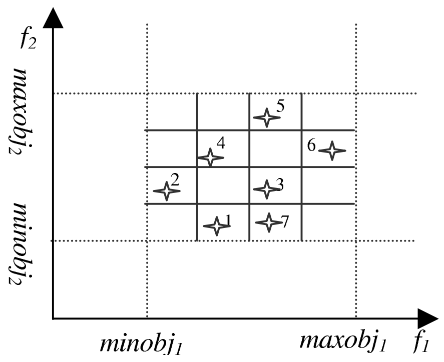


Fig. 2. The distribution of individual

There is a case that several individuals have the same sum of subspace index and the selection operator only selects part ones. We design a method that is adjusted from the partitioned quasi-random selection(PQRS)[10] to complete this task. The method first finds all the individuals that have the same sum of subspace index. The second step selects an individual that has the minimum subspace index of one objective and then selects the second individual that has the minimum subspace index of another objective. The second step runs repetitively until algorithm selects enough individuals.

3.4 The Flow of SDMOGA

The flow of SDMOGA is as follow:

Alogrithm : (SDMOGA Main Loop)
Input: *Popsize (population size)*
maxgen (maximum number of generations)
Output: *Pop (nondominated set)*

Step 1: **Initialization:** Generate an initial population *Pop* and initialize all parameter
Step 2: **Calculation:** Calculate the objective value of all individuals.
Step 3: **Termination:** If stopping criterion is satisfied then go to Step 8, otherwise go to Step 4.
Step 4: **Dividing objective space:** Divide the current objective space according to equation (5) and (6).
Step 5: **crowding individual:** Run Crowding operator for all individuals.
Step 6: **selection:** Select next generation population based on the sum of subspace index.
Step 7: **Evolution operator:** Run evolution operator for current population, then go to Step 2.
Step 8: **Output:** Out the *Pop* that is nondominated population.

The main comparisons of SDMOGA locate at the crowding operator. In the worst case,this operator requires $O(N)$ comparison for all solutions in the population, where N is the size of population. The case is that all new solutions locate at one subspace, but the probability of this case is almost zero. The run-time complexity of SDMOGA is $O(N)$, where N is the size of population. It is much less than the run-time complexity $O(mN^2)$ of general MOEAs. The storage complexity is $O(mN)$, where N is the size of population and m is the number of optimal objective. The comparisons of the computational complexity of the SDMOGA and other two popular algorithms are summarized in table 1, which all assume in the worst case.

Table 1. Computational complexity of three algorithms

	NSGA-II	SPEA2[1]	SDMOGA
Run-time complexity	$O(mN^2)$	$O(M^3)$	$O(N)$
Storage complexity	$O(mN^2)$	$O(mN)$	$O(mN)$

[1]Where M is the sum of the size of current population and archive population.

4 Experiment

In this section,we apply SDMOGA to four nicely balanced test functions,which are designed by Deb in 1999[11]. The four test functions are constructed according to formula (9).

$$\begin{array}{ll} \textit{Minimize} & T(X) = (f_1(x_1), f_2(X)) \\ \textit{Subject to} & f_2(X) = g(x_2, x_3, \ldots, x_m) h(f_1(x_1), g(x_2, x_3, \ldots, x_m)) \\ \textit{Where} & X = (x_1, x_2, \ldots, x_m) \end{array} \quad (9)$$

The four test functions are as table 2.

Table 2. Four test functions

Name	Function define	Remarks
ZDT1	$f_1(x_1) = x_1$ $g(x_2, \ldots, x_m) = 1 + 9 * \sum_{i=2}^{m} x_i/(m-1)$ $h(f_1, g) = 1 - \sqrt{f_1/g}$	$m = 30$ $x_i \in [0, 1]$
ZDT2	$f_1(x_1) = x_1$ $g(x_2, \ldots, x_m) = 1 + 9 * \sum_{i=2}^{m} x_i/(m-1)$ $h(f_1, g) = 1 - (f_1/g)^2$	$m = 30$ $x_i \in [0, 1]$
ZDT3	$f_1(x_1) = x_1$ $g(x_2, \ldots, x_m) = 1 + 9 * \sum_{i=2}^{m} x_i/(m-1)$ $h(f_1, g) = 1 - \sqrt{f_1/g} - (f_1/g) * \sin(10\pi f_1)$	$m = 30$ $x_i \in [0, 1]$
ZDT4	$f_1(x_1) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$ $g(x_2, \ldots, x_m) = 1 + 9 * \left((\sum_{i=2}^{m} x_i)/(m-1)\right)^{0.25}$ $h(f_1, g) = 1 - (f_1/g)^2$	$m = 30$ $x_i \in [0, 1]$

We compare the run-time of SDMOGA, NSGA-II[6] and SPEA2[4]. For all test functions and algorithms, we set the parameters as follows. The size of population is 50. The mutation probability is 0.05 and the crossover probability is 0.9. The maximum generation is 1000. All programs are programmed based on PISA that is designed by Bleulur[12]. The program of SPEA2 and NSGA-II is copied from PISA. All program are run in PC computer with CPU 133MHZ and memory 128M.

The run-time of three algorithms is summarized in table3, where the unit is millisecond.

In table 3, the run-time of SMDOGA is the least among ones of three algorithms. This shows that the SDMOGA reduces the run-time of MOEAs and improves the efficiency by transforming the Pareto ranking into the sum of subspace index ranking.

The results of four function optimized by three MOEAs are as figure 3, 4, 5 and 6.

The programs of NSGA-II and SPEA2 are the source programs on PISA. The results of SDMOGA optimizing four functions show that SDMOGA has a good convergence performance and maintains a widely distributed set of solutions on Pareto front. Figure 3, 4, 5 and 6 respectively show the results of three MOEAs optimizing four functions. They also show three MOEAs, SDMOGA, NSGA-II and SPEA2, have almost the same convergence performance and distributed performance of solutions.

Table 3. the run-time of three algorithms

	ZDT1	ZDT2	ZDT3	ZDT4
SPEA2	20439	15866	20427	20346
NSGA-II	10489	9884	9934	10080
SDMOGA	2078	3533	2173	170

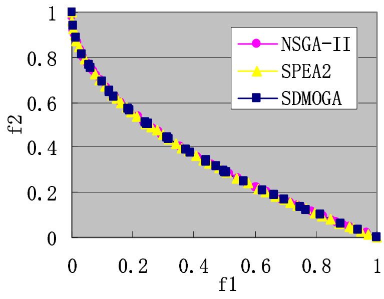


Fig. 3. The result of function ZDT1

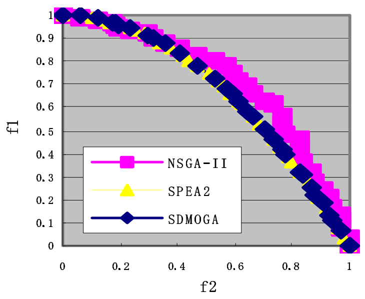


Fig. 4. The result of function ZDT2

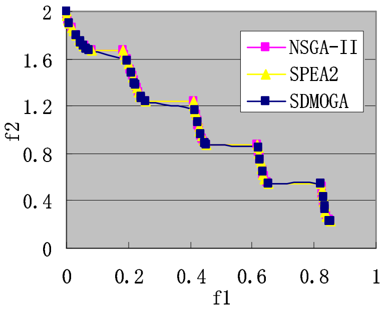


Fig. 5. The result of function ZDT3

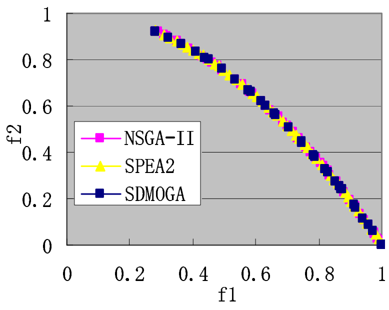


Fig. 6. The result of function ZDT4

5 Conclusion

In this paper, we proposed a fast and efficient multi-objective optimization genetic algorithm based on objective space divided. It has been proved both on the views of experiment results and computational complexity analysis that this approach could find the Pareto optimal solutions and obtain well distribution on the Pareto optimal front fast. Four test functions are used to test this approach, which the final simulation results proved that SDMOGA is not bad or better than other methods in finding better distribution on the Pareto optimal front. It is foreseeable that SDMOGA should be promising and find increasing attention and application in future.

Acknowledgements. This work was supported by the computer information processing new technology item of emphasis building subject of "211 project" Soochow University.

References

1. C.A.C.Coello. A Short Tutorial on Evolutionary Multiobjective Optimization. First International Conference on Evolutionary Multi-Criterion Optimization, editor: Eckart Zitzler and Kalyanmoy Deb and Lothar Thiele and Carlos A. Coello Coello and David Corne, Springer-Verlag. Lecture Notes in Computer Science. (2001), 21-40.
2. D.A.V.Veldhuizen,G.B.Lamont. Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art. Evolutionary Computation. (2000), 8(2), 125-147.
3. J.D.Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms, Lawrence Erlbaum, (1985) 93-100.
4. E.Zitzler,M.Laumanns, L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Computer Engineering and Networks Laboratory(TIK), Swiss Federal Institute of Technology(ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich, Switzerland, May 2001.
5. J.D.Knowles,D.W.Corne. Approximating the nondominated front using the pareto archived evolution strategy. Evolutionary Computation, 2000, 8(2), 149-172.
6. K.Deb,A.Pratap,S.Agarwal,T.Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation, April 2002, 6(2), pp: 182 197.
7. C.A.C.Coello,G.T.Pulido. A Micro-Genetic Algorithm for Multiobjective Optimization. In Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele, Carlos A. Coello Coello, and David Corne, editors, First International Conference on Evolutionary Multi-Criterion Optimization, Springer-Verlag. Lecture Notes in Computer Science. 2001, 126-140.
8. C.A.C.Coello. An Updated Survey of Evolutionary Multiobjective Optimization Techniques :State of the Art and Future Trends. In 1999 Congress on Evolutionary Computation, Washington, D.C., July 1999, Vol. 1, 3-13.
9. C.A.C.Coello. Evolutionary Multiobjective Optimization: Current and Future Challenged. in Jose Benitez, Oscar Cordon, Frank Hoffmann and Rajkumar Roy (editors), Advances in Soft Computing—Engineering, Design and Manufacturing, Springer-Verlag, ISBN 1-85233-755-9, September 2003, 243-256.
10. J.E.Fieldsend,R.M.Everson,S.Singh. Using Unconstrained Elite Archives for Multi-Objective Optimisation. TIEEE Transactions on Evolutionary ComputationT, June 2003, Vol. 7, No. 3, 305-323.
11. K.Deb. Multi-Objective genetic algorithms: Problem difficulties and construction of test problems. Evolutionary Computation. 1999, 7(30), 205-230.
12. S.Bleuler,M.Laumanns,L.Thiele,E.Zitzler. PISA - A Platform and Programming Language Independent Interface for Search Algorithms. in Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb and Lothar Thiele (editors), Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003, Springer. Lecture Notes in Computer Science. Faro, Portugal, April 2003, Volume 2632, 494-508.

Hamming Sphere Solution Space Based Genetic Multi-user Detection

Lili Lin

College of Information and Electronic Engineering, Zhejiang Gongshang University, Hangzhou, Zhejiang, 310035, China
sunshine@hzcnc.com

Abstract. Many researches on genetic algorithm based multi-user detection indicate initial population has crucial effects on the performance of detectors. Commonly used method to obtain initial population is to perturb the input chromosome randomly, which fails to fully exploit the effective information delivered by input chromosome. This paper proposes a kind of Hamming Sphere Solution Space based Genetic Multi-User Detector (HSSSGMUD), which constructs the initial population in a simple but effective manner. Firstly, select the input chromosome, and regard it as the center of a sphere in PK dimensions space, where P is data packet length and K is user numbers in Code Division Multiple Access (CDMA) system. Then, the concept of Hamming sphere space is used to obtain other chromosomes of initial population. Simulation results show the proposed HSSSGMUD not only achieves lower Bit Error Ratio (BER) and better near-far resistant ability, but also converges quickly.

1 Introduction

In the last two decades, multi-user detection has been a hot research topic for its potential to alleviate multiple access interference and resist the near-far effect in Code Division Multiple Access (CDMA) system. Since the computation complexity of the optimum multi-user detector (OMD) increases exponentially with the number of users [1], researchers have devoted themselves to finding sub-optimum detectors that can achieve better performance with less computation complexity [2], [3], [4].

Recently, many researchers have proposed Genetic Algorithm based Multi-User Detection (GAMUD) schemes [5], [6], [7] [8], [9]. Juntti firstly proposed the GAMUD for synchronous CDMA system in AWGM channels, and he pointed out it is necessary to provide better initial population for genetic algorithm to achieve better detection results [5]. The GAMUD proposed by Yen K and Hanzo [8] can obtain near single user performance. In their method, a local search is performed ahead, and then genetic algorithm is used to evolve the local search results. To improve the convergence speed of genetic algorithm, Ergün C and Hacioglu K [7] took the multistage detector as a "genetic operator" to process the detection results further at each generation, with the aim to provide a better solution assembly for next generation, and thus to accelerate the convergence speed. Evidently, better initial or medium-term population is an important premise for GAMUD to yield promising results.

According to this train of thoughts, this paper proposes a Hamming sphere solution space based genetic multi-user detector (HSSSGMUD), which is distinguished for using the concept of sphere solution space to construct the initial population. This new initial population construction method can fully exploit the effective information embedded in the input chromosome, that ensures a fast convergence speed and excellent detection performance.

2 System Model

Assume there have K active users in the asynchronous CDMA system, and each user employs the binary phase shift keying (BPSK) modulation, then baseband received signal can be written as

$$r(t) = \sum_{i=1}^{P} \sum_{k=1}^{K} A_k b_k^{(i)} s_k(t - iT_b - \tau_k) + n(t), \tag{1}$$

where A_k, $b_k^{(i)} \in \{-1, 1\}$, $s_k(t)$ and $\tau_k \in [0, T_b)$are the signal amplitude, ith bit, signature waveform and time delay of the kth user, respectively. T_b is bit interval duration, P is the length of data packet, and $n(t)$ is the gauss white noise with two-side power spectrum density $N_0/2$. Without loss of generality, the time delays of each user are assumed to satisfy the followed relation, $0 \leq \tau_1 \leq \tau_2 \leq ... \leq \tau_k < T_b$.

In the Conventional Detector (CD), the received signal $r(t)$ is passed through a bank of filters matched to the signature waveforms of each user, and then the outputs of matched filters are sampled at symbol rate. So the matched filters output vector is:

$$\mathbf{y} = \mathbf{HAb} + \mathbf{n}, \tag{2}$$

where $\mathbf{H} \in R^{PK \times PK}$ is the correlation matrix of signature waveforms, $\mathbf{A}$ is the diagonal matrix composed of signal amplitude, $\mathbf{b}$ and $\mathbf{n}$ are K-vectors of information bits and noise, respectively.

The OMD is a kind of maximum-likelihood estimate algorithm, which can achieve the theoretical minimum Bit Error Ratio (BER) value. The optimum detection result satisfies the following expression [1]

$$\hat{\mathbf{b}}_{OMD} = \arg\left\{ \max_{\mathbf{b} \in \{-1,+1\}^{PK}} (2\mathbf{y}^T\mathbf{b} - \mathbf{b}^T\mathbf{Hb}) \right\}. \tag{3}$$

Since the computation complexity of OMD increases with the number of users, most research efforts have been focused on suboptimum multi-user detectors.

Decorrelating (DEC) detector [10], [11], [12] is one of the most popular linear detectors, which uses the inverse of correlation matrix ($\mathbf{H}^{-1}$) to remove multiple access interference. The output of DEC detector can be expressed as

$$\mathbf{b}_{DEC} = sign(\mathbf{H}^{-1}\mathbf{y}) = sign(\mathbf{Ab} + \mathbf{H}^{-1}\mathbf{n}). \tag{4}$$

DEC detector can achieve optimum near-far resistance, but fail to achieve the minimum BER because it causes noise enhancement.

3 Hamming Sphere Solution Space Based Genetic Multi-User Detection

3.1 Genetic Algorithm [13], [14]

Genetic algorithm (GA) was inspired by natural selection and genetic mechanism of biology. It is a kind of massively parallel, random and self-adaptive searching algorithm, which is characterized by the strategy of population searching, and the information exchange between different chromosomes of the population. Thus, GA can solve many problems that are beyond the ability of conventional algorithms, such as chaos problems, random problems and non-linear dynamic problems.

GA starts from a population consists of a fixed number of chromosomes, and each chromosome corresponds to a possible solution of the problem to be solved. In each generation, all chromosomes in the population are evolved, and then are evaluated according to the fitness function. Generally, the chromosomes with high fitness values have more chances to be selected as parent chromosomes to give birth to offspring chromosomes by crossover and mutation. The newly produced offspring chromosomes compose a new population. Then the new population repeats the above steps until a certain number of generations, or a satisfactory solution is produced.

3.2 Hamming Sphere Solution Space Based Population Initialization

As described in Section 1, the performance of GAMUD depends heavily on the selection or construction of initial population. So far as the GAMUDs in the literatures the author can find are concerned, the initial populations are obtained by the method of perturbing the input chromosome randomly, which usually fails to exploit the useful information delivered by the input chromosome. In this paper, the author proposes a simple but effective construction method of initial population, which can make full use of the effective information embedded in the input chromosome, and ensure excellent detection performance of GAMUD.

Firstly, the output of a detector, such as the DEC detector, minimum mean square error detector, multistage detector, interference cancellation detector and etc, is selected as the input chromosome $\mathbf{b}_{Input}$of genetic algorithm. Then, the rest chromosomes $\mathbf{b}_{Initial}$ in the initial population are selected according to the following equation

$$S_d(\mathbf{b}_{Input}) = \left\{ \mathbf{b}_{Initial} \in \{-1, +1\}^{PK} \, | \| \mathbf{b}_{Initial} - \mathbf{b}_{Input} \| \leq d \right\} . \qquad (5)$$

where $S_d(\mathbf{b}_{Input})$is the Hamming sphere solution space with the center $\mathbf{b}_{Input}$ and radius d, PK is the dimension of the sphere solution space. Therefore, the

Hamming distance between the input chromosome $\mathbf{b}_{Input}$ and the other selected initial chromosomes $\mathbf{b}_{Initial}$ is d.

In this paper, we focus on d=1. Then, the Hamming distance between the input chromosome and the other chromosomes in the initial population is 1, and the maximum Hamming distance between arbitrary two chromosomes is 2. Since the input chromosome is already a better solution, the initial population constructed by the aforementioned method can exploit the effective information embedded in the input chromosome more completely than the initial population obtained by randomly perturbing all the genes of the input chromosome. So the Hamming sphere space based initial population ensures the subsequent evolution towards the advantaged direction.

3.3 Hamming Sphere Solution Space Based Genetic Multi-User Detection (HSSSGMUD)

The implementation of HSSSGMUD consists of the followed procedures.

(1) Encoding

Binary encoding and float encoding are two approaches to transform the possible solution to chromosome. Binary encoding is used widely for its simplicity and systematic theories. In binary encoding, each gene in the chromosome has only two possible values, +1 or –1. In multi-user detection, since the binary solution vector $\mathbf{b}$ of length PK is a chromosome by origin, there is no need to encode.

(2) Hamming Sphere Space Based Population Initialization

The number of chromosomes in the population is denoted as population size N_p, which depends on the complexity of the problem to be solved. To keep the balance between the variety of population and the computing time, the value of N_p should be moderate. In this paper, N_p is set to 20.

The initial population can be constructed by selecting several chromosomes from the solution space randomly. Generally, prior knowledge is incorporated in the construction of initial population to provide a better start for the evolution.

In this paper, the output of DEC detector is selected as the input chromosome, i.e., $\mathbf{b}_{DEC}= \mathbf{b}_{Input}$. Using the Hamming sphere space based initial population discribed in Section 3.2, and let the radius of sphere space d=1, the rest N_p-1 chromosomes in the initial population can be obtained.

If the radius of sphere space d=PK, the Hamming sphere space based population initialization is equivalent to the completely random population initialization.

(3) Establishment of Fitness Function

To assess the performance of chromosomes, it is necessary to establish a non-negative fitness function. The better the chromosome is, the higher its fitness value should be. From the output of OMD in equation (3), it is easy to find the purpose of multi-user detection is to maximize the followed cost function,

$$c(\mathbf{b}) = 2\mathbf{y}^T\mathbf{b} - \mathbf{b}^T\mathbf{H}\mathbf{b}. \qquad (6)$$

Since the cost function $c(\mathbf{b})$ is not non-negative, the following fitness function is defined

$$f(\mathbf{b}) = K + (c(\mathbf{b}) - c_w), \tag{7}$$

where K is a positive constant, c_w is the lowest fitness value in the current population.

(4) Selecting Genetic Operator
Genetic operators including selection, crossover and mutation, they determine the evolution fashion of the chromosomes in the population, and directly affect the speed and performance of the evolutions.

This paper uses two kind selection operators, elitist model and fitness proportional model. Firstly, the chromosome with the highest fitness value is directly copied to the next generation according to the elitist model. Then, uses the roulette wheel in fitness proportional model to select parent chromosomes to give birth to offspring chromosomes for next generation.

For the crossover operator concerned, the one-point crossover scheme is adopted in the paper. Firstly, a crossover point, which is less than the length of the chromosomes, is selected randomly. Then the portions after the crossover point are exchanged between two parent chromosomes.

In this paper, the mutation operator is the simple but widely used one-point mutation. Similar to crossover operator, one or more mutation points are selected randomly. Then the genes on these mutation points are flipped from +1 to −1 one by one, and vice versa.

After the selection, crossover and mutation, the newly obtained offspring chromosomes are ready for next generation.

(5) Termination condition
The termination condition of HSSSGMUD can be a maximum evolution generations or the best chromosome in the population satisfies the requirements.

4 Simulation Results and Analysis

In the following three sets of simulations, the Gold sequences of length L=31 are used as spreading sequences, and there are 10 users in the asynchronous DS-CDMA system in AWGN channels. The first user is assumed to be the desired one. All detectors that incorporated with genetic algorithm use the following default parameters: the population size N_p=20, crossover probability p_c=0.85, mutation probability p_c=0.05, and the maximum evolution generations of the first two sets of experiments N_g=10. For simplicity, experiments are mainly focused on the case that the output of DEC detector is selected as the input chromosome for GAMUD. Similar results can be obtained for minimum mean square error detector, multi-stage detector, decision-feedback detector, etc.

In the first set of experiments, assume all users have the same signal power. The cumulative BERs' varying with the SNR values are plotted in Fig.1 for the CD detector, DEC detector, GAMUD and HSSSGMUD with the output of

DEC detector as the input chromosome (denoted as GADEC and HSSSGDEC respectively). For comparison, the BER curve for single user case is also plotted in Fig.1. Obviously, the HSSSGDEC detector achieves the lowest BERs, it can provide about 1dB gains over GADEC detector when the BER is 10^{-3}.

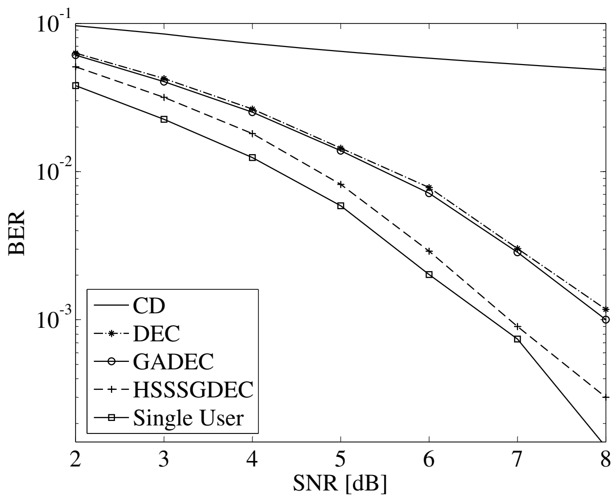


Fig. 1. BERs of CD, DEC, GADEC and HSSSGDEC detectors versus SNR

Furthermore, the cumulative BERs' of CD, MMSE, GAMMSE, HSSSGMMSE detectors varying with the SNR values are plotted in Fig.2. This figure indicates the HSSSGMMSE detector possesses the best BERs performance. And the performance of HSSSGMMSE detector and HSSSGDEC detector shown in Fig.1 are close.

It should be noted the HSSSGDEC and HSSSGMMSE are still random search. The main difference between HSSSGDEC/HSSSGMMSE and GADEC/ GAMMSE is the populatin initialization method.

In the second set of experiments, the near-far resistance ability of CD, DEC, GADEC and HSSSGDEC detectors are compared. Assume the SNR of the first user is fixed on 10dB, while the signal power of other users (interfering users) are varying. Then the BER curves of the first user are depicted in Fig.3 for the above four detectors.

As expected, the HSSSGDEC detector with the initial population constructed by the method introduced in Section 3.2 apparently outperforms the common GADEC with the initial population constructed randomly. The HSSSGDEC detector possesses the best near-far resistance ability among all the detectors.

In the third set of experiments, assume all users have the same signal power, and the SNR is fixed at 6dB. Then, the convergence performance of GADEC and HSSSGDEC detectors are compared in Fig.3. This figure indicates the HSSSGDEC detector converges to a satisfying solution after 10-15 generations, and

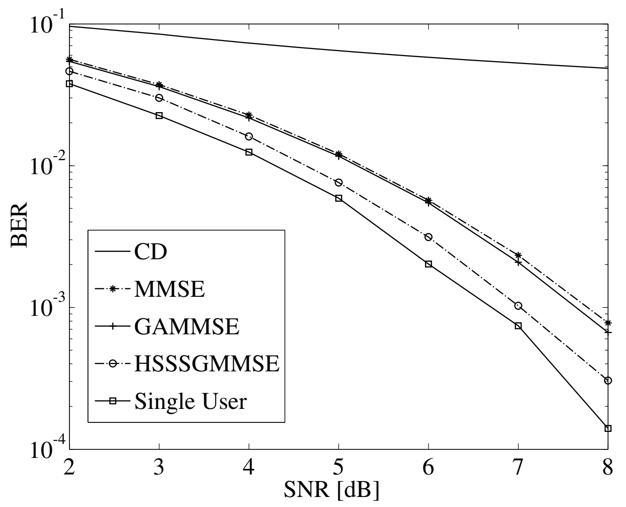


Fig. 2. BERs of CD, MMSE, GAMMSE and HSSSGMMSE detectors versus SNR

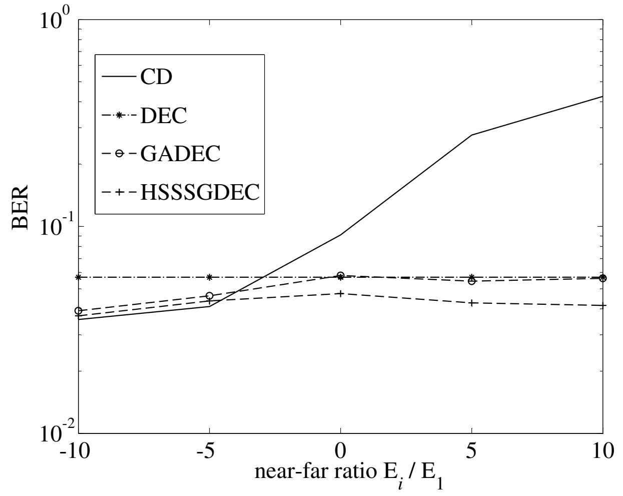


Fig. 3. BERs of CD, DEC, GADEC, HSSSGDEC detectors versus the near-far ratio

this convergence speed is much faster than that of GADEC. For example, the performance of GADEC after 40 generations is only equivalent to that of HSSS-GDEC after 6 generations. In other words, the BER performance curves tell us the HSSSGDEC detector can find much better suboptimal solution vectors than GADEC while with less computing complexity.

In a word, the aforementioned experiment results demonstrate the efficiency of HSSSGADEC detector, and verify the important role of initial population

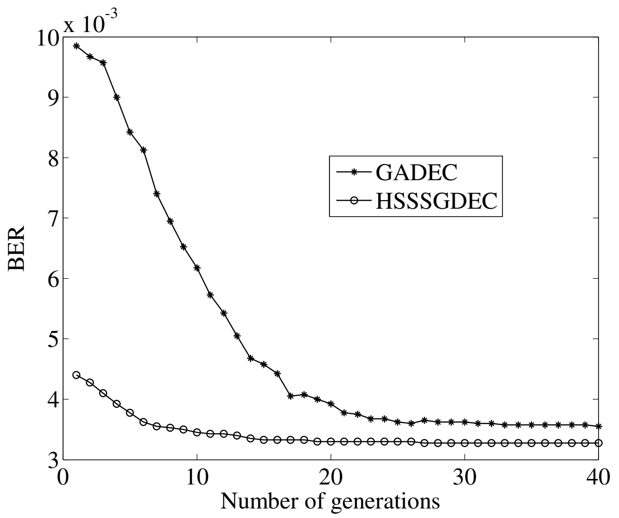


Fig. 4. BERs of GADEC and HSSSGDEC detectors versus the number of generations

in GAMUD. Good initial population is an important guarantee for excellent performance and fast convergence.

5 Conclusions

The concept of Hamming sphere solution space is used to construct initial population, and a kind of HSSSGMUD is proposed. Using different detectors' outputs as the input chromosome leads to a series of HSSSGMUDs. This paper mainly focuses on the HSSSGDEC detector, which uses the output of DEC detector as input chromosome. Simulation results demonstrate with moderate population size and 10 generations evolution, HSSSGDEC detector can achieve excellent performance in BER and near-far resistance ability. Compared with common GADEC, HSSSGDEC converges quickly that is propitious to the real-time implementation. The results of HSSSGDEC detector can be extended to other HSSSGMUDs.

References

1. Verdu, S.: Minimum Probability of Error for Asynchronous Gaussian Multiple-Access Channels. IEEE Transactions on Information Theory. **IT-32(1)** (1986) 85–96
2. Varansi, M. K., Aazhang, B.: Multi-stage Detection in Asynchronous Code-Division Multiple Access Communications. IEEE Transactions on Communication **38(4)** (1990) 509–519
3. Honig, M., Madhow, U., and Verdu, S.: Blind Multiuser Detection. IEEE Transactions on Information Theory. **41(7)** (1995) 944–960

4. Mitra, U., Poor, H. V.: Adaptive Receiver Algorithm for Near-Far Resistant CDMA. IEEE Transactions on Communication. **43(4)** (1995) 1713–1724
5. Juntti, M. J.: Genetic Algorithms for Multiuser Detection in Synchronous CDMA. Proc. of IEEEE International Symposium on Inform. Theory. (1997) 492
6. Wang, X. F., Lu, W. S., and Antoniou, A.: A Genetic-algorithm-based Multiuser Detector for Multiple access Communications. Proc. of IEEE International Symposium on Circuits and Systems. **4** (1998) 534–537
7. Ergün, C., Hacioglu, K.: Multiuser Detection Using a Genetic Algorithm in CDMA Communications Systems. IEEE Transactions on Communication. **48(8)** (2000) 1374–1383
8. Yen, K., Hanzo, L.: Hybrid Genetic Algorithm Based Detection Schemes for Synchronous CDMA Systems. Proc. of 2000 IEEE 51st Vehicular Technology Conference. **2** (2000) 1400–1404
9. Abedi, S., Tafazolli, R.: Genetically Modified Multiuser Detection for Code Division Multiple Access Systems. IEEE Journal on Selected Areas in Communications. **20(2)** (2002) 463–473
10. Kohno, R., Hatori, M., and Imai, H.: Cancellation Techniques of Co-Channel Interference in Asynchronous Spread Spectrum Multiple Access Systems. Elect. And Commun, in Japan. **66-A(5)** (1983) 20–29
11. Lupas, R., Verdu, S.: Linear Multi-user Detectors for Synchronous Code-Division Multiple Access Channels. IEEE Trans. Info. Theory. **35** (1989) 123–126
12. Lupas, R., Verdu, S.: Near-far resistance of multi-user detectors in asynchronous channel. IEEE Trans. Commun. **38** (1989) 496–508
13. Husbands, P.: Advances in Parallel Algorithm. L. Kronsjö and D. Shumsheruddin, Eds. Oxford, U.K.: Blackwell. (1992) 227–269
14. Michalewicz, Z.: Genetic algorithm + data structures = evolution programs. Berlin: Springer-Verlag Berlin Heidelberg. (1996)

UEAS: A Novel United Evolutionary Algorithm Scheme

Fei Gao and Hengqing Tong

Department of Mathematics, Wuhan University of Technology,
122 Luoshi Road, Wuhan, Hubei 430070, P.R. China
tonghq2005@mail.whut.edu.cn
http://public.whut.edu.cn/slx/

Abstract. How to detect global optimums of the complex function is of vital importance in diverse scientific fields. Though stochastic optimization strategies simulating evolution process are proved to be valuable tools, the balance between exploitation and exploration of which is difficult to be maintained. In this paper, some established techniques to improve the performance of evolutionary computation are discussed firstly, such as uniform design, deflection and stretching the objective function, and space contraction. Then a novel scheme of evolutionary algorithms is proposed to solving the optimization problems through adding evolution operations to the searching space contracted regularly with these techniques. A typical evolution algorithm differential evolution is chosen to exhibit the new scheme's performance and the experiments done to minimize the benchmark nonlinear optimization problems and to detect nonlinear map's unstable periodic points show the put approach is very robust.

1 Introduction

Evolutionary algorithm is an umbrella term used to describe computer-based problem solving systems which use computational models of some of the known mechanisms of evolution as key elements in their design and implementation. Although simplistic from a biologist's viewpoint, these algorithms are sufficiently complex to provide robust and powerful adaptive search mechanisms [1,2].

A variety of evolutionary algorithms (EAs) have been proposed, such as genetic algorithms, evolution strategies, evolutionary programming, particle swarm optimization (PSO) and differential evolution (DE) algorithm, Bayesian optimization algorithm (BOA) et al.[1,2,3]. They have been successfully applied for tackling diverse optimization problems during the past few years without the assumptions on the continuity and differentiability of the objective function while those are indispensable to deterministic approaches such as Feasible Direction and Generalized Gradient Descent methods.

EAs maintain a population of structures that evolve according to rules of genetic operators, such as reproduction, recombination and mutation. Each individual in the population receives a measure of its fitness in the environment. Much EA research has assumed that the two processes that most contribute to evolution are crossover and fitness based selection/reproduction. Evolution, by definition, absolutely requires diversity in order to work. And in nature, an important source of diversity is mutation[2,3].

Mutation prevents the GA from converging prematurely, by introducing diversity in the population, but the mutation rate has to be carefully chosen: a too small mutation rate makes this operator ineffective, while a too large mutation rate may destroy the good genetic material found by the other search operators (selection and crossover)[2,3].

Reproduction focuses attention on high fitness individuals, thus exploiting (cf. eploitation) the available fitness information. Recombination and mutation perturb those individuals, providing general heuristics for exploration[3]. How to maintain a proper balance of exploration-exploitation as the problems solved need become a dilemma in studies.

To overcome this dilemma and to decrease the EAs' tendency towards local optima, firstly some established techniques to improve the performance of evolutionary computation are discussed to design EAs to obtain a more effective search mechanism by combining exploitation with exploration of the search space. Then a novel scheme of evolutionary algorithms is proposed and as the experiments done the put scheme is valid.

The rest of this paper is organized as follows. In Section 2, we introduce some established techniques as uniform design, deflection and stretching the objective function, and space contraction. In Section3 Details of the novel scheme are proposed. In Section4 experiments done with the benchmark nonlinear optimization problems and nonlinear map's unstable periodic points are given and the results are analyzed. And Section 5 summarizes the paper.

2 Some Established Techniques

In this section, some established techniques as uniform design[4], deflection and stretching the objective function [5] is introduced, and space contraction to EA is proposed.

2.1 Uniform Design

We uses uniform design method [4] to generate the initial population in feasible field so as to have the property of convergence in large scale without better approximation of the unknown parameter as iterative initial point.

Suppose u_{ij} is the element of uniform design table $U_n\left(n^N\right), a_{ij} = (2u_{ij} - 1)/2n, j = 1, \ldots, N$, then set $P_M = \{a_k = (a_{k1}, \ldots, a_{kN}), k = 1, \ldots, M\}$ contains M points uniformly distributed[4] in $[0,1]^N$.

It is known that set generated by uniform design method is better than by random method statistically in reflecting the objective function's distribution property [4] just as Fig.1. shows.

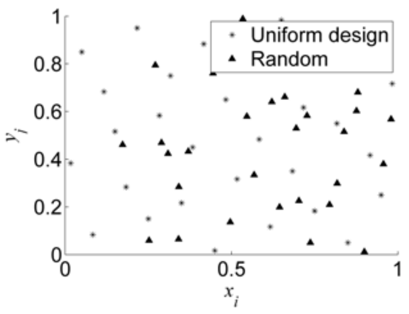


Fig. 1. Sets generated by uniform design and random method

2.2 Deflection and Stretching

We restrain the normal EA's local convergence limitation virtually through deflection and stretching[5] of objective function.

If the objective function $f(x)$ is full of local optimums and more than one minimizer is needed, we choose another established techniques to guarantee the detection of a different minimizer, such as deflection and stretching are introduced. Suppose objective function is $f(x)$, we use deflection technique[5] as below to generate the new objective function $F(x)$:

$$F(x) = \prod_{i=1}^{k} \left[\tanh(\lambda_i \left\| x - x_i^* \right\|)\right]^{-1} f(x) \tag{1}$$

where x_i^* $(i = 1, 2, \cdots k)$ are k minimizes founded, $\lambda_i \in (0, 1)$.

We also introduce stretching technique[5] to generate the new objective functions $G(x)$ and $H(x)$ as new objective functions:

$$G(x) = f(x) + \beta_1 \cdot \|x - x_i^*\| \cdot [1 + \operatorname{sgn}(f(x) - f(x_i^*))] \tag{2}$$

$$H(x) = G(x) + \beta_2 \cdot \frac{1 + \operatorname{sgn}(f(x) - f(x_i^*))}{\tanh\left[\delta\left(G(x) - G(x_i^*)\right)\right]} \tag{3}$$

where $\lambda_1, \lambda_2, \delta > 0$.

Fig.2. shows deflection and stretching effects on $f(x) = \cos x$ at $x = \pi$. In this way, we see that the searching algorithms will not locate $x = \pi$.

2.3 Space Contraction

To avoid exploitation excessively in redundant space and searching efficiency in the whole feasible space, make EA with relative fewer generations as a step, we put a novel technique through a technique space contraction simulating the idea of sequential number theoretic optimization (SNTO)[6] as below.

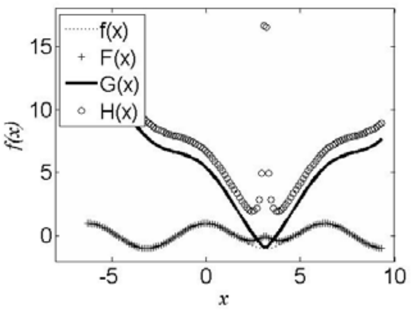


Fig. 2. Deflection and stretching effects on $f(x)$

If a local optimum Q_g is found, we define a new searching space $D^{(t+1)} = [a^{(t+1)}, b^{(t+1)}]$ centering Q_g from the current searching space $D^{(t)} = [a^{(t)}, b^{(t)}]$ as below:

$$\begin{cases} a_i^{(t+1)} = \max(x_i^{(t)} - \gamma c_i^{(t)}, a_i^{(t)}), \\ b_i^{(t+1)} = \min(x_i^{(t)} + \gamma c_i^{(t)}, b_i^{(t)}), \end{cases} (i = 1, 2, \cdots, s) \tag{4}$$

where γ in (0,1) is pre-given contraction ratio. Then we use EA to search in the new space to get a new optimum $Q\prime_g$, save the better one in $Q\prime_g$ and Q_g .

3 A Novel United Evolutionary Algorithm Scheme

With these techniques, we can put a novel United Evolutionary Algorithms Scheme (UEAS) to progress the EAs.

Algorithm 1. United Evolutionary Algorithms Scheme (UEAS)
Step0: Initialization. $t = 0, D(0) = D, a(0) = a$, $b(0) = b, \gamma \in (0, 1)$;
Step1: Outer cycle Termination condition Judging. If the optimums wanted are found, output them and terminate, otherwise deal with the objective function with the deflection and stretching techniques;
Step2: Generate the initial set $A(t)$ on $[a(t), b(t)]$ by uniform design method, evaluate the fitness and note the best one Q_g ;
Step3: Use EA to get a current optimum; update the best one Q_g;
Step4: Inner cycle Termination condition Judging. Given $\delta > 0$ enough small, $c(t) = (b(t) - a(t))/2$, if max $c(t) < \delta$, then $x(t), M(t)$ are accept, go to Step1, otherwise go to Step5;
Step5: Space contraction. Define a new region $D(t + 1) = [a(t + 1), b(t + 1)]$ by (4) with $\gamma = 0.5$, $t = t + 1$, go to Step2.

As many experiment results reported suggest that too many generations do not bring the optimum better than the local one [7], thus the EA in Step3 of Algorithm1 has relatively fewer generations contrast to the normal EAs, usually 30 to 1000.

Through Algorithm 1, firstly UEAS can detect the objective function's character as far as possible with the initial set by uniform design method. Secondly UEAS can get more optimums and avoid premature through the deflection and stretching techniques. Thirdly UEAS can avoid exploitation excessively in redundant space and search in the most prospective space of the feasible field, so it can jump the local optimum easier. Fourthly with the help of Outer cycle Termination condition Judging UEAS can find all the optimums sequentially. And lastly if EA in Step3 is valid enough, UEAS will not contract the searching space, and in this sense EAs are the special cases of UEAS, then UEAS can combine most of current stochastic optimization strategies such as such as Genetic Algorithms, Evolutionary Programming, PSO, DE algorithm, BOA et al.

Now we choose a typical stochastic optimization strategies DE [8,9] as the Step3 of UEAS to show its advantages.

DE algorithm grew out of Price's attempts to solve the Chebychev Polynomial fitting Problem that had been posed to him by Storn [9]. It utilizes Mn–dimensional vectors, $x_i = (x_{i1}, \cdots, x_{in}) \in S, i = 1, \cdots, M$ as a population for each iteration, called a generation, of the algorithm. At each generation, two operators, namely mutation and crossover (recombination), are applied on each individual, thus producing the new population. Then, a selection phase takes place, where each individual of the new population is compared to the corresponding individual of the old population, and the best between them is selected as a member of the population in the next generation [8,10]. The details of the DE are given as below [8]:

Algorithm 2 [8]. Differential Evolution (DE) Algorithm
Step1. Initialization. Random generate M individuals in feasible region S, $G = 0$, crossover constant $CR > 0$, mutation constant $CF = 0.5, G_{max}$, define a fitness function $f(x)$, value the population and label the best individual in current population as Q.
Step2. DE Evolution. $g = g + 1$, for each $x_i = (x_{i\,1}, x_{i\,2}, \cdots, x_{i\,n})$:

(1) Mutation. Random choose four mutually different individuals xa, xb, xc, xd in the current population to get a vector $D_{abcd} = (x_a - x_b) + (x_c - x_d)$, use it to generate new vector $\xi_i = (\xi_{i\,1}, \ldots, \xi_{i\,n})$ as below:

$$\xi_i = Q + CF \times D_{abcd} \tag{5}$$

(2)Crossover. To get a new testing vector $U_i = (u_{i\,1}, \ldots, u_{i\,n})$ with ξ_i :

$$u_{i\,j} = \begin{cases} \xi_{i\,j}\ , & if\ (randb(j) < CR)\ \ or\ \ (j = rnbr(i)); \\ x_{i\,j}\ , & if\ (randb(j) \geq CR)\ \ and\ \ (j \neq rnbr(i)). \end{cases} \tag{6}$$

where $rnbr(i)$ is random integer in $\{1, 2, \ldots, n\}$, $randb(j)$ is j-th random real in $[0, 1]$, $j = 1, 2, \ldots, n$.

(3) Replacement. Remain the better one between x_i and U_i:

$$x_i = \begin{cases} U_i\ , & if\ f(U_i) < f(x_i); \\ x_i, & if\ f(U_i) \geq f(x_i). \end{cases} \tag{7}$$

Step3. Updating. Find the current best $Q\prime$ and remain the better between Q and $Q\prime$ as the new Q.
Step4. Termination. If $g > G_{max}$, then export the Q, else go back to Step2.

4 Experiments

With DE as Step3 of UEAS, we will deal with the benchmark nonlinear optimization problems and nonlinear map's unstable periodic points and analyze the results.
eg.1. Rastrigrin function[11] $f(x) = \sum_{i=1}^{20} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$, $\|x\|_2 \leq 100$

For this function's optimization, make the inner cycle 200, $\gamma = 0.5$ in Algorithm1 and combine DE with Deflection and Stretching techniques and uniform design method to initial set, we can get optimums with the same value 0 as reported as below:

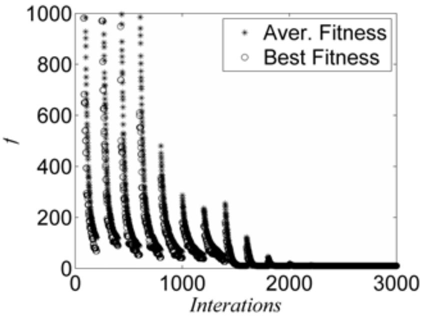


Fig. 3. Comparison of the best and average fitness in each contraction

From Fig.3, it can be concluded that UEAS can avoid local optimum easily and locate the best one through contraction.
eg.2 $f_2(x, y) = -[20 + x\sin(9\pi y) + y\cos(25\pi x)]$, where $(x, y) \in D = \{(x, y)|x^2 + y^2 \leq 81\}$, its minimum value -32.71788780688353, correspond global minim point is $(-6.44002582194051, -6.27797204163553)$. Fig.4 is f_2 on D, where its value out of D is put as -40.

f_2 is a multi–modal function difficult to be optimized. Its definition region D is big, $x\sin(9\pi y)$ and $y\cos(25\pi x)$ oscillate in different directions in their ways and it has deep valley clatters near to 4 points.

For this function's optimization, make the outer cycle just 1, and combine DE with Deflection and Stretching techniques and uniform design method to initial set, we can get 6 optimums with the same value -32.7178878068835 as reported as below:

$(-6.44002582207099, -6.2779720144943)$, $(-6.44002582208887, -6.2779720122268)$,
$(-6.44002582235322, -6.2779720135692)$, $(-6.4400258222786, -6.2779720141167)$,
$(-6.44002582226788, -6.2779720144792)$, $(-6.44002582235673, -6.2779720134721)$

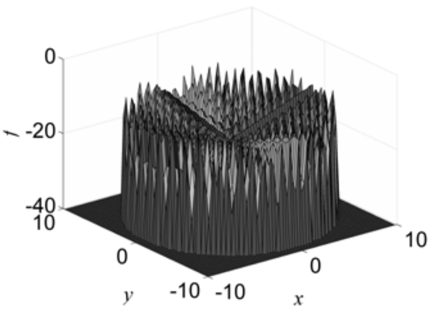


Fig. 4. f_2 on D

eg.3 Hénon map

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \phi \begin{pmatrix} x_n \\ y_n \end{pmatrix} = \begin{pmatrix} a + by_n - x_n^2 \\ x_n \end{pmatrix} \tag{8}$$

When a and b of system (8) is $a^* = 1.4, b^* = 0.3$, (8) is chaotic, Hénon chaos has strange attractor with a unstable fixed point in it [12,13].

With a new defined function (9) below as fitness function we can use UEAS with DE to get its different unstable period orbits.

$$F(X) = \|\phi^p(X) - X\|_2 \tag{9}$$

where parameter p is the order of period orbits, $X = (x_n, y_n)^T$. And Fig.5 shows the figure of function $\left\|\phi^{11}(X) - X\right\|_2$, we can see it is so difficult to optimize.

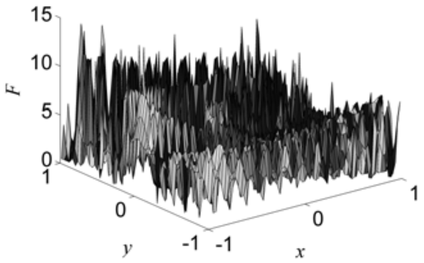


Fig. 5. The figure of function $\left\|\phi^{11}(X) - X\right\|_2$

With the parameters above for UEAS with DE, various period points found by UEAS are given in Table 1, where D is value of $F(X)$.

Fig.6 shows the correspondent unstable period orbits in Table 1 in two dimensions.

Experiments done above illustrate that UEAS with DE for optimization problems can improve the global convergence and have the advantages of high precision and robustness to such a certain extent.

Table 1. Hénon map's unstable period points

p	Period points	D
1	(0.883896267925307, 0.883896267925306)	$1.23259516440783\times10^{-32}$
11	(0.462401211021121,0.943311629120628)	$1.97215226305253\times10^{-31}$
13	(0.554806969558445,0.870801140362593)	$5.47204354507602\times10^{-21}$
23	(0.535985487833863,0.886448189518229)	$6.04937684913244\times10^{-6}$

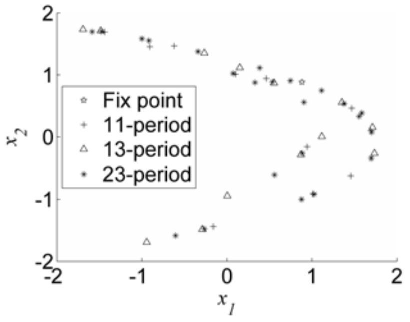


Fig. 6. Hénon map's period orbits

5 Conclusions

In this paper we propose a novel scheme UEAS combining with the concept of evolutionary calculation technique and some established techniques. And the experiments done show the proposed scheme is robust.

Though experiments are done only by UEAS with DE, we can derive from it that UEAS with the other evolutionary algorithms straightforwardly.

Acknowledgment

We thank the anonymous reviewers for their constructive remarks and comments.

This work is partially supported by Innovation Fund of the Ministry of Science & Technology of China (Grant No. 02C26214200218), the National Natural Science Foundation of China (Grant No. 30570611) to Tong H. Q., the basic Foundation (Grant No.XJJ20041130), Project of educational research, the UIRT Projects (Grant No.A156, No.A157) to Gao F. granted by Wuhan University of Technology in China.

References

1. Whitley, D.: An overview of evolutionary algorithms: Practical issues and common pitfalls. Information and Software Technology. **43**(14) (2001) 817–831
2. Eiben, A. E., Smith, J. E.: Introduction to Evolutionary Computing (Natural Computing Series). Springer (2003)

3. Yao, X., Xu, Y.: Recent Advances in Evolutionary Computation. J. of Computer Science and Technology **21**(1) (2006)1–18
4. Ma, C. X.: Uniform design Based on Centered L2 Discrepancy $U_n\,(n^s)$. http://www.math.hkbu.edu.hk/UniformDesign. (1999)
5. Parsopoulos, K. E., Vrahatis, M. N.: On the Computation of All Global Minimizers through Particle Swarm Optimization. IEEE Tran. on evolutionary computation **8**(3) (2004) 211–224
6. Hua, L. K., Wang, Y.: Applications of Number theory to Numerical analysis. Berlin & Beijing: Springer–Verlag & Science Press (1981)
7. Pan, Z.J., Kang, L.S., Chen Y. P.: Evolutionary Computation. Beijing: Tsinghua University Press (1998)
8. Storn, R., Price, K.: Differential evolution-a simple and efficient adaptive scheme for global optimization over continuous spaces. Journal of Global Optimization **11** (1997) 341–359
9. Price, K., Storn, R., Lampinen, J.: Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series). Springer (2005)
10. Gao, F., Tong, H. Q.: Control a Novel Discrete Chaotic System through Particle Swarm Optimization. Proceedings of the 6th World Congress on Control and Automation, Dalian, China, (2006) 3330–3334
11. Michalewicz, Z., Fogel, D. B.: How to Solve It: Modern Heuristics. Berlin: Springer–Verlag (2000)
12. Henon, M.: Numerical study of quadratic area–preserving mappings. Quart. Appl. Math. **27** (1969) 291–311
13. Biham, O.,Wenzel, W.: Unstable periodic orbits and the symbolic dynamics of the complex Henon map. Physical Review A **42**(10) (1990) 4639–4646

Implicit Elitism in Genetic Search

A.K. Bhatia* and S.K. Basu**

Department of Computer Science
Banaras Hindu University
Varanasi-221005 (India)

Abstract. We introduce a notion of implicit elitism derived from the mutation operator in genetic algorithms. Probability of mutation less than $1/l$ (l being the chromosome size) along with probability of crossover less than one induces implicit elitism in genetic search. It implicitly transfers a few chromosomes with above-average fitness unperturbed to the population at next generation, thus maintaining the progress of genetic search. Experiments conducted on one-max and 0/1 knapsack problems testify its efficacy. Implicit elitism in combination with traditional explicit elitism enhances the search capability of genetic algorithms.

Keywords: Genetic Algorithms, Premature Convergence, Elitism.

1 Introduction

Genetic algorithms (GAs) [8] are general purpose optimization procedures based on law of survival of the fittest in natural genetics. GAs explore and exploit a population of potential solutions through crossover, mutation and selection operators over a number of generations and march toward the global optimum. But it stops progressing toward the global optimum after a few generations and gets trapped at a sub-optimal solution, which is termed *premature convergence*.

Loss of diversity in the population is commonly regarded as the cause for occurrence of premature convergence [1]. Suitable genetic representation [13], choice of suitable values of the genetic parameters such as the population size, the probability of crossover and the probability of mutation are important to avoid the phenomenon. Generally, the parameter values are tuned and fixed before the start of GA run. Adaptation of genetic parameters has also been used with mixed results [5]. In addition, several methods involving extra computation such as crowding, sharing, restricted mating, etc. have been used with genetic algorithms to avoid premature convergence [15].

Traditionally, mutation operator has been given a background role of maintaining diversity in the population. Recently, importance of the operator in successful genetic search has been recognized [12]. In this paper, we introduce a notion of *implicit elitism* based on mutation operator in GAs [2]. Implicit elitism causes some of the high-fitness chromosomes to pass on unperturbed to the population at the subsequent generations, maintaining the progress of genetic search. We report the results obtained from the

* Current address: National Bureau of Animal Genetic Resources, Karnal - 132001 (India) Email: avnish@lycos.com

** Corresponding author. Email: swapankb@bhu.ac.in; Fax: +91-542-2368285 experiments on one-max and 0/1 knapsack problems to demonstrate efficacy of implicit elitism.

One-max problem consists of finding a binary string of length n so as to maximize the number of '1' bits in the string. The problem is frequently used to verify theoretical aspects of genetic algorithms [10].

0/1 knapsack problem is a well studied real-world problem [3, 4, 11]. It is defined as: given a profit vector p_i, a weight vector w_i and knapsack capacity C

$$\text{Maximize} \sum_{i=1}^{n} p_i x_i \quad \text{subject to} \quad \sum_{i=1}^{n} w_i x_i \leq C$$

where $x_i = 0 \, or \, 1$, $(i = 1, \ldots, n)$, n is the number of items.

The paper is organized as: section 2 introduces the notion of implicit elitism, section 3 explains the experimental setup and the results obtained, and section 4 contains our concluding remarks.

2 Implicit Elitism

A randomly generated chromosome represents a random sample from the search space of the candidate objective function [12]. A function value corresponding to a point in the search space can be considered as a cumulative effect of the n variables of the function. The sum or cumulative effect of mutually independent random variables with a common distribution follows normal distribution approximately under certain conditions (central limit theorem [7]). The theorem holds for large classes of dependent variables also [7] and even if the individual random variables have different distributions [9]. For normal distribution, the maximum frequency occurs at the mean value and it has a crowding of values around the mean. This leads us to the inference that the function value of a randomly generated chromosome will be around the mean function value. Likewise, random perturbations in the strings caused by crossover and mutation operators push the strings toward the mean function value.

The selection operator in a GA pulls the strings toward the optimum. GA converges to a suboptimal value when equilibrium is reached between the two opposite forces. Elitism has been used in GAs to obtain near optimal results, where the individual having the best function value in a population is retained explicitly in the population at the next generation. We call this traditional form of elitism *explicit elitism*. Elitist GA has theoretically been shown to converge to the global optimum [12, 14].

Mutation operator is applied to every gene in a chromosome with a pre-specified probability[1]. A $p_m = 1/l$ indicates that on an average one gene will be changed in a string of length l. Mutation with $p_m \geq 1/l$ perturbs gene(s) in each chromosome, thus perturbing all the chromosomes in the population. With $p_m \geq 1/l$, even the elite copied to the new population with explicit elitism is subject to perturbation by the mutation operator and has a tendency to move toward the mean function value. Thus, with the

[1] We denote the probability of mutation as p_m, the probability of crossover as p_c and the string length as l.

$p_m \geq 1/l$, a GA with or without explicit elitism is likely to lose its strength to march toward the global optimum after a few generations.

There is a non-zero probability of all the genes remaining unperturbed in a string with $p_m < 1/l$ and hence that of the string remaining unperturbed. Mutation operator leaves a few strings unperturbed in the population with this value of the p_m. Crossover operator with $p_c < 1$ also leaves a few strings unperturbed. Overall, a few strings pass on unperturbed to the new population with the above values of p_m and p_c. Further, the selection operator is fitness biased and selects the high fitness strings with higher probability. Thus, the probability of a few high fitness strings being copied without any change to the new population is non-zero. We call the process *implicit elitism*.

With implicit elitism, the population at the next generation is a mixture of the perturbed chromosomes and the chromosomes passed on unperturbed from the previous generation. If the perturbed chromosomes have higher fitness than the unperturbed chromosomes, they get selected to form the population at the subsequent generation. Otherwise, the unperturbed chromosomes are passed on to the population at the subsequent generation. The elite retained through explicit elitism also has a non-zero probability of passing on unperturbed to the subsequent population. Thus, the GA with implicit elitism maintains its strength till it reaches the global optimum, because it is either at the same level of fitness or it is moving ahead.

Let $p_m = h.\frac{1}{l}$, where h is a mutation factor. Value of $h < 1$ makes the $p_m < 1/l$. Crossover operator is applied after selection of parents for reproduction. The selected chromosomes undergo crossover with a pre-specified value of p_c. After selection and crossover, all the chromosomes in the population are subjected to a gene-level mutation operator. Values of $h < 1$ and $p_c < 1$ imply occurrence of implicit elitism in GAs.

3 Experiments and Results

3.1 One-Max Problem

Experiments have been conducted on problem instances of sizes 100, 500 and 1000.

We use a binary-coded GA with single-point crossover, $p_c = 0.5$, linear rank selection, population size equal to 60 and number of generations equal to 2000. GA is executed with and without the explicit elitism. Values of the mutation factor h are taken as 0.7, 1.0 and 1.5. Implicit elitism occurs when values of both p_c and h are less than one. A value of $h = 1.0$ is at the boundary of occurrence and non-occurrence of implicit elitism. We take still higher value of $h = 1.5$ for comparison purposes.

Table 1 shows the results of 10 experiments. Effect of implicit elitism is evident from the results. GA with explicit elitism as well as without explicit elitism provides the best results when $h = 0.7$. Explicit elitism has a positive effect on the results. The GA with implicit elitism alone can provide better results compared with the GA using explicit elitism only. GA without explicit elitism provides an average function value of 965.9 with $h = 0.7$ (implicit elitism), which is better than the average value of 962.6 reached by the GA with explicit elitism and $h = 1.5$ (no implicit elitism).

Figures 1 and 2 show the search progress of the GA without and with explicit elitism repectively on the one-max problem instance of size 1000 (single experiment). The

Table 1. Result of GA execution on One-max problem. (W-worst, B-best, A-average, SD-standard deviation) The best results for the categories 'best' and 'average' are highlighted. Explicit elitism is the same as the traditional form of elitism in genetic algorithms. Implicit elitism occurs when $h < 1$ and $p_c < 1$.

n		Without explicit elitism			With explicit elitism		
		$h = 0.7$	$h = 1.0$	$h = 1.5$	$h = 0.7$	$h = 1.0$	$h = 1.5$
100	W	98	95	90	100	100	100
	B	100	98	95	100	100	100
	A	99.5	96.3	91.5	**100**	100	100
	SD	0.6	0.8	0.9	0	0	0
500	W	487	464	440	500	498	489
	B	495	472	446	*500*	500	499
	A	490.3	469.1	443.6	**500**	499	494
	SD	2.0	2.6	1.8	0	0.6	2.6
1000	W	960	927	871	996	983	955
	B	977	934	883	*1000*	992	969
	A	965.9	930.2	876.3	**998.1**	988.1	962.6
	SD	4.7	2.4	3.6	1.1	2.7	3.9

GA with implicit elitism ($h = 0.7$) can reach much better maximum function value compared with the other two values of h that don't exhibit implicit elitism.

3.2 0/1 Knapsack Problem

Test data for 0/1 knapsack problem are generated for n equal to 100, 500 and 1000 using the values: $w_i = \text{uniform}[1,\ 100]$, $p_i = \text{uniform}[1,\ 100]$ and $C = 0.5 \sum_{i=1}^{n} w_i$.

Binary coding is used for the 0/1 knapsack problem. Being a constrained optimization problem, the strings may represent infeasible region of the search space. GAs can be used to solve such problems by incorporating a penalty term in the objective function

Table 2. Result of GA execution on 0/1 Knapsack problem instances

n	Optimum		Without explicit elitism			With explicit elitism		
			$h = 0.7$	$h = 1.0$	$h = 1.5$	$h = 0.7$	$h = 1.0$	$h = 1.5$
100	4345	W	4105	3979	3796	4279	4254	4166
		B	4182	4120	3954	*4326*	4318	4274
		A	4143	4030	3887	**4310**	4278	4240
		SD	22	38	56	14	19	32
500	20602	W	18696	18228	17619	19801	19438	19101
		B	19149	18576	18163	*20041*	19875	19357
		A	18978	18437	17780	**19927**	19660	19226
		SD	136	107	155	69	108	90
1000	40887	W	36481	35520	34550	38250	37791	36657
		B	37095	36643	35402	*38752*	38284	37624
		A	36844	36064	34854	**38491**	38007	37215
		SD	178	324	223	159	142	264

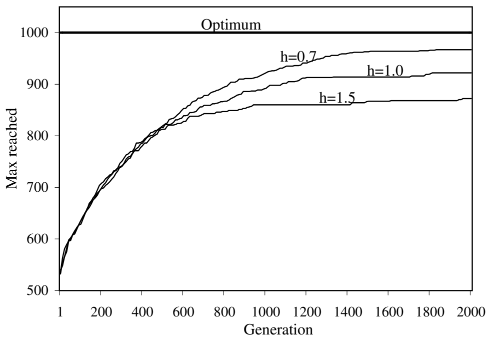


Fig. 1. Progress of the GA without explicit elitism on One-Max Problem instance of size 1000

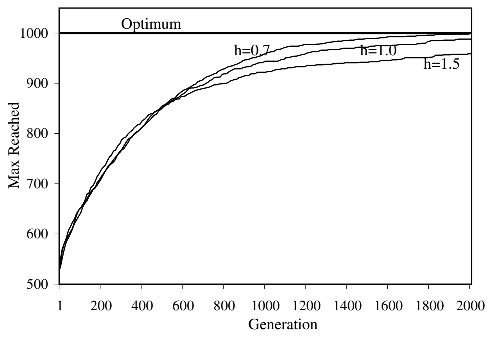


Fig. 2. Progress of the GA with explicit elitism on One-Max Problem instance of size 1000

[11]. Alternately, the infeasible strings may be removed from the population altogether resulting in death penalty. Repair algorithms can also be used to convert the infeasible strings into feasible ones. We use the method of death penalty where an infeasible string in the population is replaced with a newly generated feasible string.

All the genetic operators and parameter values taken for the 0/1 knapsack problem are the same as described for the One-Max problem in the subsection 3.1. Table 2 shows the results obtained for the three test instances in 10 experiments. The best results are obtained with $h = 0.7$. Average profit values obtained are equal to 18978 with $h = 0.7$, 18437 with $h = 1.0$ and 17780 with $h = 1.5$ for the instance of size 500 when the GA is run without explicit elitism. Overall, the best results are obtained for all the three problem instances when explicit elitism is combined with implicit elitism ($h = 0.7$), demonstrating synergetic role of the two forms of elitism in this problem also.

A single experiment of the GA has been conducted to study the progress of GA with the three values of h on the 0/1 knapsack problem instance of size 1000. Figures 3 and 4 show the progress of GA without and with explicit elitism respectively. GA with higher values of h starts faster due to exploration of the search space of this combinatorial problem. But the performance with higher values of h is ultimately subdued. The best profit values are reached when $h = 0.7$.

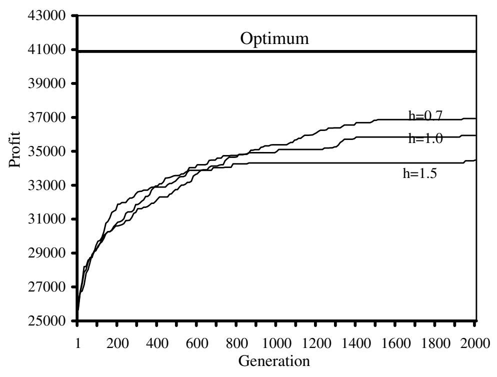


Fig. 3. Progress of the GA without explicit elitism on 0/1 Knapsack Problem instance of size 1000

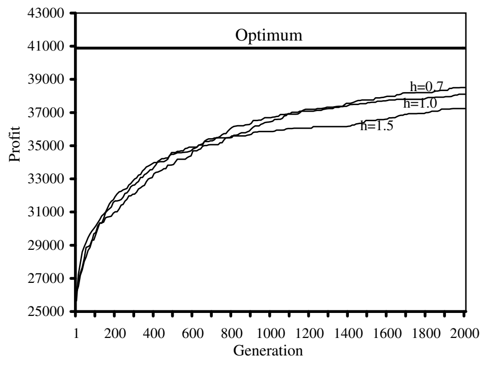


Fig. 4. Progress of the GA with explicit elitism on 0/1 Knapsack Problem instance of size 1000

3.3 Effect of Mutation Factor (h)

Genetic algorithms require a balance between exploration and exploitation of the search space for effective problem solving [6]. GA with a very small value of mutation probability can't explore the search space. We have conducted experiments with the values of h varying between 0.1 and 2.0 to observe the point where the GA can explore as well as exploit the seach space to the maximum. We take one instance of size 1000 for both the test problems for this purpose. GA has been executed for 5000 generations. All the other genetic operators and parameters are the same as described in the subsection 3.1. Ten experiments have been conducted for each value of h.

Figure 5 shows the results obtained for the One-Max problem instance using the GA with and without explicit elitism. The GA without explicit elitism can reach the global optimum with values of h equal to 0.1, 0.2 and 0.3. Performance of the GA starts deteriorating from $h = 0.4$ onward. GA can reach a value of 846 only when $h = 2.0$. The GA with explicit elitism can reach the global optimum in all the experiments with values of h equal to 0.1 to 0.8. The average function value reached with $h = 2.0$ is 980.

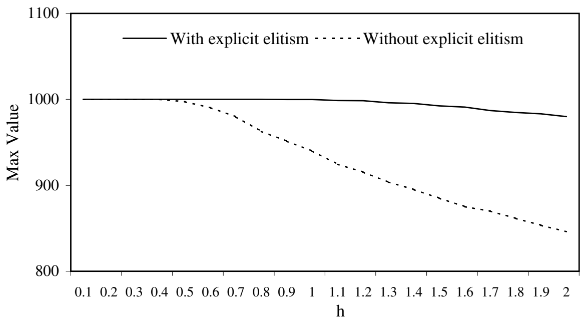


Fig. 5. Performance of the GA with values of h varying from 0.1 to 2.0 on One-Max Problem instance of size 1000

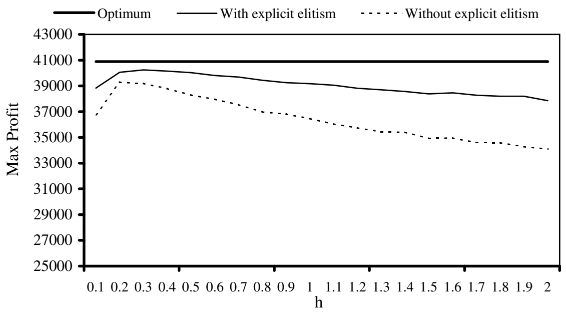


Fig. 6. Performance of the GA with values of h varying from 0.1 to 2.0 on 0/1 Knapsack Problem instance of size 1000

Figure 6 shows the performance of GA with and without explicit elitism on the 0/1 knapsack problem instance of size 1000. GA without explicit elitism provides the maximum average function value of 39276 against the global optimum equal to 40887 when $h = 0.2$. The function value reached declines with increase in the value of h. The GA with $h = 2.0$ provides an average profit value equal to 34095. The GA with explicit elitism provides the maximum average function value of 40244 when $h = 0.3$. The average profit value reached is equal to 37858 when $h = 2.0$.

The best combination of exploration and exploitation occurs when value of h is very small for both the problems. What value of h will be most effective for solving a problem remains open for further investigation.

4 Conclusion

A GA with probability of mutation less than $1/l$ (l being the chromosome size) and the probability of crossover less than one exhibits implicit elitism. GA with these values of the parameters copies some of the high-fitness chromosomes unperturbed to the subsequent generation, thus maintaining the progress of genetic search. Experiments on one-max and 0/1 knapsack problems demonstrate efficacy of implicit elitism. Traditional elitism (referred explicit elitism in this paper) provides the best results when combined with the implicit elitism, thus demonstrating synergetic role of the two forms of elitism. The experiments demonstrate that GAs have an effective balance of exploration and exploitation when the probability of mutation is very small, making use of implicit elitism.

References

[1] H. B. Amor and A. Rettinger. Using self-organizing maps in evolutionary computation. In *Proc. of Genetic and Evolutionary Computing Conference* (GECCO'05). ACM, 2005.

[2] Avnish K. Bhatia. *Studying the effectiveness of genetic algorithms in solving certain classes of problems*. PhD thesis, Department of Computer Science, Banaras Hindu University, Varanasi, India, 2005.

[3] A. K. Bhatia and S. K. Basu. Tackling 0/1 knapsack problem with gene induction. *Soft Computing*, 8(1):1–9, 2003.

[4] N. Christofides, A. Mingozzi, P. Toth, and C. Sandi, editors. *Combinatorial Optimization*, chapter 9. John Wiley & Sons, 1979.

[5] A. E. Eiben, R. Hinterding, and Z. Michalewicz. Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2):124–141, 1999.

[6] A. E. Eiben and C. A. Schippers. On evolutionary exploration and exploitation. *Fundamenta Informaticae*, 35:1–16, 1998.

[7] W. Feller. *An Introduction to Probability Theory and its Applications*, volume 1. John Wiley & Sons, third edition, 1968.

[8] D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning*. Addison Wesley, 1989.

[9] C. M. Grinstead and J. L. Snell. *Introduction to Probability*. American Mathematical Society, 1997.

[10] G. Harik, E. Cantú-Paz, D. E. Goldberg, and B. L. Miller. The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation*, 7(3):231–253, 1999.

[11] Z. Michalewicz. *Genetic algorithms + Data structures = Evolution programs*. Springer-Verlag, second edition, 1994.

[12] C. A. Murthy, D. Bhandari, and S. K. Pal. ϵ - optimal stopping time for genetic algorithms. *Fundamenta Informaticae*, 35:91–111, 1998.

[13] J. Rowe, D. Whitley, L. Barbulescu, and J.-P. Watson. Properties of gray and binary representations. *Evolutionary Computation*, 12(1):47–76, 2004.

[14] J. Suzuki. A markov chain analysis on simple genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(4):655–659, 1995.

[15] D. Thierens. Scalability problems of simple genetic algorithms. *Evolutionary Computation*, 7(4):331–352, 1999.

The Improved Initialization Method of Genetic Algorithm for Solving the Optimization Problem

Rae-Goo Kang and Chai-Yeoung Jung*

Dept. Computer Science & Statistic, Chosun University,
375 Seoseok-dong Dong-gu Gwangju, 501-759 Korea
kangrg@hanmail.net, cyjung@chosun.ac.kr

Abstract. TSP(Traveling Salesman Problem) used widely for solving the optimization is the problem to find out the shortest distance out of possible courses where one starts a certain city, visits every city among N cities and turns back to a staring city. At this time, the condition is to visit N cities exactly only once. TSP is defined easily, but as the number of visiting cities increases, the calculation rate increases geometrically. This is why TSP is classified into NP-Hard Problem. Genetic Algorithm is used representatively to solve the TSP. Various operators have been developed and studied until now for solving the TSP more effectively. This paper applied the new Population Initialization Method (using the Random Initialization method and Induced Initialization method simultaneously), solved TSP more effectively, and proved the improvement of capability by comparing this new method with existing methods.

Keywords: Genetic Algorithm, GA, Optimization, Initialization.

1 Introduction

To solve the optimization problem effectively, this paper used TSP as an experiment model. TSP is one of basic and important problems which is used widely in modern industrial fields such as physical distribution, the network of telephone, the design of integrated circuit, industrial robot-programming, optimization of network etc.

TSP is the optimization problem of sequence mixture finding the shortest course. The shortest course means the minimum course visiting every city only once among N cities in two dimension. This problem is defined easily, however, as the number of cities increases, the calculation rate increases geometrically. Therefore, this is classified into NP(Nondeterministic Polynomial)-Hard problem.

Owing to this problem, Genetic Algorithm(GA) proposed by John Holland is used representatively to obtain the optimal solution. [1],[2],[3]

The investigation space of TSP is $\{T_1, T_2, ..., T_n\}$, the set of all traveling, and the size of it is $N!$. The solution is the shortest traveling distance.

This paper drew the new Mixture Initialization method using both Random Initialization method and Induced Initialization method at the same time for Population Initialization which should be preceded to apply GA, applied this to Population Initialization and experimented to get the nearest value to the optimal solution.

* Correspondent author.

2 Operator Used in This Paper for Experiment

In Table 1, there are operators used to prove the capability of methods proposed in this paper.

Table 1. Operators used in experiment

Selection Operator	Roulette wheel Rank-based selection
Crossover Operator	PMX Edge Recombination One-Point Crossover Multi-Point Crossover
Mutation Operator	Inversion Swapping mutation

2.1 Selection Operator

The common rule is that the probability for superior solution to be chosen should be high although various Selection Operators have been presented so far. This paper used Roulette wheel selection and Rank-based selection operators.

Roulette wheel selection operator is the most representative Selection Operator. This operator estimates the quality of each solution and adjusts the fitness of the best solution to be k times than that of the worst solution. The fitness of solution i in the set of solutions is calculated like this:

$$fi = (Ct - Ci) + (Ct - Ce)/(k-1), k > 1 \tag{1}$$

Ct: Cost of the worst solution in group

Ce: Cost of the best solution in group

Ci: Cost of i

If the value of k is made high, the choice probability becomes high. Generally, the commonest value of k is 3~4. This is chosen by the standard of fitness value.

By adjusting k value, Roulette wheel selection can prevent good-quality solution and bad-quality solution from having excessive difference of fitness, but cannot adjust the distribution of solutions. Rank-based selection makes a rank in the order of qualities of solutions in solution group, and then allocates fitness first-functionally from the best solution. The formula (2) is showing the allocation function of fitness of Rank-based selection. The fitness of the ith chromosome among n chromosomes can be calculated like this: In this formula, choice probability can be adjusted through changing the difference of max and min values. The fitness of solutions is distributed regularly between max and min. [3],[4].

$$fi = \max + (i-1) \times (\min - \max)/(n-1) \tag{2}$$

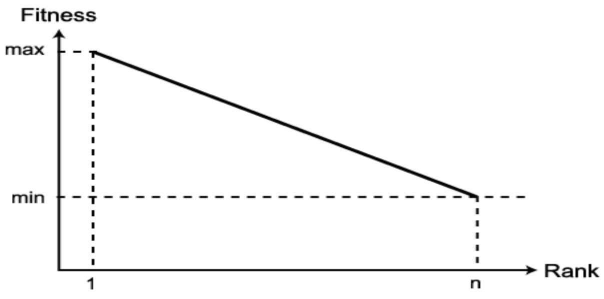
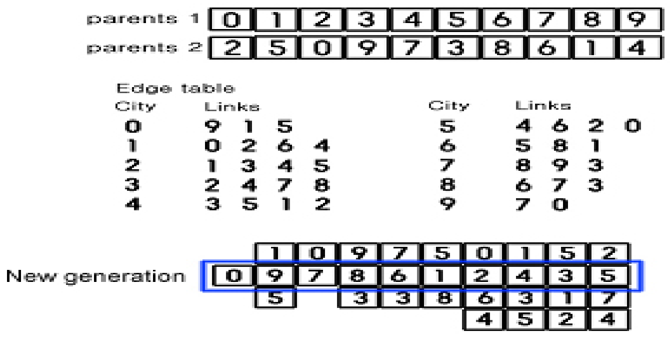


Fig. 1. The allocation function of fitness of Rank-based selection

2.2 Crossover Operator

Crossover Operator is the most various and representative operator in GA. PMX, One-Point Crossover, Multi-Point Crossover, and Edge Recombination(ER) was used in this paper. ER operator is one kind of heuristic Crossover Operator introduced by Grenfensetette and is introduced by Starkweather. [5],[6],[7]



Edge table

City	Links	City	Links
0	9 1 5	5	4 6 2 0
1	0 2 6 4	6	5 8 1
2	1 3 4 5	7	8 9 3
3	2 4 7 8	8	6 7 3
4	3 5 1 2	9	7 0

Fig. 2. Edge Recombination

Edge Recombination is a crossover operator which focuses on the adjacency relation. As illustrated in Fig 2, Edge Recombination uses an edge table to record parental edges, and then limits the generation of offspring to the edges contained in this table. In other words, the candidates of offspring edges come from parental edges principally.

With reference to the edge table, Edge Recombination builds a tour according to specific heuristics. In the original Edge Recombination(Edge-1)[8], the building process intends to select the city with the fewest links, namely, the most isolating city. Edge-1 initially generates the edge table by scanning both parents. Afterwards, Edge-1 begins the process of building the filial tour. [9]

(1) Select one of the first parental cities as the starting city.
(2) Select its next city from the remaining adjacent cities of the current city. We call these links candidates for the next city. According to the heuristic's "priority of isolating cites", the candidates with the smallest number of links is chosen.
(3) Repeat (2) until the complete tour is built.

2.3 Mutation Operator

Each population becomes stronger and more look-alike by Selection Operators and Crossover Operators. However, the more the generation goes down, the less the variety of genes is. Mutation Operators is used to compensate these faults. With Mutation Operators, new population can be made by preventing a specific bit from fixing from the early generation.

In this paper, Swapping Mutation and Inversion were used out of Mutation Operators. [1],[4]

3 Proposed Method

This paper proposes Mixture Initialization method to obtain a superior solution of TSP.

3.1 Mixture Initialization Method

There are two methods in Population Initialization. One is Random Initialization method where population is produced by extracting at random without any rules. And the other is Induced Initialization method where population is produced consistently by using background knowledge and information relating to given values.

The Population Initialization is more important than any other thing to get the nearest value to the optimal solution. Random Initialization method has been used mostly for Population Initialization of TSP.

This paper proposes Mixture Initialization using both Random Initialization method and Induced Initialization method at the same time. (Random Initialization method uses a random generator and Induced Initialization method is based on background knowledge or experience.)

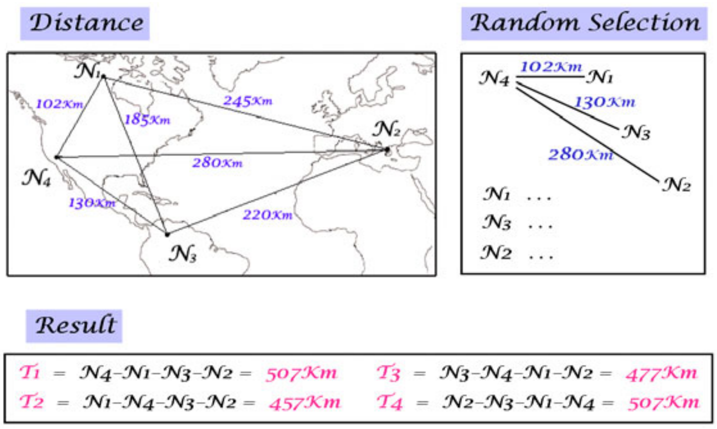


Fig. 3. Mixture Initialization Method

Like Fig 3, one chooses a starting city through random generator, and lists cities orderly from the city with the shortest distance to the city with the farthest distance, referring already-known information about distance among cities. If N cities are listed like this order, $N \times N$ matrix is formed. This matrix is considered as the first population.

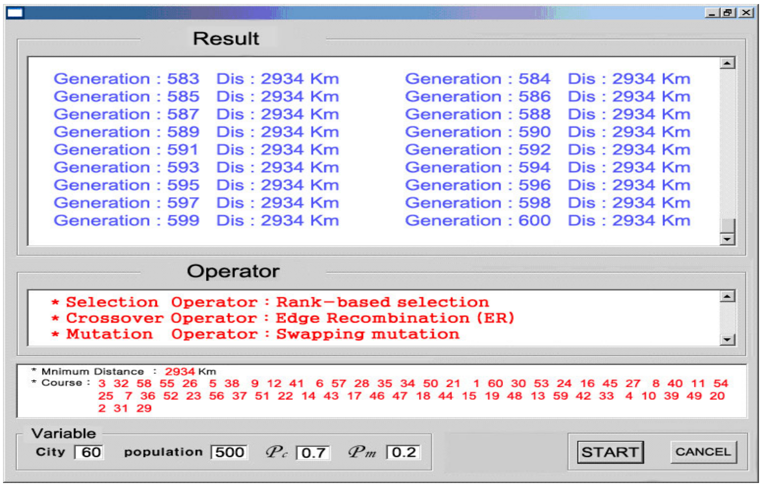
4 Consequence of Experiment

To measure the capability of Mixture Initialization method for TSP, this paper used 2 Selection Operators, 4 Crossover Operators, and 2 Mutation Operators and preserved 2 superior genes by using elitism in each generation. Used values of variables are shown in Table 2.

Table 2. Used values of variables

Variables	Values
Total cities	60
Population size	500
Total generations	600
Probability of Crossover(P_c)	0.7
Probability of Mutation (P_m)	0.2

This experiment was realized by using PowerBuilder6.5 based on Windows2000 in P-4 2.0GHz and data were saved and analyzed by Oracle 8i.



```
Generation : 583  Dis : 2934 Km      Generation : 584  Dis : 2934 Km
Generation : 585  Dis : 2934 Km      Generation : 586  Dis : 2934 Km
Generation : 587  Dis : 2934 Km      Generation : 588  Dis : 2934 Km
Generation : 589  Dis : 2934 Km      Generation : 590  Dis : 2934 Km
Generation : 591  Dis : 2934 Km      Generation : 592  Dis : 2934 Km
Generation : 593  Dis : 2934 Km      Generation : 594  Dis : 2934 Km
Generation : 595  Dis : 2934 Km      Generation : 596  Dis : 2934 Km
Generation : 597  Dis : 2934 Km      Generation : 598  Dis : 2934 Km
Generation : 599  Dis : 2934 Km      Generation : 600  Dis : 2934 Km
```

```
* Selection  Operator : Rank-based selection
* Crossover Operator : Edge Recombination (ER)
* Mutation   Operator : Swapping mutation
```

```
* Mnimum Distance  :  2934 Km
* Course :  3 32 58 55 26 5 38 9 12 41 6 57 28 35 34 50 21 1 60 30 53 24 16 45 27 8 40 11 54
            25 7 36 52 23 56 37 51 22 14 43 17 46 47 18 44 15 19 48 13 59 42 33 4 10 39 49 20
            2 31 29
```

Variable
City 60 population 500 $\mathcal{P}_c$ 0.7 $\mathcal{P}_m$ 0.2

Fig. 4. Application used in experiment

Fig 4 is the TSP program for proving the capability of Mixture Initialization method proposed in this paper.

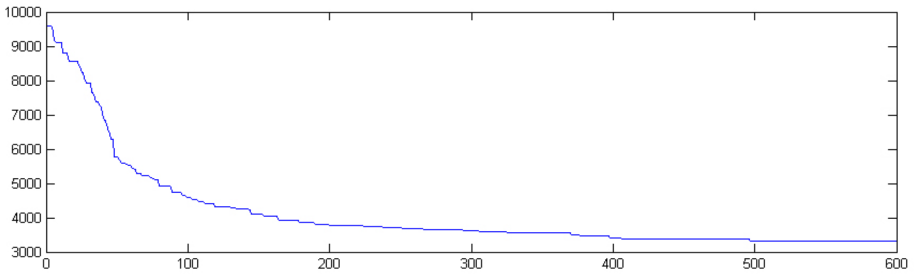


Fig. 5. Optimal value graph using Random Initialization method

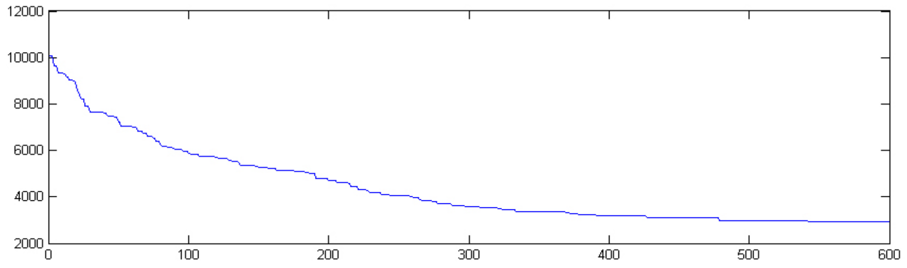


Fig. 6. Optimal value graph using newly-proposed method

Fig 5 and Fig 6 are graphs showing the most superior results of each experiment with Random Initialization method and Mixture Initialization method.

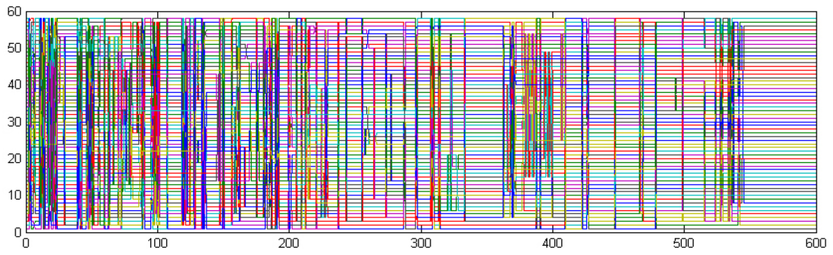


Fig. 7. Process of order change of visiting each city

Fig 7 shows the courses of visiting city in every generation, and the changes of courses of objectives with the best value of each generation among from the 1st generation to the last generation. That is, by Genetic Algorithm based on this method proposed in this paper, as generations are going by, the process of finding the shortest distance is being shown.

Table 3. Comparison of Experiment Result

Selection	Crossover	Mutation	newly-proposed method			
			Min	Gen	Min	Gen
Roulette Wheel	PMX	Inversion	3300	501	3458	554
		Swapping	3285	499	3390	509
	ER	Inversion	3598	519	3380	487
		Swapping	3478	589	3928	508
	Multi-Point Crossover	Inversion	3470	557	**3315**	**496**
		Swapping	3901	457	3903	485
	One-Point Crossover	Inversion	4002	576	4488	587
		Swapping	3502	499	3702	578
Rank-base Selection	PMX	Inversion	2967	501	3330	499
		Swapping	4157	545	4098	518
	ER	Inversion	3968	511	4012	547
		Swapping	**2934**	**546**	3123	579
	Multi-Point Crossover	Inversion	3310	497	3377	505
		Swapping	4123	457	4225	471
	One-Point Crossover	Inversion	4411	513	4507	555
		Swapping	3442	518	3518	497

ER : Edge Recombination Min : Minimum Distance Gen : Generation

Table 3 is the results of experiment with Mixture Initialization method proposed in this paper. As you know from this table, the shortest distance was 3315Km when various operators were applied to the value of initialization produced through Random Initialization method. But, better result, 2934Km, was obtained through Mixture Initialization method proposed in this paper. Also, it is shown that the distances by newly-proposed Initialization method are on the whole shorter in most operators. Experiment using various operators achieved improvement rate from max 12.9% to min -1.9% and about 4.0% on an average.

5 Conclusion

To solve the optimization problem, this paper proposed Mixture Initialization method using Random Initialization method and Induced Initialization method at the same time.

It is known that Mixture Initialization method helps the capability improved more than Random Initialization method or Induced Initialization method. This is because Mixture Initialization method uses already-known distances among cities so makes it high probability for only superior gene to be chosen. With this method, average improvement rate, approximately 4.0% was obtained.

Mixture Initialization method is applicable to existing various operators of GA and this method proved the improvement of capability. Mixture Initialization method can be used valuably in fields such as physical distribution, the network of telephone, the optimization of network etc. classified in the optimization problem.

Another study should be progressed constantly to solve the optimization problems with more complex structures or apply newly-will-be-proposed operators.

References

1. Jin gang gou.: Genetic Algorithm and the application, kousa (2000)
2. Goldberg, D.: Genetic Algorithms in search, Optimization, and Machine Learning Addison Wesley, Reading, MA (1989)
3. K.D. Boese, "Cost Versus Distance In the Traveling Salesman Problem", Technical Report CSD-950018, UCLA (1995)
4. Mun byoung ro.: Genetic Algorithm, duyangsa(2003)
5. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag (1992)
6. Grefenstette, J. Gopal,R. Rosmaita, B and Gucht, D.: "Genetic Algorithms for the Traveling Salesman Problem", Proc. the 1st Inter. Conf. on GAs and Their Applications (1985)
7. Whitley, D. Starkweather, T and Fuquay, D.: "Scheduling problems and traveling salesman: the genetic edge recombination and operator", Proc. Third Int. Conf. G As (1989) 133-140
8. D. Whitley, T. Starkweather, and D. Fuquay. Scheduling Problems and the Traveling Salesman: the Genetic Edge Recombination Operator. Proc. Third Int. Conf. on Genetic Algorithms and Their Applications (1989)
9. Chuan-Kang Ting: Improving Edge Recombination through Alternate Inheritance and Greedy Manner, EvoCOP 2004, LNCS 3004 (2004) 210–219

Optimized Fuzzy Decision Tree Using Genetic Algorithm

Myung Won Kim[1] and Joung Woo Ryu[2]

[1] School of Computing, Soongsil University, 511, Sangdo-Dong, Dongjak-Gu, Seoul, Korea
mkim@comp.ssu.ac.kr
[2] Intelligent Robot Research Division Electronics and Telecommunications Research Institute
161 Gajeong-dong, Yuseong-gu, Daejeon, Korea
ryu0914@etri.re.kr

Abstract. Fuzzy rules are suitable for describing uncertain phenomena and natural for human understanding and they are, in general, efficient for classification. In addition, fuzzy rules allow us to effectively classify data having non-axis-parallel decision boundaries, which is difficult for the conventional attribute-based methods. In this paper, we propose an optimized fuzzy rule generation method for classification both in accuracy and comprehensibility (or rule complexity). We investigate the use of genetic algorithm to determine an optimal set of membership functions for quantitative data. In our method, for a given set of membership functions a fuzzy decision tree is constructed and its accuracy and rule complexity are evaluated, which are combined into the fitness function to be optimized. We have experimented our algorithm with several benchmark data sets. The experiment results show that our method is more efficient in performance and complexity of rules compared with the existing methods.

Keywords: fuzzy classification rule, fuzzy decision tree, genetic algorithm, Optimization.

1 Introduction

A decision tree such as ID3 and C4.5 is one of the most widely used classification methods [1], [2], [3]. One of the difficult problems in classification is to handle quantitative data appropriately. Conventionally, a quantitative attribute domain is divided into a set of crisp regions and by doing so the whole data space is partitioned into a set of (crisp) subspaces (hyper-rectangles), each of which corresponds to a classification rule describing that a sample belonging to the subspace is classified into the representative class of the subspace. However, such a crisp partitioning is not natural to human and inefficient in performance because of the sharp boundary problem. Recently, fuzzy decision trees have been proposed to overcome this problem [4], [5]. It is well known that the fuzzy theory not only provides natural tool for describing quantitative data but also generally produces good performance in many applications. However, one of the difficulties with fuzzy decision trees is determining an appropriate set of membership functions representing fuzzy linguistic terms. Usually membership functions are given manually, however, it is difficult for even an expert to determine an appropriate set of membership functions when the volume and dimensionality of data are large.

In this paper we investigate combining the fuzzy theory and the conventional decision tree algorithm for accurate and comprehensible classification. We propose an efficient fuzzy rule generation method using the fuzzy decision tree (FDT) algorithm, which integrates the comprehensibility of decision trees and the expressive power of fuzzy sets. We also propose the use of genetic algorithm for optimal set of fuzzy rules by determining an appropriate set of fuzzy sets for quantitative data. In our method for a given fuzzy membership function a fuzzy decision tree is constructed and it is used to evaluate classification accuracy and rule complexity. Fuzzy membership functions evolve so that they optimize the fitness function combining both classification accuracy and rule complexity.

This paper is organized as follows: in Section 2 we describe fuzzy rules and fuzzy inference we use for classification, followed by fuzzy decision tree construction. In Section 3, we describe the use of genetic algorithm for determining an appropriate set of membership functions. In Section 4, we briefly describe some related works [6]. In Section 5 we describe experiments of our algorithm with a variety of the benchmark data sets. In Section 6 we conclude the paper.

2 Fuzzy Inference

2.1 Fuzzy Classification Rules

We use a simple form of fuzzy rules and inference for better human understanding. Each fuzzy rule is of the form "*if A then B*" where *A* and *B* are called an antecedent and a consequent, respectively. In our approach the antecedent is simple conditions conjoined while the consequent is "*Class is k.*" A simple condition is of the form "*Att is Val*" where *Att* represents an attribute name and *Val* represents a value of the attribute. Each fuzzy rule is associated with a *CF* (Certainty Factor) to represent the degree of belief that the consequent is drawn from the antecedent satisfied. Rule (1) is a typical form of fuzzy classification rules used in our approach.

$$R_j : if\ (A_{i1}\ is\ V_{il})\ and\ (A_{i2}\ is\ V_{i2}) \cdots and\ (A_{im}\ is\ V_{im})\ then\ 'Class'\ is\ k(CF_i) \tag{1}$$

In the rule A_{ik} represents an attribute and V_{ik} represents a fuzzy linguistic term represented by a fuzzy set associated with attribute A_{ik}. Application of the rule to a sample *X* results the confidence with which *X* is classified into class *k* given that the antecedent is satisfied. In this paper among a variety of fuzzy inference methods we adopt the standard method as described in the following.

For a given sample *X*, the confidence of class *k* is obtained as

$$conf_k(X) = \max_{R_j \in R(k)} \left\{ \left(\min_j \mu_{V_{ij}}(x_j) \right) \cdot CF_i \right\} \tag{2}$$

Where x_j is the value for attribute A_{ij} of *X*

In Equation (2) $\mu_V(x)$ represents the membership degree that *x* belongs to fuzzy set *V*, *R*(*k*) represents the set of all rules that classify samples into class *k* (their consequent parts are '*Class* is *k*') . The class of the maximum $Conf_k(X)$ is the final classification of *X*.

2.2 Fuzzy Decision Tree

Optimized fuzzy rules are often derived from fuzzy decision tree. A fuzzy decision tree is similar to a (crisp) decision tree. It is composed of nodes and arcs representing attributes and attribute values or value sets, respectively. The major difference is that in a fuzzy decision tree each arc is associated with a fuzzy linguistic term, which is usually represented by a fuzzy set. Also in a fuzzy decision tree a leaf node represents a class and it is associated with a certainty factor representing the confidence of the decision corresponding to the leaf node. In a fuzzy decision tree a decision is made by aggregating the conclusions of multiple rules (paths) fired as Equation (2) describes while in a crisp decision tree only a single rule is fired for a decision.

Let us assume that $A_1, A_2, , , A_d$ represent attributes in consideration for a given data set, where d represents the dimension of the data. The whole data space W can be represented as $W = U_1 \times U_2 \times ... \times U_d$, where U_i represents the domain of attribute A_i. A sample X can be represented as a point in W as $X = (x_1, x_2, , , x_d)$, where $x_i \in U_i$. In a fuzzy decision tree each arc (l, m) from node l to node m is associated with a fuzzy set $F(l, m)$ representing a fuzzy linguistic term as a value of the attribute selected for node l. Suppose we have a fuzzy decision tree and let n be a node and P_n be the path from the root node to node n in the tree. Then we can consider that node n is associated with a fuzzy subspace W_n of W defined as follows.

$$W_n = S_1 \times S_2 \times , , , \times S_d$$

$$\text{where } S_i = \begin{cases} F(l,m) & \text{if } A_i = att(l) \text{ for an arc } (l,m) \text{ in } P_n \text{ ;} \\ U_i & \text{otherwise.} \end{cases}$$

Here, $F(l,m)$ is a fuzzy set corresponding to arc (l,m) and $att(l)$ represents the attribute selected for node l. Let $v_n(X)$ represent the membership that X belongs to W_n, then we have the following according to the above definitions:

$$\begin{aligned} v_n(X) &= \mu_{W_n}(X) \\ &= \min_{(l,m) \text{ in } P_n} \mu_{F(l,m)}(x_i) \text{ where } A_i = att(l). \end{aligned}$$

$$E(m, A_i) = \sum_{n \in child(m, A_i)} (q_{mn} I_n) \tag{3}$$

$$I_n = -\sum_{k \in C} \left(p_n(k) \log_2 p_n(k) \right) \tag{4}$$

$$p_n(k) = \sum_{Y \in D_k} v_n(Y) / \sum_{X \in D} v_n(X) \tag{5}$$

$$q_{mn} = \sum_{Y \in D} v_n(Y) / \sum_{X \in D} v_m(X) \tag{6}$$

Now we define some important parameters that we use in fuzzy decision tree construction. The entropy of attribute A_i for node *m*, $E(m, A_i)$ is a measure indicating how good it is if attribute A_i is selected as the test attribute for node *m* and it is defined as in Equation (3). Here, *child*(m, A_i) represents the set of children nodes of node *m,* each of which corresponds to a value of attribute A_i, *C* denotes the set of classes, and *D* and D_k denote the set of training samples and the set of training samples of class *k*, respectively. In the above equations, *n* is a child node of node *m,* which corresponds to a value of the attribute selected for node *m,* and arc (*m*, *n*) corresponds to a fuzzy membership function for attribute A_i. Equation (4) calculates the entropy for node *n*, Equation (5) represents the probability that node *n* represents class *k*, and Equation (6) represents the degree that a sample belongs to the subspace corresponding to node *n* compared with other sibling nodes for a given attribute selected for node *m.*

Our fuzzy decision tree construction algorithm is as follows.

<Step 1>
Starting with the root node, continue to grow the tree as following until the termination conditions are satisfied. If one of the following conditions is satisfied, then make node *m* a leaf node.

(1) $\frac{1}{|D|}\sum_{X\in D} v_m(X) \le \theta_e$

(2) $\sum_{Y\in D_{k^*}} v_m(Y) / \sum_{X\in D} v_m(X) \ge \theta_d$ where $k^* = \arg\max(\sum_{Y\in D_k} v_m(Y))$

(3) no more attributes are available $CF = \dfrac{\sum_{Y\in D_{k^*}} v_m(Y)}{\sum_{X\in D} v_m(X)}$

In condition (2) class *k** represents the representative class of the corresponding fuzzy subspace. In this case it is the leaf node whose class is *k** and the associated *CF* is determined by the left hand side of the inequality in condition (2). Otherwise,

<Step 2>
(1) Let $E(m, A_{i^*}) = \min_i(E(m, A_i))$.

(2) For each fuzzy membership functions of attribute A_{i^*}, make a child node of node *m.*

(3) Go to Step 1 and apply the algorithm to all newly generated nodes, recursively.

Node expansion ((2) in Step 2) corresponds to partitioning the fuzzy subspace corresponding to node *m* into fuzzy subspaces each of which corresponds to a value of the attribute selected for the node. In Step 1 the threshold parameters θ_e and θ_d determine when partitioning terminates. Condition (1) prohibits further partitioning sufficiently sparse fuzzy subspaces while condition (2) prohibits further partitioning fuzzy subspaces having sufficiently large portion of a single class samples. The parameters θ_e and θ_d are used to control overfitting by prohibiting too much detail rules to be generated.

After constructing a fuzzy decision tree, we name each fuzzy membership function with an appropriate linguistic term.

3 Membership Function Optimization Using Genetic Algorithm

In fuzzy rules membership functions are important since they affect both of accuracy and comprehensibility of rules. However, membership functions are usually given manually and it is difficult even for an expert to determine an appropriate set of membership functions when the volume and dimensionality of data are large. In this paper, we propose the use of genetic algorithm to determine an optimal set of membership functions for a given classification problem. For simplicity we adopt a triangular membership function and it is represented by a triple of real numbers (l, c, r), where l, c and r represent the left, the center and the right points of the triangular membership function, respectively.

3.1 Genetic Algorithm

Genetic algorithm is an efficient search method simulating natural evolution, which is characterized by survival of the fittest.

Our fuzzy decision tree construction using genetic algorithm is following.

(1) Generate an initial population of chromosomes of membership functions;
(2) Construct fuzzy decision trees using the membership functions;
(3) Evaluate each individual set of membership functions corresponding to a chromosome by evaluating the performance and tree complexity of its corresponding fuzzy decision tree;
(4) Test if the termination condition is satisfied;
(5) If yes, then exit;
(6) Otherwise, generate a new population of chromosomes of membership functions by applying genetic operators, and go to (2).

We use genetic algorithm to generate an appropriate set of membership functions for quantitative data. Membership functions should be appropriate in the sense that they result in a good performance and they result in as simple a decision tree as possible.

In our genetic algorithm a chromosome is of the form $<\phi_1,\phi_2,...,\phi_d>$ where ϕ_i represents a set of membership functions associated with attribute A_i, which is, in turn, of the form $\{f_1(A_i), f_2(A_i),,,f_l(A_i)\}$ where $f_j\ (A_i)$ represents a triplet of membership function for attribute A_i. The number of membership functions for an attribute may not necessarily be fixed.

3.2 Genetic Operations and Evaluation

We have genetic operations such as crossover, mutation, addition, and merging as described in the following.

1) ***Crossover:*** it generates new chromosomes by exchanging the whole sets of membership functions for a randomly selected attribute of the parent chromosomes.
2) ***Mutation:*** we combine random mutation and heuristic mutation. In random mutation membership functions randomly selected from a chromosome are

mutated by adding Gaussian noise to the individual membership functions. In heuristic mutation membership functions are adjusted to classify correctly the misclassified data.

3) ***Addition:*** for any attribute in a chromosome, the set of associated membership functions are analyzed and new appropriate membership functions are added if necessary. For example, when some attribute values are not covered or poorly covered by the current membership functions, new membership functions are added to cover those values properly.
4) ***Merging:*** any two close membership functions of any attribute of a chromosome are merged into one.

We apply the roulette wheel method for selecting candidate chromosomes for the crossover operation. We also adopt the elitism in which the best fit chromosome in the current population is selected for the new population.

If membership functions are determined for each attribute, we generate an fuzzy decision tree according to the algorithm described in Section 2.3. We use the performance of the generated fuzzy decision tree in fitness evaluation of a chromosome. Suppose a fuzzy decision tree $\tau(e)$ is generated from the membership functions represented by chromosome e. The fitness score of chromosome e is given by

$$Fit(e) = wP(\tau(e)) - (1-w)C(\tau(e)),\ 0 \le w \le 1 \tag{7}$$

where $P(\tau(e))$ represents the performance of decision tree $\tau(e)$ and $C(\tau(e))$ represents the complexity of $\tau(e)$, measured in terms of the number of nodes of $\tau(e)$ in this paper.

We also can use genetic algorithm to generate fuzzy rules by evolving the form of rules (selection of attributes and their associated values) and membership functions simultaneously. However, genetic algorithm is generally time-consuming and our method trades off between classification accuracy and computational time.

4 Related Works

A fuzzy decision tree is more powerful, efficient, and natural to human understanding, particularly compared with crisp decision trees such as in ID3 and C.4.5.

[4] proposed fuzzification of CART (Classification And Regression Tree) using sigmoidal fuzzy splits replacing Boolean tests of a crisp CART tree and applying the back-propagation algorithm to learn parameters associated with fuzzy splits to optimize the global impurity of the tree. However, the structure of fuzzy decision tree proposed in [4] is different from that we propose in this paper and it is difficult to directly compare them.

[5] proposed a method for automatically generating fuzzy membership functions for continuous valued attributes based on the principle of maximum information gain in cut point selection for each attribute.

Hisao Ishibuchi proposed the evaluation standard to extract candidate fuzzy classification rules and the genetic rule selection using evolutionary multi-objective optimization to select useful fuzzy classification rules from among candidate fuzzy classification rules [7]. The error correcting learning method is proposed to learn the membership functions by generating highly accurate rules from predefined membership functions in [8]. This method adjusts membership functions of a fuzzy classification rules that cause misclassification of an input datum in an error-correction manner. Another method in [9] predefines triangular membership functions, divides the space, then generates fuzzy classification rules, and finally learns the certainty factor of the generated rules to adjust classification boundary to increase accuracy of the rule.

J. Abonyi et. al. suggested a method of generating rules by generating initial fuzzy classification rules according to class distribution without predetermining membership functions and optimizing the conditional terms of the rules with evolution algorithm so that the initial rules can be generated accurately and concisely. In their approach they assume that each class is described by a single, compact construct in the input space. Therefore the number of clusters is limited to the number of classes. If this is not the case, accurate rules cannot be generated because there are an insufficient number of rules. To supplement this problem in [10], the input space with the C4.5 decision tree algorithm is divided and one initial fuzzy classification rule for each grid area is generated.

A fuzzified ID3 (FID3) have been proposed in [5] and [6] and fuzzy membership functions are generated automatically based on based on the principle of maximum information gain in selection of cut points for each continuous valued attribute. We compared our method with FID3 in [11].

5 Experiment

We have conducted several experiments with various benchmark data sets in the UCI machine learning databases [12] (see Table 1). We cited simulation results in [13] and [10] for comparison of our method with the existing ones. Our experiments are described in the following.

First, we compared the accuracy and compactness of the fuzzy rules generated with different weights w used in (8) and the results are shown in Table 2. The accuracy is represented as the recognition rate and the compactness is represented as the number of rules and the total number of terms in the antecedents of the rules. In our fuzzy decision tree, two prepruning parameters θ_d (density) and θ_e (exception) are used to determine when we terminate growing a tree. The density threshold prohibits generating rules for sparse fuzzy subspaces while the exception threshold prohibits generating rules for minor classes having insufficient number of data in a fuzzy

Table 1. Benchmark data sets

Data Name	# Data	# Attributes	# Classes	# Data Per Class
iris	150	4	3	(50, 50, 50)
pima	768	8	2	(500, 268)
bcw[1]	683	9	2	(444, 239)
glass	214	9	6	(70, 76, 17, 13, 9, 29)
ionosphere	351	34	2	(126, 225)
thyroid	215	5	3	(150, 35, 30)
heart	270	13	2	(150, 120)

[1] Wisconsin Breast Cancer data.

Table 2. Perpormance comparison for varying weights

Data	w = 0.1			w = 0.5			w =0.9		
	Accuracy (%)	# of Rules	# of Terms	Accuracy (%)	# of Rules	# of Terms	Accuracy (%)	# of Rules	# of Terms
iris	97.3	3	4	97.3	3	4	98.0	3	4
pima	75.0	2	2	75.8	17	55	80.1	74	313
bcw	97.1	2	4	97.1	2	5	97.7	4	14
glass	79.0	54	114	80.8	65	153	95.3	182	441
thyroid	94.9	3	5	97.7	8	21	97.2	11	36
heart	78.9	3	10	82.6	7	29	81.9	5	22
Average	87.0	11.2	23.2	88.6	17.0	44.5	91.7	46.5	138.3

subspace. In general the prepruning parameters are used to control overfitting by prohibiting too much detail rules to be generated. Table 2 shows the results when we have θ_d = 0.02 and θ_e=1.0. In the experiment, the population size was 20, the maximum number of generations was set to 100, and the crossover probability and the mutation probability were 0.9 and 0.1, respectively.

As w is close to 1.0 in (8), it places more weight on the accuracy of rules than the complexity of rules. Each time the weight increases the average accuracy increased by about 2.0% but the average numbers of rules and terms get nearly doubled. Accordingly the weight was set to 0.1 to satisfy both accuracy and compactness.

In the next experiment, the weight is fixed to 0.1 while the prepruning parameters θ_d and θ_e were varied as in Table 3. For the glass data, as θ_e was lowered from 1.0 to 0.85 and θ_d increased from 0.02 to 0.04, the numbers of rules and terms decreased by about 50.0% so that it was possible to generate rules with better compactness; however, due to the difficulty of classification with a small number of data, the accuracy decreased by about 11.0%.

Table 4 compares the performance of our method with those of two existing methods. [13] proposed a method for generating a fuzzy decision tree with heuristic information using predefined membership functions. The table shows the results of determining three membership functions for each attribute in advance and generating fuzzy classification rules using the proposed method. Another existing method proposed in [10] uses evolutionary algorithm to evolve the initial fuzzy classification rules into accurate and concise rules. Initial fuzzy classification rules were generated by C4.5.

Table 3. Perpormance comparison with varying prepruning parameters

Data	w = 0.1					w = 0.1				
	Accuracy (%)	# of Rules	# of Terms	θ_d	θ_e	Accuracy (%)	# of Rules	# of Terms	θ_d	θ_e
iris	97.3	3	4	0.02	1.00	98.0	3	4	0.06	1.00
bcw	97.1	2	4	0.02	1.00	96.6	2	4	0.01	1.00
pima	75.0	2	2	0.02	1.00	75.7	3	4	0.01	0.85
thyroid	94.9	3	5	0.02	1.00	93.5	3	3	0.04	0.85
heart	78.9	3	10	0.02	1.00	81.1	3	9	0.03	0.85
glass	79.0	54	114	0.02	1.00	69.6	26	48	0.04	0.85
Average	**87.0**	**11.2**	**23.2**	**-**	**-**	**85.8**	**6.7**	**12.0**	**-**	**-**

Fuzzy classification rules generated in [10] produce a bit lower accuracy but the better compactness than those generated in [13]. However, when comparing the proposed method in [10] with our method, the average accuracy is about 3.0% lower, the number of rules is similar, and there are twice as many antecedent terms. In the generated fuzzy decision tree, our method generated the tree less depth and more breadth than [10] because our method creates the tree by a global discretization method while [10] used a local discretization method. Therefore, for the glass data, our method has more rules but fewer antecedent terms than [10].

The efficiency of our method was examined by comparing it with the existing method in terms of accuracy and conciseness of the fuzzy classification rules generated with several benchmark data sets.

Table 4. Performance comparison of our method with two existing methods

Data	Accuracy (%)			# of Rules			# of Terms		
	[13]	[10]	Our method	[13]	[10]	Our Method	[13]	[10]	Our method
iris	97.0	96.1	98.0	9.7	3.0	3.0	16.0	4.0	4.0
pima	80.0	73.0	75.6	34.7	11.2	3.0	53.2	40.0	4.0
ionosphere	-	86.4	91.4	-	3.4	3.0	-	10.2	6.0
glass	-	66.0	69.6	-	19.2	26.0	-	90.8	48.0
bcw	-	96.8	96.6	-	2.0	2.0	-	3.0	4.0
Average	**-**	**83.7**	**86.2**	**-**	**7.8**	**7.4**	**-**	**29.6**	**13.2**

6 Conclusion

Our method to generate fuzzy classification rules focus on both the accuracy and the compactness. The method optimizes membership functions using evolutionary algorithm so that the generated fuzzy rules perform well. Initial membership functions for evolutionary algorithm are generated by a supervised clustering algorithm in which the number of clusters is determined according to class distribution using evolutionary algorithm. The experiment results show that our method is more efficient in classification accuracy and compactness of rules compared with the existing methods. We consider adopting measures proposed in [14] for elaborate evaluation of fuzzy decision trees.

Acknowledgments. This work was supported by Korean Research Foundation Grant (KRF-2004-041-D00627).

References

1. Fayyad, U., Mannila, H., and Piatetsky-Shapiro, G., Data Mining and Knowledge Discovery. Kluwer Academic Publishers (1997)
2. Quinlan, J.R., Induction of decision trees, Machine Learning, 1(1), (1986) 81-106

3. Janikow, C.Z., Fuzzy Decision Tree : Issues and Methods, IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics, vol. 28, no. 1, (1998) 1-14
4. Suárez, A. and Lutsko, J.F., Globally Optimal Fuzzy Decision Trees for Classification and Regression, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, no. 12, (1999) 1297-1311
5. Zeidler, J. and Schlosser M., Continuous-Valued Attributes in Fuzzy Decision Trees., Proc. of the 6th Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems, (1996) 395-400
6. Janikow, C.Z. and Faifer, M., Fuzzy Partitioning with FID3.1, Proc. of the 18th International Confernece of the North American Fuzzy Information Processing Society, IEEE 1999, (1999) 467-471
7. Ishibuchi, H. and Yamamoto, T., Effects of Three-Objective Genetic Rule Selection on the Generalization Ability of Fuzzy Rule-Based Systems, Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, vol. 2632, (2003) 608-622
8. Nakashima, T., Nakai, G. and Ishibuchi, H., Improving the Performance of Fuzzy Classification Systems by Membership Function Learning and Feature Selection, FUZZ-IEEE '02 Proceedings of the 2002 International Conference on vol.1, (2002) 488-493
9. Nozali, K., Ishibuchi, H. and Tanaka, H., Adaptive Fuzzy Rule-Based Classification Systems, IEEE Transactions on Fuzzy Systems, vol.4, no.3, (1996) 238-250
10. Abonyi, J., Roubos, J. and Szeifert, F., Data-driven Generation of Compact, Accurate, and Linguistically Sound Fuzzy Classifiers based on a Decision-tree Initialization, International Journal of Approximate Reasoning, 31(1), (2003) 1-21
11. Kim, M. W. and Ryu, J. W., Optimized Fuzzy Classification using Genetic Algorithm, International Conference on Fuzzy Systems and Knowledge Discovery (FSKD '05), Lecture Notes in Computer Science, vol. 3613, (2005) 392-401
12. http://www.ics.uci.edu/~mlearn/MLRepository.html
13. Wang, X. Z., Yeung, D. S. and Tsang, E. C. C., A Comparative Study on Heuristic Algorithms for Generating Fuzzy Decision Trees, IEEE Transactions on Systems, Man, and Cybernetics, vol. 31, no. 2, (2001) 215-226
14. Mitra, S., Konwar, K. M. and Pal, S. K., Fuzzy Decision Tree, Linguistic Rules and Fuzzy Knowledge-Based Network: Generation and Evaluation, IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, vol. 32, no. 4, (2002) 328-339

A Genetic-Inspired Multicast Routing Optimization Algorithm with Bandwidth and End-to-End Delay Constraints

Sanghoun Oh[1], ChangWook Ahn[2], and R.S. Ramakrishna[1]

[1] Department of Information and Communications
Gwangju Institute of Science and Technology (GIST)
{oosshoun, rsr}@gist.ac.kr
[2] Communication Lab.
Samsung Advanced Institute of Technology (SAIT)
cwan@evolution.re.kr

Abstract. This paper presents a genetic-inspired multicast routing algorithm with Quality of Service (i.e., bandwidth and end-to-end delay) constraints. The aim is to efficiently discover a minimum-cost multicast tree (a set of paths) that satisfactorily helps various services from a designated source to multiple destinations. To achieve this goal, state of the art genetic-based optimization techniques are employed. Each chromosome is represented as a tree structure of Genetic Programming. A fitness function that returns a tree cost has been suggested. New variation operators (i.e., crossover and mutation) are designed in this regard. Crossover exchanges partial chromosomes (i.e., sub-trees) in a positionally independent manner. Mutation introduces (in part) a new sub-tree with low probability. Moreover, all the infeasible chromosomes are treated with a simple repair function. The synergy achieved by combing new ingredients (i.e., representation, crossover, and mutation) offers an effective search capability that results in improved quality of solution and enhanced rate of convergence. Experimental results show that the proposed GA achieves minimal spanning tree, fast convergence speed, and high reliability. Further, its performance is better than that of a comparative reference.

1 Introduction

In multi-hop networks such as Internet, Wireless Mesh, and Mobile Ad-hoc networks, the design of multicast routing algorithms (MRAs) with a view to finding a minimal spanning tree (MST) with satisfying Quality of Service (QoS) (i.e., bandwidth, end-to-end delay, and delay-jitter) from a specified source to a set of destinations is a very important enterprise [1,2,7,14]. This is because MRAs have significant influences on the capacity of networks. In recent days, MRAs have attracted due to attention from the research community because multicast services (e.g., video broadcasting, multimedia teleconference, and massive mailing) are becoming common place [1,2,3,4,5,9,10,11,12,13,14]. The point-to-multipoint services need to transmit multiple copies of a message from one node (i.e., a source) to a subset of nodes (i.e., destinations). In other words, they fall within the purview of multiple destination routing (MDR) problems [2,5,12].

In the MDR problems, a good routing algorithm finds low-cost tree that connects all of the routers that have attached host members of multicast group and then route packets along this tree from a source to multiple destinations according to the multicast routing tree [1,2,3,4,5,6,10,11,12,13,15,16,17]. Finding the link tree with the minimum cost is known as Steiner tree problem in graph theory [6,9,16,17]. On the other hand, there are two reasons why a tree structure efficiently supports multicast routing [14]. First, the data can be transmitted in parallel (to the multiple destinations) along the branches of the tree. Second, a minimal number of copies of the data are transmitted in this way. From the above reasons, many MRAs have been evolved to construct such tree with minimum cost. These approaches can be classified into two categories: *Group-shared tree*, where only a single routing tree is constructed for the entire multicasting group, and *Source-based tree*, where a tree is constructed for each individual sender in the group. In general, the former is efficient for large-scale stable networks; and the latter is useful for small-scale dynamic networks. Due to the increasing importance of wireless networks, the source-based tree approach appears to be more attractive. Moreover, multicast routing problems consist of a single QoS constraint [3,10,11,16] or multiple QoS constraints [4,6,13]. In wireless networks, multiple-QoS scenario is more promising.

In this paper, a genetic-inspired MRA that discovers a multicast tree that satisfies the bandwidth and end-to-end delay constraints (i.e., multiple QoS constraints) is presented. The focus is on determining a multicast routing tree, (in the source based tree category), from a designated source to a set of destinations with strict end-to-end delay requirements (in multicast group) and specified bandwidth (between adjacent nodes). MDR-friendly encoding and variation operators are devised. In the encoding, a sequence of integers (i.e., nodes' IDs) forms a chromosome. Each chromosome represents a (spanning) tree and its length is variable. Every locus (of the chromosome) is related to an order of the node (indicated by the gene of the locus) in multiple routing paths. The crossover generates new chromosomes by exchanging partial chromosomes (i.e., sub-trees) without positional consistency of potential crossing site between two chromosomes. It maximally increases the exploratory power of the proposed algorithm. The mutation that promotes genetic diversity introduces a new sub-tree from the mutation node to selected destination. It maintains a fair measure of genetic diversity (of population) so as to escape from local optima. Finally, a repair function treats infeasible chromosomes possibly generated after crossover. The proposed algorithm discovers a minimum-cost multicast tree while satisfying the bandwidth and end-to-end delay constraints.

The rest of the paper is organized as follows. Section 2 describes a mathematical model for the representing multicast routing problem. The proposed MRA is presented in Section 3. Section 4 shows experimental results. The paper concludes in Section 5.

2 Problem Formulation

A communication network is modeled as an undirected, connected weighted graph $G = (V, E)$, where V and E denote the set of nodes and the set of connected links, respectively. Further, $|V|$ and $|E|$ denote the number of nodes and links in the network respectively. Each link e_{ij} connects two nodes i and j in V. It is characterized by an ordered triple (B_{ij}, D_{ij}, C_{ij}) representing capability of bandwidth, delay, and cost between nodes i and j. An example is shown in Fig. 1. A multicast tree is defined by $T = (V_T, E_T)$, where $V_T \subseteq V$, $E_T \subseteq E$, and $T \subseteq G$, and there exists a path $P_T(s, d_k)$ from the source node s to each destination node $d_k \in \mathcal{D} = \{d_1, \cdots, d_n\}$ in T. Here, n is the number of destinations.

This paper considers two QoS constraints: the bandwidth constraint between adjacent nodes and end-to-end delay constraint from a source to each destination. These are generally employed in real-world applications:

– **QoS Constraints**
  1. *Bandwidth*: It is a basic requirement for transmitting information, and directly influences the balance of network load and the routing request success ratio for large-scale multicast sessions (i.e., video on demand). It is required that the minimum value of link bandwidth in the multicast tree T must be greater than the required bandwidth (B_{req}), along the path from a source node s to each destination node $d \in \mathcal{D}$. That is,

$$B_T = \min_{\{i,j|e_{ij} \in E_T\}} B_{ij} \geq B_{req}. \tag{1}$$

  2. *Delay bound*: It represents an upper bound on the acceptable end-to-end delay along each path from a source to a set of destinations. One should ensure that the maximum value of path delays (from a source to each destination) is smaller than the required path delay (D_{req}), i.e.,

$$D_T = \max_{\{k|k \in \mathcal{D}\}} \left(\sum_{\substack{\{i,j|e_{ij} \in \\ P_T(s,d_k)\}}} D_{ij} \right) \leq D_{req}. \tag{2}$$

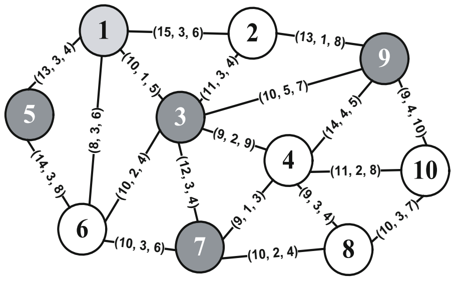


Fig. 1. Network topology

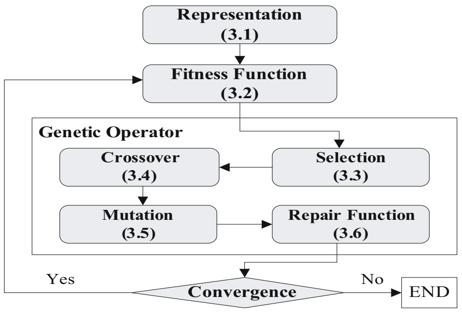


Fig. 2. Overall procedure of the proposed algorithm

– **Objective Function**
  Tree Cost: The total cost of multicast tree must be minimized (while satisfying the above two QoS constraints):

$$\min \sum_{\{i,j|e_{ij}\in T\}} C_{ij}. \tag{3}$$

3 Proposed Genetic Algorithm

This section describes a genetic algorithmic multicast routing technique with bandwidth and end-to-end delay constraints. It consists of several key components: representation, fitness function, selection, variation operators (i.e., crossover and mutation), and repair function. Chromosomes are expressed by tree data structure. Fitness function returns the sum of link costs between all the connected nodes. Variation operators (i.e., crossover and mutation) efficiently promote the search capability, and repair function fixes the infeasible solutions. Note that every step takes account of the bandwidth and end-to-end delay constraints. Fitness function and genetic operators iterate until the termination conditions are satisfied. Overall procedures of the algorithm are depicted in Fig. 2.

3.1 Representation

We design an efficient chromosome representation scheme; a tree structure of the genetic programming (GP) [1]. Variable-length chromosomes encoded by positive integers are employed, and the integers (of genes) express IDs of nodes. Each locus of the chromosome represents the node's order in the corresponding routing path. However, the length of a chromosome is not more than the total number of nodes. The procedure assembles a multicast tree while taking into account the connection between nodes. The gene of the first locus is always reserved for the source node. A chromosome (i.e., a set of routing paths) encodes the IDs of the nodes from the source to each of destinations. Fig. 3 shows an example of the proposed encoding method in the context of the Fig. 1. The first locus in each chromosome is assigned by the source node set to '1'. Each multicast tree extends the source s to the set of destinations $\mathcal{D} = \{3, 5, 7, 9\}$. The routing paths in the example are expressed as follows: the first path is $(1 \rightarrow 2 \rightarrow 3)$, the second path is $(1 \rightarrow 5)$, the third path is $(1 \rightarrow 5 \rightarrow 6 \rightarrow 7)$, and the last path is $(1 \rightarrow 2 \rightarrow 9)$.

In the course of constructing the multicast tree, the nodes that are already included in the current (sub-)tree are excluded, thereby avoiding reentry of the same node. Moreover, chromosomes are encoded under the (link) bandwidth and (path) delay constraints. In case they are violated, the encoding process is usually repeated in part so as to satisfy the requirements. Thus, it effectively constructs a multicast routing tree whose root is the source node and whose terminal nodes are a specified set of destinations with bandwidth constraint (B_{req}) and end-to-end delay constraint (D_{req}). In the chromosome in Fig. 3, it is seen that $B_T = 10$ and $D_T = 7$. The number of branches (of a tree) is proportional to the number of destinations because terminal nodes always specify destinations.

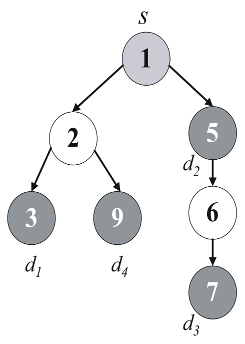


Fig. 3. Encoding a chromosome by a tree

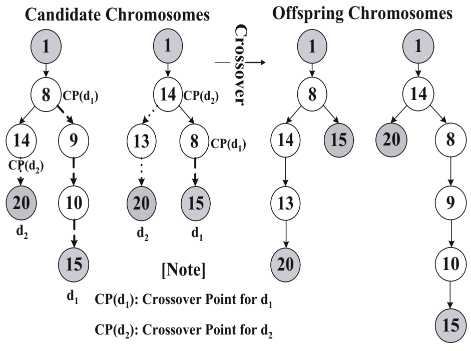


Fig. 4. Overall procedure (crossover)

3.2 Fitness Function

This function must accurately evaluate the quality of the solution (i.e., a multicast routing tree) encoded by its chromosome in the population. The fitness function is crucial as the MDR computation amounts to finding the minimum-cost multicast tree. The proposed fitness function only involves network link costs; in other words, the objective function (3) is considered. The QoS constraints are directly incorporated in the course of constructing and assembling the trees.

The value of the fitness function for the solution (e.g., a multicast tree) is represented by the chromosome. Given an initial population $\mathcal{H} = \{h_1, h_2, \ldots, h_N\}$, the fitness value of each chromosome is computed as follows. Let T_k be a multicast tree represented by the chromosome h_k, and C_{T_k} be the sum of the link costs of the tree T_k. The fitness value of the chromosome h_k, denoted as $F(h_k)$, is given by

$$F(h_k) = [C_{T_k}]^{-1} = \left[\sum_{\{i,j|e_{ij} \in T_k\}} C_{ij} \right]^{-1} . \tag{4}$$

3.3 Selection

Selection (i.e., reproduction) plays an important role in improving the average quality of the population by passing the high quality chromosomes to the next generation. The selection of chromosome is based on the value of fitness function. Note that high selection pressure leads to premature convergence. In this regard, ordinal selection such as tournament selection is preferable. In the tournament selection, the selection pressure increases with the tournament size. The proposed GA employs pair-wise tournament selection without replacement as it is effective in adjusting the selection pressure and coping with the selection noise [1]. The selection scheme randomly picks two different chromosomes and chooses the better one from a parent.

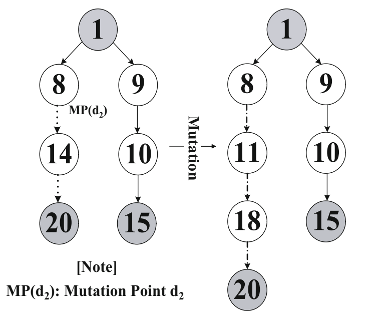


Fig. 5. Overall procedure (mutation)

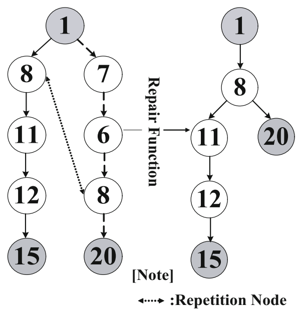


Fig. 6. Overall procedure (repair function)

3.4 Crossover

Crossover processes the current solutions so as to find better ones [8]. The crossover in the proposed GA produces diverse chromosomes by exchanging the partial chromosomes (i.e., sub-trees) without positional consistency of potential crossing sites between two chromosomes [1]. This dictates one-point crossover as a good candidate scheme for the proposed GA. The crossover between two (dominant) chosen chromosomes should have at least one common gene (node) except in the case of source and a set of destination nodes. The common nodes are called "potential crossover points." There is no requirement that they be located at the same locus. One partial routing path connects an origin to an intermediate node, and the other partial routing path from the intermediate node to a designated destination node. The crossover exchanges the latter partial path (the intermediate node → the destination node). In two dominant chromosomes, they have common genes in each branch. If the (chosen) crossover point satisfies probability of crossover p_c, the crossover will exchange the partial route path (from selected ID's node to each destination node) between two candidate chromosomes. However, it ensures that the chromosomes are valid in view of the required constraints, B_{req} and D_{req}. To sum up, the crossover performs an effective global search for discovering a multicast routing tree. Fig. 4 shows an example of the crossover procedure. In the Fig. 4, the selected gene number is '8' (i.e., node 8) with respect to the first destination set to 15. The sub-trees are (8 → 9 → 10 → 15) and (8 → 15). The crossover swaps the two sub-trees if each end-to-end delay is less than path delay constraint. With regard to the other destination, i.e., node 20, the same steps are repeated.

3.5 Mutation

The population undergoes mutation by a slight change or flipping of genes of the candidate chromosomes. This also helps it keep away from local optima [1,8]. Physically, it generates an alternative partial route from the mutation node (i.e., mutation point) to the chosen destination (in the proposed GA) according to the mutation probability of p_m. If there are several branches from the mutation node, one of them is chosen in a random fashion. The mutation replaces the sub-branch by a randomly generated partial route. It also keeps the two QoS constraints. After the mutation, it produces a new path from the mutation point to the selected destination. In principle, it enhances local search capability of the GA by maintaining the genetic diversity of population. Fig. 5 gives the overall procedure of the operation of mutation. As can be seen in the Fig. 5, a gene (i.e., node 8) is randomly selected. The partial route between the intermediate node 8 and the destination node 20 is replaced.

3.6 Repair Function

The repair function treats infeasible chromosomes that contain lethal genes that possibly form a loop. Often, they are occurred by variation operator (i.e., crossover and mutation). It means that the tree condition is violated, and thus it needs to fix. Note that no chromosome (in the initial population) is infeasible. The repair function in [1] is extended to the case of multiple routes. Note that all the chromosomes regardless of their validity satisfy the QoS constraints because they have already been taken into consideration. Thus, the proposed repair function always returns better constraint conditions. The proposed method is exhibited in Fig. 6. It shows that an offspring produced after crossover contains lethal genes, i.e., nodes 7, 6 and 8 that form a loop. By a simple search procedure, the loop is discovered. It is found to be due to the presence of node 8 in two possible routes to the destination (i.e., node 20). The first and second branches lead to $(1 \rightarrow 8 \rightarrow 11 \rightarrow 12 \rightarrow 15)$ and $(1 \rightarrow 8 \rightarrow 20)$, respectively.

4 Experimental Results

In this section, the proposed GA is compared with a comparative reference through computer simulation. The proposed GA uses a sub-quadratic population size, viz., $k \times N^{1.5}$, where N is the total number of nodes in the network, and k is a fixed number [1]. In the experiments, k is set to 3. Pair-wise tournament selection (i.e., tournament size s = 2) without replacement is employed. In all the experiments, crossover and mutation probabilities are set to 0.75 and 0.15, respectively. Each experiment is terminated when all the chromosomes in the population have converged to the same solution [1,8].

The proposed algorithm is compared with *Chen's algorithm* [12], with a view to investigating its search capability. Chen's algorithm uses the fitness function (4), with identical constraints. In the algorithm, each locus constitutes a possible path from a source to each destination (i.e., $d_1, d_2, \ldots, d_n$, where n is the number of destinations) without repeating any node in the chromosomes [5,10]. Thus, the length of all the chromosomes is equal to the number of destinations. In the algorithm, differences in the genetic operator lie in the encoding method and the mechanisms of crossover and mutation. Each gene encodes a path; the nth gene represents a route from a source to the nth destination. The crossover operation interchanges a gene between two dominant chromosomes, viz., pair-wise recombination. Mutation replaces the (end-to-end) path represented by the

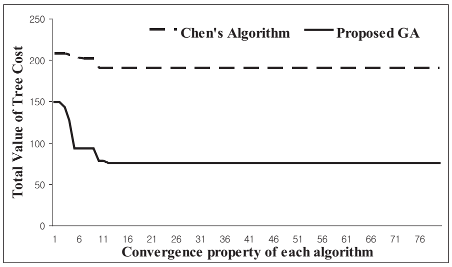


Fig. 7. Convergence property

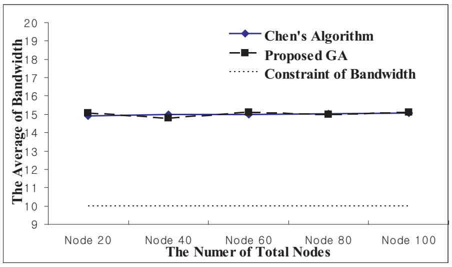


Fig. 8. Average value of bandwidth

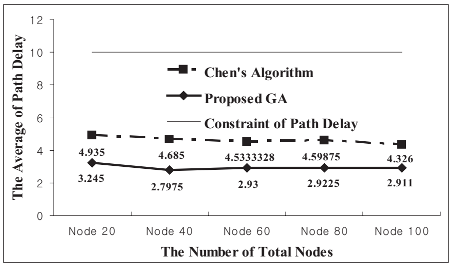


Fig. 9. Average end-to-end delay

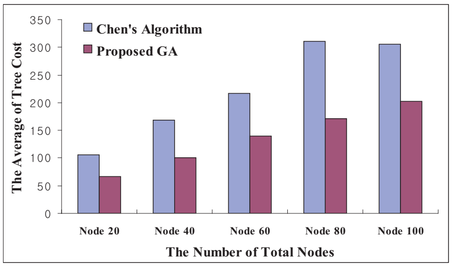


Fig. 10. Average tree cost

selected gene with a feasible path from the source to the designated destination. Crossover and mutation probabilities are set to 0.9 and 0.2, respectively [12].

Fig. 7 compares the objective function (3) values (i.e., tree costs) returned by the algorithms as applied to the network with 60 nodes. Both bandwidth and delay constraints are set to 10. It is seen that the proposed GA has a higher rate of convergence than that of Chen's algorithm. The final objective function value returned by the proposed GA has minimal cost.

The quality of the solutions is compared in Figs. 8, 9 and 10 for a range of networks with 20-100 nodes (i.e., coarse to dense networks). All the results are averaged over 100 runs. Fig. 8 illustrates the average bandwidth. The proposed GA and Chen's algorithm achieve similar performance in each network. Fig. 9 shows that the proposed GA involves a smaller end-to-end delay than does Chen's algorithm. Fig. 10 illustrates that the tree-cost found by the proposed approach is far less than that of Chen's algorithm. It is concluded that the proposed GA is superior to Chen's algorithm in respect of total tree cost. The reason is explained below. First, the encoding method does not allow of any redundancy when constructing a multicast tree, due to preventing reentry of the nodes that are already included in the current (sub-)tree. Second, the genetic operators newly designed provide higher exploratory power and a fair measure of genetic diversity. Third, the repair function cures all the infeasible trees by simply removing lethal (sub-)paths forming a loop in each tree.

To sum up, the proposed GA logs better cost values, less aggregate path delay and comparable (link) bandwidth performance; that is, it finds the minimum-cost multicast (routing) tree while satisfying QoS constraints (i.e., bandwidth and end-to-end delay).

5 Conclusion

This paper has presented a genetic-inspired multicast routing algorithm. It can find a minimum-cost multicast tree with bandwidth and end-to-end delay constraints from a designated source to multiple destinations. The chromosome is formed by a tree data structure of Genetic Programming (GP). Variable-length chromosomes expressed by positive integers are employed. Since the integers express the nodes' ID, their sequences directly represent possible routes. The fitness function returns the total tree costs about each chromosome. It plays a key role in searching for multicast routing tree with minimizing cost of multicast tree. The proposed GA is applied with tournament selection without replacement. Variation operators (i.e., crossover and mutation) efficiently promote the search capability, and a repair function fixes the infeasible solutions. The crossover exchanges the partial paths between intermediate node and destination, and this is independent on the location of the crossing site. It maximally increases the exploratory power of the proposed algorithm. The mutation maintains the genetic diversity of the population by producing a new sub-tree with a low probability. It maintains a fair measure of genetic diversity (of population) so as to escape from local optima. The synergy achieved by integrating the new components (i.e., representation, crossover, mutation, and repair function) provides a search capability that results in improved quality of the solution and enhanced rate of convergence.

Experimental results demonstrated that the proposed algorithm discovers the minimum-cost multicst tree more quickly than does Chen's algorithm. Further, it satisfies the QoS (i.e., bandwidth and end-to-end delay) constraints for diverse networks.

It is thought that the proposed GA provides a conservative tool to solve multicast routing problems with diverse QoS constraints. However, further elaboration on generalizing the idea is necessary.

References

1. Chang Wook Ahn., and R.S. Ramakrishna.: A Genetic Algorithm for Shortest Path Routing Problem and the Sizing of Populations. IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL.6, NO.6, DECEMBER 2002.
2. Yee Leung., Guo Li., and Zong-Ben Xu.: A Genetic Algorithm for the Multiple Destination Routing Problems. IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL.2, NO.4, NOVEMBER 1998.
3. Leonard Barolli., Akio Koyma., and Norio Shiratori.: A QoS Routing Method for Ad-Hoc Networks Based on Genetic Algorithm. Proceedings of the 14th International Workshop on Database and Expert Systems Applications(DEXA'03).

4. Fei Xiang., Luo Junzhou., Wu Jieyi., and Gu Guanqun.: QoS Routing based on genetic algorithm. Computer Communications 22, pp. 1392-1399, 1999.
5. REN-HUNG HWANG., WEI-YUAN DO., and SHYI-CHANG YANG.: Multicast Routing Based on Genetic Algorithms. JOURAL OF INFORMATION SCIENCE AND ENGINEERING 16, pp.885-901, 2000.
6. Abolfazl Toroghi Haghighat., Karim Faez., Mehdi Dehghan., Amir Mowlaei., and Y. Ghahremani.: GA-based heuristic algorithms for bandwidth-delay constrained least-cost multicast routing. Computer Communications 27, pp.111-127, 2004.
7. Zhou Xianwei., Chen Changja., and Zhu Gang.: A Genetic Algorithm for Multicasting Routing Problem. Proc. Int. Conf. Communication Technology (WCC-ICCT 2000), pp. 1248-1253, 2000.
8. David Beasley., David R. Bull., and Ralph R. Martin.: An Overview of Genetic Algorithms: Part 1, Fundamentals. University Computing 15(2), pp.58-69, 1993.
9. Lin Chen., ZhiYun Yang., and ZhengQuan Xu.: A Degree-Delay-Constrained Genetic algorithm for Multicast Routing Tree. Proceddings of the Fourth International Conference on Computer and Information Technology (CIT'04).
10. Jorge Crichigno., and Benjamin Baran.: A Multicast Routing Algorithm Using Multiobjective Optimization. ICT 2004, LNCS 3124, pp. 1107-1113, 2004
11. Moussa Hamdan., and Mo El-Hawary.: Multicast Routing with Delay and Delay Variation Constraint Using Genetic Algorithm. CCECE 2004- CCGEI 2004, Niagara Falls, May 2004.
12. Hua Chen., and Baolin Sun.: Multicast Routing Optimization Algorithm with Bandwidth and Delay Constraints Based on GA. Journal of Communication and Computer, Volume 2, No.5 (Serial No. 6), May 2005.
13. Xunxue Cui., Chuang Lin., and Yaya Wei.: A Multiobjective Model for QoS Multicast Routing Based on Genetic Algorithm. Proceedings of the 2003 International Conference on Computer Networks and Mobile Computing (ICCNMC'03)
14. Abdullah M.S. Alkahtani., M.E. Woodward., and K. Al-Begin.: An Overview of Quality of Service (QoS) and QoS Routing in Communication Networks. PGnet, 2003.
15. Mohamed F. Mokbel., Wafaa A. El-Haweet., and M. Nazih El-Derini.: An Efficient Algorithm for Shortest Path Multicast Routing under Delay and Delay Variation Constraints. In Proceedings of the Symposium on Performance Evaluation of Computer and Telecomm. Systems. SPECTS 2000, Vancouver, Canada, Jul., 2000.
16. Vachaspathi P. Kompella., Joseph C. Pasquale., and George C.Polyzos.: Multicast Routing for Multimedia Communication. IEEE/ACM TRANSACTION ON NETWORKING, VOL.1, NO.3, JUNE 1993.
17. Xiaola Lin., and Lionel M. Ni.: Multicast Communication in Multicomputer Networks. IEEE TRANSACTION ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL.4, NO.10, OCTOBER 1993.

Integration of Genetic Algorithm and Cultural Algorithms for Constrained Optimization

Fang Gao, Gang Cui, and Hongwei Liu

School of Computer Science and Technology, Harbin Institute of Technology, 150001 Harbin, China
gaofang@hit.edu.cn, {cg, liuhongwei}@ftcl.hit.edu.cn

Abstract. In this paper, we propose to integrate real coded genetic algorithm (GA) and cultural algorithms (CA) to develop a more efficient algorithm: cultural genetic algorithm (CGA). In this approach, GA's selection and crossover operations are used in CA's population space. GA's mutation is replaced by CA based mutation operation which can attract individuals to move to the semi-feasible and feasible region of the optimization problem to avoid the 'eyeless' searching in GA. Thus it is possible to enhance search ability and to reduce computational cost. This approach is applied to solve constrained optimization problems. An example is presented to demonstrate the effectiveness of the proposed approach.

1 Introduction

Genetic algorithm is a highly paralleled and efficient method based on natural evolution. It has enjoyed increasing popularity in solving optimization problems. The traditional GA can be divided into two categories: binary-encoding GA and real coded GA (called RGA), where the real coded version has the advantages of high precision and easy to search in large space, meanwhile it avoids the troublesome encoding and decoding process of computing the objective function. It has been reported that RGA outperforms binary-coded GA in many design problems [1]. Therefore RGA has been widely studied and applied in different fields, such as controller design [2, 3], engineering optimization [4, 5, 6] and neural network training [7]. Chang put forward a three-chromosome multi-crossover RGA, which can provide a more precise adjusting direction for finding problem solutions [8]. This improved GA has been successfully utilized to estimate the parameters of nonlinear process systems. Hrstka and Kucerova proposed to introduce differential operators into real coded genetic algorithm to prevent it to fall into local extremes [9]. Alba and Luna et al. addressed the physical parallelization of a gradual distributed multi-subpopulation real-coded GA for continuous optimization problems [10].

However, how to solve nonlinear constrained optimization problem remains a difficult and open question for real coded GA. Many constraint-handling methods resort to the concept of penalty functions, which imposes penalties on infeasible individuals that violate the problem constraints. The major difficulty in using penalty functions is still how to design an appropriate penalty function for a specific problem. Another alternative method is to add additional compulsory mutation operation on the infeasible individuals after three standard GA operations, but the new created individuals are possibly infeasible. Both the two methods seem “eyeless” to avoid violating the constraint, and therefore their efficiencies will be decreased for much result less operations. It seems necessary to add intelligence to real coded GA to solve the constrained optimization problem.

Cultural algorithm (CA) is an intelligent technique that incorporates domain knowledge obtained during the evolutionary process as to make the search process more efficient [11, 12]. In cultural algorithms, the information acquired by an individual can be shared with the entire population, unlike most evolutionary techniques, where the information can be only shared with the individual’s offspring [13]. Cultural algorithm has been successfully applied to solve optimization problems and promises to overcome some shortcomings of the above optimization methods [14], such as data mining and job-shop scheduling etc [15, 16]. Here, we propose to integrate real coded genetic algorithm and cultural algorithm to develop a more efficient algorithm, called CGA, in which GA’s selection and crossover are still used, and its mutation operation is replaced by CA based mutation, which are guided by the constraint knowledge of cultural algorithms and able to ‘attract’ the individuals to move to the feasible region of the optimization problem. It is possible to enhance the global search ability and to reduce the computational cost.

2 Cultural Algorithm

2.1 Overview

Cultural algorithm, proposed by Reynolds [11], is an evolutionary computation model derived from the cultural evolution process. It has become a candidate for many optimization applications due to its flexibility and efficiency [17]. In detail, cultural algorithm utilizes culture as a vehicle for storing relevant information that is accessible to all members of the population over the course of many generations. A cultural algorithm contains two main parts: a population space and a belief space. The population space consists of a set of possible solutions to the problem, and can be modeled using any population based technique. The belief space is the information repository in which the individuals can store their experiences for other individuals to learn them indirectly. A framework for a CA can be depicted in Fig. 1[11, 14]. CA takes two levels of evolution in the population space level and the belief space level respectively. The two spaces are connected together by an explicit communication protocol composed of two functions: an acceptance function accept() and an influence function influence(). The function accept() is used to collect the experience of the selected individuals from the population; then the contents of the belief space can be added via an update function, denoted as update(); Next, the function influence() can make use of the problem-solving knowledge in the belief space to guide the evolution of the population space [14]. In the population space, like traditional evolutionary population models, individuals are first evaluated by a performance function obj(). Then, the new individuals are created by a generation function generate(). Further, a selection function select() is used to select the population for next generation. The two feedback paths of information, one through the accept() and influence() functions and the other through individual experience and the obj() function create a system of dual inheritance [15].

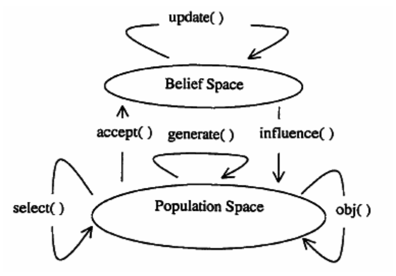


Fig. 1. Framework of cultural algorithms

2.2 Cultural Algorithm Using Regional Schemata

Jin and Reynolds explored the use of cultural algorithms for global optimization with very encouraging results [14]. They proposed a new framework based on cultural algorithms and regional schemata to solve constrained optimization problems. The regional schemata is an extension of the classic symbolic schemata, and it represents the regional knowledge that can be used to provide the functional landscape information and to guide the algorithm search in the population. In detail, an n-dimensional regional-based schema, called belief-cell, is as an explicit mechanism that supports the acquisition, storage and integration of knowledge about non-linear constraints in cultural algorithms. This belief-cell can be used to guide the search of an evolutionary computation technique by pruning the infeasible individuals in the population space and promoting the exploration of promising regions of the search space[14,15].

3 CGA Algorithm for Constrained Optimization

When CA is used for a constrained optimization problem, the domain space of the optimization problem can be partitioned into many feasible, infeasible and semi-infeasible regions, which is stored and continuously updated in belief space as constraint knowledge [14]. The knowledge is further used to conduct individual evolution in population space by adjust it into feasible regions of problem. By contrast with CA, GA is a random search algorithm lacking of similar intelligence technique of CA. In our cultural genetic algorithm (CGA), GA is supervised by CA, and intelligently adjusts its individuals into feasible and semi-feasible regions to avoid blindness searching and to reduce computation cost.

3.1 Parameter Encoding and Fitness Value

GA is a random search technique simulating natural evolution. It deals with a population, and each individual represents a candidate solution. In the real-coded GA, encoding of the parameter set are just the parameters themselves. For example, assume x_1 , x_2 ,…, x_n to be the design variables, so the corresponding coding is the set of { x_1 , x_2 ,…, x_n }. After encoding, an initial population is randomly generated. Each individual from the population is evaluated and assigned a fitness value. Herein, individuals are evaluated by a fitness function $f(X_i)$, which can be same as the performance function *obj*(). The function *obj*() is the sum of objective function and a penalty function.

3.2 Genetic Operators

Our CGA algorithm also uses three operators, namely selection (reproduction), crossover and mutation in the population space.

Selection. The selection operation is a random selection process. However, the individuals with best fitness values would have the more chance to be chosen for reproduction. By means of **roulette wheel selection** approach, the i th individual is chosen according to probability P_s as

$$P_s = f(X_i) / \sum_{i=1}^{N} f(X_i), \tag{1}$$

where N is population size of GA, $f(X_i)$ is fitness function for the i th individual as mentioned above.

The individuals among the population starting with the highest values are selected with higher probability for reproduction.

Crossover. After selection operations, two parent individuals exchange information to produce a new offspring. This process is called crossover. Crossover is governed by a crossover probability P_c, which tells the GA program whether to execute crossover. In this operation, two individuals X_i and X_j are randomly selected as parents from the pool of individuals formed by the selection procedure and crossover operations between them are executed according to the following equations:

$$\begin{aligned} X_i' &= \alpha \cdot X_i + (1-\alpha) \cdot X_j, \\ X_j' &= (1-\alpha) \cdot X_i + \alpha \cdot X_j, \end{aligned} \tag{2}$$

where α is a random number extracted from region (0, 1). X_i' and X_j' are new offsprings created by crossover.

A crossover operation can thus yield better solutions by combining the good features of existing solutions. If no crossover is performed due to the crossover probability, the offspring is the exact copy of the parent solution.

Mutation. Each offspring created from crossover is altered in a process called mutation. The effect of this operation is to create diversity in the solution population, thus the operation can avoid trapping in local minimum. In the real coded GA, mutation alters each offspring as the follow:

$$X_i'' = X_i' + \beta \cdot d \ , \tag{3}$$

where X_i' is the parent individual selected for mutation. β and d are the step size and random direction of mutation, respectively. X_i'' is the new offspring by mutation.

Using the above mutation operation, the offspring generated could violate the constraints of the problem. In this paper, we introduce the intelligent mutation scheme used in the regional-based sliding window cultural algorithm [14, 15]. Instead of Eq. (3), the new offspring can be created as

$$\boldsymbol{X_i'''} = \begin{cases} \boldsymbol{moveTo(choose(Cell[l]))} & \boldsymbol{if\ X_i'' \in \{infeasible\ cells\}} \\ \boldsymbol{X_i'' + \gamma \cdot (u_j - l_j) \cdot N_{i,j}(0,\ 1)} & \boldsymbol{if\ X_i'' \in \{otherwise\}} \end{cases}, \tag{4}$$

where l_j and u_j represent the current lower bound and upper bound, respectively, for the j th dimension of X_i. γ is a certain positive number. $Cell[\bullet]$ is a r-dimensional template, called regional schemata, which is maintained in the belief space. This template is used to record the constraint characteristics of a certain region in the search space. For a given cell i, namely a small region in domain space of optimization problem, it may be:

(1) a feasible cell including only valid individuals;
(2) an infeasible cell including only invalid individuals;
(3) a semi-feasible cell including both valid individuals and invalid ones;
(4) a unknown cell with no individuals in it, so no knowledge about it now.

Based on which types all individuals in the cell belongs to, the i th cell can be assigned a different weight as follows[14]:

$$W_i = \begin{cases} w_1 & if\ Cell[i] \in \{unknown\ cells\} \\ w_2 & if\ Cell[i] \in \{feasible\ cells\} \\ w_3 & if\ Cell[i] \in \{semi-feasible\ cells\} \\ w_4 & if\ Cell[i] \in \{infeasible\ cells\} \end{cases}. \tag{5}$$

The *choose* (cell[*l*]) function in Eq. (4) is a function to be used to select a target cell from all cells for the *moveTo*() function. This process can be implemented by roulette wheel selection based on the weights values in Eq. (5), just similar to the previous selection operation.

Assuming that the *k*th cell C_k is selected by roulette wheel selection, *moveTo* (C_k) creates a new offspring as follows[14]:

$$X_i''' = Left_k + \text{uniform}(0,1) \cdot Csize_k \tag{6}$$

where $Left_k$ is a $1 \times r$ array which denotes the left-most position of cell C_k, $Csize_k$ is a $1 \times r$ array which represents the sizes of C_k in all dimensions, and $\text{uniform}(0,1)$ generates a $1 \times r$ array of numbers within [0,1] by uniform distribution.

It can be seen that the above mutation operation is guided by constraint knowledge preserved in belief space. The individuals continuously adjust its direction to semi-feasible and feasible area of the problem space. These cultural based intelligences will enhance the search ability of CGA.

3.3 Pseudo-code of CGA

The Pseudo-code of the cultural genetic algorithm is as follows:

```
Step 1. t=0;
Step 2. Initialize the population $P^t$ ;
Step 3. Initialize the Belief Space $B^t$=<$N^t$ ,$C^t$>;
Step 4. Evaluate each individual's performance of $P^t$;
Step 5. Update the Belief Space which includes:
        Update the normative knowledge component $N^t$,
        Update the constraint knowledge component $C^t$ ;
Step 6. Execute selection operation according to Eq.(1);
Step 7. Execute crossover operation according to Eq.(2);
Step 8. Execute mutation operation according to Eq.(4);
Step 9. t=t+1;
Step 10. Repeat to Step 4 until the termination condition is
         achieved.
```

3.4 Discussion

Genetic algorithm is modeled based on simulating of natural evolution, whereas the computational model of cultural algorithm derived from the cultural evolution process of human society. In GA, individuals are replicated in the population for the next generation according to their relative fitness. New individuals are created for the next generation by operations that simulate biological crossover and mutation. Good genes are continuously inherited for the whole population by selection and crossover. Besides, mutation is used to keep the diversity of the population and occasionally create new better genes. It can be seen that vertical inheritance is a noticeable feature in GA. In CA, culture is used as a vehicle for storing knowledge that influences all members of the population space, and in the mean time, is accumulated by each member. All members share with culture and "co-evolve". This definite "interactive" dual evolution mechanism can make the individual evolution with more definite direction, and meanwhile speed up the transverse search ability of cultural algorithm. Thus, the integration of GA and CA can synthesize both their advantages, and shows good ability both in depth and breadth search.

For constrained optimization problems, the belief space can store constraint information about the solved problem, and guides the evolution of GA in the population space, which can make full use of GA's search ability and CA's intelligence.

4 Numerical Example

The general optimization problems, minimization or maximization, can occur in many economic, technical and scientific projects. Solving it remains a difficult and open problem for evolutionary algorithms. The common opinion about evolutionary algorithms is that they are good optimization methods but can't handle constraints well [14, 15]. Cultural algorithms can learn constrain knowledge during the search instead of having to acquire it beforehand and benefit from this acquired knowledge. The example of nonlinear constrained optimization problem in literature [14] is used to illustrate the approach. It is given as follows:

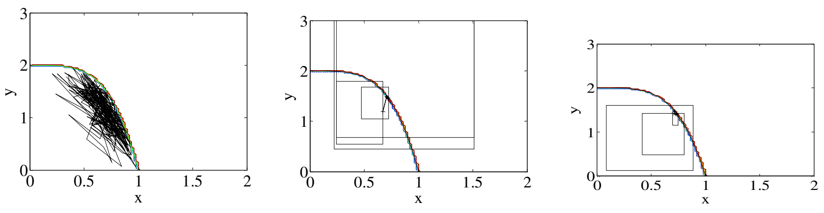
Objective function: $\min f(x,y) = -12x - 7y + y^2$,

Constraints: $0 \le x \le 2$;

$0 \le y \le 3$;

$y \le -2x^4 + 2$.

The GA population size N= 60, and other main parameters setting are: P_c=0.3, γ=0.3, $w_1 = w_2 = 2$, w_3=3, $w_4 = 1$. Simulations are performed with Matlab on 2.6G Pentium PC. The illustrated test run is shown in Fig. 2-(c). The proposed algorithm is tested statistically using the above example as listed in Table 1. To examine the performance, a standard real coded GA (called RGA) and CA by X. D. Jin are used as the comparison as shown in Fig. 2-(a),(b) and Table 1.



(a) RGA (b) CA (c) CGA

Fig. 2. Iteration process of algorithms

Table 1. Statistical contrast results of algorithms

Algorithms	Mean of number of generations	Standard deviation of number of generations	Mean of running times
RGA	728	223	5.2
CA	5.4	1.58	4.8
CGA	4.1	1.26	3.4

From Table 1, it can be seen that RGA needs a large number of iteration generations because of its "blindness" in search process, but its running takes no long time since its simple algorithm results in lower computation cost of each iteration. The improved CA has much fewer iteration generations and less time because its population space evolves under guidance of the belief space and has better adjusting direction to the optimal solution. Further, the CGA shows better performance to some extent because it synthesizes both the advantages of GA and CA, which is a GA-based search scheme meanwhile under intelligent guidance.

5 Conclusions

We present a cultural real coded genetic algorithm by integrating GA and CA. GA's selection and crossover operators are used in CA's population space, and GA's mutation is replaced by CA based mutation under guidance of the constraint knowledge of cultural algorithm.

It can synthesize both the advantages of genetic algorithm and cultural algorithm. Compared with the traditional RGA, the proposed approach improves mutation direction in search process. Compared with CA by X. D. Jin et al., the proposed approach also increases the search ability and takes less computation cost.

Compared with GA, CA and CGA both employ a dual evolution mechanism and its algorithm is relative complicated. Computation cost is relative more than GA in their each iteration, which is compensated by relative less iteration times. The division of belief cell usually costs more computation, which becomes more noticeable, especially for high dimensional optimization problem, Thus how to design a more time-saving belief-cell division scheme is worth to further analyzing.

Acknowledgement

This work is supported by the National Science Foundation of China under grant No. 60503015.

References

1. Janikow, C. Z., Michalewicz, Z.: An Experimental Comparison of Binary and Floating Point Representations in Genetic Algorithms, in: Proceedings of the 4th International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, San Mateo, (1991)31–36.
2. Lin, F., Shieh, H., Shyu, K., Huang, P.: On-line Gain-tuning IP Controller Using Real-Coded Genetic Algorithm. Electric Power Systems Research. 72(2004)157–169
3. Arfiadi, Y., Hadi, M. N. S.: Optimal Direct (static) Output Feedback Controller Using Real Coded Genetic Algorithms. Computers and Structures. 79(2001)1625–1634
4. Oyama, A., Obayashi, S., NakaMura, T.: Real-coded Adaptive Range Genetic Algorithm Applied to Transonic Wing Optimization. Applied Soft Computing. 1(2001)179–187
5. Ha, J., Fung R., Han, C.: Optimization of an Impact Drive Mechanism Based on Real-coded Genetic Algorithm. Sensors and Actuators. 121(2005)488–493
6. Yan, S.Z., Zheng, K., Zhao, Q., Zhang, L.: Optimal Placement of Active Members for Truss Structure Using Genetic Algorithm. Lecture Notes in Computer Science. 3645(2005) 386–395.
7. Blanco, A., Delgado, M., Pegalajar, M. C.: A Real Coded Genetic Algorithm for Training Recurrent Neural Networks. Neural Networks. 14(2001)93–105
8. Chang, W.: An Improved Real Coded Genetic Algorithm for Parameters Estimation of Nonlinear Systems. Mechanical Systems and Signal Processing. 20(2006)236–246.

9. Hrstka, O., Kucerova, A.: Improvements of Real Coded Genetic Algorithms Based on Differential Operators Preventing Premature Convergence. Advances in Engineering Software. 35(2004)237–246
10. Alba, E., Luna, F., Nebro, A.J., Troya, J.M.: Parallel Heterogeneous Genetic Algorithms for Continuous Optimization. Parallel Computing. 30(2004)699–719
11. Reynolds, R. G.: An Introduction to Cultural Algorithms. Proceedings of the 3rd Annual Conference on Evolutionary Programming. World Scientific Publishing. (1994)108–121
12. Reynolds, R. G., Chung, C. J.: A Self-adaptive Approach to Representation Shifts in Cultural Algorithms. IEEE. 3(96)94–99
13. Becerra, R. L., Carlos A. Coello Coello.: Culturizing Differential Evolution for Constrained Optimization. Proceedings of the Fifth Mexican International Conference in Computer Science, IEEE, (2004)304–311
14. Jin, X.D., Reynolds, R. G.: Using Knowledge-Based Evolutionary Computation to Solve Nonlinear Constraint Optimization Problems: a Cultural Algorithm Approach, IEEE, (1999)1672–1678
15. Jin, X.D., Reynolds, R. G.: Mining Knowledge in Large Scale Databases Using Cultural Algorithms with Constraint Handling Mechanisms, Proceeding of the 2000 congress on evolutionary computation, IEEE , (2000)1498–1506
16. Ho, N. B., Tay. J. C.:GENACE: An Efficient Cultural Algorithm for Solving the Flexible Job-Shop Problem, Proceeding of 2004 Congress on Evolutionary Computation, 2(2004) 1759–1766
17. Yuan, X. H., Yuan, Y. B.; Application of Cultural Algorithm to Generation Scheduling of Hydrothermal Systems. Energy Conversion and Management. 47(2006) 2192–2201

Neuro-genetic Approach for Solving Constrained Nonlinear Optimization Problems

Fabiana Cristina Bertoni and Ivan Nunes da Silva

University of São Paulo, Department of Electrical Engineering, CP 359
CEP 13566.590, São Carlos, SP, Brazil
{bertoni, insilva}@sel.eesc.usp.br

Abstract. This paper presents a neuro-genetic approach for solving constrained nonlinear optimization problems. Genetic algorithm must its popularity to make possible cover nonlinear and extensive search spaces. On the other hand, artificial neural networks have high computational rates due to the use of a massive number of simple processing elements and the high degree of connectivity between these elements. Neural networks with feedback connections provide a computing model capable of solving a large class of optimization problems. The association of a modified Hopfield network with genetic algorithm guarantees the convergence of the system to the equilibrium points, which represent feasible solutions for constrained nonlinear optimization problems. Simulated examples are presented to demonstrate that proposed method provides a significant improvement.

1 Introduction

Mathematical optimization problems have been widely applied in practically all knowledge areas. The nonlinear optimization plays a fundamental role in many problems involved with the areas of sciences and engineering, where a set of parameters is optimized subject to inequality and/or equality constraints [1].

In the neural networks literature, there exist several approaches used for solving constrained nonlinear optimization problems. The first neural approach applied in optimization problems was proposed by Tank and Hopfield in [2], where the network was used for solving linear programming problems. More recently, it was proposed in [3] a recurrent neural network for nonlinear optimization with a continuously differentiable objective function and bound constraints. In [4], it was developed a multilayer perceptron for nonlinear programming, which transforms constrained optimization problems into a sequence of unconstrained ones by incorporating the constraint functions into the objective function of the unconstrained problem. Basically, most of these neural networks presented in the literature for solving nonlinear optimization problems contain some penalty parameters. The stable equilibrium points of these networks, which represent a solution of the constrained optimization problems, are obtained only when those parameters are properly adjusted, and in this case, both the accuracy and the convergence process can be affected. In this paper, we propose a neuro-genetic hybrid system for solving constrained nonlinear optimization problems.

More specifically, a modified Hopfield network is associated with a genetic algorithm to guarantee convergence of the system to the equilibrium points. The

Hopfield network performs the optimization of constraints, whereas the genetic algorithm is responsible to minimize the objective function. The neuro-genetic system has been applied to several nonlinear optimization problems and has shown promise for solving such problems efficiently.

The organization of the present paper is as follows. In Section 2, the modified Hopfield network is presented. Section 3 describes the genetic algorithm applied to optimization of the objective function. In Section 4, a mapping of constrained nonlinear optimization problems is formulated using the neuro-genetic architecture. Simulation results are presented in Section 5 to demonstrate the advanced performance of the proposed approach. In Section 6, the main results of the paper are summarized and conclusions are presented.

2 The Modified Hopfield Neural Network

Besides providing a new approach for solving constrained optimization problems, artificial neural networks provide a method for exploring intrinsically parallel and adaptive processing architectures. In this paper, a modified Hopfield network with equilibrium points representing the problem solution has been developed. As introduced in [5], Hopfield networks are single-layer networks with feedback connections between nodes. In the standard case, the nodes are fully connected, i.e., every node is connected to all others nodes, including itself. The node equation for the continuous-time network with *N*-neurons is given by:

$$\dot{u}_i(t) = -\eta . u_i(t) + \sum_{j=1}^{N} T_{ij} . v_j(t) + i_i^b \; . \tag{1}$$

$$v_i(t) = g(u_i(t)) \; . \tag{2}$$

where:

$u_i(t)$ is the current state of the *i*-th neuron.
$v_j(t)$ is the output of the *j*-th neuron.
i_i^b is the offset bias of the *i*-th neuron.
$\eta . u_i(t)$ is a passive decay term.
T_{ij} is the weight connecting the *j*-th neuron to *i*-th neuron.

In Equation (2), $g(u_i(t))$ is a monotonically increasing threshold function that limits the output of each neuron to ensure that network output always lies in or within a hypercube. It is shown in [5] that the network equilibrium points correspond to values $\boldsymbol{v}(t)$ for which the energy function (3) associated with the network is minimized:

$$E(t) = -\frac{1}{2} \boldsymbol{v}(t)^T . \boldsymbol{T} . \boldsymbol{v}(t) - \boldsymbol{v}(t)^T . \boldsymbol{i}^b \; . \tag{3}$$

The mapping of constrained nonlinear optimization problems using a Hopfield network consists of determining the weight matrix $\boldsymbol{T}$ and the bias vector $\boldsymbol{i}^b$ to compute equilibrium points. A modified energy function $E^m(t)$ is used here, which is defined by:

$$E^{m}(t) = E^{conf}(t) + Min\, f(\boldsymbol{v})\,. \tag{4}$$

where $E^{conf}(t)$ is a confinement term that groups all the constraints imposed by the problem, and $Min\, f(\boldsymbol{v})$ refers to the minimization of the objective function associated with the constrained optimization problem in analysis, which conducts the network output to the equilibrium points. Thus, the minimization of $E^{m}(t)$ of the modified Hopfield network is conducted in two stages:

i) Minimization of the Term $E^{conf}(t)$

$$E^{conf}(t) = -\frac{1}{2}\boldsymbol{v}(t)^{T}.\boldsymbol{T}^{conf}.\boldsymbol{v}(t) - \boldsymbol{v}(t)^{T}.\boldsymbol{i}^{conf}\,. \tag{5}$$

where: $\boldsymbol{v}(t)$ is the network output, $\boldsymbol{T}^{conf}$ is weight matrix and $\boldsymbol{i}^{conf}$ is bias vector belonging to E^{conf}. This corresponds to confinement of $\boldsymbol{v}(t)$ into a valid subspace that confines the inequality constraints imposed by the problem. An investigation associating the equilibrium points with respect to the eigenvalues and eigenvectors of the matrix $\boldsymbol{T}^{conf}$ shows that all feasible solutions can be grouped in a unique subspace of solutions. As consequence of the application of this subspace approach, which is named the valid-subspace method, a unique energy term can be used to represent all constraints associated with the optimization problem. A detailed explanation on the valid-subspace technique can be found in [6].

ii) Minimization of the Objective Function $f(\boldsymbol{v})$
After confinement of all feasible solutions to the valid subspace, a genetic algorithm (GA) is applied in order to optimize the objective function by inserting the values $\boldsymbol{v}(t)$ into the chromosomes population. Thus, a great variety of constrained optimization problems can be solving by the modified Hopfield network in association with the genetic algorithm.

The operation of this hybrid system consists of three main steps as shown in Fig. 1:

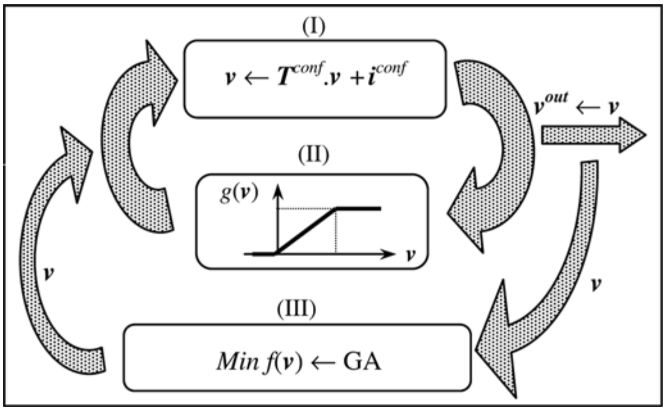

(I)
$\boldsymbol{v} \leftarrow \boldsymbol{T}^{conf}.\boldsymbol{v} + \boldsymbol{i}^{conf}$
$\boldsymbol{v}^{out} \leftarrow \boldsymbol{v}$
(II)
$g(\boldsymbol{v})$
$\boldsymbol{v}$
$\boldsymbol{v}$
$\boldsymbol{v}$
(III)
$Min\, f(\boldsymbol{v}) \leftarrow$ GA

Fig. 1. The modified Hopfield network

Step (I): Minimization of E^{conf}, corresponding to the projection of $\boldsymbol{v}(t)$ in the valid subspace defined by:

$$\boldsymbol{v}(t+1) = \boldsymbol{T}^{conf}.\boldsymbol{v}(t) + \boldsymbol{i}^{conf} . \tag{6}$$

where: $\boldsymbol{T}^{conf}$ is a projection matrix ($\boldsymbol{T}^{conf}.\boldsymbol{T}^{conf} = \boldsymbol{T}^{conf}$) and ($\boldsymbol{T}^{conf}.\boldsymbol{i}^{conf} = \boldsymbol{0}$). This operation corresponds to an indirect minimization process of $E^{conf}(t)$ using orthogonal projection de $\boldsymbol{v}(t)$ on the feasible set.

Step (II): Application of a nonlinear 'symmetric ramp' activation function constraining $\boldsymbol{v}(t)$ in a hypercube:

$$g_i(v_i) = \begin{cases} lim_i^{\text{inf}} & , \text{ if } \quad v_i < lim_i^{\text{inf}} \\ v_i & , \text{ if } \quad lim_i^{\text{inf}} \le v_i \le lim_i^{\text{sup}} \\ lim_i^{\text{sup}} & , \text{ if } \quad v_{\text{i}} > lim_i^{\text{sup}} \end{cases} \tag{7}$$

where $v_i(t) \in [lim_i^{\text{inf}}, lim_i^{\text{sup}}]$.

Step (III): Minimization of $f(\boldsymbol{v})$, which involves the application of a genetic algorithm to move $\boldsymbol{v}(t)$ towards an optimal solution that corresponds to network equilibrium points, which are the solutions for the constrained optimization problem considered in a particular application.

As seen in Fig. 1, each iteration represented by the above steps has two distinct stages. First, as described in Step (III), $\boldsymbol{v}$ is updated using the genetic algorithm. Second, after each updating given in Step (III), $\boldsymbol{v}$ is projected directly in the valid subspace by the modified Hopfield network. This second stage is an iterative process, in which $\boldsymbol{v}$ is first orthogonally projected in the valid subspace by applying Step (I) and then thresholded in Step (II) so that its elements lie in the range defined by [lim_i^{inf} , lim_i^{sup}] . This process moves the network output to the equilibrium point that corresponds to the optimal solution for the constrained optimization problem. The convergence process is concluded when the values of $\boldsymbol{v}^{out}$ during two successive loops remain practically constant, where the value of $\boldsymbol{v}^{out}$ in this case is equal to $\boldsymbol{v}$.

3 Genetic Algorithm for Objective Function Optimization

The components and parameters of the genetic algorithm employed in the hybrid system are presented in this section. To this end, one begins with a chromosomal representation of the individuals of the population. Each individual is a possible solution for the problem. The algorithm begins by randomly developing the first population. From this point, the fitness value in relation to each individual is computed. Based on this value, the elements that will belong to the next generation are selected (by election based on probabilistic criteria). To complete the population, the selected parents are reproduced through the implementation of genetic operators such as Crossing and Mutation. Each genetic operator has a proper occurrence rate expressed as a biological metaphor. The process is repeated until a specified stop condition is met. This condition may be either the successive repetition of the best individual or a maximum limit of generations.

Codification: In this stage, the chromosomes $C_i=(c_{i1}, c_{i2},\dots, c_{im})$ are encoded into sequences of binary digits and have a fixed size m, which represents the number of bits necessary to codification of a real number into the interval $[lim_i^{\text{inf}}, lim_i^{\text{sup}}]$. In our simulations, the value of m was assumed as 16.

Population Size: The size of the population used here was one hundred individuals, which allowed for a better coverage of the search space and proved efficient in our experiment.

Initial Population: The initial population is generated by introducing a chromosome that represents the values $\boldsymbol{v}(t)$ previously obtained into Steps (I) and (II) described at Section 2. The remaining chromosomes are generated randomly.

Number of Generations: As stop criterion, it is verified the variation of the best individual from a generation to another one, and when there is no variation, the algorithm must finish its execution. Associated to this criterion, a maximum number of one hundred generations was also established, being enough for reach the minimum value to the objective function of a constrained nonlinear optimization problem. Therefore, there is no significant difference in the best individual after one hundred generations.

Fitness Function: The fitness function evaluates each chromosome verifying its environment adaptation. The fitness function for constrained optimization problems is the own objective function to be minimized. The most adapted individual will have the small fitness value.

Intermediate Population: Given a population in which each individual has received a fitness value, there are several methods to select the individuals upon whom the genetic algorithms of crossing and mutation will be applied. The selected individuals will make up a population called intermediate population. The selection method used here to separate the intermediate population was the Roulette method [7] and the Crossing and Mutation rates were defined, respectively, at 70% and 1%, as recommended in the literature [7]. An elitism percentage of 10% was also used.

4 Formulation of the Nonlinear Optimization Problem

Consider the following constrained optimization problem, with m-constraints and n-variables, given by the following equations:

$$\text{Minimize } f(\boldsymbol{v}) \tag{8}$$

$$\text{Subject to: } h_i(\boldsymbol{v}) \le 0,\ i \in \{1..m\} \tag{9}$$

$$\boldsymbol{z}^{\text{mim}} \le \boldsymbol{v} \le \boldsymbol{z}^{\text{max}} \tag{10}$$

where $\boldsymbol{v}$, $\boldsymbol{z}^{\text{min}}$, $\boldsymbol{z}^{\text{max}} \in \Re^n$, $f(\boldsymbol{v})$ and $h_i(\boldsymbol{v})$ are continuous, and all first and second order partial derivatives of $f(\boldsymbol{v})$ and $h_i(\boldsymbol{v})$ exist and are continuous. The vectors $\boldsymbol{z}^{\text{min}}$ and $\boldsymbol{z}^{\text{max}}$ define the bounds on the variables belonging to the vector $\boldsymbol{v}$. The parameters $\boldsymbol{T}^{conf}$ and $\boldsymbol{i}^{conf}$ are calculated by transforming the inequality constraints in (9) into equality constraints by introducing a slack variable $\boldsymbol{w} \in \Re^N$ for each inequality constraint:

$$h_i(\boldsymbol{v}) + \sum_{j=1}^{m} q_{ij}.w_j = 0\,. \tag{11}$$

where w_j are slack variables, treated as the variables v_i , and q_{ij} is defined by the Kronecker impulse function:

$$q_{ij} = \begin{cases} 1, \text{ if } i = j \\ 0, \text{ if } i \neq j \end{cases} \tag{12}$$

After this transformation, the problem defined by equations (8), (9) and (10) can be rewritten as:

$$\text{Minimize}\, f(\boldsymbol{v}^{+})\,. \tag{13}$$

$$\text{Subject to: } \boldsymbol{h}^{+}(\boldsymbol{v}^{+}) = 0\,. \tag{14}$$

$$\boldsymbol{z}^{\text{mim}} \leq \boldsymbol{v}^{+} \leq \boldsymbol{z}^{\text{max}},\ \boldsymbol{i} \in \{1..n\}\,. \tag{15}$$

$$0 \leq \boldsymbol{v}^{+} \leq \boldsymbol{z}^{\text{max}},\ \text{i} \in \{n{+}1..N^{+}\}\,. \tag{16}$$

where $N = n + m$, and $\boldsymbol{v}^{+T} = [\boldsymbol{v}^T \ \ \boldsymbol{w}^T] \in \Re^N$ is a vector of extended variables. Note that $f(\boldsymbol{v})$ does not depend on the slack variables $\boldsymbol{w}$. In [8] has been shown that a projection matrix to the system (9) is given by:

$$\boldsymbol{T}^{conf} = \boldsymbol{I} - \nabla\boldsymbol{h}(\boldsymbol{v})^{T}.(\nabla\boldsymbol{h}(\boldsymbol{v}).\nabla\boldsymbol{h}(\boldsymbol{v})^{T})^{-1}.\nabla\boldsymbol{h}(\boldsymbol{v})\,. \tag{17}$$

where:

$$\nabla\boldsymbol{h}(\boldsymbol{v}) = \begin{bmatrix} \frac{\partial h_1(\boldsymbol{v})}{\partial v_1} & \frac{\partial h_1(\boldsymbol{v})}{\partial v_2} & \cdots & \frac{\partial h_1(\boldsymbol{v})}{\partial v_N} \\ \frac{\partial h_2(\boldsymbol{v})}{\partial v_1} & \frac{\partial h_2(\boldsymbol{v})}{\partial v_2} & \cdots & \frac{\partial h_2(\boldsymbol{v})}{\partial v_N} \\ \vdots & \vdots & \ddots & \\ \frac{\partial h_m(\boldsymbol{v})}{\partial v_1} & \frac{\partial h_m(\boldsymbol{v})}{\partial v_2} & & \frac{\partial h_m(\boldsymbol{v})}{\partial v_N} \end{bmatrix} = \begin{bmatrix} \partial h_1(\boldsymbol{v})^T \\ \partial h_2(\boldsymbol{v})^T \\ \vdots \\ \partial h_m(\boldsymbol{v})^T \end{bmatrix} \tag{18}$$

Inserting the value of (17) in the expression of the valid subspace in (6), we have:

$$\boldsymbol{v}^{+} \leftarrow [\boldsymbol{I} - \nabla\boldsymbol{h}(\boldsymbol{v}^{+})^{T}.(\nabla\boldsymbol{h}(\boldsymbol{v}^{+}).\nabla\boldsymbol{h}(\boldsymbol{v}^{+})^{T})^{-1}.\nabla\boldsymbol{h}(\boldsymbol{v}^{+})].\ \boldsymbol{v}^{+} + \boldsymbol{i}^{conf} \tag{19}$$

Results of the Lyapunov stability theory [9] should be used in (19) to guarantee the stability of the nonlinear system, and consequently, to force the network convergence to equilibrium points that represent a feasible solution to the nonlinear system. By the definition of the Jacobean, when $\boldsymbol{v}$ leads to equilibrium point implicates in $\boldsymbol{v}^e = \boldsymbol{0}$. In this case, the value of $\boldsymbol{i}^{conf}$ should also be null to satisfy the equilibrium condition, i.e., $\boldsymbol{v}^e = \boldsymbol{v}(t) = \boldsymbol{v}(t + 1) = \boldsymbol{0}$. Thus, $\boldsymbol{h}(\boldsymbol{v}^{+})$ given in equation (19) can be approximated as follows:

$$\boldsymbol{h}(\boldsymbol{v}^{+}) \approx \boldsymbol{h}(\boldsymbol{v}^{e}) + \boldsymbol{J}.(\ \boldsymbol{v}^{+} - \boldsymbol{v}^{e})\,. \tag{20}$$

where $\boldsymbol{J} = \nabla\boldsymbol{h}(\boldsymbol{v}^{+})$ and $\boldsymbol{h}(\boldsymbol{v}^{+}) = [h_1(\boldsymbol{v}^{+})\ \ h_2(\boldsymbol{v}^{+})\ \ldots\ h_m(\boldsymbol{v}^{+})]^T$.

In the proximity of the equilibrium point $\boldsymbol{v}^e = \mathbf{0}$, we obtain the following equation related to the parameters $\boldsymbol{v}^+$ and $\boldsymbol{h}(\boldsymbol{v}^+)$:

$$\lim_{\boldsymbol{v}^+ \to \boldsymbol{v}^e} \frac{\| \boldsymbol{h}(\boldsymbol{v}^+) \|}{\| \boldsymbol{v}^+ \|} = \mathbf{0} . \tag{21}$$

Finally, introducing (20) and (21) in equation given by (19), we obtain:

$$\boldsymbol{v}^+ \leftarrow \boldsymbol{v}^+ - \nabla \boldsymbol{h}(\boldsymbol{v}^+)^T.(\nabla \boldsymbol{h}(\boldsymbol{v}^+).\nabla \boldsymbol{h}(\boldsymbol{v}^+)^T)^{-1}.\boldsymbol{h}(\boldsymbol{v}^+) . \tag{22}$$

Therefore, equation (22) synthesizes the valid-subspace expression for treating systems of nonlinear equations. In this case, for constrained nonlinear optimization problems the original valid-subspace equation given in (6), which is represented by Step (I) in Fig. 1, should be substituted by equation (22). Thus, according to Fig. 1, successive applications of the Step (I) followed by the Step (II) make $\boldsymbol{v}^+$ convergent to a point that satisfies all constraints imposed to the nonlinear optimization problem.

To demonstrate the advanced behavior of the neuro-genetic system derived in this section and to validate its properties, some simulation results are presented in the next section.

5 Simulation Results

In this section, the neuro-genetic hybrid system proposed in previous sections has been used to solve nonlinear optimization problems. We provide two examples to illustrate the effectiveness of the proposed architecture.

Problem 1. Consider the following constrained optimization problem, which is composed by inequality and equality constraints:

$$Min\, f(\boldsymbol{v}) = v_1^3 + 2v_2^2 \cdot v_3 + 2v_3$$

$$\text{Subject to:} \quad v_1^2 + v_2 + v_3^2 = 4$$

$$v_1^2 - v_2 + 2v_3 \le 2$$

$$v_1 , v_2 , v_3 \ge 0$$

The optimal solution for this problem is given by $\boldsymbol{v}^* = [0.00 \quad 4.00 \quad 0.00]^T$, where the minimal value of $f(\boldsymbol{v}^*)$ at this point is equal to zero. Figure 2 shows the trajectories of the system variables starting from the initial point $\boldsymbol{v}^0 = [1.67 \quad 1.18 \quad 3.37]^T$.

The trajectory of the objective function starting from initial point presented above is illustrated in Fig. 3. The system has also been evaluated for different values of initial conditions. All simulation results obtained by the neuro-genetic system show that the proposed architecture is globally asymptotically stable at $\boldsymbol{v}^*$.

These results show the ability and efficiency of the neuro-genetic system for solving nonlinear optimization when equality and inequality constraints are simultaneously included in the problem.

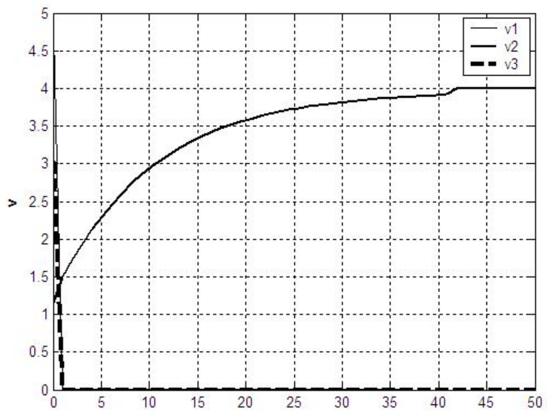


Fig. 2. Evolution of the neuro-genetic system output (Problem 1)

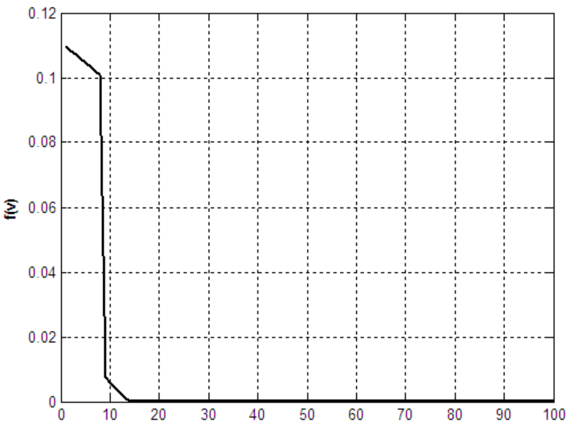


Fig. 3. Objective function behavior

Problem 2. Consider the following constrained nonlinear optimization problem, which is composed by inequality constraints and bounded variables:

$$Min\, f(\boldsymbol{v}) = e^{v_1} + v_1^2 + 4v_1 + 2v_2^2 - 6v_2 + 2v_3$$

Subject to: $v_1^2 + e^{v_2} + 6v_3 \le 15$

$$v_1^4 - v_2 + 5v_3 \le 25$$

$$v_1^3 + v_2^2 - v_3 \le 10$$

$$0 \le v_1 \le 4$$

$$0 \le v_2 \le 2$$

$$v_3 \ge 0$$

This problem has a unique optimal solution $\boldsymbol{v}^* = [0.00 \quad 1.50 \quad 0.00]^T$ and the minimal value of $f(\boldsymbol{v}^*)$ at this point is equal to –3.50. All simulation results provided by the neuro-genetic system show that it is convergent to $\boldsymbol{v}^*$.

Figure 4 shows the trajectories of the neuro-genetic system starting from $\boldsymbol{v}^0 = [9.50 \; 2.31 \; 6.07]^T$ and converging towards the equilibrium point $\boldsymbol{v}^*$. The bound constraints represented by the last three equations are directly mapped through the piecewise activation function defined in (7).

The network has also been evaluated for different values of initial conditions. All trajectories lead toward the same equilibrium point. These results show also the ability and efficiency of the neuro-genetic system for solving constrained nonlinear optimization composed by inequality constraints and bounded variables.

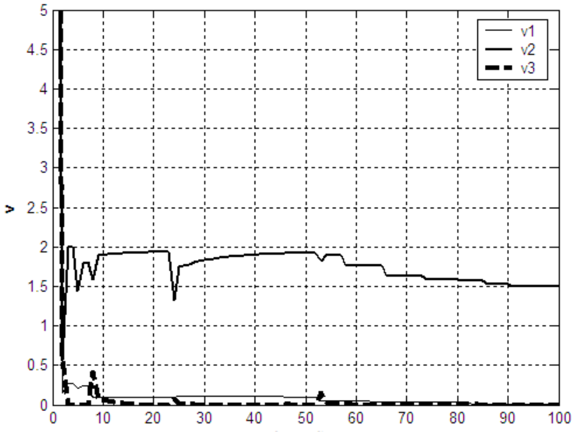


Fig. 4. Evolution of the neuro-genetic system output (Problem 2)

6 Conclusions

This paper presents a neuro-genetic approach for solving constrained nonlinear optimization problems. In contrast to the other neural approaches used in these types of problems, the main advantages of using the proposed approach in nonlinear optimization are the following: i) consideration of optimization and constraint terms in distinct stages with no interference with each other, i.e., the modified Hopfield network performs the optimization of constraints and the genetic algorithm is responsible to minimize the objective function, ii) unique energy term (E^{conf}) to group all constraints imposed on the problem, iii) the internal parameters of the modified Hopfield network are explicitly obtained by the valid-subspace technique of solutions, which avoids the need to use training algorithm for their adjustments; and iv) optimization and confinement terms are not weighted by penalty parameters, which could affect both precision of the equilibrium points and their respective convergence processes.

Some particularities of the neuro-genetic approach in relation to primal methods normally used in nonlinear optimization are the following: i) it is not necessary the computation, in each iteration, of the active set of constraints; ii) the initial solution used to initialize the network can be outside of the feasible set defined from the constraints. The simulation results demonstrate that the neuro-genetic system is an alternative method to solve constrained nonlinear optimization problems efficiently.

References

1. Bazaraa, M.S., Sherali, H.D., Shetty, C.M.: Nonlinear Programming. Wiley, New York (1993)
2. Tank, D.W., Hopfield, J.J.: Simple Neural Optimization Networks: An A/D Converter, Signal Decision Network, and a Linear Programming Circuit. IEEE Trans. on Circuits and Systems. 33 (1986) 533–541
3. Liang, X.B., Wang, J.: A Recurrent Neural Network for Nonlinear Optimization With a Continuously Differentiable Objective Function and Bound Constraints. IEEE Trans. on Neural Networks. 11 (2000) 1251–1262
4. Reifman, J., Feldman, E.E.: Multilayer Perceptron for Nonlinear Programming. Computers & Operations Research. 29 (2002) 1237–1250
5. Hopfield, J.J.: Neurons With a Graded Response Have Collective Computational Properties Like Those of Two-State Neurons. Proc. of the National Academy of Science. 81 (1984) 3088–3092
6. Aiyer, S.V., Niranjan, M., Fallside, F.: A Theoretical Investigation into the Performance of the Hopfield Network. IEEE Trans. on Neural Networks. 1 (1990) 53–60
7. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press, Cambridge (1996)
8. Luenberger, D.G.: Linear and Nonlinear Programming. Springer, New York (2003)
9. Vidyasagar, M.: Nonlinear Systems Analysis. Prentice-Hall, Englewood Cliffs (1993)

An Improved Primal-Dual Genetic Algorithm for Optimization in Dynamic Environments

Hongfeng Wang and Dingwei Wang

Institute of Systems Engineering, Northeastern University, P.R. China, 110004
{hfwang, dwwang}@mail.neu.edu.cn

Abstract. Inspired by the complementary and dominance mechanism in nature, the Primal-Dual Genetic Algorithm (PDGA) has been proved successful in dynamic environments. In this paper, an important operator in PDGA, primal-dual mapping, is discussed and a new statistics-based primal-dual mapping scheme is proposed. The experimental results on the dynamic optimization problems generated from a set of stationary benchmark problems show that the improved PDGA has stronger adaptability and robustness than the original for dynamic optimization problems.

1 Introduction

As a kind of robust optimization techniques for stationary optimization problems where the fitness landscape or objective function does not change during the course of computation, genetic algorithms (GAs) have extended their application areas to time-varying system as the necessary for solving real-world non-stationary problems increases recently. For these time-varying problems, the goal of GAs is no more to find a satisfactory solution, but to track the trajectory of the moving optimum point in the search space. This presents serious challenge to conventional GA because they fail to track the changing optimal solution once the population converges to one point.

To improve GA's performance in dynamic environments, researchers have developed many approaches [1, 2] to make GA maintain sufficient diversity for continuously adapting to fitness changes. Among these researches, the multiploidy and dominance mechanism that broadly exist in nature has been introduced into GAs and achieved some success for dynamic optimization problems. The multiploid representation that store redundant information in the genotype can provide a latent source of diversity in the population and allows population to respond more quickly to changing landscape [3]. Inspired this complementarity and dominance mechanism in nature, a genetic algorithm called Primal-Dual Genetic Algorithm [4, 5] (PDGA) is proposed, which is proved well suited for dynamic discrete optimization problems especially in binary-encoded space. In PDGA, a chromosome is defined a dual chromosome that is calculated through an operator called primal-dual mapping (PDM). In this work, a new primal-dual mapping scheme that can adaptively adjust the distance in genotype between a chromosome and its implement is proposed for PDGA instead of maximum distance in a given distance space. And we compare the performance of the PDGA with adaptive PDM scheme over the original in the following experiments on some dynamic optimization problems.

The paper's outline is as follows: Section 2 reviews relevant literature in brief. The next section the new adaptive primal-dual mapping scheme and the framework of modified PDGA are described. Then, a set of dynamic optimization problems are introduced in Section 4 and the experimental results and analysis are reported in Section 5. The conclusions are drawn in the final section.

2 Related Researches

In dynamic environments, the main problem with traditional GAs appears to be that they eventually converge to an optimum and thereby loose their diversity necessary for efficiently exploring the search space. Over the past few years, some researchers have begun to address this problem in the complement-based ways. The related works will be surveyed in this section.

The most prominent approach to complementary mechanism seems to be multiploidy. Goldberg and Smith [6, 7] report their examinations on a genetic algorithm with using diploid and dominance which achieves better adaptive qualities than a simple GA on a time-varying knapsack problem. Hadad and Eick [8] use multiploidy and a dominance vector as an additional part of an individual that breaks ties whenever there are an equal amount of 0's and 1's at a specific gene location. Ryan [9] uses additive multiploidy, where the genes determining one trait are added in order to determine the phenotypic trait. Lewis [10] et al. have compared several multiploid approaches and observed some interesting results: a simple dominance scheme is not sufficient to track the optimum well. If the methods are extended with a dominance change mechanism, much better result can be obtained.

Collard and his co-workers propose their Dual Genetic Algorithm [11] (DGA), which has exhibited the complementary mechanism in nature too. The DGA operates on a search space by introducing a meta-gene added in front of the regular bits that, when set to'0', has no effect, but all regular bits are translated to their compliment for fitness evaluation when set to '1'. Thus, each point in search space has two complementary representations, e.g. the two individuals [0 011] and [1 100] have the same phenotypic meaning. The added meta-bit undergoes the same genetic operations within DGA as other regular bits do.

3 Improved Primal-Dual Genetic Algorithm

3.1 Primal-Dual Genetic Algorithm

Inspired by the complementary mechanism in nature, Yang has proposed his PDGA for dynamic 0-1 optimization problems. Here we first introduce its framework simply. A chromosome recorded explicitly in the population is called a primal chromosome in PDGA. The chromosome that has maximum distance to a primal chromosome in a distance space is called its dual chromosome. In binary-encoded space, the Hamming distance (the number of locations where the corresponding bits of two chromosomes differ) is usually used as the definition of distance. Given a primal chromosome $x = (x_1, x_2, ..., x_L) \in I = \{0,1\}^L$ of fixed length *L*, its dual or its implement $x' = dual(x) = (x_1', x_2', ..., x_L') \in I = \{0,1\}^L$, where $dual(\cdot)$ is the primal-dual mapping

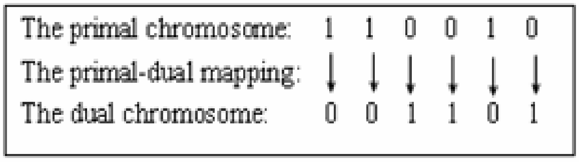

The primal chromosome: 1 1 0 0 1 0
The primal-dual mapping:
The dual chromosome: 0 0 1 1 0 1

Fig. 1. An Example Operation of PDM

function and $x_i^{'} = 1 - x_i$, Fig. 1 shows an example of applying primal-dual mapping operator to a 6-bit string chromosome.

With above definition, a set of low fitness primal chromosomes are selected to evaluate their duals during the run of PDGA and before the next generation starts. For every candidate, it is replaced with the dual if the dual is evaluated to be fitter; otherwise it is survived into the next generation.

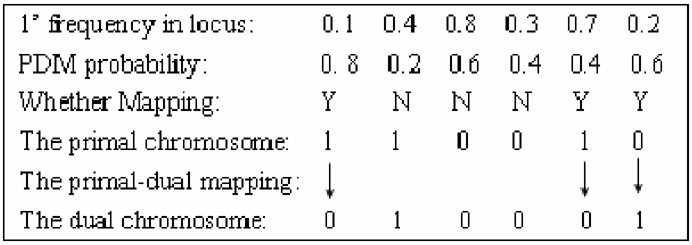
3.2 Adaptive Primal-Dual Mapping Operator

In PDGA, the primal-dual mapping function is a crucial operator, which is originally designed the maximum distance in Hamming distance space. That is, each bit of a chromosome string is translated to its complement during the course of PDM calculation. However, this hyper-mapping scheme however might become helpless, e.g. because the individuals converging into the high performance area in the late iterations makes their dual become inferior, or because the system changing little makes PDM become invalid. And the invalid PDM is usually not also able to benefit GA's performance. Here a new adaptive PDM operator, called statistic-based primal-dual mapping (SPDM), is proposed to improve mapping validity. In SPDM, not each bit participates in the PDM calculation any more and whether they are calculated is determined by a statistic probability in the population.

Within SPDM, the mapping probability of each locus is an important variant. Here an adaptive scheme to decide the value of mapping probability by the statistic information of each gene locus in the population is introduced. Let $p(i)$ denote the mapping probability of the locus i; f_{ki} denote the frequency of k's in the gene locus i over the population, where i=1,2,…,L and k is one of the gene values. Then in the 0-1 encoding space, we have f_{1i}+f_{0i}=1, $0 \le f_{0i}, f_{1i} \le 1$, and f_{1i} can be regarded as the tendency to "1" for the locus i. If each f_{1i} tends to equal to 1 or 0, the whole population is converging to one point. Thus the one-dimension vector $\{ f_{11}, f_{12}, \ldots, f_{1L} \}$ can express the convergence degree of the population. Then the calculation equation from f_{1i} to $p(i)$ is given as follows:

$$p(i) = \begin{cases} 0.5 - f_{1i}, f_{1i} \le 0.5 \\ f_{1i} - 0.5, f_{1i} > 0.5 \end{cases} \tag{1}$$

Obviously, the same result also can be gained through f_{0i} in a binary-encoded space. From equation 1, we can find that $p(i)$ can achieve the minimum value when f_{1i} is equal to 0.5 and the maximum value while equaling to 1 or 0. That is, the more the allele value in a gene locus converges, the more the mapping probability is. It is mostly considered that a diverse, spread-out population can adapt to changes more easily. From the statistics-based method, PDM can be adjusted by the convergence degree of population. And the original mapping scheme is looked as a special instance when $p(1)=p(2)=\ldots=p(L)=1$. Fig. 2 shows an example of applying the statistic operator to the same individual as Fig. 1.



1' frequency in locus:	0.1	0.4	0.8	0.3	0.7	0.2
PDM probability:	0. 8	0.2	0.6	0.4	0.4	0.6
Whether Mapping:	Y	N	N	N	Y	Y
The primal chromosome:	1	1	0	0	1	0
The primal-dual mapping:	↓				↓	↓
The dual chromosome:	0	1	0	0	0	1

Fig. 2. An Example operation of SPDM

In Yang's PDGA, the dual chromosomes' representation is unique and only the chromosomes selected into the PDM set have the chance to jump to their complements. However, there are multiple genotypes for an individual's complement in improved PDGA (IPDGA) with SPDM operator. This mechanism is similar to the polyploidy in nature but a pseudo-multiploidy is used actually since the encoding is not multi-stranded but single-stranded chromosome and it works at the chromosome level. Thus, IPDGA can keep more diversity and help to explore the solution space more effectively than the original. And another difference between the improved algorithm and the original lies that all chromosomes are requested to evaluate their duals in IPDGA in order to enforce the sufficient diversity in the population.

4 The Test Problems

A set of well studied stationary problems is selected to compare the performance between two kinds of PDGA, DGA and SGA. The dynamic problems are constructed from these stationary problems by a special dynamic problem generator.

4.1 Stationary Optimization Problems

1 Bit-Matching Problem

The bit-matching problem is one of One-Max problems where the individual's function is the number of bits matching a given template. The template is a 100-bit binary string for this study. So the optimum solution's fitness is 100.

2 Royal-Road Function

This function is studied by Michel, Forrest and Holland [12], which is defined on a 64-bit string consisting of eight continuous building blocks (BBs) of eight bits. The fitness is calculated by summing the contributions c_i corresponding to each of given BBs s_i, of which x is an instance (denote $x \in s_i$). That is, the fitness of a bit string x is calculated as follows:

$$f(x) = \sum_{i=1}^{8} c_i \cdot \delta_i(x) \qquad (2)$$

where c_i=8, and $\delta_i(x)$={1, if x is an instance of s_i; 0, otherwise}. And the fitness value of optimum string is 64.

3 Deceptive Function

The deceptive functions are a family of functions where there exist low-order BBs that do not combine to form the higher-order BBs. Here a deceptive function consisting of 10 copies of the order-3 fully deceptive function [13] is constructed for this study. And the optimum fitness value is 300.

4.2 Dynamic Problem Generator

In this paper, a dynamic test problem generator from a given stationary problem is introduced as follows. Given a binary-encoded stationary problem $f(x)$, a binary template T of the chromosome length is created first randomly or in a controlled way. Second, each chromosome x in the population performs the operation $x \oplus T$ where $\oplus$is a bit-wise exclusive-or operator (i.e. $0\oplus0=0$, $1\oplus0=1$, $1\oplus1=0$). If the environment changes in the generation t, we have $f(x,t+1)=f(x\oplus T,t)$ at the generation $(t+1)$.

The advantage of this way is the fitness landscape can still keep the certain properties in the original landscape after a change occurs since the environmental change is brought by a bit operation to the individuals. Moreover, when a bit is one in the template T, the corresponding allele in the chromosome x is reversed (i.e. form 0 to 1, or form 1 to 0), that does not take place when it is zero. So the total number of 1's in the template T can be looked as the severity of change. Let s devote the percent of 1's the template T contains, which allows controlling the severity of a change. In our experiments, s is set to 0.1, 05 and 0.9 which denotes the slightly changing environment, the randomly changing environment and the strongly changing environment respectively.

5 Experiments

For our experiments, we will compare IPDGA with original PDGA, traditional SGA and Collard and his co-worker's DGA. The population size N is set to 100 for all the GAs and typical genetic operators are used and parameter settings: one-point crossover with a fixed probability p_c=0.6, bit mutation with the mutation probability p_m=0.001, and fitness proportionate selection with roulette wheel. The best N chromosomes among all the parents and children are always transferred to the next generation. And each experiment result is averaged over 100 runs with different random seed.

Since for dynamic optimization problems a single, time-invariant optimal solution does not exist, the goal is not to find the extrema but to track their progression through the space as closely as possible. We will here report an offline performance function as measuring performance, which is the average of the best solutions at each time step, i.e. $x_T^* = (\sum_{t=1}^{T} e_t^*)/T$ with e_t^* being the best solution at time t. Note that the number of values that are used for the average grows with time, thus the curves tend to get smoother and smoother.

5.1 Experiments and Analysis on Bit-Matching Problem

We first compare and analyse the experimental results of different GAs on the periodically shifting Bit-Matching problem. The maximum allowable generation is set to 2000 and the fitness landscape is shifted every 200 generations. That is, there are 10 change periods for all GAs. Fig. 3 shows the results on the Bit-Matching problems in different severity environments.

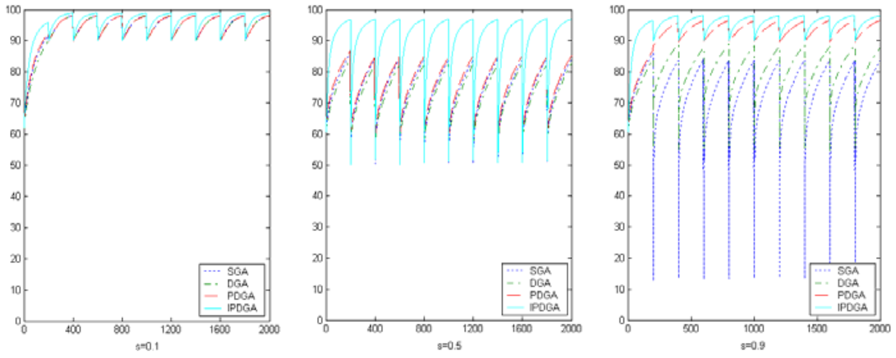


Fig. 3. Experimental results on Bit-Matching Problem in dynamic environments

From Fig. 3, IPDGA outperforms the other GAs in all periods especially when *s* equals 0.5. For PDGA, its performance curves almost overlap with DGA's and SGA's when the environment change is slight while it performs much better when the change becomes very strong. For DGA, the similar results can also be obtained but it is beaten by PDGA in the high-severity environments.

In IPDGA, the SPDM operator can keep sufficient diversity in the population and make the individuals to jump quickly to the new optimum point or nearby when the environment changes. Though the original PDM operator also enforces some diversity in PDGA, the efficiency responding to a change becomes very low since most primal-dual mapping operations can become valid if the environment change is not strong. And DGA behaves much less efficiently than the above algorithms due to the blindness in mutating the meta-bit. Within SGA, the performance curves gain a little improvement since a re-initialization method is used that 10% individuals in the population is generated again randomly after a change occurs.

5.2 Experiments and Analysis on Royal Road Function

The same parameters on Royal Road function are set as on Bit-matching problem. The experimental results are shown in Fig. 4.

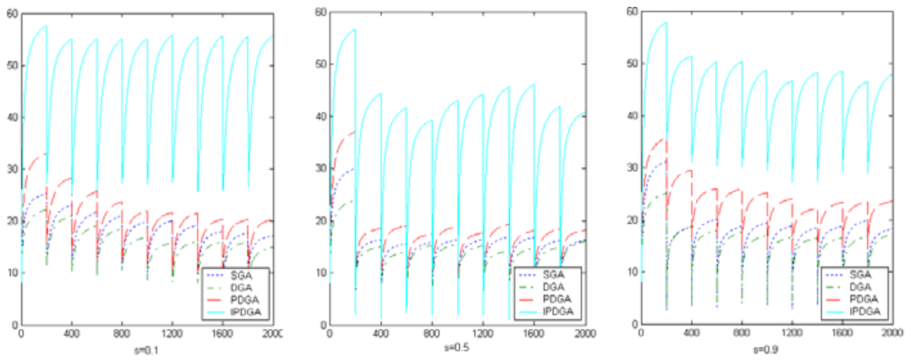


Fig. 4. Experimental results on Royal Road Function in dynamic environments

From Fig. 4, IPDGA now outperforms PDGA, DGA and SGA with a much higher degree on the Royal Road function than it does on the Bit-Matching problem. PDGA has always achieved more BBs than DGA and SGA in all period, but cannot find the optimum solution for the dynamic periods as IPDGA does. DGA is disappointing here and even performs worse than SGA.

The basis BBs in the Royal Road function are of order-8, which is very hard to be searched for the GAs. In IPDGA, the adaptive complementary mechanism can offer any block the chance translating to a building block while only the block, where all the bits are equal to zero, has such the chance in PDGA. So IPDGA can adapt more effectively to the environment than PDGA. In DGA, the efficiency of diversity-keeping is poor and even worse than that of SGA.

5.3 Experiments and Analysis on Deceptive Function

Fig. 5 shows the experimental results on the Deceptive function, which seems a little similar to the situation in Fig. 3. IPDGA performs best and SGA performs worst and PDGA outperforms DGA. The reason is explained on the Bit-matching problem and the Royal Road function.

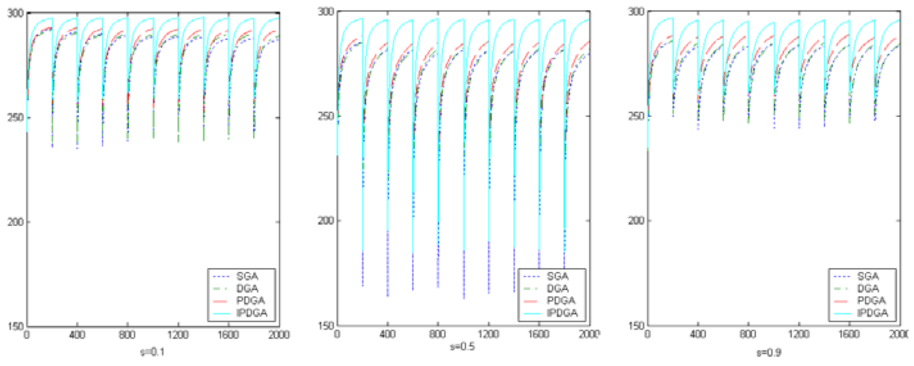


Fig. 5. Experimental results on Deceptive Function in dynamic environments

All the experimental results show that IPDGA has stronger robustness and adaptability than PDGA in the dynamic environments. The meta-bit scheme used in DGA also improves its adaptability under dynamic environments. However, the effect is limited due to blindness in applying the complementary mechanism.

6 Conclusions

In this paper, a genetic algorithm, Primal-Dual Genetic Algorithm, for dynamic optimization problems is introduced and the key operator with PDGA, the primal-dual mapping operator, is discussed and improved by an adaptive statistics-based scheme which uses the statistic information of allele distribution in the current population to adjust the operation probability for each gene locus. Experiment study over the dynamic optimization problems suggests that IPDGA can solve the complex dynamic problems more effectively than original PDGA, traditional GA and Collard and coworkers' DGA. Experimental results show that the mechanism of primal-dual chromosomes in PDGA is a very effective diversity-keeping idea. So many questions retain open and are worthy for future study in PDGA. For example, the idea of dualism can be extended to real-code GAs and also generalized to other optimization algorithms, which are very interesting works.

References

1. Cobb, H.G..: An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environment. Technical Report AIC-90-001, Naval Research Laboratory, Washington, USA, (1990)
2. Grefenstette, J.J.: Genetic algorithms for changing environments. In: Maenner, R. Manderick, B.(eds.): Parallel Problem Solving from Nature 2,. North Holland, (1992) 137-144
3. Lewis, J., Hart, E., Ritchie, G.: A Comparison of Dominance Mechanisms and Simple Mutation on Non-Stationary Problems. In: Eiben, A.E., Back, T., Schoenauer, M., Schwefel, H.P.(eds.): Proceeding of the 5th Int. Conf. on Parallel Problem Solving from Nature, (1998) 139-148
4. Yang, S.: The Primal-Dual Genetic algorithm. In Proc. of the 3rd Int. Conf. on Hybrid Intelligent System. IOS Press. (2003)
5. Yang, S.: Non-stationary problem optimization using the primal-dual genetic algorithm. Proceeding of the 2003 Congress on Evolutionary Computation, (2003) 2246-2253
6. Goldberg, D.E., Smith, R.E.: Nonstationary function optimization using genetic algorithms with dominance and diploidy. In: Grefenstette, J.J.(eds.): Second International Conference on Genetic Algorithms, Lawrence Erlbaum Associates, (1987) 59-68
7. Smith, R.E.: Diploid genetic algorithms for search in time varying environments. In Annual Southeast Regional Conference of the ACM, New York, (1987) 175-179
8. Hadad, B.S., F.Eick, C.: Supporting polyploidy in genetic algorithms using dominance vectors. In: Angeline, PJ, Reynolds, R.G., Mcdonnell, J.R., Eberhart, R.(eds.): Proceedings of the Sixth International Conference on Evolutionary Programming, (1997) 223-234
9. Ryan, C.: Diploidy without dominance. In: Alander, J.T.(eds.): Third Nordic Workshop on Genetic Algorithms, (1997)63-70

10. Lewis, J., Hart, E., Ritchie, G.: A comparison of dominance mechanisms and simple mutation on non-stationary problems. In Eiben, A.E., Back, T., Schoenauer, M., Schwefel, H.P. (eds.): Parallel Problem Solving from Nature, (1998) 139-148
11. Collard, P., Escazut, C., Gaspar, A.: An evolutionary approach for time dependant optimization. International Journal on Artificial Intelligence Tools, 6 (1997) 665-695
12. Mitchell, M., Forest, S., Holland, J.H.: The Royal Road for Genetic Algorithms: Fitness Landscape and GA performance. In: Varela, F.J., Bourgine, P. (eds.): Proceeding of 1st European Conference on Artificial Life, MIT-Press (1992)245-254
13. Goldberg, D.E.: Genetic Algorithms and Walsh Function: Part I, a Gentle Introduction. Complex System, (1989) 129-152

Multiobjective Optimization Design of a Hybrid Actuator with Genetic Algorithm

Ke Zhang

School of Mechanical and Automation Engineering, Shanghai Institute of Technology
120 Caobao Road, 200235 Shanghai, China
zkwy@hotmail.com

Abstract. A hybrid mechanism is a configuration that combines the motions of two characteristically different electric motors by means of a mechanism to produce programmable output. In order to obtain better integrative performances of hybrid mechanism, based on the dynamics and kinematic analysis for a hybrid five-bar mechanism, a multi-objective optimization of hybrid five bar mechanism is performed with respect to four design criteria in this paper. Optimum dimensions are obtained assuming there are no dimensional tolerances and clearances. By the use of the properties of global search of genetic algorithm (GA), an improved GA algorithm is proposed based on real-code. Finally, a numerical example is carried out, and the simulation result shows that the optimization method is feasible and satisfactory in the design of hybrid actuator.

1 Introduction

Hybrid mechanism is a new type of planar controllable mechanism. A hybrid mechanism is to combine the motion of a large constant speed motor, with a small servomotor via a two degree of freedom (DOF) mechanism, where, the constant speed motor provides the main torque and motion requirements, while the servomotor contributes to modulations on this motion. Such machines will introduce to users greater flexibility with programmability option, and energy utilization will be realized at maximum. Although some points are partially explored, there is still a need for optimal design studies to guide potential users for possible industrial applications with hybrid machines.

Previous work in hybrid machine can be found in some studies and publications. Tokuz and Jones [1] have used a hybrid machine configuration to produce a reciprocating motion. A slider crank mechanism was driven by a differential gearbox having two separate inputs; constant speed motor and a pancake servomotor to simulate a stamping press. A mathematical model was developed for the hybrid machine-motor system. The model results were compared with the experimental ones, and model validation was achieved. Later Greenough et al. [2] have presented a study on optimization design of hybrid machines, and a Svoboda linkage is considered as a two degree of mechanism. Kinematic analysis of Svoboda linkage was presented with inverse kinematics issue. Kireçci and Dülger [3] have proposed a different hybrid arrangement driven by two permanent magnet DC servomotors and a constant speed motor. In the configuration, a slider crank mechanism having an adjustable crank was used together with two power screw arrangements to produce a motion in x-y plane. A mathematical model was prepared and the simulation results were presented for different motion requirements including electrical drive properties. Kireçci [4] has then offered a hybrid drive system involving a servomotor and a constant speed motor for a printing machine. Kireçci and Dülger [3] had given description of a different hybrid actuator configuration consisting of a servomotor driven seven link mechanism with an adjustable crank. Wang [5] has design a variable structure generalized linkage, the linkage is optimized to perform exactly the complicated motion required.

The above design of hybrid mechanism employed mainly traditional optimization design methods. However, these traditional optimization methods have drawbacks in finding the global optimal solution, because it is so easy for these traditional methods to trap in local minimum points [6].

GA refer to global optimization technique based on natural selection and the genetic reproduction mechanism [7], [8]. GA is a directed random search technique that is widely applied in optimization problems. This is especially useful for complex optimization problems where the number of parameters is large and the analytical solutions are difficult to obtain [9]. GA can help to find out the optimal solution globally over a domain. In this paper, the standard GA is modified and new genetic operators are introduced to improve its performance.

Optimal dynamic design is an important subject in designing a hybrid mechanism. The aim of our study is to optimize the synthesized mechanism. It is necessary to take into account not only the kinematic performance but also the dynamic performance. Hybrid five bar mechanism is the most representative one of hybrid mechanism, and is presented in this paper. The paper presents kinematic analysis, dynamic analysis, and the multi-objective optimization design of hybrid mechanism by deriving its mathematical model. By means of four optimization goals, optimal design for hybrid five bar mechanism is taken by using GA. The calculation results of the example are obtained in this system herein.

2 Hybrid Mechanical Description

Figure 1 represents five link mechanism configuration having all revolute joints except one slider on output link. Fig.2 shows the positional, the geometrical and the dynamic relationships. Notations shown in Fig. 1 and 2 are applied throughout the study. The hybrid mechanism has an adjustable link designed to include a power screw mechanism for converting rotary motion to linear motion by means of a small slider. The slider is assumed to move on a frictionless plane.

The crank is driven by a DC motor (the main motor) through a reduction gearbox; the slider is driven by a lead screw coupled the second servomotor (the assist motor). Here the main motor is applied as a constant speed motor, and the constant speed motor profile is applied. Point-to-point positioning is certainly achieved for both motors, and the system output is taken from the last link.

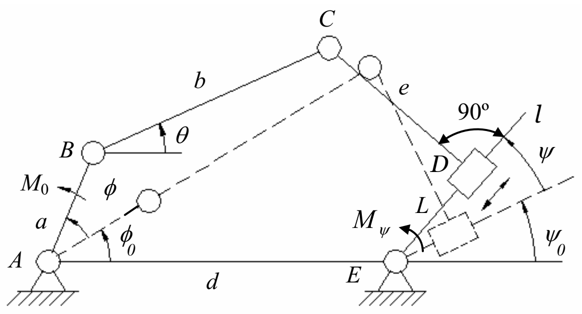


Fig. 1. Schematic diagram of five link mechanism

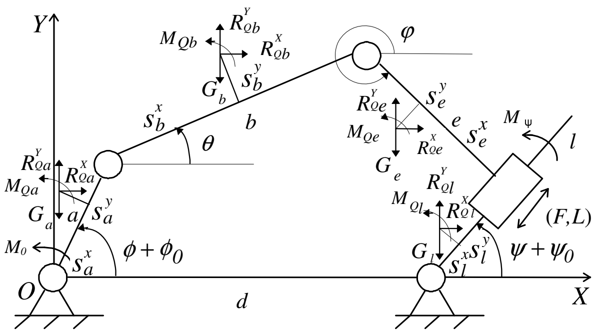


Fig. 2. Configuration of five link mechanism

a, b, d, e link lengths of the mechanism (m)

$\phi, \theta, \varphi, \psi$ angular displacement of the links (rad)

$\dot{\phi}, \dot{\theta}, \dot{\varphi}, \dot{\psi}$ angular velocity of the links (rad/s)

$\ddot{\phi}, \ddot{\theta}, \ddot{\varphi}, \ddot{\psi}$ angular acceleration of the links (rad/s^2)

S_i^x, S_i^y ($i = a, b, e, l$) positions to the centre of gravity in local coordinates (m)

m_i masses of the links (kg)

G_i gravity of the links (N)

J_{is} link moment of inertias on the mass centre of the links(kgm^2)

$L, \dot{L}, \ddot{L}$ displacement, velocity, and acceleration of the slider on output link (m, m/s, m/s^2)

F the assist driving force (N)

M_0 the main driving torque (Nm)

M_ψ drag torque on output link (Nm)

R_{Qi}^X, R_{Qi}^Y inertia forces of the links (N)

M_{Qi} inertia torques of the links (N)

X_i, Y_i positions to the mass centre of the links in fixed coordinates (m)

3 Kinematic Analysis of Hybrid Mechanism

Kinematic analysis of five bar linkage is needed while carrying out derivations for the mathematical model. The mechanism is shown with its position vectors in Fig. 1. The output of system is dependent on two separate motor inputs and the geometry of five bar mechanism.

By referring to Fig. 1, the loop closure equation is written as:

$$\overrightarrow{AB} + \overrightarrow{BC} + \overrightarrow{CD} = \overrightarrow{AE} + \overrightarrow{ED} \tag{1}$$

By solving vector loop equation (1), we can obtain angular position of the each link. Having found the angular displacements of each linkage in the five bar linkage, time derivatives can be taken to find angular velocity and accelerations. They are also definitely needed during the analysis of dynamic model.

4 Dynamic Analysis of Hybrid Mechanism

In general, the model of a mechanical system can simply be considered as inertial rigid system. Simplifying assumptions are required while developing the mathematical model. Friction and clearance in all joints are neglected. The mechanism operates in vertical plane and gravity effects are included. Figure 3 shows the bond graph model of hybrid five-bar mechanism [10]. It is composed of three parts: (1) Multi-ports element MCHANISM, (2) Inertial field, (3) Source field.

In Fig. 3, there exist *N* 1-Junctions corresponding to velocities vector of mechanism, $\dot{q}_K$ ($\dot{q}_K = [\dot{q}_{KI}\dot{q}_{KD}]^T$). According to bond graph theory [11], an algebraic sum of all efforts (e_i) on the bonds attached to a 1-Junction is zero. From the bond graph in Fig. 3, the effort summations at two 1-Junctions associated with the independent generalized velocities $\dot{q}_{KI}$ vectors are written as follows:

$$e_{SI} = e_{KD} + e_{PI} \tag{2}$$

The effort summations at (*N*-2) 1-Junctions associated with dependent velocity $\dot{q}_{KD}$ vectors are written as follows:

$$e_{SD} = -e_{KI} + e_{PD} \tag{3}$$

where e_{KI} , e_{PI} , e_{SI} are 2-effort vector , and e_{KD} , e_{PD} , e_{SD} are (*N*-2)-effort vector.

According to literature [10] and equations (2), (3), we can get

$$(e_{SI} - e_{PI}) = T^T(q_K, \Psi)(e_{PD} - e_{SD}) \tag{4}$$

From (4), we may found the dynamic equation of hybrid mechanism in the form

$$A_1\ddot{q}_{KI} + A_2\dot{q}_{KI} + A_3U_1 + A_4U_2 = 0 \tag{5}$$

where A_1, A_2, A_3 and A_4 are coefficient matrixes of dynamic equation, U_1 represent input torques(forces) vector, and U_2 represent other torques(forces) on hybrid mechanism. Thus, the 2-vector U_1 can be found from equation (5)

$$U_1 = (-1)A_3^{-1}(A_1\ddot{q}_{KI} + A_2\dot{q}_{KI} + A_4U_2 + A_5) \tag{6}$$

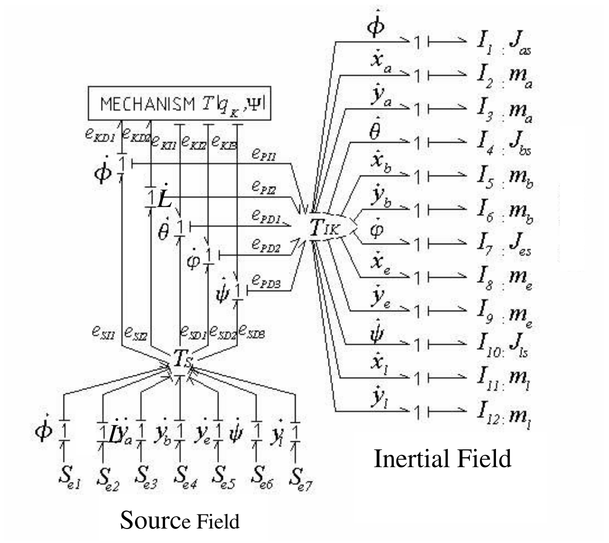


Fig. 3. Bond graph model of hybrid five bar mechanism

5 Optimum Dynamics Design of Hybrid Mechanism

5.1 Design Variables

Hybrid mechanism can be determined by selecting a design vector as follows：

$$x = [x_1, x_2, x_3, x_4, x_5]^T \tag{7}$$

where, $x_1 = a/d, x_2 = b/d, x_3 = e/d, x_4 = \phi_0, x_5 = \psi_0$.

5.2 Objective Functions

The problem of determining mechanism dimensions can be expressed as a constrained optimization problem. In order to ensure the synthetical performance of the hybrid mechanism, five performance criteria are simultaneously considered in the optimization of the mechanism as follows.

$$f_1(x) = L_{\max} - L_{\min} \tag{8}$$

$$f_2(x) = \max(\dot{L}) \tag{9}$$

$$f_3(x) = \max(F) \tag{10}$$

$$f_4(x) = \max(F\dot{L}) \tag{11}$$

Considering the difference on unit of each sub-objective function value, order of magnitude of them must be harmonized in the course of optimization. Thus the multi-objective optimization function is designed as follows.

$$f(x) = \omega_1 f_1'(x) + \omega_2 f_2'(x) + \omega_3 f_3'(x) + \omega_4 f_4'(x) \tag{12}$$

where $f_i'(x)$ is sub-objective function harmonized order of magnitude, and there is a relationship among weighting coefficients ω_i,

$$\sum_{i=1}^{4} \omega_i = 1 \quad (i=1,2,3,4) \tag{13}$$

5.3 Constraint Functions

These functions consist of inequality constraints with stand type according to Matlab optimization toolbox. They are functions of design variables.

1) Inequality constraint related to the movable condition of hybrid mechanisms

To ensure existence of the hybrid five-bar mechanism, the follow inequality constraints are to be satisfied.

$$\begin{aligned} &a + d - b - \sqrt{e^2 + L^2} < 0 \\ &a + \sqrt{e^2 + L^2} - b - d < 0 \\ &a + b - \sqrt{e^2 + L^2} - d < 0 \\ &a - \min(b, \sqrt{e^2 + L^2}, d) < 0 \end{aligned} \tag{14}$$

2) Inequality constraint due to the transmission angle

$$\begin{aligned} &\frac{b^2 + c^2 + d^2 - (d - a)^2}{2 \cdot b \cdot \sqrt{e^2 + L^2}} - \cos[\gamma] \le 0 \\ &\frac{b^2 + c^2 + d^2 - (d + a)^2}{2 \cdot b \cdot \sqrt{e^2 + L^2}} - \cos[\gamma] \le 0 \end{aligned} \tag{15}$$

where $[\gamma]$ is allowable transmission angle of mechanism.

5.4 Improved GA

5.4.1 Initial Population

The initial population is a potential solution set Q. The first set of population is usually generated randomly.

$$Q = \{q(1), q(2), \cdots, q(i), \cdots, q(N)\} \tag{16}$$

$$q(i) = [q(1,i) \quad q(2,i) \quad \cdots \quad q(j,i) \quad \cdots \quad q(p,i)] \tag{17}$$

$$q_{\min}(j,i) \le q(j,i) \le q_{\max}(j,i) \tag{18}$$

where N denotes the population size; p denotes the number of variables to be tuned; $q(j,i)$ $(i = 1,2,\ldots,N;\ j = 1,2,\ldots,p)$ are parameters to be tuned; It can be seen from equations (16)–(17) that the potential solution set Q contains some candidate solutions (chromosomes). The chromosome contains some variables $q(j,i)$ (genes).

5.4.2 Evaluation

Each chromosome in the population will be evaluated by a defined fitness function. The better chromosomes will return higher values in this process. According to characteristics of objective functions, the fitness function to evaluate a chromosome in the population can be written as

$$\text{fitness}(i) = 1/[\delta + f(i)] \tag{19}$$

where δ is an adjustment parameter for preventing that denominator is zero. The form of the fitness function depends on the application.

Check every individual whether meet constraints function, if they meet constraints function, then calculate their objective function value and fitness function value ; if some or other individual dissatisfy constraints function, then it may bring on overflow problem of numerical calculation, optimization may not be performed. So, the algorithm is improved as follows : we do not calculate objective function value of the individual, and directly set fitness value for zero, the individual can also be called lethal gene.

5.4.3 Selection

Two chromosomes in the population will be selected to undergo genetic operations for reproduction by the method of spinning the roulette wheel. It is believed that high potential parents will produce better offspring. The chromosome having a higher fitness value should therefore have a higher chance to be selected. The selection can be done by assigning a probability p_i to the chromosome $q(i)$,

$$p_i = f(q(i)) / \sum_{i=1}^{N} f(q(i)) \quad (i = 1,2,\cdots,N) \tag{20}$$

The cumulative probability p'_i for the chromosome p_k is defined as

$$p'_i = \sum_{k=1}^{i} p_k \quad (i = 1,2,\cdots,N) \tag{21}$$

The selection process starts by randomly generating a nonzero floating-point number, $d \in [0 \quad 1]$. Then, the chromosome p_i is chosen if $p'_{i-1} < d \le p'_i$ $(p'_0 = 0)$. It can be observed from this selection process that a chromosome having a larger $f(q(i))$ will have a higher chance to be selected. Consequently, the best chromosomes will get more offspring, the average will stay and the worst will die off.

5.4.4 Genetic Operations

1) Crossover: the crossover operation is mainly for exchanging information from the two parents, chromosomes $q(j,i_1)$ and $q(j,i_2)$, obtained in the selection process. The two parents will produce one offspring. First, four chromosomes will be generated according to the following mechanisms:

$$q_1^c(j,i) = (q(j,i_1) + q(j,i_2))/2 \tag{22}$$

$$q_2^c(j,i) = q_{\max}(j,i)(1-\omega) + \max(q(j,i_1), q(j,i_2))\omega \tag{23}$$

$$q_3^c(j,i) = q_{\min}(j,i)(1-\omega) + \min(q(j,i_1), q(j,i_2))\omega \tag{24}$$

$$q_4^c(j,i) = (q_{\max}(j,i) + q_{\min}(j,i))(1-\omega) + (q(j,i_1) + q(j,i_2))\omega)/2 \tag{25}$$

where $\omega \in [0\ \ 1]$ denotes the weight to be determined by users, $\max(q(j,i_1), q(j,i_2))$ denotes the vector with each element obtained by taking the maximum among the corresponding element of $q(j,i_1)$ and $q(j,i_2)$. Similarly, $\min(q(j,i_1), q(j,i_2))$ gives a vector by taking the minimum value. Among $q_1^c(j,i)$ to $q_4^c(j,i)$, the one with the largest fitness value is used as the offspring of the crossover operation. The offspring is defined as $q_{k_{os}}^c(j,i)$, k_{os} denotes the index *k* which gives a maximum value of $f(q_k^c(j,i))$, $k = 1,2,3,4$.

2) Mutation: the offspring will then undergo the mutation operation. The mutation operation is to change the genes of the chromosomes. Consequently, the features of the chromosomes inherited from their parents can be change. Three new offspring will be generated by the mutation operation

$$q_k^m(j,i) = q_{k_{os}}^c(j,i) + b_j \Delta q^m(j,i) \quad k = 1,2,3 \tag{26}$$

where b_j, $j = 1,2,\cdots,p$, can only take the value of 0 or 1, $\Delta q^m(j,i)$, $j = 1,2,\cdots,p$, are randomly generated numbers such that $q_{k_{os}}^c(j,i) + \Delta q^m(j,i) \in [0\ \ 1]$. The first new offspring (*j* = 1) is obtained according to equation (26) with that only one (b_j being randomly generated within the range) is allowed to be one and all the others are zeros. The second new offspring is obtained according to equation (26) with that some b_j randomly chosen are set to be one and others are zeros. The third new offspring is obtained according to equation (26) with all $b_j = 1$. These three new offspring will then be evaluated using the fitness function of (19). A real number will be generated randomly and compared with a user-defined number $p_a \in [0\ \ 1]$. If the real number is smaller than p_a, the one with the largest fitness value among the three new offspring will be replace the chromosome with the smallest fitness f_s in the population. If the real number is lager than p_a, the first offspring $q_1^m(i,j)$ will replace the chromosome with the smallest fitness value f_s in the population if $f(q_1^m(j,i)) > f_s$; the second and the third offspring will do the same. p_a is effectively the probability of accepting a bad offspring in order to reduce the chance of converging to a local optimum. Hence, the possibility of reaching the global optimum is kept.

After the operation of selection, crossover, and mutation, a new population is generated. This new population will repeat the same process. Such an iterative process can be terminated when the result reaches a defined condition; a defined number of iteration has been reached.

6 Numerical Examples

In order to prove the effect of the proposed optimization design procedure numerical examples have been performed. Mechanical properties of five link mechanism, link lengths, positions to the center of gravity of each link, link masses, and link inertias on the masses center are shown in Table. 1. These link lengths and angle values for hybrid five bar mechanism in the studies of optimal kinematic design were obtained by Wang and Gao [5]. The output motion profile is designed as the law of sine acceleration.

Table 1. Mechanical properties of five link mechanism

a=0.04m, b=0.3902m, e=0.0694m, d=0.4m;
$\phi_0 = 8.8\times\pi/180\text{rad}$, $\dot{\phi} = 240\times\pi/30$ rad/s, $\ddot{\phi} \approx 0$ rad/s^2;
m_a=12.9 kg, m_b=2.3 kg, m_e=1.8kg, m_l=15.0 kg ;
J_{aS}=34.2×10^{-3} kgm^2, J_{bS}=70.4×10^{-3} kgm^2, J_{eS}=3.6×10^{-3} kgm^2, J_{lS}=447×10^{-3} kgm^2 ;
S_a^x =0.0m, S_a^y =0.0m, S_b^x =0.2405m, S_b^y =0.0m, S_e^x =0.0227m,
S_e^y =0.0m, S_l^x =0.0m, S_l^y =0.0 m ;

Experimental model in Fig. 1 have been designed that kinematic sizes can be adjusted statically relative to crank, the masses of links, inertias and positions to the masses center of links in the local coordinates are independent of the static adjusment.

Using the improved GA for the multi-objective optimization design, a computer program has been written. Initial values of design varibles can be obtained with link lengths of link and angle ϕ_0 in Table.1. If we choose [γ] = 45°, and $\omega_i = 0.25$ (i=1,2,3,4), mechanism dimensions obtained for hybrid five bar mechanism in this studies of multi-objective optimization design are shown in Table.2. Here, we analyze kinematic and dynamic performance of the multi-objective optimization design mechanism, compared with single objective kinematic and dynamic optimization design using objective functions as equations (9) and (11).

For kinematic performance index of the actuator, we choose the actuator manipulability index μ , $\mu = \sqrt{\det(\boldsymbol{J}\cdot\boldsymbol{J}^T)}$ [12], where matrix $\boldsymbol{J}$ is known as the Jacobin. Most of the measures of kinematic performance of linkage actuators are based on the Jacobin matrix. According to kinematic analysis of hybrid five bar actuator in Fig. 1, the Jacobin matrix can be found as follows

$$\boldsymbol{J} = \begin{bmatrix} \dfrac{a\cdot sin(\phi+\phi_0-\theta)}{e\cdot sin(\varphi-\theta)+L\cdot sin(\psi+\psi_0-\theta)} & \dfrac{cos(\psi+\psi_0-\theta)}{e\cdot sin(\varphi-\theta)+L\cdot sin(\psi+\psi_0-\theta)} \end{bmatrix} \tag{27}$$

Figure 4 shows the change of manipulability index μ values of the actuator. As shown in these graphs, manipulability index μ obtained by using multi-objective optimization is satisfactory, compared with kinematic and dynamic optimization. Hybrid actuator can get good kinematic performance.

Table 2. Optimal results of hybrid five bar mechanism

	a	*b*	*e*	*d*	ϕ_0 (°)
Optimal results	0.04	0.3862	0.0755	0.395	9.8

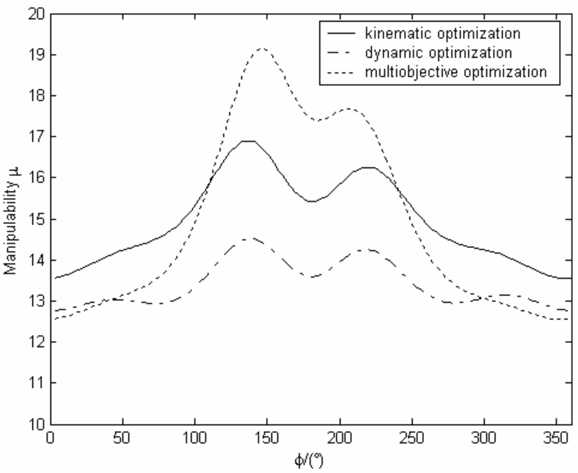


Fig. 4. The manipulability index of the hybrid five bar mechanism

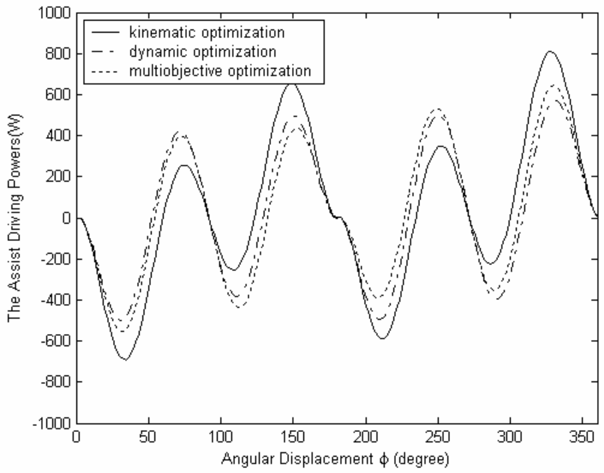


Fig. 5. The assist driving power of the hybrid five bar mechanism

For dynamic performance index of the actuator, we choose the assist driving power of the actuator. Figure 5 shows the assist driving power for the assist motor driving the slider on the lead screw. In Fig. 5, three curves represent respectively the calculations results for optimal kinematic design, optimal dynamic design and multi-objective optimization design. As seen in Fig. 5, multi-objective optimization can reduce the peak power of the assist drive, compared with optimal kinematic design. According to the above analysis, hybrid mechanism can obtain better integrative performance by using multi-objective optimization design.

7 Conclusions

The main aim of this study was while optimization hybrid mechanism, to consider simultaneously their kinematic performance, dynamic performance. Hybrid mechanism dimensions have been determined via multi-objective optimization based with four design goals: minimum displacement of the assist motion, minimum of the maximum velocity of the assist motion, minimum of the maximum driving force of the assist motion, and minimum of the maximum driving power of the assist motion. The optimal design for a hybrid five bar mechanism is performed by using an improved GA in this paper. Although some simplifications are made during derivations, this study illustrates how well the method. As a result of the comparisons, better integrative performances have been obtained in terms of multi-objective functions.

References

1. Tokuz, L. C., Jones, J. R.: Hybrid Machine Modelling and Control. PhD thesis, Liverpool Polytechnic (1992)
2. Greenough, J. D., Bradshaw, W. K., Gilmartin, M.J.: 9th World Congress on the Theory of Machines and Mechanisms 4 (1995) 2501-2505
3. Dülger, L. C., Kireçci, A., Topalbekiroglu, M.: Modeling and Simulation of a Hybrid Actuator. Mechanism and Machine Theory 38 (2003) 395-407
4. Kireçci, A., Dülger, L. C.: A Study on a Hybrid Actuator. Mechanism and Machine Theory 35 (8) (2000) 1141-1149
5. Wang S. Z., Gao, X.: Research on Kinematics Design for a Variable Structure Generalized linkage. Proceedings of Young Scientists Conference on Manufacturing Science and Technology for New Century Wuhan, China, (1998) 404-408
6. Singiresu, S.R. Engineering optimization. John Wiley & Sons New York (1996)
7. Pham, D.T. and Karaboga, D.: Intelligent Optimization Techniques, Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks. Springer-Verlag, Berlin Heidelberg New York (2000)
8. Michalewicz, Z.: Genetic Algorithm + Data Structures = Evolution programs, 3nd extended ed. Springer-Verlag, Berlin Heidelberg New York (1999)
9. Lee, H. W. J., Ali, M. M., Wong, K. H.: Global Optimization for a Special Class of Discrete-valued Optimal Control Problems. Dynamics of Continuous, Discrete and Impulsive Systems Series B: Algorithm and Applications (11) (2004) 735-756
10. Zhang, K., Wang, S. Z.: Dynamics Analysis of a Controllable Mechanism. Proceedings of the ASME International Engineering Technical Conferences and Computers and Information in Engineering Conference 7A (2005) 133-140
11. Karnopp, D. C., Margolis, D. L., Rosenberg, R. C.: System Dynamic: A Unified Approach. Wiley New York (1990)
12. Yoshikawa, T.: Manipulability of Robotic Mechanism. International Journal of Robotic Research 4 (2) (1985) 3-9

Human Hierarchical Behavior Based Mobile Agent Control in ISpace with Distributed Network Sensors

SangJoo Kim[1], TaeSeok Jin[2], and Hideki Hashimoto[3]

[1] Coresystem, Dong-eui Institute of Technology, San 72, Yangjung-dong, Busan, Korea
ksj_elec@hanmail.net
[2] Dept. of Mechatronics Engineering, DongSeo University,
San 69-1 Churye-2-dong, Sasang-Gu, Busan 617-716, Korea
jints@dongseo.ac.kr
[3] IIS, the University of Tokyo, 4-6-1 Komaba, Meguro-ku, Tokyo 153-8505, Japan
hashimoto@iis.u-tokyo.ac.jp

Abstract. The aim of this paper is to investigate a control framework for mobile robots, operating in shared environment with humans. The Intelligent Space (iSpace) can sense the whole space and evaluate the situations in the space by distributing sensors. The mobile agents serve the inhabitants in the space utilizes the evaluated information by iSpace. The iSpace evaluates the situations in the space and learns the walking behavior of the inhabitants. The human intelligence manifests in the space as a behavior, as a response to the situation in the space. The iSpace learns the behavior and applies to mobile agent motion planning and control. This paper introduces the application of fuzzy-neural network to describe the obstacle avoidance behavior learned from humans. Simulation and experiment results are introduced to demonstrate the efficiency of this method.

1 Introduction

The Intelligent Space (iSpace) is a space (room, corridor or street), which has ubiquitous distributed sensory intelligence (various sensors, such as cameras and microphones with intelligence) actuators (TV projectors, speakers, and mobile agent) to manipulate the space [1], [3].

The iSpace propagates mobile robots in the space, which act in the space in order to change the state of the space. These mobile robots are called mobile agents. Mobile Agents cooperating with each other and core of the iSpace to realize intelligent services to inhabitants. The intelligence in iSpace has capability of evaluation of situations inside the space [2]. The evaluated situations are applied for learning the behavior of inhabitants. The evaluated behaviors are given to the control system of mobile agent. There are many definitions of the intelligence. The intelligence can be considered as a reaction against a given action. Behavior is a generalized mapping between situations (state of the space) and actions. But the intelligence is also means capability of learning. The iSpace integrates both types of definitions. Inhabitants in the iSpace are producing intelligent reactions against instantaneous situation.

The iSpace evaluates situations (actions-reactions) from sensed information [4]. The evaluated situations are given to the learning system, where behaviors are concluded from situations. The mobile agents serve the inhabitants in the space utilizes the evaluated information by iSpace [5]. The mobile agents have sensors and/or actuators with computational devices and computer network communication capabilities. The iSpace senses the space and acting in the space. The sensing is done through distributed sensory network, and the acting is done by global actuators like projectors or speaker systems, or by local actuators like mobile agents. The mobile agents can sense the space and can act in the space locally.

The rest of this paper is organized as follows. The following section summarizes the pedestrian behavior models and proposes a mobile agent control framework. Section II explains the obstacle avoidance behavior and introduces a mathematical model to describe the particular behavior. Obstacle avoidance behavior is modeled by artificial potential fields. Fuzzy-Neural Non-linear Function Approximation is applied to describe the artificial potential functions. Section III introduces the evaluation and learning capability of Fuzzy Neural Network. The evaluation is done by walking path extraction from spatially distributed camera sensors. Section IV introduces some simulation examples to demonstrate the effectiveness of this method.

2 Modeling Obstacle Avoidance Behavior

Let us consider two typical styles (Figure 1.). One, main navigation behavior of an aircraft carrying dangerous material is to keep "as far from the mountains as possible" Two, remaining in secret while seeking a mouse leads to the opposite behavior for a cat, namely, "get as close to the object as possible"

A simple combination of these basic behavior styles can characterize the main rule of a traffic system: "keep close to the right or the left side". Let’s consider a simple example to illustrate the importance of this knowledge. Let's assume that Japanese and American person are walking towards each other. Recognizing this situation, they try to avoid each other. Using their national traffic rule, the Japanese person keeps left and the American keeps right and they are again in front of each other. It might be ended in a collision. (see Figure 2.).

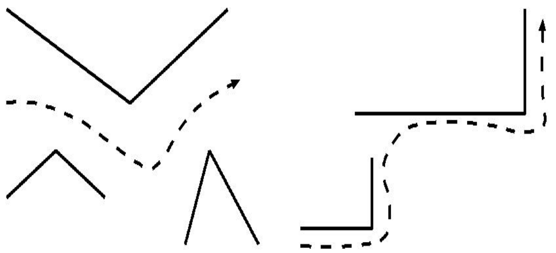


Fig. 1. Basic obstacle avoidance strategies: “As Far As Possible” (left) and “As Close As Possible” (right)

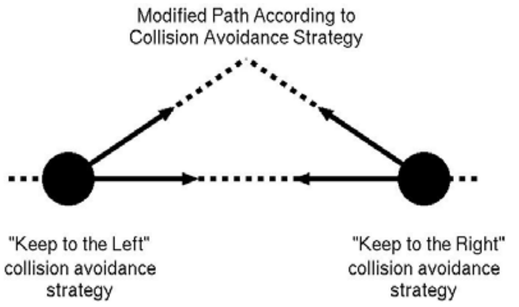

Modified Path According to Collision Avoidance Strategy
"Keep to the Left" collision avoidance strategy
"Keep to the Right" collision avoidance strategy

Fig. 2. Two different obstacle avoidance strategy may result dangerous situation

2.1 Direct Evaluation of Sensor Information

Artificial potential based guiding approach is applied to handle the dynamic and uncertain environment around the robot ([6] and [7]). The robot can detect objects in the scanned area (Figure 3.).

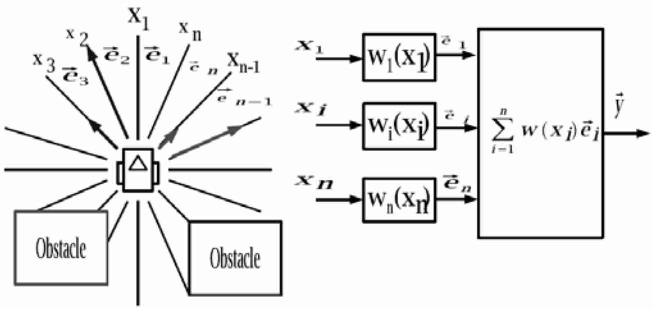


Fig. 3. Scanning area of the robot (left), direct evaluation of sensor information (right)

The scanned area is divided into n scanned lines that are pointed into directions of $\vec{e}_i$ (unique vectors, where $i = 1 \ldots n$). The radial scanned lines structure has an important advantage that spatial density of the scanning is growing with the decreasing distance between the obstacle and the robot. The sensor system provides the distance between the robot and the object on the scanned lines [10]. The main idea of the potential based guiding is to repulse (or attract) the robot from/to the obstacles [8]. The objects and the target generate imaginary forces (y_i , $i=1 \ldots n$) acting on the robot. Summing the effect of these virtual forces, the desired moving direction can be obtained. The virtual vectors must be calculated for each location as quickly as possible to achieve a smooth and reactive guiding. The magnitudes of the repulsive forces are usually inversely proportional to the distance between the obstacles and the robot but they can be described by any non-linear functions.

The virtual force along the scanned line:

$$\vec{y}_i = w_i(x_i)\vec{e}_i \tag{1}$$

where $i = 1 \ldots n$ (n is the number of scanned lines) from the measured distances (x_i) to each scanned lines. The $w_i(x_i)$ is the weight function of the scanned line. The virtual force vectors are pointed into the opposite of the scanned direction (key idea of potential based guiding), and their absolute values depending on the detected distances are: $|\vec{y}_i| = w_i(x_i)$. The overall force is the summation of the virtual forces along the directions of the scanned lines:

$$\vec{y} = \sum_{i=1}^{n} w_i(x_i)\vec{e}_i \tag{2}$$

In many cases this kind of evaluation is not effective. For example let the weight function on each scanned line the same. Applying (2) to symmetrically located obstacles, will result attractive and repulsive force, and the sum results zero vector (see Figure 4.). The attractive force represents the goal reaching behavior, while the repulsive force represents the obstacle avoidance behavior. Choosing one of the $\vec{y}_i$, what is perpendicular to the attractive and repulsive force, in the evaluation would lead to escape from the local minimum.

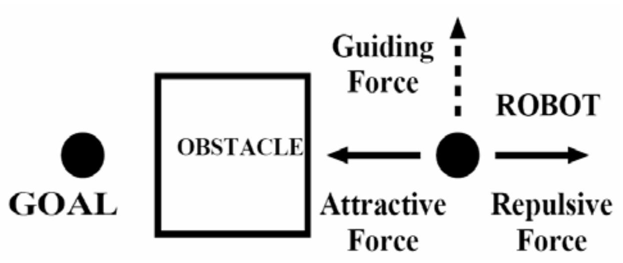


Fig. 4. Local minimum point of the potential based guiding

2.2 Indirect Evaluation of Sensor Information

To avoid the local minimum problem (Figure 4.) an extension of the above mentioned method is introduced. All sensor information is propagated to all outputs (Figure 5.). Weight function is introduced between scanned inputs *i* and the output nodes *j* (*j=1… m*):

$$\vec{y} = \sum_{i=1}^{n} w_{j,i}(x_i)\vec{e}_i \tag{3}$$

The summarized vector output is calculated as in (2), but with extended weight functions as in (3):

$$\vec{y} = \sum_{j=1}^{m}\sum_{i=1}^{n} w_{j,i}(x_i)\vec{e}_i \tag{4}$$

2.3 Printing Area Fuzzy-Neural Approximation

The weight functions are approximated by fuzzy sets. The fuzzy approximation gives piece-wise linear approximation in case of triangular antecedent fuzzy set. The number of antecedent fuzzy sets are denoted with *k*, where *k=1… l*. Fuzzy approximation of direct sensor evaluation is shown first. The weight function of direct evaluation of sensor input [9]:

$$w_i(x_i) = \sum_{k=1}^{l} \mu_{A_{i,k}}(x_i) b_{i,k} \tag{5}$$

The $\mu_{A_{i,k}}(x_i)$ is the membership values of the sensor value, x_i case of antecedent set *k*, and direction of scanned line *i*. The $b_{i,k}$ is the consequent set for antecedent set *k*, and direction of scanned line *i*. In this model the consequent set is only one value set. The virtual vector along the scanned line *i* is generated by:

$$\vec{y}_i = \sum_{k=1}^{l} \mu_{A_{i,k}}(x_i) b_{i,k} \vec{e}_i \tag{6}$$

Where $\vec{e}_i$ is a unique vector pointed into the direction of scanned line as in (2). Summarized vector output (2) approximated by fuzzy sets:

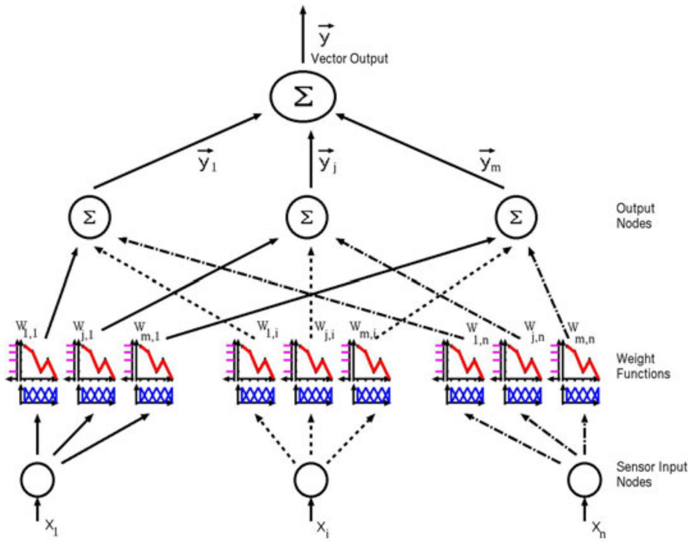


Fig. 5. Fuzzy approximation of indirect evaluation of sensor information

Summarized vector output of the fuzzy-neural network:

$$\vec{y}_i = \sum_{j}^{m} \sum_{i=1}^{n} \sum_{k=1}^{l} \mu_{A_{i,k}}(x_i) b_{j,i,k} \vec{e}_i \tag{7}$$

Figure 5 illustrates the applied fuzzy neural network architecture. Each sensor data ($X_i, i=1...n$) is distributed to each sensor node ($\vec{y}_i$) via the weight function, $W_{j,i}(X_i)$. Weight functions are piece-linear approximated by fuzzy sets. The input fuzzy set are is Ruspini partitions in our case. The consequent fuzzy sets are one valued fuzzy sets. This simple architecture enables fast computation, and simple implementation algorithm.

3 Evaluation and Learning of Pedestrian Behaviors

3.1 Evaluation and Learning Framework

This section illustrates the learning capability of the obstacle avoidance behavior of mobile robot. The learning capability enables by the fuzzy-neural network which is applied for approximation of direct and indirect sensor evaluation. Figure 6 shows the actual configuration of learning. The picture of the human walking is taken by the DIND and sent to the Human Localization Module. The module calculates the human position and sends to the Learning module. The result of the leaning is a potential function, what is given to the robot control module.

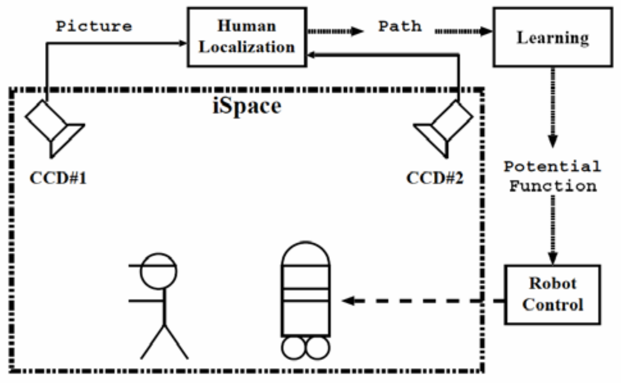


Fig. 6. Learning and evaluation framework in iSpace

3.2 Learning Method of Fuzzy Neural Network

Learning method is introduced for indirect sensor evaluated control (Section 2.1). The human walking path (*p[t]*) calculated by Human Localization module. The walking path is a discrete series of position, along time series *t:={t=t(k)|k=1... z}*. The path is scaled to the obstacle avoidance control:

$$\vec{d}[t] = |\vec{y}[t]| \frac{\vec{p}[t]}{|\vec{p}[t]|} , \tag{8}$$

where $\vec{\bullet}[t]$ denotes vector value at *t=t(k)* time instance. $|\vec{y}[t]|$ is the absolute value of obstacle avoidance initial rule base. The training algorithm does not tune all sets, but the absolute value of the consequent vectors, namely values $b_{j,i,k}[t]$. The *t-th* training pattern contains input values $x_i[t]$ and the desired output direction $\vec{d}[k]$. The error criteria is the instantaneous error between the reference vector and the robot (Figure 7.). Consequently, the tuned consequent sets:

$$b_{j,i,k}[t+1] = b_{j,i,k}[t] + p\mu_{A_{i,k}}(x_i[t])\,|\vec{\varepsilon}[t]|\cos(\vartheta_i[t]) \tag{9}$$

where $\vartheta_i[i]$ is the angle of the error vector $\vec{\varepsilon}[t]$ and the unique vector $\vec{e}_i$.

Figure 7 shows the obstacle avoidance behavior learning method. The error defines as the difference between, the reference moving direction (Reference vector) (walking habit from the observed path) and the Robot's moving direction (Robot vector). This error vector is evaluated back to the direction of the sensors, and tunes the weight constants, $b_{j,i,k}[t]$.

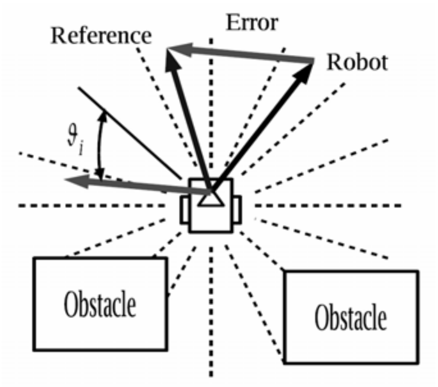


Fig. 7. Evaluation of error vector, difference between the reference vector and the robot vector

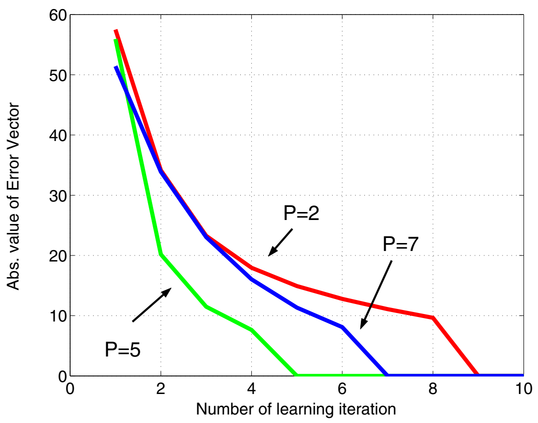


Fig. 8. Speed of Learning: fall of error vector absolute value at different learning parameter (P)

Figure 8 shows the convergence of the training procedure with different values of learning parameters, *P*. When *P=2* almost 10 learning iterations is necessary with the same training data for small error. Increasing learning parameter, *P* may not means faster learning. In this training session, *P=5* gives faster learning, than *P=7*. The learning parameter should be tuned for each training session, as a conclusion of training process.

4 Examples

Tactical Level Control of Mobile Agent is considered in this section. The control framework for tactical control is shown in Figure 9. The output of this layer is Moving Vector which points toward the moving direction, and its absolute value represents the desired instantaneous traveling speed.

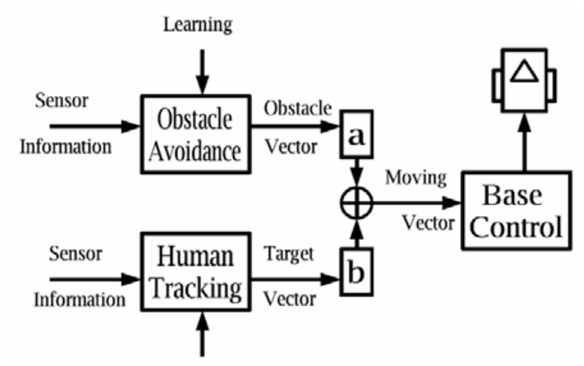


Fig. 9. Tactical Level Control of Mobile Agent

The Moving Vector ($\vec{M}$) is weighted summary of the Obstacle Vector ($\vec{y}$) and the Target Vector ($\vec{T}$):

$$\vec{M}(t) = a\vec{y}(t) + b\vec{T}(t) \tag{10}$$

To approach the target and avoid objects behavior can be tuned by the weight parameter *a* and *b*. If *b* is positive, then the mobile agent approaches the target even there is no obstacles $\vec{y} = 0$. If *b* is negative, than the mobile agent is pushed by the target.

Figure 10 shows a basic example of obstacle avoidance. The robot moves from start position to goal position. The robot can not move directly form start to goal position because of corner. Figure 10(a) shows the resulted path according to (17). The resulted path and the resulted behavior can be changed by parameter *a* and *b*.

Figure 11 shows three cases of trained obstacle avoidance behavior. The basic obstacle avoidance behaviors of manual control were: 1) keep on left side. 2) keep on right side. 3) get as far from the objects as necessary. Figure 11 shows the obstacle avoidance behavior of the three trained mobile agent among the new set of objects.

We concluded that the robot is able to pick up the main human obstacle avoidance behaviors. The next demonstration illustrates the limitation of the presented method to describe the obstacle avoidance behavior. The thick lines represent wall-type objects.

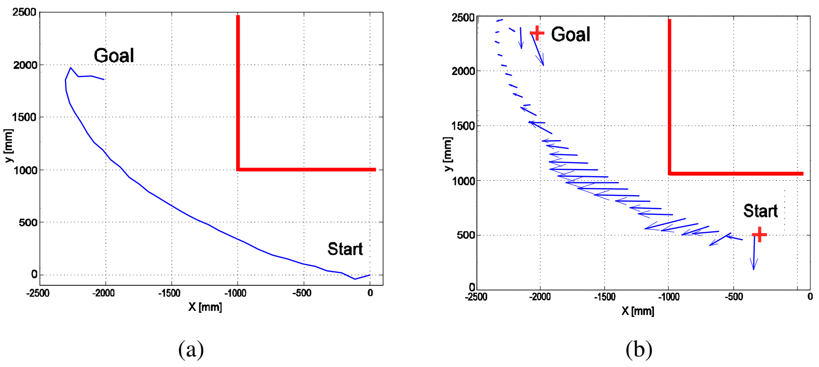


Fig. 10. Path of the Mobile Robot (a) and the Obstacle Vectors along the Path (b)

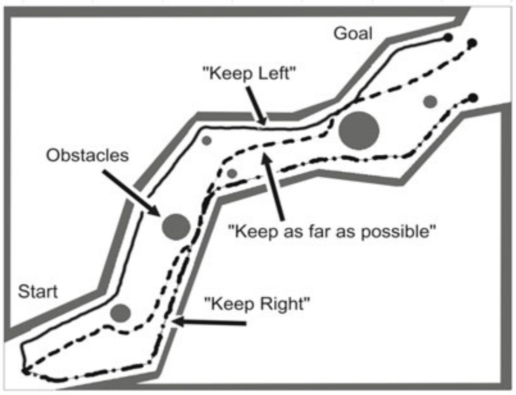


Fig. 11. Path of the Mobile Robot (left) and the Obstacle Vectors along the Path (right)

5 Conclusion

The aim of this paper is to investigate a control framework for mobile robots, operating shared environment with humans. The principle of control framework is derived from pedestrian behavior model. The obstacle avoidance behavior is a characteristic feature of the proposed framework. Virtual potential based obstacle avoidance method is applied to describe the obstacle avoidance behavior. The virtual potential method is approximated by fuzzy-neural network. The learning capability of fuzzy-neural network, and learning methods is also presented. The learning methods, and the learning configuration in the iSpace will be revised as a future work.

References

1. Lee, J.H., Hashimoto, H.: Intelligent Space -concept and contents. Advanced Robotics, Vol.16, No.3 (2002) 265-280.
2. Akiyama, T., Lee, J.H., Hashimoto, H.: Evaluation of CCD Camera Arrangement for Positioning System in Intelligent Space. International Symposium on Artificial Life and Robotics, (2001) 310-315.
3. Kazuyuki, M, Lee, J.H., Hashimoto, H.: Human Centered Robotics in Intelligent Space. IEEE International Conference on Robotics \& Automation (ICRA'02), (2002) 2010-2015.
4. Kazuyuki M., Lee, J.H., Hashimoto, H.: Physical Agent for Human Following in Intelligent Sensor Network. IEEE/RSJ International Conference on Intelligent Robotics and Systems (IROS'02), (2002) 1234-1239.
5. Lee, J.H., Ando, N., Yakushi, T., Nakajima, K., Kagoshima, T., Hashimoto, H.: Adaptive Guidance for Mobile Robots in Intelligent Infrastructure. IEEE/RSJ International Conference on Intelligent Robots and Systems, (2001).
6. Dozier, G., Homaifar, A., Bryson, S., Moore, L.: Artificial Potential Field based Robot Navigation Dynamic Constrained Optimization, and Simple Genetic Hill-Climbing. in *Proc. ICEC'98*, (1998) 189–194.
7. Koren, Y., Borenstein, J.: Potential Field Methods and Their Inherent Limitations for Mobile Robot Navigation. Proc. of the IEEE Int. Conf. on Robotics and Automation, (1991) 1398-1404.
8. Bronstein, J., Koren, Y.: Real-time Obstacle Avoidance for Fast Mobile Robots. IEEE Trans. on Systems Man and Cybernetics vol. 19, no. 5, (1989) 1179-1187.
9. Mizik, S., Baranyi, P., Korondi, P., Sugiyama, M.: Virtual Training of Vector Function based Guiding Styles" Transactions on Automatic Control and Computer Science, vol. 46(60) No.1 (2001) 81-86.
10. Nagy, I., Fung, W.K., Baranyi, P.: Neuro-Fuzzy based Vector Field Model: An Unified Representation for Mobile Robot Guiding Styles. IEEE Int. Conf. System Man and Cybernetics (IEEE SMC'2000), Nashville, Tennessee, USA, (2000) 3538-3543.
11. Hoogendoorn,S.P., Bovy, P.H.L.: Pedestrian Route-Choice and Activity Scheduling Theory and Models. Transportation Research Part B-38, Pergamon, Elsevier Science Ltd. (2004) 169-190.
12. Hoogendoorn, S., Bovy, P.H.L.: Simulation of Pedestrian Flows by Optimal Control and Differential Games. Optimal Control Applications and Methods, (2003) 153-172.

Evolvable Viral Agent Modeling and Exploration

Jingbo Hao, Jianping Yin, and Boyun Zhang

School of Computer Science, National University of Defense Technology,
Changsha 410073, China
{hjb, jpyin}@nudt.edu.cn, hnjxzby@yahoo.com.cn

Abstract. A computer virus is a program that can generate possibly evolved copies of itself when it runs on a computer utilizing the machine's resources, and by some means each copy may be propagated to another computer in which the copy will have a chance to get executed. And we call a virus instance as a viral agent since it is autonomous during its execution by choosing what action to perform in the computer without a user's intervention. In the paper we develop a computational model of viral agents based on the persistent Turing machine (PTM) model which is a canonical model for sequential interaction. The model reveals the most essential infection property of computer viruses well and overcomes the inherent deficiency of Turing machine (TM) virus models in expressing interaction. It is conceivable that viral agents have much potential to evolve in various environments according to the model. Therefore we also discuss the evolution of viral agents with two existing relevant works.

Keywords: Viral Agent, Persistent Turing Machine, Evolution.

1 Introduction

A computer virus is a program that can generate possibly evolved copies of itself when it runs on a computer utilizing the machine's resources, and by some means each copy may be propagated to another computer in which the copy will have a chance to get executed. And we call a virus instance as a viral agent since it is autonomous during its execution by choosing what action to perform in the computer without a user's intervention. Computer viruses are always a serious threat against information security. Although various countermeasures have hitherto been adopted, computer viruses can not be prevented radically.

To understand computer viruses in essence, a proper computational models of viral agents is certainly necessary. There have been a few such models [1–3] which leave out an important property of viral agents – interaction. This is due to the inherent limitations of the classical computational models, such as the most typical Turing machines (TMs). A TM can only model a function-based transformation of an input to an output. When the area of computation is extended from algorithms to processes, TMs are no longer appropriate to capture all the features of computation including interaction. A computing agent has the interaction property if it has input and output actions that interact with an external environment not under its control [4]. To model interactive computation, Wegner [4] proposed the notion of interaction machines (IMs) which extend TMs with input and output actions that interact dynamically with the external environment. In terms of concurrency of interaction, IMs can be divided into sequential interaction machines (SIMs) and multiple-stream interaction machines (MIMs). Persistent Turing machines (PTMs) are a canonical model for sequential interaction and are equivalent to SIMs in expressiveness [5]. The concepts of PTMs have been well defined in [6] and we can use PTMs to model sequentially interactive agents. In general a viral agent interacts with its environment sequentially because no viral agent works like an engaged web server, and so PTMs are quite suitable for viral agent modeling.

In the paper we develop a computational model of viral agents based on PTMs which are a canonical model for sequential interaction. The model reveals the most essential infection property of computer viruses well and overcomes the inherent deficiency of TM virus models in expressing interaction. It is conceivable that viral agents have much potential to evolve in various environments according to the model. Therefore we also discuss the evolution of viral agents with two existing relevant works. The rest of the paper is organized as follows. Firstly in section 2 we develop a computational model of viral agents based on PTMs. Then in section 3 we explore the evolution of viral agents. Finally we draw the conclusion.

2 Modeling Viral Agents with PTMs

2.1 Modeling Viral Agents with PTMs

A PTM is a nondeterministic 3-tape TM (N3TM) with a read-only input tape, a read/write work tape, and a write-only output tape [6]. Upon receiving an input token from its environment on its input tape, a PTM computes for a while and then outputs the result to the environment on its output tape, and this process is repeated forever. A PTM performs persistent computations in the sense that the work tape contents are maintained from one computation step to the next, where each PTM computation step represents an N3TM computation.

Definition 2.1. An N3TM is a quadruple $<K,\Sigma,\delta,s_0>$, where:

- K is a finite set of states.
- Σ is a finite alphabet containing the blank symbol #, but not containing L (left) and R (right).
- $\delta \subseteq K\times\Sigma\times\Sigma\times\Sigma\times(K\cup\{h\})\times(\Sigma\cup\{L,R\})\times(\Sigma\cup\{L,R\})\times(\Sigma\cup\{L,R\})$ is the transition relation.
- $s_0 \in K$ is the initial state.
- $h \notin K$ is the halting state.

Definition 2.2. A PTM is N3TM having a read-only input tape, a read/write work tape, and a write-only output tape.

Definition 2.3. Let M be a PTM having alphabet Σ, and let w_i, w, w' and w_o be words over Σ. We say that $w \xrightarrow[M]{w_i / w_o} w'$ (yields in one macrostep) if M, when started in its initial control state with its heads at the beginning of its input, work, and output tapes containing w_i, w, and Є respectively, has a halting computation that produces w_i, w' and w_o as the respective contents of its input, work, and output tapes. Should M's computation diverge, we write $w \frac{w_i / \mu}{M} > s_{div}$, where $s_{div}, \mu \notin \Sigma$ and s_{div} is a special "divergence state" such that $s_{div} \frac{w_i / \mu}{M} > s_{div}$ for all inputs w_i; and μ is a special output symbol signifying divergence.

Definition 2.4. Let M be a PTM with alphabet Σ, then reach(M), the reachable states of M, is defined as:

$$\{\varepsilon\} \cup \{w \in \Sigma^* \cup \{s_{div}\} \mid \exists k \geq 1, \exists w^1{}_i, ..., w^k{}_i \in \Sigma^*,$$
$$\exists w^1{}_o, ..., w^k{}_o \in \Sigma^* \cup \{\mu\}, \exists w^1, ..., w^k \in \Sigma^* \cup \{s_{div}\}:$$
$$\varepsilon \frac{w^1{}_i / w^1{}_o}{M} > w^1, w^1 \frac{w^2{}_i / w^2{}_o}{M} > w^2, ..., w^{k-1} \frac{w^k{}_i / w^k{}_o}{M} > w^k \wedge w = w^k\}.$$

Definition 2.5. Let U be a PTM with alphabet Σ_U, let M be a PTM with alphabet Σ_M, let w_i, w_o, w and w' be strings over Σ_M, and let $\eta : M, \Sigma^*{}_M \to \Sigma^*{}_U$ be a one-to-one encoding function for the transition relation and alphabet of M. Then, U is a universal PTM simulating M if:

- U has an initial macrostep $\varepsilon \frac{<\eta(M), \eta(w)> / \varepsilon}{U} >< \eta(M), \eta(w) >$.
- If M has a halting computation $w \frac{w_i / w_o}{M} > w'$, the U has a halting computation $< \eta(M), \eta(w) > \frac{\eta(w_i) / \eta(w_o)}{U} >< \eta(M), \eta(w') >$.
- If M diverges, written $w \frac{w_i / \mu}{M} > s_{div}$, then U diverges, written $< \eta(M), \eta(w) > \frac{\eta(w_i) / \mu}{U} > s_{div}$.
- If $s_{div} \frac{w_i / \mu}{M} > s_{div}$, then $s_{div} \frac{\eta(w_i) / \mu}{U} > s_{div}$.

Theorem 2.1. There is a PTM U that is a universal PTM. (Refer to [6] for the proof)

2.2 Viral Agent Modeling

Modern general-purpose computers are implemented in accordance with the idea of universal TMs. However, as the theoretical basis of computers universal TMs are unable to describe a continuous computational process of a computer during which unpredictable interaction events sequentially happen. It is conceivable that universal PTMs can be used to make up the deficiency since a universal PTM can simulate a computational process of an arbitrary PTM which can represent a program properly. Upon that we model viral agents with PTMs based on our previous work in [7].

Definition 2.6. For a given universal PTM U, the set of all its programs $TP = \{\eta(M) \mid M \text{ is a PTM}\}$.

Definition 2.7. For a given universal PTM U, V is a program set iff $V \subseteq TP \text{ and } V \neq \phi$.

Definition 2.8. For a given universal PTM U, the set of all the program sets $TS = \{V \mid V \subseteq TP \text{ and } V \neq \phi\}$.

Definition 2.9. For a given universal PTM U, for $V \in TS$ and $v = \eta(M) \in V$, $v \overset{U}{\Rightarrow} V$ iff $\exists w \exists w' \exists w_i \exists v' : w, w' \in reach(M), w_i \in \Sigma^*_M, v' \in V$:
$< v, \eta(w) > \frac{\eta(w_i)/v'}{U} >< v, \eta(w') >$.

Definition 2.10. With respect to all kinds of universal PTMs, the whole viral set $WS = \{(U,V) \mid U \text{ is a universal PTM}, V \in TS \text{ for } U, \text{and } \forall v \in V, v \overset{U}{\Rightarrow} V\}$.

Definition 2.11. V is a viral set with respect to a universal PTM U iff $(U,V) \in WS$.

Definition 2.12. v is a viral agent with respect to a universal PTM U iff $v \in V \text{ and } (U,V) \in WS$.

These definitions recursively model a viral agent as a PTM which can generate possibly evolved isogenous PTMs through interaction with the environment. In this way the model adequately exhibits the infection essence of a viral agent. And it is conceivable that viral agents have much potential to evolve in various environments according to the model. Several relevant definitions and theorems are given below.

Definition 2.13. v directly evolves into v' for a universal PTM U iff v and v' are both viral agents with respect to U and $v \overset{U}{\Rightarrow} \{v, v'\}$.

Definition 2.14. v' is directly evolved from v for a universal PTM U iff v directly evolves into v' for U.

Definition 2.15. v' is a evolution of v for a universal PTM U iff $(U,V) \in WS$ and $\exists V' \subseteq V$ such that $\forall v_k \in V', v_k \text{ directly evolves into } v_{k+1}, \text{and } \exists l, m \in N$ such that $l < m, v_l = v \text{ and } v_m = v'$.

Definition 2.16. A smallest viral set with respect to a given universal PTM is a viral set of which no proper subset is a viral set.

Theorem 2.2. For an arbitrary universal PTM there exists a smallest viral set which is a singleton set.

Proof. For an arbitrary universal PTM U, we can construct a PTM M so that:

$$\forall w_i \in \Sigma^*_M, \forall w \in reach(M), \exists w' \in reach(M), let\ v = \eta(M):$$
$$< v, \eta(w) > \frac{\eta(w_i)/v}{U} >< v, \eta(w') >.$$

Therefore $v \overset{U}{\Rightarrow} \{v\}, (U, \{v\}) \in WS$, and so {v} is just a singleton viral set for U.

Theorem 2.3. For any finite number n and an arbitrary universal PTM U, there exists a smallest viral set with n elements.

Proof. For any finite number n and an arbitrary universal PTM U, we can construct n PTMs $M_1, \ldots, M_n$, so that:

$$\forall w_{i1} \in \Sigma^*_{M_1}, \ldots, \forall w_{in} \in \Sigma^*_{M_n}, \forall w_1 \in reach(M_1), \ldots, \forall w_n \in reach(M_n),$$
$$\exists w_1' \in reach(M_1), \ldots, \exists w_n' \in reach(M_n):$$
$$< \eta(M_1), \eta(w_1) > \frac{w_{i1} / \eta(M_2)}{U} >< \eta(M_1), \eta(w_1') >, \ldots,$$
$$< \eta(M_{n-1}), \eta(w_{n-1}) > \frac{\eta(w_{in-1}) / \eta(M_n)}{U} >< \eta(M_{n-1}), \eta(w_{n-1}') >,$$
$$< \eta(M_n), \eta(w_n) > \frac{\eta(w_{in}) / \eta(M_1)}{U} >< \eta(M_n), \eta(w_n') > .$$

Therefore $\forall v \in \{\eta(M_1), \ldots, \eta(M_n)\} = V, v \overset{U}{\Rightarrow} V, (U, V) \in WS$ and no proper subset of V can be a viral set, and so V is a smallest viral set with n elements.

Theorem 2.4. Any union of viral sets is also a viral set, i.e., if $(U, V_1) \in WS$ and $(U, V_2) \in WS$ then $(U, V_1 \cup V_2) \in WS$.

Proof. $\forall v \in V_1 \cup V_2$,

$$if\ v \in V_1, since\ (U, V_1) \in WS, then\ v \overset{U}{\Rightarrow} V_1, so\ v \overset{U}{\Rightarrow} V_1 \cup V_2;$$
$$if\ v \in V_2, since\ (U, V_2) \in WS, then\ v \overset{U}{\Rightarrow} V_2, so\ v \overset{U}{\Rightarrow} V_1 \cup V_2.$$
$$Therefore\ \forall v \in V_1 \cup V_2, v \overset{U}{\Rightarrow} V_1 \cup V_2, and\ so\ (U, V_1 \cup V_2) \in WS.$$

These theorems may not be much comprehensive, but may be helpful for further understanding of the evolution of viral agents.

3 Evolution of Viral Agents

Generally speaking, evolution is a gradual process in which something changes into a different and usually more complex or better form. It has been argued that whether computer viruses are a form of artificial life to which evolution is a prerequisite [8, 9]. However, no computer virus has shown to be really evolvable since evolution implies changes in functionality while polymorphic or metamorphic viruses represent only cases of random changes in structure but not functionality.

As for living organisms, evolution can be divided into two types: accidental and pre-programmed [9]. Accidental evolution might be the result of a stray cosmic ray striking some organism and altering its genetic structure. Such alternations could be harmful, neutral or even beneficial. In a beneficial situation, the mutated organism would reproduce successfully, and possibly replace the original in a large number of generations. Pre-programmed evolution infers that some technique of modifying the genetic structure of an organism from generation to generation is built into its very coding. For example, human genetic information is broken up into chromosomes, and each one of which has an equal chance to get selected at conception. In addition, a crossover phenomenon occasionally occurs in which two adjacent chromosomes break apart and combine with each other to form two different new chromosomes. Computer viruses clearly can evolve and use evolution to overcome challenges to their survival similarly [9]. In the following two different efforts towards exploring the evolution of viral agents are presented.

3.1 Computer Virus Project 2.0

Nechvatal's Computer Virus Project 2.0 [10] has brought his earlier computer virus project into the realm of artificial life, that is, into a synthetic system which exhibits behaviors characteristic of natural living systems. With Computer Virus Project 2.0, a virus is modeled to be autonomous agents living in an image, thereby introducing elements of artificial life. The project simulates a population of active viral agents functioning as an analogy of a viral biological system.

In Computer Virus Project 2.0, the world is modeled as an image via a set of pixels. Every pixel's color is defined by RGB vectors which represent the red, green and blue components of every pixel's color. The image world has no edges. Every square on the edge of the image is adjacent to another on the opposite edge. The behavior of a viral agent is modeled as a generated looping activity in which the agent will pick up information from its environment, decide on a course of action, and carry it out. A viral agent will perceive the pixel which the agent is on and the eight adjacent ones, and it can get information on its color and on the possible presence of other agents. In order to decide on a course of action, each viral agent is programmed with a set of randomized instructions of different kinds. As the viral agent executes, it moves to one of the adjacent squares and changes the current pixel. It can even reproduce itself and its genome-program changes with the mutation operator. In addition to these changes, every cycle produces a change in the energy level of the viral agent. The agent will lose a set amount of energy with every run, and when it runs out of energy it dies. In order to survive, a viral agent needs to pick up energy, which it can only do by degrading the image. The more it changes the color of a pixel, the more energy it acquires. A viral attack will generally develop as follows (see Fig. 1):

- A world is created from an image.
- A population of viral agents is generated randomly and introduced into the image. Each agent takes on its very own behavior as its program is defined randomly.
- Once the viral agents have been placed in an image, the attack can start. It will consist of a series of action cycles that will only come to an end when there is no viral agent left alive (or after a given time limit).

The instruction 'divide' will reproduce slightly mutated replicas of a viral agent. The creation of these replicas will immediately trigger a considerable loss of energy in the agent. This means that a viral agent that is not capable of drawing energy from its environment will not survive much longer. On the other hand, the fact that these replicas are not identical to the original offers the possibility of examining new types of behavior. When an adapted individual appears, it can remain in the image for some time. If it executes the 'divide' instruction, its descendants will most probably be equally adapted. The number of these agents will generally increase exponentially, and thereby create a large population of active viral agents.

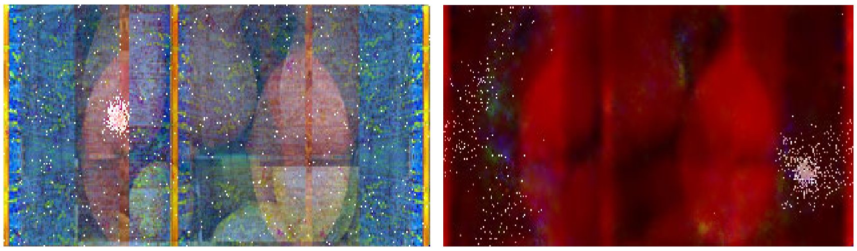


Fig. 1. Viral agent dynamics in an image (left at the start point and right at the end point) [10]

3.2 Genetic Programming in Computer Viruses

ValleZ in 29A which is the most famous organization of computer virus writers in the world proposed to use genetic programming in computer viruses [11]. To use genetic programming, a virus program must be composed of a set of high level operational elements, such as a ChangeDir element that changes the current directory, an Infector element that infects files, a Worm element or a Payload element, etc. These elements of code must work by themselves and in any direction of memory, and could be named as blocks. Two important genetic programming mechanisms can be similarly applied into virus programming:

- Mutation. If we have a set of operational blocks we generate randomly a set of generation 0 programs and we take one:

gen0->ChangeDir-ChangeDir-Infector-Payload-Infector-Payload-ChangeDir-Payload

We could program an Infector block with ability of mutating current code. It could change a block by another or change the order of blocks. For example, this Infector block infects a file and changes the order of two blocks:

gen1->ChangeDir-Infector-ChangeDir-Payload-Infector-Payload-ChangeDir-Payload

The gen1 viral agent can infect two directories in one execution. Now it infects a file, but this time it mutates by changing a block by another:

gen2->ChangeDir-Infector-ChangeDir-Antidebug-Infector-Payload-ChangeDir-Payload

The gen1 viral agent may also infect a file with a bad combination of blocks:

gen2->ChangeDir-Payload-ChangeDir-Payload-Infector-Payload-ChangeDir-Payload

This gen2 viral agent will have the less probability of surviving. Mutations must not occur always because if we have a good individual and when it reproduces it always mutates, it will lose its advantages rapidly.

- Crossover. For crossover, we need two viral progenitors. Crossover may occur when a viral agent tries to infect an already infected file. If a viral agent detects it's trying to infect an infected file, it will not infect it. However, it could "learn" a block (randomly) from the infected file and keep it to use it in next infection. In this manner, the next generation of the viral agent will be a descendant from two viral progenitors.

4 Conclusion

In the paper we develop a computational model of viral agents based on PTMs which are a canonical model for sequential interaction. The model reveals the most essential infection property of computer viruses well and overcomes the inherent deficiency of TM virus models in expressing interaction. It is conceivable that viral agents have much potential to evolve in various environments according to the model. Therefore we also discuss the evolution of viral agents with two existing relevant works. But due to the limitation of time, we still have many problems left to solve which will be done in our subsequent work.

Acknowledgments. This work has been sponsored by the National Natural Science Foundation of China under Grant No. 60373023.

References

1. Cohen, F.: Computational Aspects of Computer Viruses. Computers & Security, Vol. 8, No. 4 (1989) 325–344
2. Adleman L.M.: An Abstract Theory of Computer Viruses. In: Goldwasser, S. (ed.): Advances in Cryptology – CRYPTO'88. Lecture Notes in Computer Science, Vol. 403. Springer-Verlag, Berlin Heidelberg (1990) 354–374
3. Leitold, F.: Mathematical Model of Computer Viruses. In: EICAR 2000 Best Paper Proceedings, Brussels (2000) 194–217
4. Wegner, P.: Towards Empirical Computer Science. The Monist, Vol. 82, No. 1 (1999) 58–108
5. Wegner, P., Goldin, D.: Mathematical Models of Interactive Computing. Technical Report, CS-99-13, Brown University, Providence (1999)
6. Goldin, D., Smolka, S., Attie, P., Sonderegger, E.: Turing Machines, Transition Systems, and Interaction. Information and Computation, Vol. 194, No. 2 (2004) 101–128
7. Hao, J., Yin, J., Zhang, B.: Proof of the Virus Decidability Theorem Based on a Universal Turing Machine Model (in Chinese). In: Proceedings of the National Annual Conference on Theoretical Computer Science, Qinhuangdao (2005) 243–245
8. Spafford, E.H.: Computer Viruses – A Form of Artificial Life. Technical Report, CSD-TR-985, Purdue University, West Lafayette (1991)
9. Ludwig, M.A.: Computer Viruses, Artificial Life and Evolution. American Eagle Publications, Inc., Tucson (1993)
10. Nechvatal, J., Sikora, S.: Red Ovoid Attack - Computer Virus Project 2.0. (2001) http://www.computerfinearts.com/collection/nechvatal/redattack/
11. ValleZ: Genetic Programming in Virus. (2004) http://vx.netlux.org/29a/29a-7/Articles/29A-7.016

Mobile Robot Control Using Fuzzy-Neural-Network for Learning Human Behavior

TaeSeok Jin[1], YoungDae Son[2], and Hideki Hashimoto[3]

[1] Dept. of Mechatronics Engineering, DongSeo University,
[2] Dept. of Electronics Engineering, DongSeo University,
San 69-1 Churye-2-dong, Sasang-Gu, Busan 617-716, Korea
{jints, ydson}@gdsu.dongseo.ac.kr
[3] Institute of Industrial Science, the University of Tokyo,
4-6-1 Komaba, Meguro-ku, Tokyo 153-8505, Japan
hashimoto@iis.u-tokyo.ac.jp

Abstract. The knowledge of human walking behavior has primary importance for mobile agent in order to operate in the human shared space, with minimal disturb of other humans. This paper introduces such an observation and learning framework, which can acquire the human walking behavior from observation of human walking, using CCD cameras of the Intelligent Space. The proposed behavior learning framework applies Fuzzy-Neural Network(FNN) to approximate observed human behavior, with observation data clustering in order to extract important training data from observation. Preliminary experiment and results are shown to demonstrate the merit of the introduced behavior.

1 Introduction

The iSpace propagates mobile robots in the space, which act in the space in order to change the state of the space. These mobile robots are called mobile agents. Mobile Agents cooperating with each other and with the core of the iSpace to realize intelligent services to inhabitants. Mobile robots become more intelligent through interaction with the iSpace. Moreover, robots can understand the requests (e.g. gestures) from people, so that the robots and the space can support people effectively. The Intelligent Space can physically and mentally support people using robot and VR technologies; thereby providing satisfaction for people. These functions will be an indispensable technology in the coming intelligence consumption society (Fig. 1).

The Intelligent Space proposes an infrastructure for collection of distributed sensory data, and offers adaptive data filtering to extract the necessary information for the mobile agent. All sensing nodes cover a fixed partial area of the space. The mobile agents only acquire the observation data from that sensing node, which is related with the robot instantaneous position. The intelligent space shares its intelligence to the mobile robots, to extend the functionalities both the mobile robot and the intelligent space. The acquired data is projected to an intention-free representation of the mobile agent's local area.

This paper is organized as follows. This section introduces the concept of Intelligent Space. Section 2 introduces the mobile agent and human behavior model, which is used to understand human behavior directly from observation. Section 3 shows experiment for observation of human behavior using the distributed sensory intelligence of the Intelligent Space and control the mobile agent using the acquired behavior.

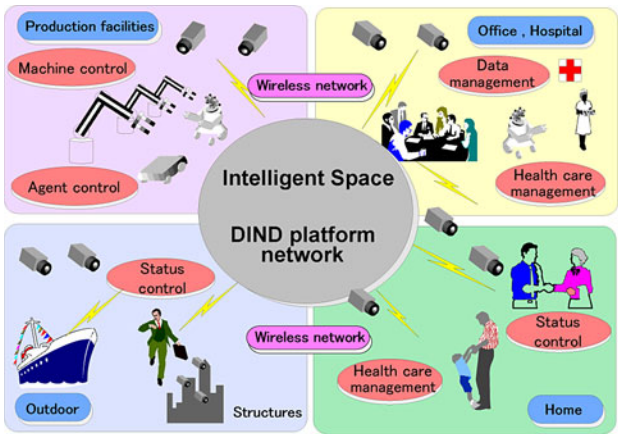


Fig. 1. Vision of intelligent space, as a human support system for more comfortable life

2 Mobile Agent and Human Walking Behavior Model

2.1 Input of the Behavior: Local Space

This paper uses the following definition:

Definition 1 (Mobile Agent): Mobile Agent is a mobile robot integrated in the Intelligent Space. The mobile agent utilizes the intelligence of the intelligent space: receives the information of the state of the space, and sends information, using his own sensor system.

Definition 2 (Behavior): The behavior is a process of nonlinear mapping between instantaneous sensory input, and the output (action).

Definition 3 (Local Space): Local Space is an intention-free representation of the environment surrounded by the observed object, such as human or mobile robot. The collected sensor data is projected into the Local Space and the conditions of the space is represented by features. Occupancy, Mobility and Speed features are used to describe other object's state.

Fig. 2 shows the idea of the Local Space. The sensor data are projected to a polar occupancy space. The occupancy space is divided into sectors. All the important features in the local space are described in the coordinate system of the observed object, i.e. $x^{rcs} - y^{rcs}$. Fig. 2 shows three kind of typical objects that may exist in the space. The first is a wall, which is a large static object, occupies several sectors, second a static object and a moving object.

The sectors of the local space contain the following information:

- **Occupancy:** Proportional with the occupied area of the sector.
- **Relative speed:** The relative speed of the obstacle in the local space.
- **Mobility:** describes the mobility of the given obstacle.

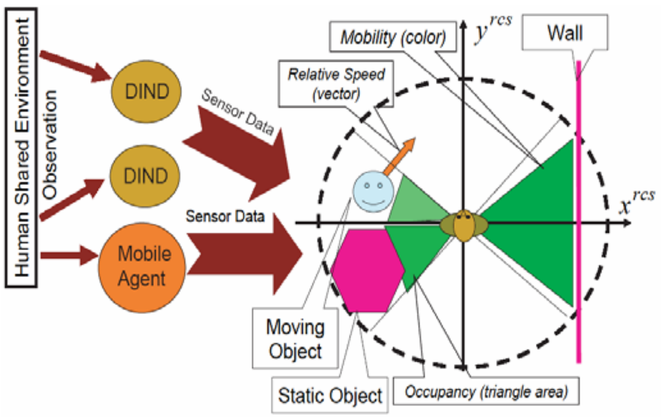


Fig. 2. Local space, the data Structure for sensor data collection. The observation data is collected by many nodes in the Intelligent Space, and classified by the viewpoint of each observed individual.

2.2 Output of the Behavior: MOT Movement Model

Two efforts, (sub-behaviors) are considered in this walking behavior model, obstacle avoidance effort and target tracing effort. The name of MOT movement model is defined by the movement vectors, what are considered in this model: moving, object and target vector. One may notice that the human walking behavior is far more complex than the mentioned two sub-behaviors, but these two behaviors can easily be observed from human walking. The imagination of object vector ($\vec{O}$) is related to the effort of human to adapt itself to the instantaneous state of the surrounding environment. The obstacle vector is a result of complex decision making mechanism (behavior) where all the features of the surrounding environment are considered. This mechanism is approximated with one object vector. The target vector ($\vec{T}$) points to the target area or target point, beyond the perception area. The goal or target area of human walking can be observed in the Intelligent Space. After the explanation of object and target imaginary vectors, we may define the moving vector.

Definition 4 (Moving Vector): Moving Vector is an imaginary object (represented as a speed vector) related to the effort of movement to the target area, while adapting to the surrounding instantaneous features of the perception area (Local Space).

The general equation of MOT movement model is:

$$\vec{M} = {}_{sc}\vec{O} + {}_{sc}\vec{T}\ , \tag{1}$$

where the notation of vectors are shown on Fig. 3. For the actual calculation of these vectors, the following assumption is defined in the movement model:

The following additional vectors and notation are introduced to calculate the vectors (Fig. 3). The following equations introduce the method to calculate vector component from observation of human movement. This way called the indirect way, because the aim is to calculate the components of (1), i.e. obstacle vector ${}_{sc}\vec{O}$, and target vector ${}_{sc}\vec{T}$ from observation in order to learn human walking behavior. The relative target vector (${}_{rel}\vec{T}$) is calculated from the instantaneous position ($\vec{R}$) of the robot or human location (Fig. 3.(left)):

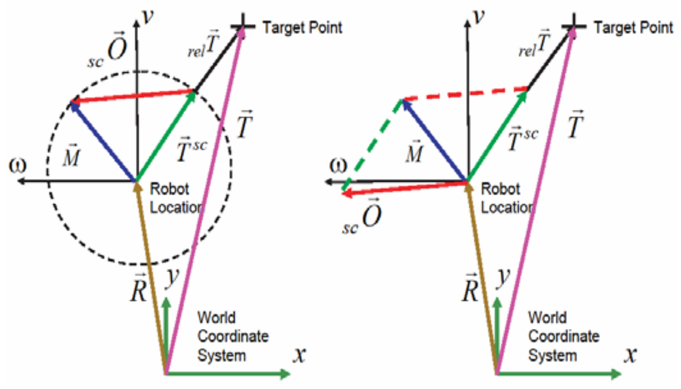


Fig. 3. MOT movement model. The model helps to understand human movement from observation, involving obstacle avoidance and target tracing. The model can be used for indirect way for observation (left) and direct way for control (right).

$$ {}_{rel}\vec{T} = \vec{T} + \vec{R} , \tag{2} $$

where $\vec{T}$ is the location of the target area. The target tracing vector is scaled to the moving vector and the relative target vector. The absolute value of the target tracing vector is scaled to the absolute value of the moving vector. The direction of the target tracing vector is scaled to the relative target vector. Finally, the object vector is calculated as follows:

$$ {}_{sc}\vec{O} = \vec{M} - {}_{sc}\vec{T} \tag{3} $$

The vectors are represented into $v-\omega$ coordinate system of the observed individual.

2.3 Behavior Approximation Framework

The proposed behavior approximation framework is shown on Fig. 4. The framework contains adaptive blocks, and fixed connections. The adaptive blocks approximate the observed human behavior, namely, the Obstacle Avoidance block approximates the obstacle avoidance behavior, which calculates a direction to avoid collision with the local obstacles ${}_{sc}O_{\phi}^{rcs}$; the Target Tracing block approximates target tracking behavior, and calculates a direction to cruise toward the target zone ${}_{sc}T_{\phi}^{rcs}$; the Action Selection block controls the balance between the two behaviors by calculating the absolute values of the obstacle avoidance and target tracing vectors, ${}_{sc}O_{r}^{rcs}$ and ${}_{sc}T_{r}^{rcs}$.

The fix connection part calculates the moving vector $\vec{M}$ from output of the sub-behavior's block. The basic equation of the wired part of the approximation framework, based on (1), is:

$$\begin{bmatrix} M_{\upsilon}^{rcs} \\ M_{\omega}^{rcs} \end{bmatrix} = \begin{bmatrix} {}_{sc}O_r^{rcs} \cos {}_{sc}O_{\phi}^{rcs} \\ {}_{sc}O_r^{rcs} \sin {}_{sc}O_{\phi}^{rcs} \end{bmatrix} + \begin{bmatrix} {}_{sc}T_r^{rcs} \cos {}_{sc}T_{\phi}^{rcs} \\ {}_{sc}T_r^{rcs} \sin {}_{sc}T_{\phi}^{rcs} \end{bmatrix}, \tag{4}$$

where $\bullet_r^{rcs}$ and $\bullet_{\phi}^{rcs}$ denote the r and ϕ coordinate value in the $r-\phi$ coordinate system, what is a polar coordinate system of $\upsilon-\omega$ coordinate system. Equ. (4) is the direct way to calculate moving vector $\vec{M}$ as shown on Fig. 3(right).

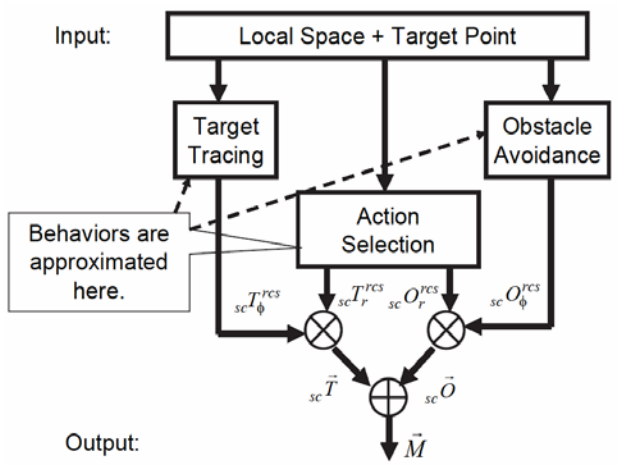


Fig. 4. The behavior approximation framework for human walking behavior. The input of the framework is the local space and the desired target point, and the output is the moving vector $\vec{M}$. Target Tracing, Action Selection and Obstacle Avoidance blocks contain Fuzzy-Neural Networks, which approximate the observed behavior.

2.4 Behavior Approximation with Fuzzy-Neural Network

Fuzzy-Neural Netwoks (FNN) is applied in the behavior approximation framework (Fig. 4.) in order to handle the non-linear mapping of Target Tracking, Obstacle Avoidance and Action Selection. The FNN is class of adaptive network that is functionally equivalent to fuzzy inference systems. Takagi-Sugeno fuzzy inference system (TS-FIS) is an effort to develop a systematic approach to generating fuzzy rules from a given input-output data set [7]. A typical fuzzy rule of the TS-FIS model:

$$\begin{aligned} R = IF(x_1\, is\, A_{i1}) AND \ldots AND(x_j\, is\, A_{ij}) AND \ldots AND(x_n\, is\, A_{in}) \\ THEN(y_i = w_{i0} + w_{i1}x_1 + \cdots + w_{ij}x_j + \cdots + w_{in}x_n) \end{aligned} \tag{5}$$

where R_i denotes the i^{th} fuzzy rule, $(i = 1\ldots r)$, r is the number of fuzzy rules, $\vec{x}$ is the input vector, $\vec{x} = [x_1 \ldots, x_j \ldots, x_n]^T$, $A_{i,j}$ denotes the antecedent fuzzy sets, $(j = 1\ldots n)$, y_i is the output of the i^{th} linear subsystem, and w_{ij} are its parameters, $(l = 0\ldots n)$. The nonlinear system of FNN forms a collection of loosely coupled multiple linear models. The degree of firing of each rule is proportional to the level of contribution of the corresponding linear model to the overall output of the TS-FIS model. For Gaussian-like antecedent fuzzy set, the degree of membership is

$$\vec{\mu}_{ij} = \vec{e}^{-\| x_j - x_{ij}^* \|^2}, \tag{6}$$

where x_j is the j^{th} input, x_{ij}^* denotes the center of A_{ij} membership function, $\alpha = 4/r^2$ and r is positive constant, which defines the spread of the antecedent and the zone of the influence of the i^{th} model (radius of the neighborhood of a data point); too large value of r leads to averaging. The firing level of rules are defined as Cartesian product or conjunction of respective fuzzy sets for this rule, The output of the TS-FIS model is calculated by the weighted averaging of individual rules' contributions,

$$y = \sum_{i=1}^{r} \lambda_i y_i = \sum_{i=1}^{r} \lambda_i x_e^T \pi_i , \tag{7}$$

where $\lambda_i = (\tau_i / \sum_{j=1}^{r} \tau_j)$ is the normalized firing level of the i^{th} value, y_i represents the output of the i_{th} linear model, $\pi_i = [w_{i0}, w_{i1}, \ldots, w_{ij}, \ldots, w_{in}]^T$ is the vector of parameters of the i^{th} linear model, and $x_e = [1 \quad x^T]^T$ is the expanded data vector.

2.5 Training of Fuzzy-Neural Network

The FNN network output is linear in the parameters of the consequent linear models, thus these linear parameters can be identified by linear least-squares-method. This approach leads to the hybrid learning rule [10], which combines steepest descent and the least-squares-method for fast identification of parameters. For fixed antecedent parameters the second subtask, estimation of the parameters of the consequent linear models can be transformed into a least-square-problem [9]. This is accomplished by eliminating the summation operation in (7) and replacing with an equivalent vector expression of y,

$${}_kE = \frac{1}{2}({}_ky^* - {}_ky)^2 , \tag{8}$$

where ${}_ky^*$ is the k^{th} training data. The aim of the delta rule is to minimize ${}_kE$. The partial derivative of ${}_kE$ with respect to ${}_kw_{ij}$ is as follows,

$$\frac{\partial_k E}{\partial_k w_{ik}} = \frac{\partial_k E}{\partial_k y}\frac{\partial_k y}{\partial_k w_{ik}} = -\left({}_k y^* - {}_k \psi^T k^{\theta}\right)_k \lambda_{ik} x_j \tag{9}$$

To update the $(k+1)^{th}$ consequence parameter of the i^{th} rule as,

$${}_{k+1}w_{ij} = {}_k w_{ik} + L\frac{\partial_k E}{\partial_k w_{ij}} = {}_k w_{ik} + L({}_k y^* - \sum_{i=1}^{r} {}_k \lambda_{i\,k} x_e^T {}_k \pi_i)_k \lambda_{i\,k} x_j\,, \tag{10}$$

where $L < 0$ is the learning parameter.

3 Experiment: Learning-Evaluation-Control

A human is walking straight toward the position of the teacher. The teacher avoids collision to change its heading and velocity. The knowledge of this basic behavior is fundamental for mobile agent control in human shared environment. Fig. 5 shows the teacher and human path in world coordinate system. The figure shows the navigation of the teacher who turns right to avoid the collision. The teacher returns its original path, when the observed human leaves the teacher forward zone. The following important points are selected from the teachers' path, what contains characteristic information about the observed behavior. Subtractive clustering has been applied on the observed data [8].

The following situations are happened at the characteristic points,

- **Point 14.** The behavior starts from this point.
- **Point 35.** The teacher stops turning and goes parallel to the avoided human.
- **Point 51.** The teacher goes parallel to the avoided human.
- **Point 59.** The teacher starts to turn back toward to the original target point.
- **Point 75.** The teacher goes straight toward to the target area.

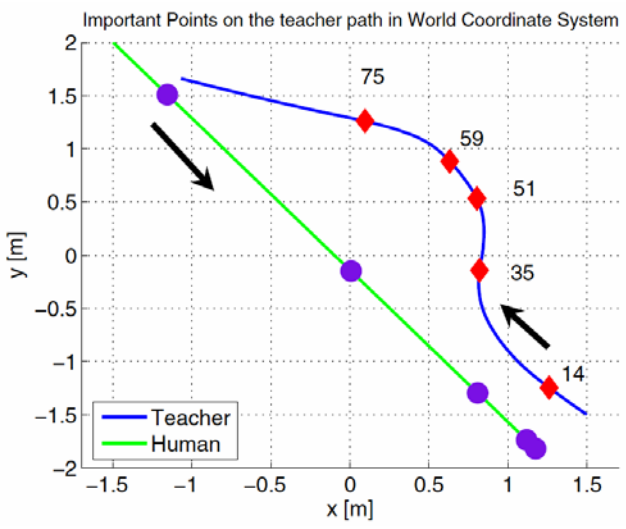


Fig. 5. Teacher and human path in world coordinate system

After the selection of characteristic observation point, the training of the fuzzy neural network takes place. According to the proposed behavior approximation framework (Fig. 4), the Obstacle Avoidance, Target Tracing, and Action Selection blocks are contain a fuzzy neural network to learn the nonlinear mapping between Local Space (Fig. 2) and MOT behavior parameters (Fig. 3(right)). The following Fuzzy Neural Networks are used: The inputs of the fuzzy neural networks are features of the local space, extended with the relative target vector in polar coordinate, $({}_{rel}T_r^{rcs}, {}_{rel}T_\phi^{rcs})$ for target tracing and action selection behavior. The output of the fuzzy neural networks is the direction of the scaled obstacle vector, ${}_{sc}O_\phi^{rcs}$ for obstacle avoidance; the direction of the scaled target vector, ${}_{sc}T_\phi^{rcs}$ for target tracing; and the length of scaled target vector $({}_{rel}T_r^{rcs})$ and scaled obstacle vector $({}_{rel}O_r^{rcs})$ for action selection.

The teacher and the robot path in world coordinate system are shown on Fig. 6. The mobile agent follows the teacher's path quite close, if we neglect the initial heading difference. The mobile agent produces more sharp turn in the process of navigation, than the human.

As regards the experimental results, Fig. 7 presents the path deviation error with respect to the teacher and the robot path in world coordinate system. During the moving time of 3〜9 sec and 9〜12 sec, the path deviation error is at most 0.02m and 0.05m, respectively. The response of the position is reasonably smooth, whereas the orientation continues to be very noisy during the moving time 10sec. This is due to distortions of the images caused by the human's motion and walking speed.

The application of tracking control is effective and the target trajectory of a mobile robot is continuous and smooth when the usual tracking control is applied. The human-walking trajectory that is to be tracked by the robot is generally also stable.

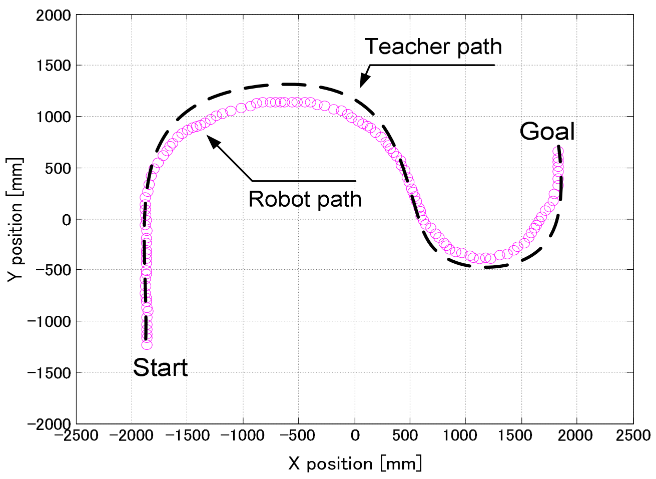


Fig. 6. Teacher and mobile agent path in world coordinate system

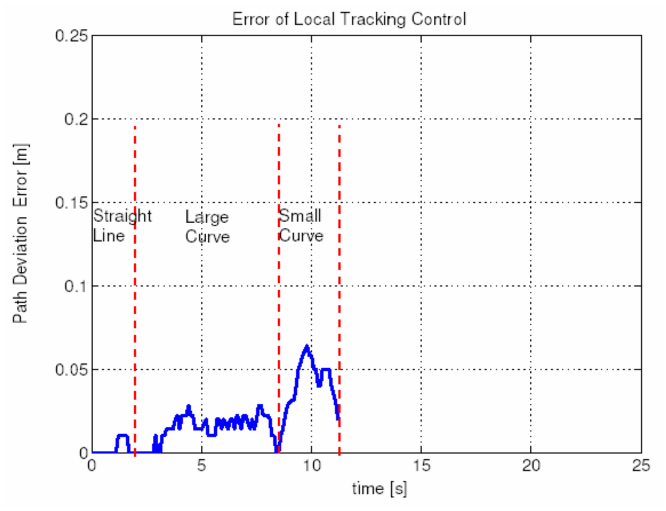


Fig. 7. Error of local tracking control

4 Conclusion

This paper investigates a behavior learning framework to acquire human walking behavior directly from observation of walking humans. The observed walking behavior is decomposed of obstacle avoidance and target tracing using MOT movement model. Obstacle avoidance and target tracing is approximated with Fuzzy-Neural Networks. The whole walking behavior approximation framework consider the target of the observed human and the instantaneous situation around the human as an input, and calculates the appropriate movement of the pedestrian as an output. The introduced behavior approximation framework is used to train the observation of walking human using the technology of the Intelligent Space. The trained behavior approximation framework controls the mobile agent, and performs identical as the observed human in similar situation.

Future studies will involve improving the tracking accuracy for the mobile robot and applying this system to complex environments where many people, mobile robots and obstacles coexist. Moreover, it is necessary to survey the influence of the mobile agent which maintains a flexible distance between the robot and the human, and introduces the knowledge of cognitive science and social science.

References

1. Morioka, K., Lee, J.H., Hashimoto, H.: Human-Following Mobile Robot in a Distributed Intelligent Sensor Network. IEEE Trans. on Industrial Electronics, Vol.51, No.1 (2004) 229-237.
2. Appenzeller, G., Lee, J.H., Hashimoto, H.: Building Topological Maps by Looking at People: An Example of Cooperation between Intelligent Spaces and Robots. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, (1997) 1326-1333.

3. Home Page of Inter Process Communication Package at: http://www.ri.cmu.edu/ projects/project_394.html
4. Home Page of Player/Stage Project at: http://playerstage.sourceforge.net/
5. McFarland, D.: What it means for a robot behavior to be adaptive. In. J.-A. Meyer and S.W. Wilson (Eds.), From animals to animates: Proc. of the First Int. Conf. on Simulation of Adaptive Behavior, Cambridge, MA, MIT Press (1990) 22-28.
6. Kanayama, Y.: Two Dimensional Wheeled Vehicle Kinematics. Proc. of IEEE Conf. on Robotics and Automation, San-Diego, California (1994) 3079-3084.
7. Takagi, T., Sugeno, M.: Fuzzy Identification System and its Application to Modeling and Control. IEEE Trans. on Systems, Man and Cybernetics, 15, (1985) 116-132.
8. Chiu, S.L.: Fuzzy model identification based on cluster estimation. Journal of Int. Fuzzy System, vol. 2, (1994) 267-278.
9. Ljung, L.: System Identification, Theory for the User. Englewood Cliffs, NJ: Prentice-Hall, 1987.
10. Jang, J.-S. R.: ANFIS: Adaptive-Network-based Fuzzy Interference Systems. IEEE Trans. on Systems, Man, and Cybernetics, 23(03): (1993) 665-685.

EFuNN Ensembles Construction Using a Clustering Method and a Coevolutionary Multi-objective Genetic Algorithm

Fernanda L. Minku and Teresa B. Ludermir

Federal University of Pernambuco, Informatics Center,
Recife-PE 50732-970, Brazil, P.O.Box 7851
{flm, tbl}@cin.ufpe.br

Abstract. This paper presents the experiments which where made with the Clustering and Coevolution to Construct Neural Network Ensemble (CONE) approach on two classification problems and two time series prediction problems. This approach was used to create a particular type of Evolving Fuzzy Neural Network (EFuNN) ensemble and optimize its parameters using a Coevolutionary Multi-objective Genetic Algorithm. The results of the experiments reinforce some previous results which have shown that the approach is able to generate EFuNN ensembles with accuracy either better or equal to the accuracy of single EFuNNs generated without using coevolution. Besides, the execution time of CONE to generate EFuNN ensembles is lower than the execution time to produce single EFuNNs without coevolution.

1 Introduction

Several approaches have been developed to optimize parameters of Evolving Connectionist Systems (ECoSs) [1] using evolutionary algorithms, e.g. [2], [3], [4], and [5]. Nevertheless, ensembles of learning machines have been formally and empirically shown to generalize better than single predictors [6]. Instead of utilizing just one neural network to solve a specific problem, an ensemble of neural networks combines a set of neural networks. In order to improve the accuracy of a particular type of ECoSs called Evolving Fuzzy Neural Network (EFuNN), a multi-module classifier called multiEFuNN has been proposed in [7].

However, the construction of ensembles of neural networks is not an easy task [6]. Besides, the choice of the best EFuNN parameters set is also a difficult task and the execution time of evolutionary algorithms to optimize the EFuNN parameters is high. Therefore, a new approach to construct ensembles of neural networks has been proposed in [8] and experiments have been made using a Coevolutionary Genetic Algorithm to generate a particular type of EFuNN ensembles. These experiments have shown that CONE is able to generate EFuNN ensembles with accuracy either better or equal to the accuracy of single EFuNNs generated using a Genetic Algorithm (GA). Moreover, the execution time of CONE to produce EFuNN ensembles is lower than the execution time of the GA.

However, the Coevolutionary GA used in [8] demands the predefinition of some parameters for the fitness function. These parameters have great influence on the results of the evolutionary process. Besides, CONE needs to be evaluated using other coevolutionary algorithms in order to be validated. Thus, a coevolutionary multi-objective GA has been used with CONE in [9] to perform experiments on four classification problems. Nevertheless, CONE needs to be evaluated using not only classification problems, but also time series prediction problems in order to be validated. Thus, this paper presents experiments which have been made with CONE and a coevolutionary multi-objective GA on two classification problems and two time series prediction problems.

This paper is organized as follows: Sect. 2 contains an explanation about ECoSs and EFuNNs. Section 3 presents CONE and explains a particular instance of it (i.e. a clustering method, a particular type of EFuNN ensemble and a coevolutionary multi-objective GA which can be used by the approach). Section 4 presents the results of the experiments made with this instance of CONE. Section 5 presents the conclusions and future works.

2 ECoS and EFuNNs

The ECOSs presented in [1] are systems constituted by one or more neural networks. Some of their characteristics are that their learning is on-line, incremental, fast and local [4]. EFuNNs [10] are a class of ECOSs which join the neural networks functional characteristics to the expressive power of fuzzy logic.

The EFuNN learning has some predefined parameters. Using different parameters sets, EFuNNs attain different performances and different weights are learned. The optimal parameters set usually depends on the input and output data presented. Thus, it is important to correctly choose the parameters which define the EFuNN learning according to the data presented.

Some of the predefined parameters of the EFuNN leaning algorithm are the number of membership functions; the initial sensitivity threshold (S) of the nodes (it is also used to determine the initial radius of the receptive field of a node); the error threshold (E); the *m-of-n* value (number of highest activation nodes used in the learning); and the maximum radius of the receptive field ($Mrad$). It is recommended to read [10] to get more details about the EFuNN learning algorithm and its parameters.

3 CONE

This section briefly describes CONE [8]. The general idea of this approach is to construct neural network ensembles using a clustering method to partition the input space in clusters. The training and test patterns are used by the clustering method to create the clusters. After that, the clusters are used to separate the training and the test patterns themselves in various subsets of training and test patterns with empty intersection.Each subset is used to train/test a different population of neural networks, which composes a species that is evolved through a cooperative coevolutionary algorithm. Thus, each cluster is associated with a training subset, a test subset and a species.

At the end of the evolutionary process, the representatives of each species in the last generation are used to constitute the ensemble. In order to use/test the ensemble, the clusters to which the input test pattern belongs are determined. After that, the outputs of the EFuNNs correspondent to these clusters are calculated and combined using a predefined combining method. Examples of combining methods can be found in [11].

The patterns used by the approach are divided into 3 types: training patterns (used to create clusters and to train the neural networks), test patterns (used to create clusters and to test the neural networks during the evolutionary process), and final test patterns (used to test the neural network ensemble generated at the end of the evolutionary process).

The following sections explain the instance of CONE which has been used in the experiments to produce EFuNN ensembles: Sect.3.1 explains the clustering method used to partition the input space, Sect.3.2 explains the EFuNN ensembles created and Sect.3.3 explains the coevolutionary algorithm used.

3.1 Clustering Method

The clustering method used in the experiments is similar to the Evolving Clustering Method [7]. It is recommended to read [8] to get more details about the clustering method used. The main information about the clustering method for this paper is the *Dthrs* parameter. This is the most important parameter to determine whether a cluster could be updated to accommodate a particular pattern, or if a new cluster would have to be created to accommodate this pattern.

3.2 Creating EFuNN Ensembles

In the experiments performed with CONE, the coevolutionary algorithm was used to optimize the predefined parameters of the EFuNN learning which where cited in Sect.2, and the EFuNN learning algorithm itself was used to train the EFuNNs. A representative of a species was considered the best fit individual of the species. Two combining methods were used to combine the outputs of the EFuNNs that compose the ensemble. One of them is the arithmetic average of the outputs of the EFuNNs to which the pattern presented belongs. The other one is the weighted average of the outputs of the EFuNNs to which the pattern presented belongs. The value used as the weight of a cluster $C_j,\ j = 0, 1, ...N$ is $1/||x_i - Cc_j||)$, where x_i is the pattern presented and Cc_j is the cluster center. If a pattern does not belong to any cluster, the output of the ensemble is the output of the EFuNN correspondent to the cluster whose center is the nearest center to the pattern.

3.3 Coevolutionary Algorithm

This section describes the coevolutionary multi-objective GA which was used in the experiments. It is recommendable to read [9] to get more details about it.

The coevolutionary multi-objective GA used in the experiments has a binary representation of the EFuNN parameters to be optimized, bitwise bit-flipping mutation, one-point crossover, generational survivor selection, and the learning algorithm of the neural networks being optimized is used with the training patterns right before the calculation of the objective values.

The initial population of each species is composed by individuals created randomly choosing values for each of the EFuNN parameters to be optimized. In this population, the objectives vector of an individual i is:

$$[RMSE\_Obj_i = RMSE_i,\ SIZE\_Obj_i = size_i] \ , \tag{1}$$

where $RMSE_i$ is the Root Mean Squared Error (RMSE) obtained testing the EFuNN correspondent to the individual i using the test subset correspondent to its species, and $size_i$ is the size of this EFuNN. Thus, the objectives are calculated without considering the individuals of the other species. The size component of the objectives vector is used to penalize the size of the EFuNNs and reduce the execution time of the evolutionary algorithm, as suggested in [5].

In all generations after the initial one, the objectives of an individual i are calculated using not only the output error and the size of the EFuNN correspondent to i, but also the output error and size of the EFuNNs correspondent to the representatives of the other species in the previous generation. Thus, the objectives vector of an individual i is $[RMSE\_Obj_i,\ SIZE\_Obj_i]$, where:

$$RMSE\_Obj_i = \sqrt{\frac{SSE_i + repr\_sse}{total\_test\_patterns\_number}} \text{ and} \tag{2}$$

$$SIZE\_Obj_i = size_i + repr\_size \ . \tag{3}$$

In these equations, SSE_i is the Sum of Squared Error (SSE) obtained testing the EFuNN correspondent to the individual i with the test subset correspondent to its species; $size_i$ is the size of this EFuNN; *repr_sse* is the sum of the SSEs and *repr_size* is the sum of the sizes of the EFuNNs correspondent to the representatives of all other species in the previous generation; and *total_test_patterns_number* is the total number of test patterns, including the patterns of all species.

An individual i dominates an individual j if $(RMSE_i \le RMSE_j)$ and $(SIZE_i \le SIZE_j)$ and $(RMSE_i < RMSE_j$ or $SIZE_i < SIZE_j)$.

After the calculation of the objective values, the rank of each individual is calculated. As it is done in [12], the rank of an individual i of the population p in the generation g is the number of individuals $j \neq i,\ j \in p$ which dominate the individual i in the generation g. In this way, a pareto optimal individual (an individual which is not dominated by any other individual of the population) has always rank equal to 0.

The best individual of a population is the individual which has the lowest rank. When more than 1 individual has the same rank, the best between them is the one which has the lowest SSE obtained testing the EFuNN correspondent to it with the test subset correspondent to its species.

The parents selection is made using the roulette wheel method and is proportional to the value determined by be following equation:

$$Prob_{i,p,g} = \frac{max\_rank_{p,g} - rank_{i,p,g}}{\sum_{j=0}^{pop\_size_p - 1}(max\_rank_{p,g} - rank_{j,p,g})} \ , \tag{4}$$

where $max\_rank_{p,g}$ is highest rank of the population p in the generation g, $rank_{i,p,g}$ is the rank of the individual i of the population p in the generation g, and pop_size_p is the size of the population p.

4 Experiments

This section explains the experiments which have been made with the instance of CONE described in Sects. 3.1, 3.2 and 3.3. The experiments have utilized two classification databases (Card and Diabetes) [13] and two time series prediction problems (Mackey-Glass (MG) [14] and Gas Furnace (GF) [15]).

Section 4.1 shows the parameters which were used in the experiments and Sect. 4.2 presents the results of the experiments.

4.1 Parameters and Executions

The parameters utilized in the experiments were the same as the parameters used in [9], except the *Dthrs*. The *Dthrs* was empirically determined for each database and it was 0.40 for Card database, 0.25 for Diabetes database, 0.20 for Mackey-Glass time series and 3.00 for Gas Furnace time series. The EFuNN parameters optimized during the coevolutionary process were also the same as the optimized in [9]: m-of-n, E, $Mrad$, S and membership functions number.

Three different partitions of the training+test and final_test data sets of the classification problems were used and three different time series were used to compose 3 partitions of the training+test and final_test data sets for the time series problems:

– Mackey-Glass:
  $w(1) = [x(t-12)x(t-8)x(t-4)x(t); y(t+4)]$
  $w(2) = [x(t-18)x(t-12)x(t-6)x(t); y(t+6)]$
  $w(3) = [x(t-24)x(t-16)x(t-8)x(t); y(t+8)]$
– Gas Furnace:
  $w(1) = [y(t-1)y(t-2)x(t-1)x(t-2); y(t)]$
  $w(2) = [y(t-1)y(t-3)x(t-1)x(t-3); y(t)]$
  $w(3) = [y(t-2)y(t-3)x(t-2)x(t-3); y(t)]$

Ten executions with different random seeds were performed for each partition, thus totalizing 30 executions for each database. Executions with the above combinations of parameters were also made using a multi-objective GA to generate single EFuNNs. The multi-objective GA utilized was the same as the algorithm presented in Sect. 3.3, but using the objectives (1) for all generations and just one species. In this way, 30 executions of the multi-objective GA were made for each database.

The objective of the executions explained above was to compare:

- EFuNN ensembles generated using CONE with weighted average combining method (weighted EFuNN ensembles – WEns);
- EFuNN ensembles generated using CONE with arithmetic average combining method (arithmetic EFuNN ensembles – AEns);
- Single EFuNNs generated using multi-objective GA (Sing).

The characteristics compared were the execution times of the evolutionary approaches, and the output classification errors/sum of squared errors(SSEs) of the EFuNN ensembles and of the single EFuNNs generated.

4.2 Results

In this section, the classification errors and the SSEs are those obtained using the final_test patterns set to test the single EFuNNs or the EFuNN ensembles generated after the evolutionary processes. The classification errors are used for the classification problems and the SSEs are used for the time series problems.

Table 1 shows the classification error/SSE and execution time averages, standard deviations, minimal and maximum values, considering the 30 executions for each database. Table 2 shows the statistics of the T student tests [16] performed to compare the classification errors/SSEs and the execution times. As it can be seen, the classification error/SSE averages of the ensembles created for all databases were considered statistically equal to the classification error/SSE averages of the single EFuNNs. The classification error averages of the weighted EFuNN ensembles were also considered statistically equal to the classification error averages of the arithmetic EFuNN ensembles, for the classification problems. However, for the time series problems, the SSE averages of the weighted EFuNN ensembles were statistically lower than the SSE averages of the arithmetic EFuNN ensembles.

For all databases, the execution time average among all 30 executions of CONE to generate EFuNN ensembles was statistically lower than the execution time average among all 30 executions of the multi-objective GA to generate single EFuNNs. Table 2 shows the statistics of the T student tests made to prove this analysis.

The execution time of the CONE to generate EFuNN ensembles was lower than the multi-objective GA execution time to generate single EFuNNs probably because in the optimization process of a single EFuNN, for each pattern presented to train/test the EFuNN, the activation levels of all rule nodes of the EFuNN have to be calculated. When an EFuNN ensemble is being created, just the activation levels of the rule nodes of the correspondent EFuNNs have to be calculated. A single EFuNN is usually higher than each EFuNN which compose an ensemble because a single EFuNN has to accommodate all training patterns and a component of an ensemble has to accommodate only the patterns correspondent to a particular cluster of the input space.

Table 1. Measures related to the class error/SSEs and execution times

		Class errors		SSEs		Execution times			
		Card	Diabetes	MG	GF	Card	Diabetes	MG	GF
WEns	Av	0.1611	0.2717	0.0621	182.2633	3122.5333s	804.6s	155s	70.8333s
	SD	0.0276	0.0304	0.0188	118.8774	652.2496s	174.3222s	45.8333s	16.7396s
	Min	0.1214	0.2083	0.0323	79.9432	2063s	500s	76s	49s
	Max	0.2370	0.3281	0.1262	429.2530	4228s	1174s	232s	106s
AEns	Av	0.1611	0.2744	0.0645	184.3665				
	SD	0.0281	0.0363	0.0184	119.1835	Equal to WEns			
	Min	0.1214	0.1875	0.0357	81.7392				
	Max	0.2370	0.3594	0.1274	432.929				
Sing	Av	0.1694	0.2722	0.0536	171.2507	6717.9333s	2200.8333s	746.8333s	184.8s
	SD	0.0294	0.0289	0.0270	124.6976	1285.7797s	467.8189s	208.9156s	55.9411s
	Min	0.1272	0.2187	0.0194	70.7992	4274s	1172s	297s	101s
	Max	0.2312	0.3333	0.1028	391.727	9536s	3396s	1112s	275s

Table 2. T Student test statistics comparing the class error/SSE averages and the execution time averages, using level of significance equal to 0.05

		Card	Diabetes	MG	GF
Class error/SSE	WEns x AEns	-0.0001	-1.0523	-7.8697	-5.3195
averages comparisons	WEns x Sing	-1.5590	-0.0893	1.4294	1.3073
	AEns x Sing	-1.5548	0.3334	1.8200	1.5590
Execution time	WEns x Sing	-13.2646	-15.5206	-15.1242	-12.4373
averages comparisons	AEns x Sing	Equal to WEns x Sing			

5 Conclusions

This paper presents experiments which have been made with CONE using a coevolutionary multi-objective GA. The experiments have used two classification problems and two time series prediction problems.

The experimental results have shown that the EFuNN ensembles construction using CONE has a lower execution time than the single EFuNNs construction using a multi-objective GA. The standard deviations of the execution times are also lower for CONE. Even so, the EFuNN ensembles generalization abilities are statistically equal to the single EFuNNs ones. These results contribute to the validation of CONE, reinforcing the results presented in [8] and [9], which have shown that CONE is able to produce EFuNN ensembles with either equal or better generalization using a lower execution time than similar non-coevolutionary algorithms to produce single EFuNNs.

Future works include the use of other clustering methods and coevolutionary algorithms to create neural network ensembles using CONE.

Acknowledgment

This work was supported by CNPq – Brazil.

References

1. Kasabov, N.: Evolving Connectionist Systems. Springer, Great Britain (2003)
2. Watts, M., Kasabov, N.: Dynamic optimisation of evolving connectionist system training parameters by pseudo-evolution strategy. In: CEC'2001. Volume 2., Seoul (2001) 1335–1342
3. Watts, M., Kasabov, N.: Evolutionary optimisation of evolving connectionist systems. In: CEC'2002. Volume 1., Honolulu, Hawaii, IEEE Press (2002) 606–610
4. Kasabov, N., Song, Q., Nishikawa, I.: Evolutionary computation for dynamic parameter optimization of evolving connectionist systems for on-line prediction of time series with changing dynamics. In: IJCNN'2003. Volume 1., Oregon (2003) 438–443
5. Minku, F.L., Ludermir, T.B.: Evolutionary strategies and genetic algorithms for dynamic parameter optimization of evolving fuzzy neural networks. In: CEC'2005. Volume 3., Edinburgh, Scotland (2005) 1951–1958
6. Chandra, A., Yao, X.: Ensemble learning using multi-objective evolutionary algorithms. Journal of Mathematical Modelling and Algorithms (1) (2006)
7. Kasabov, N.: Ensembles of efunns: An architecture for a multimodule classifier. In: Proceedings of the International Conference on Fuzzy Systems. Volume 3., Australia (2001) 1573–1576
8. Minku, F.L., Ludermir, T.B.: EFuNNs ensembles construction using a clustering method and a coevolutionary genetic algorithm (to appear). In: CEC'2006, Vancoucer, Canada (2006)
9. Minku, F.L., Ludermir, T.B.: EFuNN ensembles construction using CONE with multi-objective GA (to appear). In: SBRN'2006, Ribeirao Preto, Brazil (2006)
10. Kasabov, N.: Evolving fuzzy neural networks for supervised/unsupervised on-line, knowledge-based learning. IEEE Transactions on Systems, Man and Cybernetics **31**(6) (2001) 902–918
11. Dietterich, T.G.: Machine-learning research: Four current directions. The AI Magazine **18**(4) (1998) 97–136
12. Fonseca, C.M., Fleming, P.J.: Multi-objective optimization and multiple constraint handling with evolutionary algorithms - part I: A unified formulation. IEEE Transactions on Systems, Man and Cybernetics - Part A **28**(1) (1998) 26–37
13. Prechelt, L.: PROBEN1 - a set of neural network benchmark problems and benchmarking rules. Technical Report 21/94, Fakultt fr Informatik, Universitt Karlsruhe, Karlsruhe, Germany (1994)
14. Mackey, M.C., Glass, L.: Oscillation and chaos in physiological control systems. Science **197** (1977) 287–289
15. Box, G.E.P., Jenkins, G.M.: Time Series Analysis, Forecasting and Control. Holden Day, San Francisco (1970)
16. Witten, I.H., Frank, E.: Data Mining - Pratical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann Publishers, San Francisco (2000)

Language Learning for the Autonomous Mental Development of Conversational Agents

Jin-Hyuk Hong, Sungsoo Lim, and Sung-Bae Cho

Dept. of Computer Science, Yonsei University
134 Sinchon-dong, Sudaemoon-ku
Seoul 120-749, Korea
{hjinh, lss}@sclab.yonsei.ac.kr, sbcho@cs.yonsei.ac.kr

Abstract. Since the manual construction of our knowledge-base has several crucial limitations when applied to intelligent systems, mental development has been investigated in recent years. Autonomous mental development is a new paradigm for developing autonomous machines, which are adaptive and flexible to the environment. Language development, a kind of mental development, is an important aspect of intelligent conversational agents. In this paper, we propose an intelligent conversational agent and its language development mechanism by putting together five promising techniques; Bayesian networks, pattern matching, finite state machines, templates, and genetic programming. Knowledge acquisition implemented by finite state machines and templates, and language learning by genetic programming are developed for language development. Several illustrations and usability tests show the usefulness of the proposed developmental conversational agent.

1 Introduction

Mental development is a primary human characteristic, and vision, audition, behavior, and language are the principal ingredients (Johnson & Munakata, 2005; Weng, 2000). Matured people are usually better than the younger in these areas, since they are more mentally developed. Similarly, an intelligent system that has a richer and more sophisticated knowledge base might be better than other systems. It is, however, very difficult to construct any kind of knowledge base during the initial stage of the manual development of the system (Lauria, 2001). Manual construction is apt to make a system static and stiff toward its environment. Therefore, recent research on intelligent systems has focused on an autonomous mental development (AMD) mechanism. Like humans, this mechanism develops its mental capabilities through autonomous real-time interactions with its environment. This includes interactions with human supervisors. AMD, a new paradigm for developing autonomous machines, constructs an intelligent system that is based on incremental learning (Joshi & Weng, 2003) and is adaptive and flexible to the environment.

Linguistic sense is highly related to mental development. Since adults generally accumulate more knowledge and skill than children, they can generate more sophisticated and informative sentences (Clack, 2004). A conversational agent, a popular type of intelligent system, provides users with informative answers based on such a linguistic sense. Since experiments are commonly based on pattern-response pairs, the level of the knowledge base determines the quality of the response. That is, if the knowledge-base of the conversational agent is composed of lots of qualified pattern-response pairs, the agent works well enough to respond to the user's queries. Autonomous Language Development (ALD) of a conversational agent is different from the conventional manual approach.

Computational implementations of learning mechanisms provide language acquisition researchers with important clues as to how the brain starts to use language (Plunkett, 1997). Moreover, these implementations promote the development of intelligent systems that interact with humans by using a realistic and scalable conversational interface. Lauria *et al.* proposed an instruction-based learning method for personal robots (Lauria, 2001), while Dominey and Boucher developed a system for language learning. Zhou proposed a robot learning method with GA-based fuzzy reinforcement learning (Zhou, 2003), while Jerbic *et al.* examined an autonomous agent based on reinforcement learning and an adaptive shadowed network. Joshi and Weng introduced the concept of AMD and studied the autonomous learning of speech production under reinforcement learning (Joshi & Weng, 2003).

2 The Developmental Conversational Agent

2.1 Conversational Agent Architecture

The proposed conversational agent is composed of three parts: language interpretation, language generation, and language development. Language interpretation infers the intention of the user's queries, while language generation generates answers that correspond to those queries. When it does not understand the query, language development learns the pattern by interacting with the human. Language development also

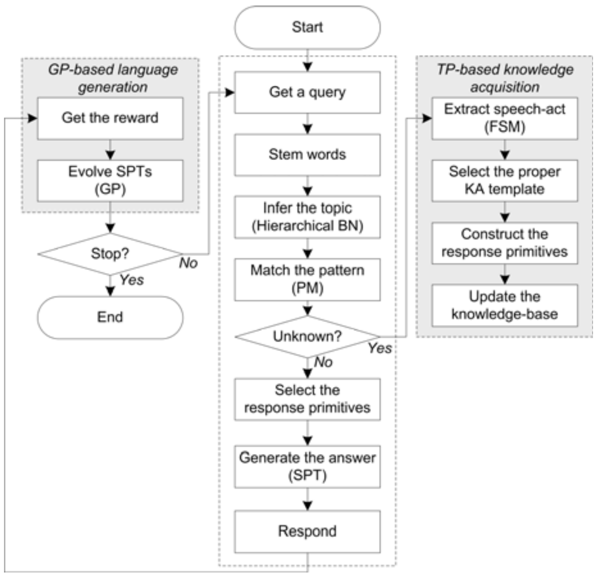

Start
GP-based language generation
Get the reward
Evolve SPTs (GP)
Stop?
No
Yes
End
Get a query
Stem words
Infer the topic (Hierarchical BN)
Match the pattern (PM)
Unknown?
Yes
No
Select the response primitives
Generate the answer (SPT)
Respond
TP-based knowledge acquisition
Extract speech-act (FSM)
Select the proper KA template
Construct the response primitives
Update the knowledge-base

Fig. 1. The process of the proposed conversational agent

evolves its linguistic sense for generating answers by using genetic programming. Fig. 1 shows the management of the input query. A pattern is analyzed and accumulated into the knowledge base, while the answer is used to evaluate the genetic programming that produces the sentence plan trees (SPTs).

1) Two-stage inference for dialogue management
At first, words that infer the user's intention are extracted from the input query. This inference is determined by two methods: the Bayesian network and keyword matching. The Bayesian network infers the user's intention, while keyword matching selects a proper pattern-response pair for generating the corresponding answer. Analyzing queries in these stages makes it feasible to infer the detailed intention of the user and to model the context of the conversation. Furthermore, dividing the knowledge base improves the scalability and portability of the conversational agent.

Topic Inference using the Bayesian Network: The Bayesian network is used to infer the topic of a user's query and model the context of dialogue. This leads to a definition of the scope of the dialogue. Since Bayesian networks are based on graph theory, they are effective in inference and representation. In this paper, the Bayesian network is hierarchically constructed with three levels based on function: keyword, concept, and topic. The keyword level consists of words related to topics in the domain. The concept level is composed of the entities or attributes of the domain, while the topic level represents entities whose attributes are defined.

Response Selection using Keyword Matching: Once the topic is selected for an input query, keyword matching (using the knowledge base associated with the topic) is performed to find the corresponding response. When there are many scripts, performance declines because of the redundancy of information. In this paper, we divide the scripts into several groups based on their topics. This reduces the number of scripts to be compared. A script is stored as an XML form. A set of candidate scripts are sequentially matched to find an appropriate response. The matching scores are calculated by the F-measure, which is a popular measurement in text classification. When there is a corresponding pattern-response pair, language generation is used to generate the answer.

2) Answer generation using the sentence plan trees (SPTs)
Once a pattern-response pair is selected, language generation is applied to the corresponding response primitives. In language generation, a SPT is selected and filled with response primitives to construct an answer. A SPT is a binary tree with leaves labeled by pre-defined templates of simple sentences (SSs), and with interior nodes labeled with joint operators (JOs) that combine two sentences (Ratnaparkhi, 2002). We define five types of JOs (suitable for the Korean language) to combine 2 sentences, as follows.

Sentence A = subject (s1) + response primitive (t1) + verb (v1).
Sentence B = subject (s2) + response primitive (t2) + verb (v2).

- **JO 1:** Combine sentences A and B by using the conjunction 'and'. The result is 's1 t1 v1, and s2 t2 v2.'
- **JO 2:** Combine two sentences that have the same subject (s1 = s2). The result is 's1 t1 v1 and t2 v2.'

- **JO 3:** Combine two sentences that have the same subject (s1 = s2) and the same verb (v1 = v2). The result is ‘s1 t1 t2 v1.’
- **JO 4:** Combine two sentences that have the same response primitive (t1 = t2) and the same verb (v1 = v2) by making a proper subject (s3). The result is ‘s3 t1 v1’ where s3 is the subject that includes s1 and s2.
- **JO 5:** Combine two sentences that have the same subject (s1 = s2) and different verbs but can be replaced by a verb v3, which includes both v1 and v2. The result is ‘s1 t1 t2 v3.’

2.2 Autonomous Language Development in the Conversational Agent

1) Dialogue-based knowledge acquisition
For the developmental conversational agent, knowledge acquisition is accomplished (as shown in Fig. 1) through dialogue. It first analyzes the pattern of the input query and then constructs the response through interaction with a human. The pattern of a script is composed of the topic and a set of words. The topic is obtained by the Bayesian network, while the words are extracted by preprocessing.

We consider the dialogue act of an input query to select a proper answer template that is used to collect the response primitives. An automaton extracts a dialogue act, and then 30 automata are designed for 30 dialogue acts, as shown in Table 1. A subsumption architecture is adopted to select one dialogue act for per query.

Table 1. Dialogue acts defined in this paper

Category	Dialogue act
Question	Ability, Description, Fact, Location, Method, Miscellaneous, Obligation, R eason, Time, WhatIf, Who, Acquisition, Comparison, Confirmation, Cost, Direction, Example, More, Possession
Statement	Act, Fact, Message, Miscellaneous, Possession, Status, Want, Cause, Cond ition, Feeling, Time

Each dialogue act has corresponding answer templates. This template technique is useful to represent static information in situations where variables change dynamically. In the answer template, “class” means the dialogue act, and “question” is a sentence to collect the response primitive. “Requirement (information collected from the user)” is the response primitive for constructing the answers. We define 64 templates: three templates for each question dialogue act, and six positive/negative templates for each statement dialogue act. Finally, the response is completed with the template and saved as the response part of the script. The pattern-response pair is generated with the patterns and the responses.

2) GP-based language generation
Genetic programming, as proposed by John R. Koza, is used to generate the adaptive structure of answers to the domain (Koza, 1994). An individual represented as an SPT denotes a structure of answers. As mentioned before, the leaf node of an SPT contains an SS that corresponds to the response primitive. The parent nodes represent JOs. SPTs can evolve by genetic programming when humans are involved.

An individual exposed to genetic programming is represented as an SPT that consists of the joint operator set {JO1, JO2, JO3, JO4, JO5} and the response primitive set $\{r_1, r_2, \ldots, r_n\}$ where n is the number of response primitives. The grammar for the SPT is: G={V={EXP, OP, VAR}, T={ JO1, JO2, JO3, JO4, JO5, $r_1, r_2, \ldots, r_n$}, P, {EXP}}, where the rule set P is as the following:

$$\begin{aligned}&\text{EXP}\rightarrow\text{EXP OP EXP} \mid \text{VAR}\\&\text{OP}\rightarrow\text{JO1} \mid \text{JO2} \mid \text{JO3} \mid \text{JO4} \mid \text{JO5}\\&\text{VAR}\rightarrow r_1 \mid r_2 \mid \ldots \mid r_n\end{aligned}$$

Crossover and mutation are employed to generate diverse sentence structures. Crossover randomly selects and changes sub-trees from two individuals, while mutation changes a sub-tree into a new one. These processes are conducted according to predefined probabilities. Through the evolution in real time, the system develops its linguistic sense for generating sentences.

3 Experimental Results

3.1 Implementation

To show the usefulness of the proposed method, we have developed an artificial secretary system using MFC. It consists of a main window for displaying information, an input text box, and the avatar system with a speech generation engine. When a user types a query, the avatar provides the answer in speech with a corresponding action. Q-avatar[1] is employed as the avatar system, while 'Voiceware[2]', a solution for speech recognition and generation, is used to provide user a realistic and convenient interface.

The target domain of the genetic programming of the proposed method is travel planning, and Table 2 describes six response primitives of the domain. When the user's query arrives, it extracts the response primitives from the query to collect information for planning a train trip. There are six SPTs based on the number of response primitives collected. If there is no information on travel, the agent uses the

Table 2. Response primitives used by a travel planning agent

Response primitive		Simple Sentence
Departure Location	Question	Where are you leaving from?
	Statement	You leave [dLocation].
Arrival Location	Question	Where are you going?
	Statement	You are going to [aLocation].
Departure Date	Question	What day do you leave?
	Statement	You are leaving on [Date].
Departure Time	Question	What time do you leave?
	Statement	You leave at [dTime].
Arrival Time	Question	What time do you want to arrive?
	Statement	You arrive at [aTime].
Seat Type	Question	What kinds of seats do you want?
	Statement	You'll take [tGrade].

[1] http://www.qavatar.com
[2] http://www.voiceware.co.kr

first SPT to generate a sentence structure, while if all travel information is collected, it locates a proper train from the database. Genetic programming evolves each SPT, while a human scores as fitness between 0 and 10 through evolution.

Fig. 2 shows the example of autonomous linguistic mental development through knowledge acquisition. Contrary to manual operations, a user only needs to provide what the agent requires through dialogue. It automatically performs all processes such as stemming words, defining the dialogue act, and constructing the answers.

User: Where is the laboratory?
Agent: Hmm, I don't know. Change mode to knowledge acquisition.
⇒ analyzing query (keyword: laboratory, where)
(topic: laboratory's location)
(dialogue act: location question)
Agent: Where is it?
User: Number 529, the 3rd engineering building, 134, Yonsei University.
⇒ analyzing answer (response primitive: 529, the 3rd engineering building, 134, Yonsei University)
⇒ generating the pattern-response pair

```
<SCRIPT>
<PATTERN> where laboratory </PATTERN>
<RESPONSE> 529, the 3rd engineering building, 134, Yonsei
 University </RESPONSE>
<TYPE> location </TYPE>
</SCRIPT>
```

User: Where is the laboratory?
Agent: Number 529, the 3rd engineering building, 134, Yonsei University

Fig. 2. Knowledge acquisition when using the proposed method

3.2 Usability Test for the Language Development

We conducted a usability test to demonstrate the proposed method. First, we collected a number of dialogues from eight subjects who perform three tasks that requested to search for information. 50 dialogues were used as training data to construct pattern-response pairs, while another 50 dialogues were used as test data. Both experts and novices performed the experiment. Table 3 shows the results of the usability test for knowledge acquisition. For queries with the same pattern, designers did not recognize them manually, while they did notice them when using the proposed method. Therefore, the size of knowledge base is optimized. In terms of construction time and user feedback, the proposed method is outstandingly superior to manual construction.

For the demonstration of autonomous language development by genetic programming, we also performed the usability test with 10 subjects iteratively interacting with the system in 90 generations. The crossover rate and mutation rate were set as 0.6 and 0.2, respectively. For each generation, the subjects evaluated the fitness of all individuals. For simplicity, we set the population size as 20.

Table 3. Results of the usability test for autonomous linguistic mental developmentmodel

		Manual construction		Proposed method	
		Experts	Novices	Experts	Novices
Pattern-response pairs generated		50	50	44	44
File size		28 KB	30 KB	24 KB	24 KB
Construction time		20 min.	1 hour	5 min.	8 min.
Accuracy	Training	92%	84%	100%	96%
	Test	84%	82%	88%	86%
Feedback (0~5)		2	1	4	4

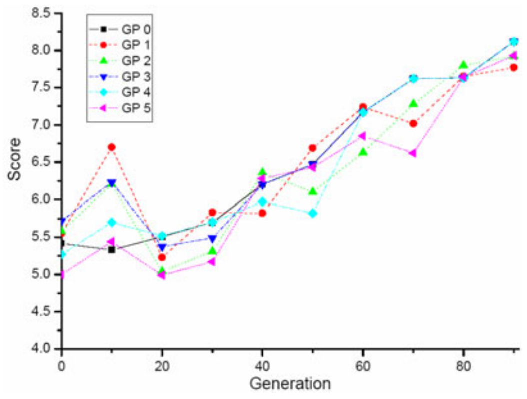


Fig. 3. Score distribution through evolution

Fig. 3 presents the changes of the average score as the generation grows. GP *n* means the STP whose number of response primitives given is *n*. We limited the generation to 90 steps because more steps cause overfitting and a decrease in diversity. As a result, the score increased rapidly during the first 10 generations and decreased between the 10th and 20th generations, then increased continuously after the 20th generation.

The system used the responses primitives of departure location, arrival location, and departure data. The initial sentence was disorderly and complex, while the sentence obtained through evolution became more refined.

- **Generation 0:** You leave Seoul on October 10th and what time do you want to arrive? And what time do you leave? And you are going to Pusan.
- **Generation 30:** What kind of train do you want to take and what time do you want to leave? And you leave Seoul on October 10th.
- **Generation 60**: What time do you leave Seoul? And you are going to Pusan.
- **Generation 90:** What time do you leave Seoul for Pusan?

4 Conclusion

We have proposed a conversational agent with autonomous language development. Since the manual construction of intelligent systems presents several problems, it was necessary to propose a developmental conversational agent that might adapt itself to the environment. In language development, we addressed the autonomous increasing of the knowledge base and linguistic sense obtained by genetic programming. In this paper, various illustrations and usability tests demonstrated the usefulness of the proposed conversational agent.

Language is very important in mental development, and people improve their linguistic sense of understanding queries and generating sentences. Research on autonomous language development might not only be useful to demonstrate various cognitive language models in cognitive science, but also provide a direction in engineering to construct an effective communication method between humans and intelligent systems. In our future work, we will develop an intelligent system that includes autonomous mental and language development.

Acknowledgement

This research was supported by Brain Science and Engineering Research Program sponsored by Korean Ministry of Commerce, Industry and Energy.

References

Clack, E. (2004). How language acquisition builds on cognitive development. *Trends in Cognitive Science*, *8*, 472-478.

Johnson, M., & Munakata, Y. (2005). Processes of change in brain and cognitive development. *Trends in Cognitive Sciences*, *9*, 152-158.

Joshi, A., & Weng, J. (2003). Autonomous mental development in high dimensional context and action spaces. *Neural Networks*, *16*, 701-710.

Koza, J. (1994). *Genetic programming, Automatic discovery of reusable programs*, The MIT Press.

Lauria, S., et al. (2001). Personal robots using natural language instruction. *IEEE Intelligent Systems*, *16*, 38-45.

Plunkett, K. (1997). Theories of early language acquisition. *Trends in Cognitive Science*, *1*, 146-153.

Ratnaparkhi, A. (2002). Trainable approaches to surface natural language generation and their application to conversational dialog systems. *Computer Speech and Language*, *16*, 435-455.

Weng, J., et al. (2000). Autonomous mental development by robots and animals. *Science*, *291*, 599-600.

Zhou, C. (2002). Robot learning with GA-based fuzzy reinforcement learning agents. *Information Science*, *145*, 45-68.

A Multi-objective Evolutionary Algorithm for Multi-UAV Cooperative Reconnaissance Problem

Jing Tian and Lincheng Shen

College of Mechatronic Engineering and Automation,
National University of Defense Technology, Changsha, P.R. China
jingtian@nudt.edu.cn

Abstract. The object of multiple Unmanned Aerial Vehicles(UAVs) cooperative reconnaissance is to employ a limit number of UAVs with different capabilities conducting reconnaissance on a set of targets at minimum cost, without violating real world constraints. This problem is a multi-objective optimization problem. We present a Pareto optimality based multi-objective evolutionary algorithm MUCREA to solve the problem. Integer string chromosome representation is designed which ensures that the solution can satisfy the reconnaissance resolution constraints. A construction algorithm is put forward to generate initial feasible solutions for MUCREA, and Pareto optimality based selection with elitism is introduced to generation parent population. Problem specific evolutionary operators are designed to ensure the feasibilities of the children. Simulation results show the efficiency of MUCREA.

1 Introduction

Multiple Unmanned Aerial vehicles (UAVs) cooperative reconnaissance remains key activities in military operations. Multi-UAV cooperative reconnaissance problem(MUCRP) can be described as that, N_V kinds of UAVs with different capabilities are required to conduct reconnaissance on N_T targets located in different sites. Mission plans should be quickly made for UAVs that meet the reconnaissance demands and can deal with the real-world constraints such as reconnaissance resolution, time windows, number of UAVs available, and capabilities of the UAVs(i.e. imagery resolution, maximum permitted travel time etc.).

Some solutions have been proposed for the multiple UAVs cooperative reconnaissance problem, such as [1][2][3]. Although those previous studies are proven to be efficient to some extent, the reconnaissance demands on the targets are not considered thoroughly, such as reconnaissance resolution and time window constraints. Further more, in previous studies, the limited number of UAVs was not considered, but in the real world only a limited number of UAVs is available, thus one of the objectives for cooperative reconnaissance should be maximizing the number of reconnoitered targets with limited number of UAVs, which has never been considered in previous researches.

In this paper, we first present a mathematical model for MUCRP. The model focuses on maximizing the number of targets reconnoitered with limited number of UAVs as well as minimizing the reconnaissance cost. Then a Pareto optimality based multi-objective evolutionary algorithm is proposed to solve the problem. The rest of the paper is organized as follows. Section 2 presents the mathematical model of MUCRP. The proposed algorithm is described in detail in section 3. Section 4 shows the simulation results and section 5 concludes the paper.

2 Problem Formulation

The reconnaissance targets set is denoted by $T_0 = \{1, 2, ..., N_T\}$, and $T = \{0, 1, ..., N_T\}$ denotes the extended targets set, where 0 indicates the base that UAVs depart from and return to, N_T is the number of targets. R_i is the reconnaissance resolution demand of target $i \in T_0$. Each target $i \in T_0$ has a time window $[e_i, l_i]$, where e_i is the earliest time allowed for reconnaissance on it and l_i is the latest time. T_i is the time to begin reconnaissance on target i. An UAV may arrive before e_i, which incurs the waiting time w_i until reconnaissance is possible. However, no UAV may arrive past l_i. $V = \{V^1, V^2, ..., V^{N_V}\}$ denotes UAVs set with different capabilities, where V^k denotes the k-th kind of UAVs set and v_q^k is the qth UAV in V^k. The maximum travel time permitted for UAVs in V^k is TL_k, and the reconnaissance resolution is r_k, $r_1 < r_2 < ... < r_{Nv}$. N_v^k is the number of UAVs available in V^k. s_i^k denotes the reconnaissance duration that an UAV in V^k conducts reconnaissance on target i. The routes set between target pairs is denoted by $A = \{(i, j) | i, j \in T, i \neq j\}$. Each route $(i, j) \in A$ is associated with a distance d_{ij} denoting the length of (i, j) and a travel time t_{ij}^k denoting the travel time for UAV in V^k between target i and j. The decision variable is

$$x_{pij}^k = \begin{cases} 1 & \text{if route } (i,j) \text{ is used by UAV } v_p^k \\ 0 & \text{otherwise} \end{cases}$$

The MUCRP model can be mathematically formulated as follows:

$$(\textbf{MUCRP}) \ \min \boldsymbol{f} = (f_1, f_2) \tag{1}$$

$$f_1 = N_T - \sum_{j \in T_0} \sum_{V^k \in V} \sum_{v_p^k \in V^k} x_{pij}^k \tag{2}$$

$$f_2 = \sum_{V^k \in V} \sum_{v_p^k \in V^k} \sum_{(i,j) \in A} f_{ij}^k x_{pij}^k \tag{3}$$

Where $f_{ij}^k = \alpha d_{ij} + \beta(T_i + s_i^k + t_{ij}^k + w_j)$.
Subject to:

$$\sum_{j \in T_0} x_{p0j}^k = 1, \sum_{j \in T_0} x_{pj0}^k = 1, \forall v_p^k \in V^k, V^k \in V \tag{4}$$

$$\sum_{v_p^k \in V^k} \sum_{(i,j)\in A} x_{pij}^k \le N_v^k, \forall V^k \in V \tag{5}$$

$$\sum_{V^k \in V} \sum_{v_p^k \in V^k} \sum_{i \in T} x_{pij}^k \le 1 \quad \forall j \in T_0 \tag{6}$$

$$e_i \le T_i \le l_i \quad \forall i \in T_0 \tag{7}$$

$$If \;\; x_{pij}^k = 1, \; then \;\; T_i + s_i^k + t_{ij}^k + w_j = T_j \tag{8}$$

$$w_j = max(0, e_j - (T_i + s_i^k + t_{ij}^k)) \tag{9}$$

$$\sum_{(i,j)\in A} x_{pij}^k (t_{ij}^k + s_j^k + w_j) \le TL_k \;\; \forall v_p^k \in V^k, V^k \in V \tag{10}$$

$$If \;\; x_{pij}^k = 1, \; then \;\; r_k \le R_j \tag{11}$$

MUCRP has two objectives to minimize. f_1 means to minimize the number of targets not be reconnoitered. f_2 means to minimize the total UAVs travel distances and the consumed time for finishing the cooperative reconnaissance mission. f_1 is called the number of unreconnoitered targets(NUT) and f_2 is called the cooperative reconnaissance cost(CRC) in the rest of the paper. Eq.(4) specifies that each UAV should begin at the base and return to the base finally. Eq.(5) constrains that the number of UAVs employed in reconnaissance should not exceed the number of UAVs available. Eq.(6) ensures that each target can be reconnoitred by at most one UAV. The time feasibility constraints are defined in equations (7)-(10). Inequality (7) imposes the time window constraints. Waiting time is the amount of time that an UAV has to wait if it arrives at a target location before the earliest time for that target. Inequality (10) states that the total travel time of an UAV could not exceed the maximum permitted travel time of it. Inequality (11) restricts that the UAV conducting reconnaissance on a target should satisfy its reconnaissance resolution demand.

3 Multiobjective Evolutionary Algorithm for MUCRP

MUCRP is a typical multi-objective optimization problem(MOP) and the optimal solutions to MOP are non-dominated solutions known as Pareto optimal solutions. Two objective vectors $\boldsymbol{u} = (u_1, u_2)$ and $\boldsymbol{v} = (v_1, v_2)$, $\boldsymbol{u}$ is said to dominate $\boldsymbol{v}$ if and only if $\forall i \in \{1,2\} : u_i \le v_i \wedge \exists j \in \{1,2\} : u_i < v_i$, which is denoted as $\boldsymbol{u} \prec \boldsymbol{v}$. A solution $\boldsymbol{x}$ is said to dominate solution $\boldsymbol{y}$ iff $\boldsymbol{f}(\boldsymbol{x}) \prec \boldsymbol{f}(\boldsymbol{y})$. $\boldsymbol{x_0}$ is Pareto optimal if there is no other solution in the search space that dominates $\boldsymbol{x_0}$. The set of all Pareto optimal solutions is called as *Pareto optimal set*.

Multi-objective Evolutionary Algorithm(MOEA) has been increasingly investigated to solve MOP and has been proven to be a general, robust and powerful search mechanism[4][5]. This section provides a MOEA based approach to find the approximate Pareto optimal set of MUCRP, Multi-UAV cooperative reconnaissance evolutionary algorithm(MUCREA). Fig.1 illustrates the flowchart of MUCREA, which provides an overall view of the algorithm.

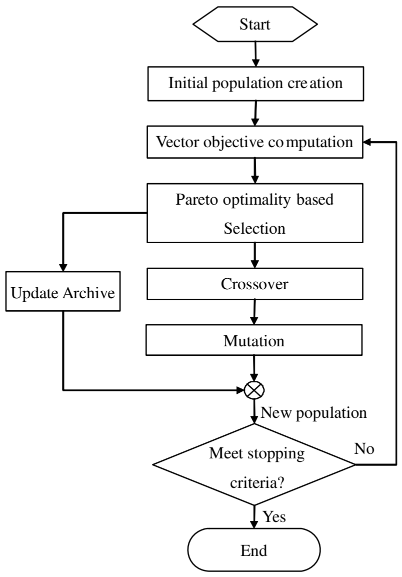

Start
Initial population creation
Vector objective computation
Pareto optimality based Selection
Update Archive
Crossover
Mutation
New population
Meet stopping criteria?
No
Yes
End

Fig. 1. The flowchart of proposed MUCREA

3.1 Chromosome Representation

To facilitate chromosome representation in MUCREA, all targets in T_0 are clustered according to their reconnaissance resolution demands and the reconnaissance resolutions of the UAVs. For a target $t \in T_0$, if $r_j < R_t < r_{j+1}$, then $t \in C_j$, which means that only UAVs in $V^1, V^2, ..., V^j$ can satisfy the reconnaissance resolution demands of target t.

A chromosome is shown in Fig.2, which consists of N_V sequences and is represented as $S = \{s_1, s_2, ..., s_{Nv}\}$. Sequence s_i is the reconnaissance targets sequence for UAVs in V^i. Each sequence $s_i = \{s_1^i, ..., s_n^i\}$ comprises several subsequences, where subsequence s_p^i is the reconnaissance targets sequence for UAV v_p^i. According to the target categorizes, all targets in C_j could only appear in one of the sequences in $\{s_i | i \leq j\}$. Thus the solution that the chromosome encodes can satisfy the constraint(11). Because only a limited number of UAVs is available, a solution of MUCRP contains some targets not be reconnoitered. Therefore, rejection pool(RP), a data structure to store the unreconnoitered targets, is associated with each chromosome.

3.2 Creation of Initial Population

In MUCREA, we generate a mix population comprising feasible solutions and random solutions. The reason for constructing this mix population is that, a population of entirely feasible members gives up the opportunity of exploring the whole regions; and a completely random population may be largely infeasible taking a long time to convergence. To generate initial feasible solutions, an

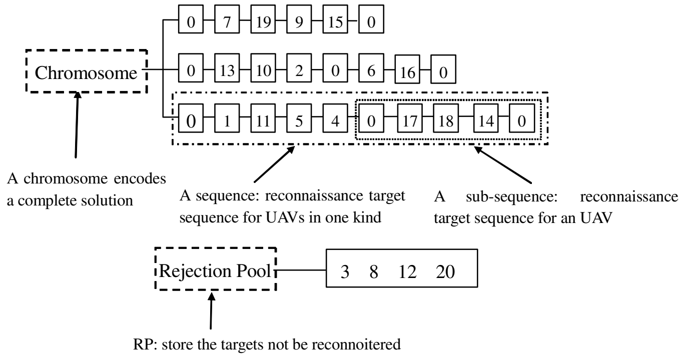

Chromosome
0 7 19 9 15 0
0 13 10 2 0 6 16 0
0 1 11 5 4 0 17 18 14 0
A chromosome encodes a complete solution
A sequence: reconnaissance target sequence for UAVs in one kind
A sub-sequence: reconnaissance target sequence for an UAV
Rejection Pool
3 8 12 20
RP: store the targets not be reconnoitered

Fig. 2. Data structure of chromosome representation and Rejection Pool

algorithm *Construction-Initial-Feasible-Solution*(CIFS), is presented. The detail steps of CIFS are shown in Fig.3. Due to the randomness in step(4), repeat running of CIFS will generate different feasible solutions.

In CIFS, Cs_k in step(1) is a data structure used to store targets that will be reconnoitered by UAVs in V^k. For a target t in C_k, $Cs_i, i \in \{1, ..., k\}$ is randomly chosen, then t is added into Cs_i. Steps(6-14) describe how to insert all targets in Cs_i to sequence s_k which is initially null. The cost function for inserting a target t as the first target to a new subsequence is as follows:

$$Cost1(t) = -\alpha t_{0t} + \beta d_{0t} \tag{12}$$

The target with the lowest cost t^* is selected as the first target in a new subsequence, see step(9). Eq.(12) implies that the desired target is the one that is distant from the base and has an early time window. The weights were derived empirically and were set to $\alpha = 0.7, \beta = 0.3$.

Once the first target t^* is selected for the current subsequence s_r^k, CIFS selects from the rest targets the target t which minimizes the total insertion cost between every two adjacent targets p and q in s_r^k without violating the time window and maximum permitted travel time constraints. The cost of inserting target t between p and q in s_r^k is:

$$Cost2(t, p, q) = \alpha D + \beta W + \gamma O + \eta T \tag{13}$$

Where, $D = \sum_{(i,j)\in \tilde{s}_r^k} d_{ij}$ is the total distance travelled by UAV v_r^k.

$W = \sum_{(i,j)\in \tilde{s}_r^k} (T_i + s_i^k + t_{ij}^k + w_j)$ is the total consumed time of UAV v_r^k.

$O = \sum_{(i,j)\in \tilde{s}_r^k} \max\{0, (T_i + s_i^k + t_{ij}^k - l_j)\}$ is the penalty for tardiness.

$T = \max\{0, W - TL_k\}$ is the penalty for exceeding the maximum travel time. Here, $\tilde{s}_r^k$ is the subsequence after t is inserted between p and q.

The target t is inserted to the least cost position between $(p*, q*) \in s_r^k$ and step(10) and step(11) are repeated until no more target can be inserted. At this stage, a new subsequence is created and steps(7-11) are repeated until all targets in Cs_k are inserted to sequence s_k.

```
Algorithm 1 Construction-Initial-Feasible-Solution(CIFS)
(1) Set all $Cs_i$=NULL, $i = 1, ..., N_V$, $k = 1$
(2) If $k > N_V$, go to step(6)
(3) If $C_k = NULL$, go to step(5)
(4) For a target $t$ in $C_k$,
        Randomly choose $i \in \{1, ..., k\}$, add $t$ into $Cs_i$.
        Delete $t$ from $C_k$. Go to Step(3)
(5) $k = k + 1$. Go to step (2).
(6) Set all sequences $s_i = NULL, i = \{1, ..., N_V\}$, $RP = NULL$, $k = 1$.
(7) $r = 1$
(8) $s_r^k = NULL$
(9) Select the target $t^* = \arg\min_{t \in Cs_k} Cost1(t)$, set $s_r^k = (0, t^*, 0)$.
    Delete target $t^*$ from $Cs_k$.
(10) If $Cs_k = NULL$, go to step(12).
(11) For a target $t$ in $Cs_k$,
        $(p^*, q^*) = \arg\min_{(p,q) \in s_r^k} Cost2(t, p, q)$
        Check the feasibilities of $\tilde{s}_r^k$=insert $t$ between $p^*$ and $q^*$
        If all the constraints are satisfied
            $s_r^k = \tilde{s}_r^k$, delete target $t$ from $Cs_k$. Go to step (12).
        Else $r = r + 1$, go to step (8)
(12) If $r > N_v^k$,
        Delete extra subsequences with the fewest targets.
        Add the targets in the deleted subsequences into RP.
        $k = k + 1$.
    Else $k = k + 1$.
(13) If $k > N_V$, go to step (14). Else go to step (7).
(14) Output $S = \{s_1, s_2, ..., s_{Nv}\}$ and RP. Stop CIFS.
```

Fig. 3. Construction-Initial-Feasible-Solution

Because the number of UAVs are limited, in each sequence s_k, only the N_v^k subsequences with the largest numbers of targets are kept. All other subsequences are removed from the solutions and the targets in these subsequences are copied to the RP, see step(12).

3.3 Pareto Optimality Based Selection with Elitism

To find the approximate Pareto set of the problem and avoid bias to any objective, we introduce a Pareto dominance based tournament selection strategy combined with diversity preservation to generate a new population.

Two individuals are randomly selected from the population. Firstly, the feasibilities of the two individuals are checked. If the two individuals are all infeasible solutions, one is randomly chosen to be the parent. If only one of the two individuals is a feasible solution, the feasible one is chosen to be the parent. If the two individuals are all feasible solutions, the vector objectives of the individuals are computed according to Eq.(1)-(3). Based on the vector objectives, the non-dominated individual is to be the parent. If none of the two individuals dominate the other, we choose the one according to diversity preservation, that is the density estimation of the individuals. In our approach, we use *crowding distance* defined by Deb[6] as the density estimation. The crowding distance is the average distance of the two points on either side of a particular point along each of the objectives, which indicates the density of solutions surrounding the point in the population. Then, for the two feasible solutions which do not dominate each other, the one with bigger crowding distance is chosen to be the parent.

Elitism is introduced to avoid losing good solutions due to random effects. A secondary population, archive is maintained during the evolutionary process, to which non-dominated solutions in the population are copied at each generation. An individual in the current population can be copied to archive if and only if that it is non-dominated in the current population and is not dominated by any of the individuals in the archive.

3.4 Evolutionary Operators

Sequence Exchange Crossover. According to the special chromosome representation described above, a novel sequence exchange crossover operator is introduced. Given two parents, $P1 = \{s1_1, ..., s1_{Nv}\}$ and $P2 = \{s2_1, ..., s2_{Nv}\}$, a random integer pc between 1 and N_V is generated. The sequence $s1_{pc}$ and $s2_{pc}$ are exchanged. Then the repeated targets in other subsequences and in the RP are deleted. The next step is to locate the best possible locations for the missing targets that do not exit in the chromosome or in the RP. A missing target t belonging to C_{pt} is inserted into a chromosome $C = \{s_1, ..., s_{Nv}\}$ or its associated RP using the algorithm *Insert-Target*(IT) described in Fig4.

```
Algorithm 2 Insert-Target(IT)
(1) Select a sequence $s_k$ randomly from $\{s_i | i \leq pt \wedge i \neq pc\}$
(2) Compute $\sigma_i^k$ of all the subsequences as in Eq.(14)
(3) Sort ascending the subsequences according to $\sigma_i^k$
(4) $r = 1$
(5) If $r > n$ then go to step (7)
(6) For current subsequence $s_r^k$
      $(p^*, q^*) = \arg\min_{(p,q) \in s_r^k} Cost2(t, p, q)$
      Check the feasibilities of $\tilde{s}_r^k$=insert $t$ between $p^*$ and $q^*$
      If all the constraints are satisfied, $s_r^k = \tilde{s}_r^k$, go to step (8)
      Else $r = r + 1$, go to step(5)
(7) If $r < N_v^k$, add a new subsequence $s_r^k = (0, t, 0)$ to $s_k$, go to step(8)
      Else add $t$ to RP. Exit.
(8) Exit. Output $s_k$
```

Fig. 4. Insert-Target

First, a sequence s_k is selected randomly from $\{s_i|i \leq pt \wedge i \neq pc\}$. Then, all the subsequences s_i^k in $s_k = \{s_1^k, ..., s_n^k\}$ are sorted according to the mission saturation σ_i^k defined as

$$\sigma_i^k = RT_i/TL_k \tag{14}$$

Where RT_i is the time spent for UAV v_i^k travelling along subsequence s_i^k, and TL_k is the maximum permitted travel time for UAV v_i^k. $\sigma_i^k > 1$ means that the subsequence s_i^k violates the constraint (9). The subsequence s_i^k with the smallest σ_i^k is selected as the subsequence that target t inserted to.

Then target t is inserted to the least cost position between (p^*, q^*) in s_i^k (the insertion cost is computed as Eq.15). If t cannot be inserted to any subsequences in s_k and no more UAVs in V^k is available, t is added to RP.

Mutation Operators. Three different problem specific mutation operators are put forward for MUCREA:

- *Relocate to RP*: Selecting a target in the chromosome randomly and adding it into the RP.
- *Relocate from RP*: Selecting a target in the RP randomly and inserting it to the chromosome using algorithm IT.
- *Exchange with RP*: Selecting a target in the chromosome randomly and adding it into the RP; Selecting a target in the RP randomly and inserting it to the chromosome using algorithm IT.

For an individual, the above three mutation operators are applied with mutation probabilities p_{mt}, p_{mf} and p_{me} respectively. The three different mutation operators ensure that the mutated individuals will still satisfy all the constraints defined in the problem formulation.

4 Simulation Results

Simulation experiments are carried out to verify the performance of MUCREA. In our experiments, 3 different kinds of UAVs with different capabilities are employed in conducting reconnaissance on 100 targets. The reconnaissance resolutions of the 3 kinds UAVs are $r_1 = 1m$, $r_2 = 5m$, $r_3 = 10m$ respectively. The locations and time windows of 100 targets are the same with those of the customers in the Solomon's Vehicle Routing Problems with Time Window instances[7]. The reconnaissance resolution demands of the targets are generated randomly between 1 and 30m, and reconnaissance duration time are generated randomly between 10 and 100 time units.

4.1 Good Convergence Trace

The convergence trend is a useful indication to validate the performance of any optimization algorithm. MUCRP has two objectives, CRC and NUT. We show how minimization of both objectives occurs throughout the generations. Fig.5(a) shows the reducing of NUT over generations and Fig.5(b) shows the reducing of

CRC. Although it is difficult to prove that we have found the optimal solution, it is reasonable to believe that MUCREA is able to optimize the two objectives of MUCRP concurrently and effectively.

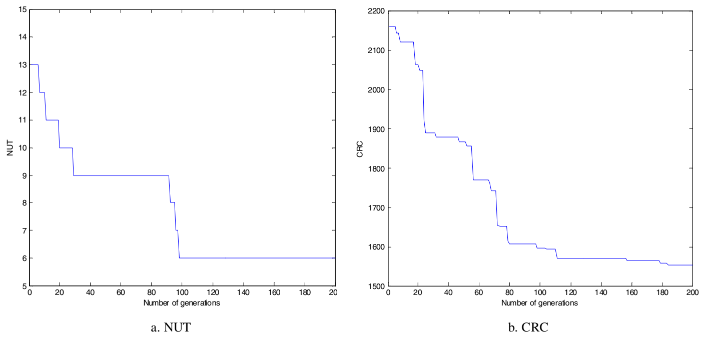
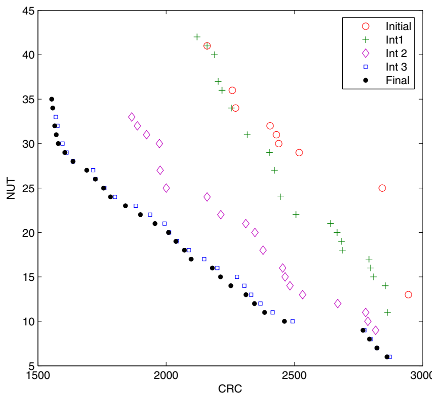


Fig. 5. NUT and CRC over generation

4.2 Good Multi-objective Optimization Effort

To verify the multi-objective optimization effort of our algorithm, the non-dominated solutions in the archive at several generations are shown in Fig.6. From the results, we can see that MUCREA moves toward Pareto optimal front through evolutions.



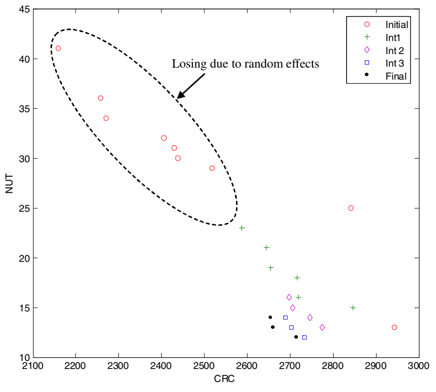


Fig. 6. Plot of elitist points

Fig. 7. Results with no elitism

The introduction of elitism is to ensure that the non-dominated solutions will not deteriorate during the evolutionary process. To prove this, an experiment is conducted in which the optimization algorithm is the same with MUCREA except that it does not adopt elitism. Fig.7 shows the results, which indicates that some non-dominated solutions in the earlier generations are lost during the evolutionary process. Comparing the results in Fig.6 and Fig.7, we can draw easily the conclusion that elitism plays an important role in the MUCREA. It prevents losing the non-dominated solutions due to random effects during the evolutionary process.

5 Conclusions

A multi-objective evolutionary algorithm is presented for Multi-UAV cooperative reconnaissance problem. MUCRP is a typical MOP and the solutions to it are Pareto optimal. The proposed algorithm MUCREA tries to find the approximate Pareto optimal set of MUCRP. The integer string representation is designed which ensures that the solution that the chromosome encodes can satisfy the reconnaissance resolution of the targets, and a construction heuristic algorithm CIFS is presented to generate initial feasible solutions for MUCREA. Then Pareto optimality based selection mechanism combined with diversity preservation is introduced to generate parent population. Archiving strategy is combined to prevent losing the non-dominated solutions during the evolutionary process due to random effects. Problem specific evolutionary operators, including a sequence exchange crossover operator and three different mutation operators are presented to ensure the feasibilities of the offsprings. The simulation results show that MUCREA has good convergence and multi-objective optimization effort.

References

1. Ryan,Joel L., Bailey,T.G., Moore,J.T., and Carlton, W.B.: Reactive Tabu Search in Unmanned Aerial Reconnaissance Simulations. Proceedings of the 1998 Winter Simulation Conference, Grand Hyatt, Washington D.C. (1998) 873-879
2. Hutchison,M.G.: A Method for Estimating Range Requirements of Tactical Reconnaissance UAVs. AIAA's 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles,Portsmouth, Virginia (2002) 120-124
3. Ousingsawat,J., and Campbell,M.E.: Establishing Trajectories for Multi-Vehicle Reconnaissance. AIAA Guidance, Navigation, and Control Conference and Exhibit, Providence, Rhode Island (2004) 1-12
4. Zitzler,E., Laumanns,M., Bleuler,S.: A Tutorial on Evolutionary Multiobjective Optimization, Metaheuristics for Multiobjective Optimisation. Lecture Notes in Economics and Mathematical Systems. Vol.535 (2004) 3-37
5. Coello, C.A.C.: Recent Trends in Evolutionary Multiobjective Optimization, Evolutionary Multiobjective Optimization: Theoretical Advances And Applications, Springer-Verlag,London (2005) 7-32
6. Deb, K., Agrawal, S., Pratap, A. etc.: Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. KanGAL report 200002, Indian Institute of Technology, Kanpur, India, 2000
7. Solomon M.M.: Algorithms for Vehicle Routing and Scheduling Problems with Time Window Constraints. Operations Reasearch. Vol.35, No.2 (1987) 254-265

Global and Local Contrast Enhancement for Image by Genetic Algorithm and Wavelet Neural Network

Changjiang Zhang and Xiaodong Wang

College of Mathematics, Physics and Information Engineering, Zhejiang Normal University, Postcode 321004 Jinhua, China
{zcj74922, wxd}@zjnu.cn

Abstract. A new contrast enhancement algorithm for image is proposed combing genetic algorithm (GA) with wavelet neural network (WNN). In-complete Beta transform (IBT) is used to obtain non-linear gray transform curve so as to enhance global contrast for an image. GA determines optimal gray transform parameters. In order to avoid the expensive time for traditional contrast enhancement algorithms, which search optimal gray transform parameters in the whole parameters space, based on gray distribution of the image, a classification criterion is proposed. Contrast type for original image is determined by the new criterion. Parameters space is given respectively according to different contrast types, which greatly shrinks parameters space. Thus searching direction of GA is guided by the new parameter space. In order to calculate IBT in the whole image, WNN is used to approximate the IBT. In order to enhance the local contrast for image, discrete stationary wavelet transform (DSWT) is used to enhance detail in an image. Having implemented DSWT to an image, detail is enhanced by a non-linear operator in three high frequency sub-bands. The coefficients in the low frequency sub-bands are set as zero. Final enhanced image is obtained by adding the global enhanced image with the local enhanced image. Experimental results show that the new algorithm is able to well enhance the global and local contrast for image.

1 Introduction

Traditional image enhancement algorithms are as following: point operators, space operators, transform operators and pseu-color enhancement [1]. S.M. Zhou gave a kind of algorithm for contrast enhancement based on fuzzy operator [2]. However, the algorithm cannot be sure to be convergent. Lots of improved histogram equalization algorithms were proposed to enhance contrast for kinds of images [3]. The visual quality cannot be improved greatly with above algorithms. Tubbs gave a simple gray transform algorithm to enhance contrast for images [4]. However, the computation burden of the algorithm was large. Existing many enhancement algorithms' intelligence and adaptability are worse and much artificial interference is required.

To solve above problems, a new algorithm employing IBT, GA and WNN is proposed. To improve optimization speed and intelligence of algorithm, a new criterion is proposed based on gray level histogram. Contrast type for original image is determined employing the new criterion. Contrast for original images are classified into seven types: particular dark (PD), medium dark (MD), medium dark slightly (MDS), medium bright slightly (MBS), medium bright (MB), particular bright (PB) and good gray level distribution (GGLD). IBT operator transforms original image to a new space. A certain objective function is used to optimize non-linear transform parameters. GA is used to determine the optimal non-linear transform parameters. In order to reduce the computation burden for calculating IBT, a new kind of WNN is proposed to approximate the IBT in the whole image.

2 IBT

The incomplete Beta function can be written as following:

$$F(u) = B^{-1}(\alpha,\beta)\times\int_0^u t^{\alpha-1}(1-t)^{\beta-1}dt, \quad 0<\alpha,\beta<10 \tag{1}$$

All the gray levels of original image have to be unitary before implementing IBT. All the gray levels of enhanced image have to be inverse-unitary after implementing IBT.

3 Contrast Classification for Image Based on Histogram

Based on gray level histogram, contrast classification criterion can be described in Fig.1:

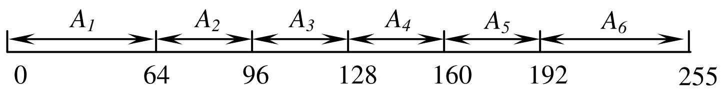


Fig. 1. Image classification sketch map based on gray level histogram

Given that original image has 255 gray levels, the whole gray level space is divided into six sub-spaces: $A_1, A_2, A_3, A_4, A_5, A_6$ .Where A_i (i=1, 2, ···, 6) is the number of all pixels which distribute in the ith sub-space. Let,

$$M = \max_{i=1}^{6} A_i, \quad B_1 = \sum_{k=2}^{6} A_i, \quad B_2 = \sum_{k=2}^{5} A_i, \quad B_3 = \sum_{k=1}^{5} A_i,$$

$$B_4 = A_1 + A_6, \quad B_5 = A_2 + A_3, \quad B_6 = A_4 + A_5,$$

Following classification criterion can be obtained:

$$if\ (M = A_1)\ \&\ (A_1 > B_1)$$

```
        Image is PB;
elseif $(B_2 > B_4)$ & $(B_5 > B_6)$ & $(B_5 > A_1)$ & $(B_5 > A_6)$ & $(A_2 > A_3)$
        Image is MD;
elseif $(B_2 > B_4)$ & $(B_5 > B_6)$ & $(B_5 > A_1)$ & $(B_5 > A_6)$ & $(A_2 < A_3)$
        Image is MDS;
elseif $(B_2 > B_4)$ & $(B_5 < B_6)$ & $(A_1 < B_6)$ & $(A_6 < B_6)$ & $(A_4 > A_5)$
        Image is MBS;
elseif $(B_2 > B_4)$ & $(B_5 < B_6)$ & $(A_1 < B_6)$ & $(A_6 < B_6)$ & $(A_4 < A_5)$
        Image is MB;
  elseif $(M = A_6)$ & $(A_6 > B_3)$
        Image is PB;
  else
        Image is GGLD;
  end
```

Where symbol & represents logic "and" operator.

4 Transform Parameters Optimization by GA

GA can find the near-global optimal solutions in a large solution space quickly. GA has been used extensively in many application areas, such as image processing, pattern recognition, feature selection, and machine learning. We will employ the GA to optimize transform parameters [5]. If the algorithm is used directly to enhance the global contrast for an image, it will result in large computation burden. The range of α and β can be determined by Tab.1 so as to solve above problems.

Table 1. Range of α and β

Parameter	PD	MD	MDS	MBS	MB	PB
α	[0 , 2]	[0 , 2]	[0 , 2]	[1 , 3]	[1 , 4]	[7 , 9]
β	[7 , 9]	[1 , 4]	[1 , 3]	[0 , 2]	[0 , 2]	[0 , 2]

Let $\mathbf{x} = (\alpha, \beta)$, $F(\mathbf{x})$ is the fitness function for GA, where $a_i < \alpha, \beta < b_i$ （$i = 1,2$）, a_i and b_i （$i = 1,2$）can be determined by Tab.1.

GA consists of the following steps (procedures):

A. Initialization. An initial population size P for a genetic algorithm should be generated, which may be user-specified or randomly selected.

B. Evaluation. Fitness of each chromosome within a population will be evaluated. The fitness function maps the binary bit strings into real numbers. The larger (or smaller) the fitness function value is, the better the solution (or chromosome) will be.

C. Selection. Based on the fitness values of bit strings in the current population, pairs of "parent" bit strings are selected which undergo the genetic operation "crossover" to produce pairs of "offspring" bit strings forming the next generation. The probability of selection of a particular chromosome is (directly or inversely) proportional to the fitness function value.

D. Crossover. Crossover exchanges information between two parent bit strings and generates two offspring bit strings for the next population. Crossover is realized by cutting individually the two parent bit strings into two or more bit string segments and then combining the two bit string segments undergoing crossing over to generate the two corresponding offspring bit strings. Crossover can produce off-springs which are radically different from their parents.

E. Mutation. Mutation is to perform random alternation on bit strings by some operations, such as bit shifting, inversion, rotation, etc. which will create new offspring bit strings radically different from those generated by the reproduction and crossover operations. Mutation can extend the scope of the solution space and reduce the possibility of falling into local extremes. In general, the probability of applying mutation is very low.

F. Stopping criterion. There exists no general stopping criterion. The following two stopping criteria are usually employed:

(1) No further improvement in the fitness function value of the best bit string is observed for a certain number of iterations or
(2) A predefined number of iterations have been reached. Finally, the best bit string obtained is determined as the global optimal solution.

Before running GA, several issues must be considered as follows.

A. System parameters. In this study, the population size is set to 20 and the initial population will contain 20 chromosomes (binary bit strings), which are randomly selected. The maximum number of iterations (generations) of GA is set to 50 or 100.

B. Fitness function. The fitness (objective) function is used to evaluate the goodness of a chromosome (solution). In this study, the fitness function is formed by Equation (2)[1]:

$$F_{ctr} = \left[\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N} g'(i,j)\right]^2 - \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N} g'^2(i,j) + C \tag{2}$$

Where M, N show width and height of original image. $g'(i,j)$ Shows gray level at (i,j) in enhanced image. In order to be sure the fitness function F_{ctr} is positive, we add a constant C to F_{ctr} and here $C = 100$. Less F_{ctr} is, more well proportioned the distribution of image gray level is.

C. Genetic operations. For GA, the three genetic operations, namely, reproduction, crossover, and mutation, will be implemented. In this study, a multi-point crossover is employed. For the multi-point crossover, the crossover-point positions of the bit string segments of pair-wise bit strings are randomly selected. Mutation is carried out by performing the bit inversion operation on some randomly selected positions of the parent bit strings and the probability of applying mutation, P_m, is set to 0.001.

The GA will be iteratively performed on an input degraded image until a stopping criterion is satisfied. The stopping criterion is the number of iterations is larger than another threshold (here they are set as 50, 50 and 80 respectively in order to enhance three images). Then the chromosome (the solution) with the smallest fitness function value, i.e., the optimal set of IBT for the input degraded image, is determined. Using the optimal set of IBT enhances the degraded image. Totally there are two parameters in the set of IBT (α and β). The two parameters will form a solution space for finding the optimal set of IBT for image enhancement. Applying GA, the total two parameters will form a chromosome (solution) represented as a binary bit string, in which each parameter is described by 20 bits. We will employ the GA to optimize continuous variables [5]. If the algorithm is used directly to enhance image contrast, it will result in large computation cost.

5 IBT Calculation with WNN

IBT is calculated pixel-to-pixel. Operation burden is very large when pixels in original image are large. Different IBT have to be calculated to different α and β. Different IBT need to be calculated one time in every iterative step during optimization. To improve operation speed during the whole optimization, a new kind of wavelet neural network is proposed.

Let $f(x) \in L^2(R^n)$, WNN can be described approximately as follows:

$$Wf(x) = \sum_{i=1}^{N} w_i \psi[(a_i x - \tau_i)] \tag{3}$$

where τ_i is translation factor, a_i is scale factor, $Wf(x)$ shows the output of WNN. The translation factor, scale factor and wavelet basis function, which are on the same line, is called wavelet unit.

Parameters to be estimated are w_i, a_i, τ_i, $i = 1,2,\cdots,N$ (where N is the number of wavelet unit.

Let

$$\boldsymbol{\theta} = [a_1, a_2, \cdots, a_N, \tau_1, \tau_2, \cdots, \tau_N] \tag{4}$$

$$z_i(x) = \psi(a_i x - \tau_i) \tag{5}$$

Equation (6) is rewritten as:

$$Wf(x) = \sum_{i=1}^{N} w_i z_i(x) = \boldsymbol{\varphi}_t^{\tau} \mathbf{W}_t \tag{6}$$

Definition

$$\boldsymbol{\varphi}_t = [z_1(t), z_2(t), \cdots, z_N(t)]^T \tag{7}$$

$$\mathbf{W}_t = [w_1(t), w_2(t), \cdots, w_N(t)]^T \tag{8}$$

Where $z_i(t)$ and $w_i(t)$ show the output of ith wavelet unit at time t and corresponding to weight respectively. T shows transpose of matrix. "Forgetting factor" algorithm can be used to estimate $\mathbf{W}$ [6]. Parameter matrix $\boldsymbol{\theta}$ can be estimated by iterative prediction error algorithm [6]. Weight, translation factors and scale factors are trained iteratively and mutually with above two algorithms.

IBT can be calculated by the above WNN. Parameters α, β, g are input to trained WNN and output g' for IBT is obtained directly. 100000 points are selected as sample sets. Parameter α and β, which are between 1 and 10, are divided into 10 parts at the same interval. Parameter x, which is between 0 and 1, is divided into 1000 parts at the same interval. 25 wavelet units are selected. The dimension number of input layer and output layer are determined according to the dimension number of input samples and output samples. Mexican hat wavelet is selected as mother wavelet:

$$\psi(x) = (1 - x^2)e^{-x^2/2} \tag{9}$$

The "forgetting factor" $\alpha = 0.97$ in the WNN. Mean square error is selected as error index and set as 0.00001.

6 Local Contrast Enhancement by Non-linear Operator

Based on DSWT, a non-linear enhancement operator, which was proposed by A. Laine in 1994, is employed to enhance the local contrast for image [7]. For convenience, let us define following transform function to enhance the high frequency sub-band images in each decomposition level respectively:

$$g[i,j] = MAG\{f[i,j]\} \tag{10}$$

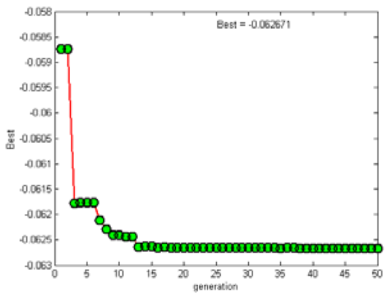
Where $g[i,j]$ is sub-band image enhanced, $f[i,j]$ is original sub-band image to be enhanced, MAG is non-linear enhancement operator, M, N is width and height of image respectively. Let $f_s^r[i,j]$ is the gray values of pixels in the $\mathbf{r}th$ sub-band in the sth decomposition level, where $s = 1,2,\cdots,L$; $r = 1,2,3$. $maxf_s^r$ is the maximum of gray value of all pixels in $f_s^r[i,j]$. $f_s^r[i,j]$ can be mapped from $[-maxf_s^r, maxf_s^r]$ to $[-1,1]$. Thus the dynamic range of a,b,c can be set respectively. The contrast enhancement approach can be described by:

$$g_s^r[i,j] = \begin{cases} f_s^r[i,j], & \left|f_s^r[i,j]\right| < T_s^r \\ a\cdot \max f_s^r\{\text{sigm}[c(y_s^r[i,j]-b)]- & \\ \text{sigm}[-c(y_s^r[i,j]+b)]\}, & \left|f_s^r[i,j]\right| \ge T_s^r \end{cases} \tag{11}$$

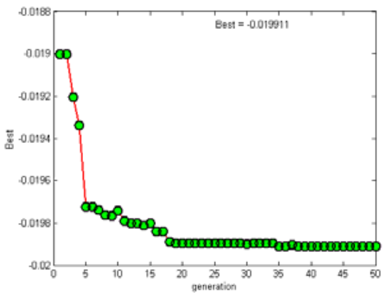
$$y_s^r[i,j] = f_s^r[i,j] / maxf_s^r \tag{12}$$

7 Experimental Results

We use two images (plane and tank) to prove the efficiency of our algorithm. Fig.2 is relationship curve between number of evolution generation and Best, where $Best = F_{ctr} - C$. Fig.3 (a) represents transform curve obtained by GA, where $\alpha = 1.9846,\ \beta = 2.3221$. Fig.3 (b) is a non-linear gain curve used to obtain the detail image, where $b = 0.15$, $c = 20$. Fig.3 is used to enhance the global contrast and obtain the detail image in Fig.4 (a). Two traditional contrast enhancement algorithms are compared with the new algorithm. They are histogram equalization (HE) and unsharpened mask algorithm (USM) respectively. Fig.4 (b)-(d) are enhanced images by using HE, USM and the new algorithm respectively. Although the global contrast is good when HE is used to enhance Fig.4 (a), background clutter is also enlarged while some detail information also lost, such as detail symbol in Fig. (b). Although local detail is well enhanced when USM is used to enhance Fig.4 (a), the global contrast is bad in Fig.4 (c). The new algorithm can well enhance the global and local contrast in Fig.4 (d), and the background clutter is also well suppressed. It is very obvious that the new algorithm is better in visual quality than HE and USM.

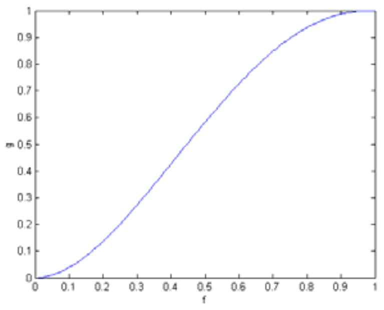


(a) Plane

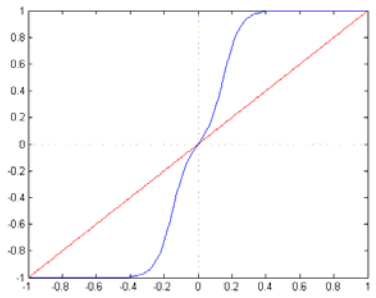


(b) Tank

Fig. 2. Relationship curve between generation and best individual fitness

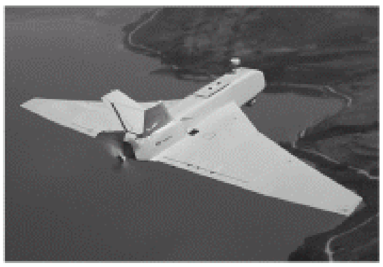


(a) IBT curve

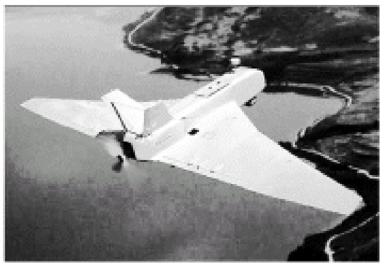


(b) Non-linear gain curve

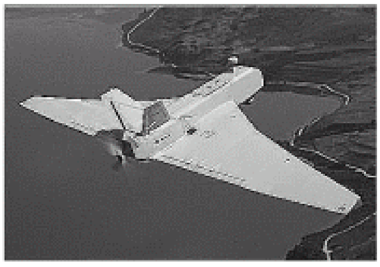
Fig. 3. IBT curve and non-linear gain curve



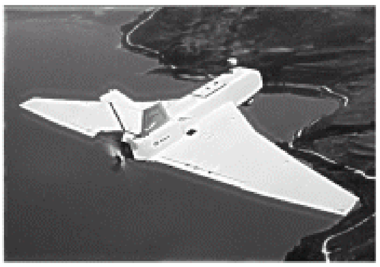
(a) Plane image



(b) Enhancing by HE



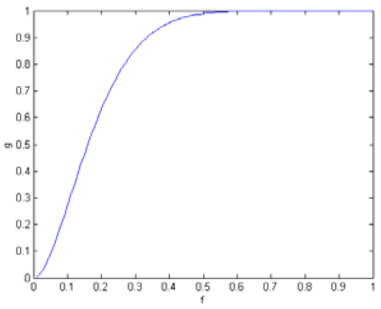
(c) Enhancing by USM



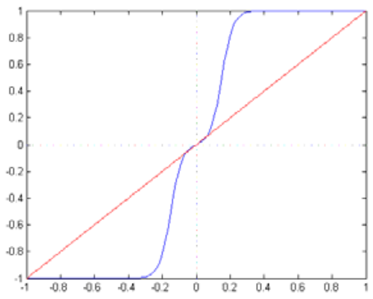
(d) Enhancing by new method

Fig. 4. Enhancement by three methods

Fig.5 (a) represents transform curve obtained by GA, where $\alpha = 1.9928$, $\beta = 8.7512$. Fig.5 (b) is a non-linear gain curve used to obtain the detail image, where $b = 0.15$, $c = 30$. Fig.5 is used to enhance the global contrast and obtain the detail image in Fig.6 (a). Fig.6 (a) represents an infrared tank image. Fig.6 (b)-(d) are enhanced images using HE, USM and the new algorithm respectively. It is very obvious that the new algorithm is better in visual quality than HE and USM.

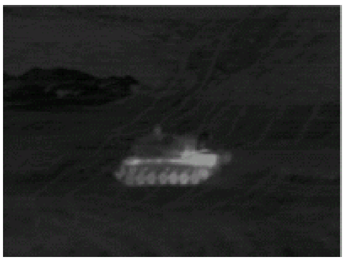


(a) IBT curve

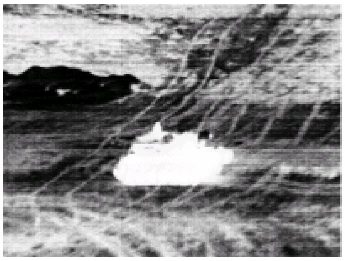


(b) Non-linear gain curve

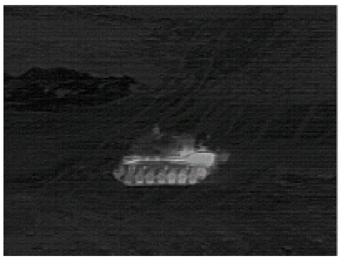
Fig. 5. IBT curve and non-linear gain curve



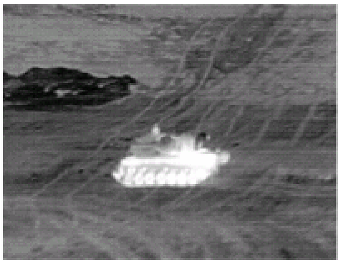
(a) Infrared tank image



(b) Enhancing by HE



(c) Enhancing by USM



(d) Enhancing by new method

Fig. 6. Enhancement by three methods

From the experimental results, it is very obvious that the new algorithm is better than HE and USM. Most of detail is lost in infrared tank when HE is used to enhance contrast in Fig.6 (b). The global contrast is bad when USM is used to enhance contrast in Fig.6 (c). The overall contrast quality is better than HE and USM when the new algorithm is used to enhance contrast in Fig.6 (d).

In order to show the efficiency of the new algorithm, Equation (2) is used to evaluate the enahnced image quality. Evaluation results are list in Tab.2. From Tab.2, it is

Table 2. Contrast evaluation of the three algorithms

Algorithms	Plane	Tank
USM	5.6011e+003	8.0439e+002
HE	5.6522e+003	5.6183e+003
NEW	-6.2671e-002	-1.9911e-002

obvious that the fitness function value of the new algorithm is the smallest, so the visual quality is better than HE and USM. This can draw the same conclusion as in Fig.4 and Fig.6.

8 Conclusion

Experimental results show that the new algorithm can adaptively enhance the global and local contrast for an image. The new algorithm is better than HE and USM in visual quality.

References

1. Azriel Rosenfield, Avinash C K. Digital Picture Processing. New York: Academic Press, (1982)
2. Shang-Ming Zhou; Qiang Gan. A new fuzzy relaxation algorithm for image contrast enhancement. Proceedings of the 3rd International Symposium on Image and Signal Processing and Analysis, **1** (2003) 11-16
3. Soong-Der Chen, Ramli, A.R. Minimum mean brightness error bi-histogram equalization in contrast enhancement. IEEE Transactions on Consumer Electronics, **48** (2003) 1201-1207
4. Tubbs J D. A note on parametric image enhancement. Pattern Recognition, **30** (1997) 616-621
5. Ming-Suen Shyu and Jin-Jang Leou. A genetic algorithm approach to color image enhancement. Pattern Recognition, **31** (1998) 871-880
6. Meiling Wang, Changjiang Zhang, Mengyin Fu. Simulation study of a kind of wavelet neural network algorithm used in approaching non-linear functions. Journal of Beijing Institute of Technology, **22** (2002) 274-278
7. A. Laine, S. Schuler: Hexagonal wavelet processing of digital mammography. In Medical Imaging 1993, Part of SPIE's Thematic Applied Science and Engineering Series, 1993: 1345-1348

A Novel Constrained Genetic Algorithm for the Optimization of Active Bar Placement and Feedback Gains in Intelligent Truss Structures

Wenying Chen[1,2], Shaoze Yan[1], Keyun Wang[2], and Fulei Chu[1]

[1] Department of Precision Instruments and Mechanology, Tsinghua University, Beijing 100084, P.R. China
cwy04@mails.tsinghua.edu.cn,
{chufl, yansz}@mail.tsinghua.edu.cn
[2] Beijing Special Vehicle Research Institution, Beijing 100072, P.R. China
cwy04@mails.tsinghua.edu.cn

Abstract. In this paper, a novel constrained genetic algorithm is proposed and also successfully applied to the optimization of the active bar placement and feedback gains in intelligent truss structures. Based on the maximization of energy dissipation due to active control action, a mathematical model with constrains is initially developed. Then, according to the characteristics of the optimal problem, a new problem-specific encoding scheme, some special "genetic" operators and a problem-dependent repair algorithm are proposed and discussed. Numerical example of a 72-bar space intelligent truss structure is presented to demonstrate the rationality and validity of this methodology, and some useful conclusions are obtained.

1 Introduction

Intelligent truss structure is a self-adaptive structure which is widely used in astronomical installations. Due to the facts that truss-type astronomical structures have high flexibility, and they must maintain micron-level geometric-shape accuracy when subjected to static, thermal and dynamic disturbances, active control using piezoelectric active bar has become one of the important and effective strategies to suppress the structure vibration. In the design of vibration control system for intelligent truss structures, there are two sets of design parameters: feedback gains and positions of active/passive members. The feedback gains are continuous variables, while positions of active/passive members only take discrete values. Thus, the resulting optimization problem has discrete-continuous design variables, which is difficult to solve through the traditional optimal algorithms. In the past, structural and control design were considered separately, as in [1], [2] and [3]. At present, simultaneous optimization method is a hot topic studied in the design of intelligent structures in order to enhance the overall performance of the controlled system. Ref. [4] presented an optimal design method for placement and gains of actuators and sensors in output feedback control systems. Their new contribution was the derivation of analytical expressions for the gradients of the performance function. The quadratic performance function was minimized using nonlinear programming. Ref. [5] developed a method of concurrent design of both placements and gains. This method was based on the maximization of dissipation energy due to control action. The solution to the optimization problem was obtained by a gradient-based nonlinear programming technique. Recently, genetic algorithms (GAs) have proved to be effective tools for solving this kind of complex optimization problems. Ref. [6] presented a float-encoded genetic algorithm and applied it to an integrated determination of actuator and sensor locations and feedback gains for a cantilever beam. Ref. [7] developed an integrated optimization method of structure and control for intelligent truss structures based on the approach of independent modal space control. In this paper, the genetic algorithm was used to solve the optimization problem. The design variable set, including structural sizing variables, control parameters and actuator/sensor placement, are encoded as a binary code string. By the Penalty Function method, a constrained optimization problem was transformed to an unconstrained optimization problem. Ref. [8] presented an integrated optimal design of vibration control system for smart beams using genetic algorithms, in which the size of piezoelectric patches was encoded as integer, while the placement of piezoelectric patches and the feedback control gains were encoded as real numbers. In addition, by introducing the exterior penalty functions, a constrained optimal problem was transformed into an unconstrained problem.

In this paper, the simultaneous optimization of the active vibration control system for intelligent truss structures, including feedback gains and positions of active/passive members, has been formulated. Based on the maximization of energy dissipation due to control action, the performance function is initially developed for the vibration suppression. A novel constrained genetic algorithm has been developed to solve the simultaneous optimization problem with discrete and continuous design variables. The results of a numerical example show that the simultaneous optimization of structure and control is feasible and effective. In addition, the proposed genetic algorithm with the problem-specific chromosome representation, special "genetic" operators and a problem-dependent repair algorithm presents the advantages of globalization and robustness of GAs.

2 Mathematical Modeling and Optimal Criterion

Following the finite element formulation described in [1], the equation of motion for an intelligent truss can be expressed as

$$[M]\{\ddot{u}(t)\}+[C]\{\dot{u}(t)\}+[K]\{u(t)\}=\{F_e(t)\}+\{B\}\{F_p(t)\}, \tag{1}$$

where $[M]$,$[C]$ and $[K]$ are the $n\times n$ mass, damping and stiffness matrices, respectively. $\{u(t)\}$,$\{\dot{u}(t)\}$ and $\{\ddot{u}(t)\}$ are structure displacement, velocity and acceleration vectors, respectively. The term $\{F_e(t)\}$ is $n\times 1$ external nodal force vector. The term $\{F_p(t)\}$ represents the $m\times 1$ control force vector and m is the number of active members. The matrix $[B]$ consists of the active bar's direction cosines.

Assuming that the system is of proportional damping and no external force, will not affect the generality of the results. With the modal approximation $\{u(t)\} = [\phi]\{q(t)\}$, the equation of motion takes the form

$$\{\ddot{q}\} + [D]\{\dot{q}\} + [\Omega]\{q\} = [\phi]^T [B]\{F_p\}, \tag{2}$$

where $[D] = diag\left[2\xi_j w_j\right]$, $[\Omega] = diag\left[w_j^2\right]$, $(j = 1, 2, \ldots n)$, $[\phi] = [\phi_1 \quad \phi_2 \quad \ldots \quad \phi_n]$, w_j and ξ_j are j th order inherent frequency and modal damping ratio of structure, respectively.

Considering direct output velocity feedback control, the output vector and control force vector can be expressed as

$$\begin{aligned}\{y\} &= [C]\{\dot{u}\} = [C][\phi]\{\dot{q}\} \\ \{F_p\} &= -[G]\{y\} = -[G][C][\phi]\{\dot{q}\}\end{aligned}, \tag{3}$$

where $[C]$ and $[G]$ represents the output and control matrices respectively. Since each active member can be considered a collocated actuator/sensor pair, the output matrix $[C]$ is the transpose of the input matrix $[B]$, i.e., $[C] = [B]^T$.

Substituting (3) into (2) yields the corresponding equation of the closed-loop system

$$\{\ddot{q}\} + \left([D] + [\phi]^T [B][G][B]^T [\phi]\right)\{\dot{q}\} + [\Omega]\{q\} = \{0\}. \tag{4}$$

In the state-space representation, the equation of motion for the closed-loop system becomes

$$\{\dot{x}\} = [A]\{x\}, \tag{5}$$

where $\{x\} = [q \quad \dot{q}]^T$ and

$$[A] = \begin{bmatrix} 0 & [I] \\ -[\Omega] & -\left([D] + [\phi]^T [B][G][B]^T [\phi]\right) \end{bmatrix}.$$

Based on the maximization of dissipation energy due to the active control action, the optimization criterion can be expressed as

$$J = \int_0^\infty \{\dot{q}\}^T [\phi]^T [B][G][B]^T [\phi]\{\dot{q}\} dt, \tag{6}$$

When the solution of (5), $\{x\} = \exp([A]t)\{x_0\}$, is used, The Eq. (6) becomes

$$J = \{x(0)\}^T [P]\{x(0)\}, \tag{7}$$

where $\{x(0)\} = \begin{Bmatrix} q_0 \\ \dot{q}_0 \end{Bmatrix}$ is the initial state and $[P]$ is the solution of the following Lyapunov equation:

$$[A]^T [P] + [P][A] = [Q], \tag{8}$$

where

$$[Q] = \begin{bmatrix} [\Omega] & 0 \\ 0 & [I] \end{bmatrix}.$$

When the method described in [9] is used, the performance function can be written as

$$J = tr[P]. \tag{9}$$

Thus, the problem can be expressed as a nonlinear optimization problem with constraints:

$$\text{Maximize: } J([B],[G]) , \tag{10}$$

Subject to the constraints:

$$[B] \subset [B]^*, [G] \subset [G]^*. \tag{11}$$

In this model, $[B]$ and $[G]$ are design variables. $[B]^*$ is the bounds of $[B]$, and $[G]^*$ is the upper bounds of the feedback gains.

3 Realization of the Constrained Genetic Algorithm

The GA is a kind of stochastic optimization method which is originally derived from the Darwinian evolutionary principle of "survival-of-the-fittest". Its basic procedures include encoding, selection, crossover and mutation. In this paper, to solve the simultaneous optimization problem of the placement of active/passive bar and feedback gains in intelligent truss structures, a novel constrained genetic algorithm, with a novel problem-specific encoding scheme, some special "genetic" operators and a problem-dependent repair algorithm, is initially presented.

3.1 Encoding

Encoding is the genetic representation of design variables. The genetic representation is called as a chromosome in GA, representing a design parameter set in an organized manner. The design of the encoding method and the related problems are discussed in detail below.

3.1.1 Code Design

In the vibration control system of intelligent truss structures, there are two sets of design parameters: feedback gains and positions of active/passive members. The feedback gains are continuous variables, while positions of active/passive members only take discrete values. Considering each design variable has its own physical meaning, a new combined encoding method is presented in this paper. Its typical chromosome representation can be illustrated in Fig.1, where feedback gain variables are encoded with floating-point numbers, while the locations of active/passive bars are encoded as binary-code string, in which 1 indicates an active bar and 0 a passive bar.

1	……	1	……	0	G_1	…	G_n

Fig. 1. A typical chromosome representation

Obviously, this binary-float-encoded chromosome representation is consistent to the description of the design parameters in the problem space. Therefore it is not only easily comprehended but also much easier to design special "genetic" operators for handling the nontrivial constraints. In addition, this encoding method shortens the length of chromosome string , and avoids the difficult encoding and decoding of the binary data that is required in standard GAs, thus decreasing the computational burden and improving the computation precision.

3.1.2 Constraint Problem

Considering the constraints of design variables presented above, some special "genetic" operators are carefully designed, which can guarantee to keep all the individuals always within the feasible space. These special "genetic" operators will be discussed in detail.

In order to simplify evolution process, one chromosome is divided into two parts, i.e., binary-code part and floating-point-number-code part. In addition, different genetic operators, i.e., mutation and crossover, are employed for the binary- and floating-point parts of one chromosome, respectively. The non-uniform mutation and the whole arithmetical crossover are adopted for the floating-point number part in order to keep evolutionary individuals always in feasible space. The non-uniform mutation is a special dynamic mutation operator, which can improve single-element tuning and reduce the disadvantage of random mutation in the floating point implementation, as in [10].

The whole arithmetical crossover produces two complimentary linear combinations of the parents as

$$\begin{aligned} \tilde{v}_k &= r v_k + (1-r) w_k \\ \tilde{w}_k &= (1-r) v_k + r w_k \end{aligned} . \tag{12}$$

where r is a random number, v_k and w_k are the parents, and $\tilde{v}_k$ and $\tilde{w}_k$ are the offspring.

The mutation and crossover for the binary-code part are the same as the standard binary-encoded GAs. Besides, in order to satisfy the placement constraint, i.e., the number of active members is unchangeable, a specific repair algorithm has been designed for this particular optimal problem. Its characteristic is by a forced mutation technique to repair an infeasible individual as a feasible individual.

3.2 Optimization Procedure

The procedure flow of the optimization of the novel problem-dependent GA is described as follows:

Step 1: Specify GA parameters, including the population size $Psize$, the crossover probability P_c and the mutation probability P_m;

Step 2: Randomly generate the initial population in the feasible space;

Step 3: Calculate the fitness value for the population;

In this step, the N_m individuals ahead are selected into the "Evolutive Memory Pool" according to the fitness value rank. The number N_m is a predetermined value.

Step 4: Generate the next generation offspring population;

Firstly, allow the N_m individuals in the "Evolutive Memory Pool" to descend directly into the next generation. The other N_c individuals are produced using the roulette selection. Here $N_c = N - N_m$.

Step 5: Execute the genetic operators, i.e., crossover and mutation;

Step 6: Terminate the optimal process according to the termination criterion.

If the termination criterion is met, the optimal process is stopped. Otherwise, the steps 3, 4 and 5 are repeated until the termination criterion is met. The termination criterion is the maximal iteration number, which is a predetermined value. Besides, it has to be pointed out that the repair algorithm is always used to repair infeasible individuals as feasible individuals through the evolution process.

4 Numerical Example

In this section, the integrate optimization of active bar placement and feedback gains for a 72-bar space intelligent truss with two active bars is used to evaluate the feasibility and effectiveness of the proposed optimal scheme, as shown in Fig. 2. The nodal masses of passive and active members are $0.29kg$ and $0.5kg$, respectively. The passive bar is aluminum alloy tube with diameter $20mm$ and thickness $1.5mm$. The Young's module of aluminum alloy is $7.0\times10^{10} N/m^2$, and density $2.74\times10^3 kg/m^3$. The lengths of two connected bars for piezoelectric active member are the same. The mass density and Young's module of piezoelectric stacks are $7.5\times10^3 kg/m^3$ and $6.3\times10^{10} N/m^2$, respectively. The equivalent stress coefficient e_{33} is $18.62 C/m^2$. The piezoelectric stack is made of circular piezoelectric patches with the diameter of $10mm$ and its corresponding length is $120mm$.

The optimization problem as previously formulated is a nonlinear optimization with constraints. In this case, besides the geometric constraints, a simple bound constraint is imposed on the feedback control gain matrix G; i.e., $0 \le G_{ij} \le 150$. The Population size, the crossover rate and the mutation rate are 100, 0.8 and 0.2, respectively.

After the stopping criterion is achieved, the most optimal individual is obtained. The optimal positions of active bars is (1), (16), the Feedback gains are 147.85 and 149.88. The curve of the maximal fitness with respect to the number of iteration, shown in Fig.3, demonstrates that the constrained genetic algorithm proposed in this paper has good global convergence performance.

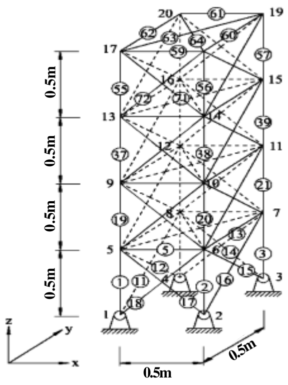


Fig. 2. The 72-bar space intelligent truss

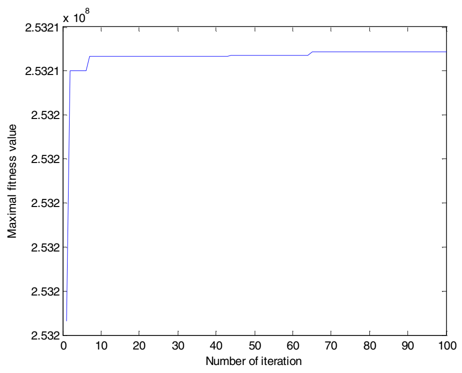


Fig. 3. Curve of the maximal fitness with respect to the number of iteration

5 Conclusions

In this paper, the integrated optimization of the active vibration control system for the intelligent truss structures, including feedback gains and positions of active/passive members, has been formulated. Based on the maximization of energy dissipation due to control action, the performance function was developed. A novel constrained genetic algorithm has been proposed to solve the simultaneous optimization problem with discrete and continuous design variables. The results of a numerical example show that the simultaneous optimization of structure and control is feasible and effective. Moreover, the proposed genetic algorithm with the problem-specific chromosome representation, special "genetic" operators and problem-dependent repair algorithm can greatly decrease computation burden and improve computation precision. In conclusion, the problem-specific optimal idea proposed in this paper is very heuristic for solving the complex optimization problem.

Acknowledgments

This work is supported by Natural Science Foundation of China (Grant No. 50425516) and the Trans-Century Training Programme Foundation for the Talents by the Ministry of Education.

References

1. Chen G-S, Bruno RJ, Salama M.: Optimal Placement of Active/Passive Members in Structures Using Simulated Annealing. AIAA Journal, 8(1991) 1327–34
2. Shaoze Yan, Kai Zheng, Qiang Zhao, and Lin Zhang: Optimal Placement of Active Members for Truss Structure Using Genetic Algorithm. Lecture Notes in Computer Science, 3645(2005) 386-395
3. Lammering R, Jia J, Rogers CA: Optimal Placement of Piezoelectric Actuators in Adaptive Truss Structures. Journal of Sound and Vibration, 3(1994) 67–85
4. K.Xu, P.Warnitchai and T.Igusa: Optimal Placement and Gains of Sensors and Actuators for Feedback Control. Journal of Guidance, Control, and Dynamics, 9-10(1994) 929-934
5. Schulz G, Heimbold G.: Dislocated Actuator/Sensor Positioning and Feedback Design for Flexible Structures. Journal of Guidance, Control, and Dynamics, 9-10 (1983) 361-366
6. Hongwei Zhang, Barry Lennox, Peter R Goulding and et al: A Float-encoded Genetic Algorithm Technique for Integrated Optimization of Piezoelectric Actuator and Sensor Placement and Feedback Gains. Smart Materials and Structures, 8(2000) 552-557
7. B.Xu, J.S.Jiang: Integrated Optimization of Structure and Control for Piezoelectric Intelligent Trusses with Uncertain Placement of Actuators and Sensors. Computational Mechanics, 1(2004) 406-412
8. Yaowen Yang, Zhanli Jin, Chee Kiong Soh: Integrated Optimal Design of Vibration Control System for Smart Beams Using Genetic Algorithms. Journal of Sound and Vibration, 4(2005)1293-1307
9. Makola M.Abdullah: Optimal Location and Gains of Feedback Controllers at Discrete Locations. AIAA Journal, 11(1998) 2109-2116
10. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. 3rd edn. Springer-Verlag, Berlin Heidelberg New York (1996)

A Double-Stage Genetic Optimization Algorithm for Portfolio Selection

Kin Keung Lai[1,2], Lean Yu[2,3], Shouyang Wang[1,3], and Chengxiong Zhou[3]

[1] College of Business Administration, Hunan University, Changsha 410082, China
[2] Department of Management Sciences, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong
{mskklai, msyulean}@cityu.edu.hk
[3] Institute of Systems Science, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100080, China
{yulean, sywang}@amss.ac.cn

Abstract. In this study, a double-stage genetic optimization algorithm is proposed for portfolio selection. In the first stage, a genetic algorithm is used to identify good quality assets in terms of asset ranking. In the second stage, investment allocation in the selected good quality assets is optimized using a genetic algorithm based on Markowitz's theory. Through the two-stage genetic optimization process, an optimal portfolio can be determined. Experimental results reveal that the proposed double-stage genetic optimization algorithm for portfolio selection provides a very feasible and useful tool to assist the investors in planning their investment strategy and constructing their portfolio.

1 Introduction

In modern portfolio theory, the mean-variance model originally introduced by Markowitz [1] has been playing an important and critical role so far. Since Markowitz's pioneering work [1] was published, the mean-variance model has revolutionized the way people think about portfolio of assets, and has gained widespread acceptance as a practical tool for portfolio optimization. But Markowitz's portfolio theory only provides a solution to asset allocation among the pre-determined assets. In the investment markets, several hundred of different assets, such as stocks, bonds, foreign exchanges, options, commodities, real estates and future contracts, are available for trading. The qualities of these assets vary from very good to extremely poor. Usually, investors are difficult to find out those good quality assets because of information asymmetry and asset price fluctuations. Therefore, it is not wise to use portfolio theory blindly for optimizing asset allocation among some low quality assets. The suitable way of constructing a portfolio is to select some good quality assets first and then to optimize asset allocation using portfolio theory.

But an obvious challenge is how to select and optimize some good assets. With focus on the business computing, applying artificial intelligence to portfolio selection and optimization is one good way to meet the challenge. Some studies have been presented to solve asset selection problem. Levin [2] applied artificial neural network (ANN) to select valuable stocks. Chu [3] used fuzzy multiple attribute decision analysis to select stocks for portfolio. Similarly, Zargham [4] used a fuzzy rule-based system to evaluate the listed stocks and realize stock selection. Recently, Fan [5] utilized support vector machine (SVM) to train universal feedforward neural networks (FNN) to perform stock selection. For portfolio optimization, Berger [6] applied tabu search to find the optimal asset allocation. While some researchers, such as Casas [7] and Chapados [8], trained neural networks to predict asset behavior and used the neural network to make the asset allocation decisions. In addition, Mulvey [9] applied dynamic programming to construct a multi-stage stochastic model for solving asset allocation problem.

However, these approaches have some drawbacks in solving the portfolio selection problem. For example, fuzzy approach [3-4] usually lack learning ability, while neural network approach [2, 5, 7-8] has overfitting problem and is often easy to trap into local minima. In order to overcome these shortcomings, we use two-stage genetic algorithm (GA) to solve the portfolio selection and optimization problem. Comparing with tabu search [6], GA is less problem-dependent and provides a high chance of reaching the global optimum. In comparison with the dynamic programming [9], GA allows the users to get the sub-optimal solution while dynamic programming cannot, which is very important for some financial problems. Since the time is a limit in financial world, the investors often use a sub-optimal but acceptable solution to allocate assets. Due to these advantages, we use GA to perform portfolio selection.

The main motivation of this study is to employ a two-stage genetic optimization algorithm for portfolio selection. In the first stage, a genetic algorithm is used to identify good quality assets in terms of asset return ranking. In the second stage, asset allocation in the selected good quality assets is optimized using a genetic algorithm based on Markowitz's theory. Through the double-stage genetic optimization process, an optimal portfolio can be determined. The rest of the paper is organized as follows. Section 2 describes the basic selection and optimization process based on the two-stage genetic algorithm in detail. In order to test the efficiency of the proposed algorithm, a simulation study is performed in Section 3. And Section 4 concludes.

2 Double-Stage Genetic Algorithm for Portfolio Selection

Generally, GA imitates the natural selection process in biological evolution with selection, crossover and mutation, and the sequence of the different operations of a genetic algorithm is shown in the left part of Fig. 1. Usually, GA is based on the survival-of-the-fittest fashion by gradually manipulating the potential problem solutions to obtain the more superior solutions in population. Optimization is performed in the representation rather than in the problem space directly. To date, GA has become a popular optimization method as they often succeed in finding the best optimum by global search in contrast to most common optimization algorithms. Interested readers can be referred to [10-11] for more details.

2.1 Stage I: Asset Ranking Using Genetic Algorithm

The aim of this stage is to identify the quality of each stock so that investors can choose some good ones for investment. Here a genetic algorithm is used as a stock ranking tool. In this study, some financial indicators of the listed companies are employed to determine and identify the quality of each stock. That is, the financial indicators of the companies are used as input variables while a score is given to rank the stocks. The output variable is stock ranking. Through the study, four important financial indicators, return on capital employed (ROCE), price/earnings ratio (P/E Ratio), earning per share (EPS) and liquidity ratio are utilized in this study. Their definition is formulated as

$$\text{ROCE} = (\text{Profit})/(\text{Shareholder's equity})*100\% \tag{1}$$

$$\text{P/E ratio} = (\text{stock price})/(\text{earnings per share})*100\% \tag{2}$$

$$\text{Earnings per share} = (\text{Net income})/(\text{The number of ordinary shares}) \tag{3}$$

$$\text{Liquidity Ratio} = (\text{Current Assets})/(\text{Current Liabilities})*100\% \tag{4}$$

When the input variables are determined, we can use GA to distinguish and identify the quality of each stock, as illustrated in Fig. 1. The detailed procedure is illustrated as follows.

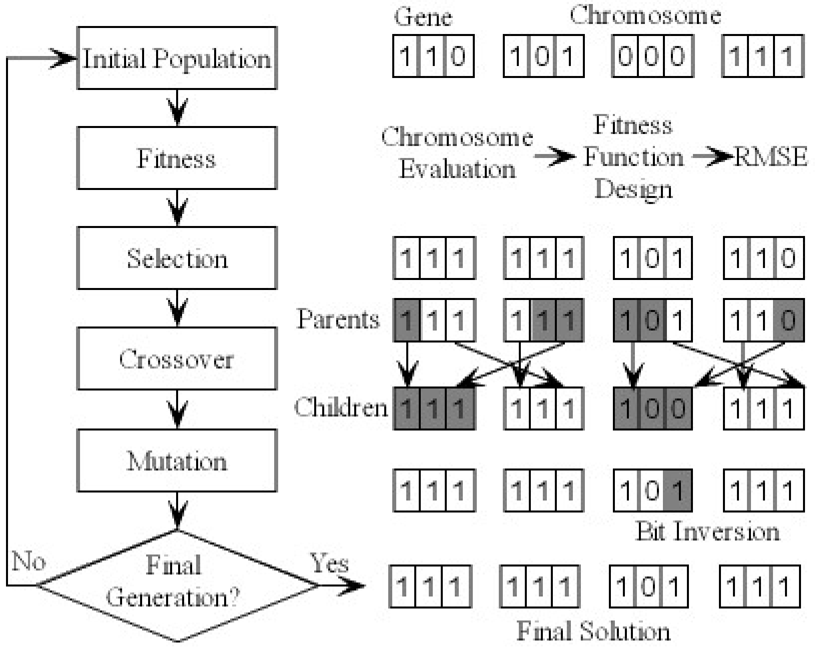


Fig. 1. Stock ranking with genetic algorithm

First of all, a population, which consists of a given number of chromosomes, is initially created by randomly assigning "1" and "0" to all genes. In the case of stock ranking, a gene contains only a single bit string for the status of input variable. The top right part of Fig. 1 shows a population with four chromosomes, each chromosome includes different genes. In this study, the initial population of the GA is generated by encoding four input variables. For the testing case of ROCE, we design 8 statuses representing different qualities in terms of different interval, varying from 0 (Extremely poor) to 7 (very good). An example of encoding ROCE is shown in Table 1. Other input variables are encoded by the same principle. That is, the binary string of a gene consists of three single bits, as illustrated by Fig. 1.

Table 1. An example of encoding ROCE

ROCE value	Status	Encoding
(-∞, -30%]	0	000
(-30%, -20%]	1	001
(-20%,-10%]	2	010
(-10%,0%]	3	011
(0%, 10%]	4	100
(10%, 20%]	5	101
(20%, 30%]	6	110
(30%,+∞)	7	111

The subsequent work is to evaluate the chromosomes generated by previous operation by a so-called fitness function, while the design of the fitness function is a crucial point in using GA, which determines what a GA should optimize. Since the output is some estimated stock ranking of designated testing companies, some actual stock ranking should be defined in advance for designing fitness function. Here we use annual price return (APR) to rank the listed stock and the APR is represented as

$$APR_n = \frac{ASP_n - ASP_{n-1}}{ASP_{n-1}} \tag{5}$$

where APR_n is the annual price return for year n, ASP_n is the annual stock price for year n. Usually, the stocks with a high annual price return are regarded as good stocks. With the value of APR evaluated for each of the N trading stocks, they will be assigned for a ranking r ranged from 1 and N, where 1 is the highest value of the APR while N is the lowest. For convenience of comparison, the stock's rank r should be mapped linearly into stock ranking ranged from 0 to 7 according to the following equation:

$$R_{actual} = 7 \times \frac{N - r}{N - 1} \tag{6}$$

Thus, the fitness function can be designed to minimize the root mean square error (*RMSE*) of the difference between the financial indicator derived ranking and the next year's actual ranking of all the listed companies for a particular chromosome, representing by

$$RMSE = \sqrt{\frac{1}{m}\sum_{t=1}^{m}\left(R_{derived} - R_{actual}\right)^2} \tag{7}$$

After evolving the fitness of the population, the best chromosomes with the highest fitness value are selected by means of the roulette wheel. Thereby, the chromosomes are allocated space on a roulette wheel proportional to their fitness and thus the fittest chromosomes are more likely selected. In the following crossover step, offspring chromosomes are created by some crossover techniques. A so-called one-point crossover technique is employed, which randomly selects a crossover point within the chromosome. Then two parent chromosomes are interchanged at this point to produce two new offspring. After that, the chromosomes are mutated with a probability of 0.005 per gene by randomly changing genes from "0" to "1" and vice versa. The mutation prevents the GA from converging too quickly in a small area of the search space. Finally, the final generation will be judged. If yes, then the optimized results are obtained. If no, then the evaluation and reproduction steps are repeated until a certain number of generations, until a defined fitness or until a convergence criterion of the population are reached. In the ideal case, all chromosomes of the last generation have the same genes representing the optimal solution [12].

2.2 Stage II: Asset Allocation Optimization Using Genetic Algorithm

In the previous stage, some good quality stocks can be revealed in terms of stock return ranking. However, portfolio management does not only focus on the return but also on risk minimization. Therefore, good stock ranking is not enough for portfolio management; risk factor must be taken into account in terms of portfolio theory.

Modern portfolio theory originally by Markowitz [1] is based on a reasonable trade-off between expected return and risk. As earlier noted by Equation (2), portfolio optimization model can be solved by quadratic programming (QP). But the QP model can also be solved by genetic algorithm. Since it is a typical optimization model, GA is suitable for this task. The basic procedure of GA for this problem is similar to Section 3.1, but a suitable chromosome representation is needed to encode its solution space and an appropriate fitness function should be designed. In order to apply the model, the values of the expected return $E(R_i)$ and covariance σ_{ij} for all i and j should be determined, which are represented by

$$\begin{cases} \text{Expected Return} & E(R_i) = \sum_{i=1}^{n} R_{it}/n \\ & R_{it} = \dfrac{SCP_{it} - SCP_{i(t-1)}}{SCP_{i(t-1)}} \\ \text{Covariance} & \sigma_{ij} = \dfrac{1}{n}\sum_{i=1}^{n}\left((R_{it} - E(R_i)) \times (R_{jt} - E(R_j))\right) \end{cases} \tag{8}$$

where R_{it} is the return of stock i for time t, SCP_{it} is stock closing price for stock i at time t, n is the number of time period for available data.

Solution for asset allocation for stock should be a composition of the stock quantity to be held so as to minimize the risk on a given level of expected return which will get the optimal solution. Thus the chromosome can be designed as follows: each of the stock weight (w) is a composite of eight bits, representing the value from 0 to 255, thus the normalized weight (x) of each stock can be calculated with the Equation (9) and the detailed chromosome representation is shown in Fig. 2.

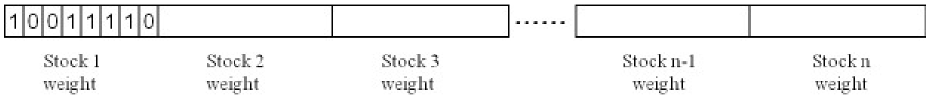


Fig. 2. The chromosome design of portfolio optimization

$$x_i = \frac{w_i}{\sum_{i=1}^{n} w_i} \tag{9}$$

The fitness function is another important issue in genetic algorithms for solving the problem. In the portfolio optimization, the fitness function must make a rational trade-off between minimizing risk and maximizing return. Thus the fitness function can be designed as follows:

$$Fitness = \sum_{i=1}^{n} \sum_{j=1}^{n} \sigma_{ij} x_i x_j + \left(\sum_{i=1}^{n} E(R_i) x_i - R_p^* \right)^2 \tag{10}$$

From Equation (10), we find that the fitness function can be broken up into two parts. The first one is required to minimize the risk while the second part also needs to be minimized so that the portfolio's overall return will stick to the expected return that we pre-defined. Therefore, the GA can be performed by minimizing this fitness function. The fitness function for each chromosome is the indicator for GA to perform the selection. After crossover and mutation, the new chromosome is generated for the next iterative evaluation procedure.

Through the optimization process of two-stage GA, the most valuable portfolio, i.e., good stock combination with optimal asset allocation can be mined and discovered to support investors' decision-making.

3 Experiment Analysis

3.1 Data Description and Experiment Design

The daily data used in this study is stock closing price obtained from Shanghai Stock Exchange (SSE) (http://www.sse.com.cn). The sample data span the period from January 2, 2001 to December, 31 2004. Monthly and yearly data in this study are obtained by daily data computation. For simulation, 100 stocks are randomly selected. In this study, we select 100 stocks from Shanghai A share, and their stock codes vary from 600000 to 600100.

In the first stage, the company financial information as the input variables is fed into the GA to obtain the derived company ranking. This output is compared with the actual stock ranking in terms of APR, as indicated by Equations (5) and (6). In the process of GA optimization, the RMSE between the derived and the actual ranking of each stock is calculated and served as the evaluation function of the GA process. The best chromosome obtained is used to rank the stocks and the top *n* stocks are chosen for the portfolio in the next stage. For experiment purpose, the top 10, 20 and 30 stocks are chosen for testing according to the ranking of stock quality using GA.

In the second stage, the top 10, 20 and 30 stocks with the highest rank derived from the previous stage are selected. The portfolio optimization is then performed for asset allocation. Expected return of the previous 12 months and covariance of return are needed to calculate according to the Equation (8) for each stock by accumulating the return of each month. Consequently, the portfolio allocation, weight of stock in the portfolio, will be obtained from GA process by minimizing the fitness function (i.e., Equation (10)). Therefore, the most valuable portfolio can be mined and discovered by the two-stage genetic optimization algorithm.

3.2 Experimental Results

In the first stage, four financial indicators of different stocks as input variables are fed into GA process to derive the stock rank and meantime the good quality stock ranks are obtained by minimizing the discrepancies between the derived rank and the actual rank. Again, the RMSE is used to measure the quality of the solution. For simulation, the RMSE results of the top 10, 20 and 30 stocks are reported in Table 2. As can be seen from Table 2, the RMSE increases with the increase of the number of stocks selected.

Table 2. The RMSE results for stock ranking using GA optimization

Number of Stocks	Top 10	Top 20	Top 30
2001	0.8756	0.9231	0.9672
2002	0.8935	0.9056	0.9247
2003	0.8542	0.9098	0.9111
2004	0.9563	0.9352	0.9793

After ranking the stock, some good quality stocks can be selected as the component of the portfolio. The selection of the good quality stocks is depended on a threshold for the stock ranking that investor pre-defined. When the number of stocks is determined by investors in terms of stock ranking, the subsequent process is that these selected stocks will be sent to the second optimization stage for finding out the proportion of investment. For testing purpose, the best 10, 20 and 30 stocks are selected as the input values for the stock allocation process. Of course, the investor's expected return is also required as an input variable. It should be noted that for a month basis evaluation process, the expected monthly return should be the result of annual return divided by 12. Based upon the algorithm proposed by Section 2.2, the optimal asset allocation for the stocks can be obtained using GA. For interpretation, two important comparisons are performed. Assumed that expected return is set to 10% and net accumulated return is used as performance evaluation criterion in the simulation.

A. Return comparison between optimal portfolio and equally weighted portfolio
In this comparison, equally weighted portfolio is that assets are equally assigned to every stock in the portfolio while optimal portfolio is obtained by GA optimization. In addition, only the top 10 stocks are included into the portfolio in this comparison. Accordingly, the performance results are shown in Fig. 3 below.

From Fig. 3, the net accumulated return of the equally weighted portfolio is found to be the worse than that of the optimal portfolio. This implies that if one investor with no experience randomly chooses a portfolio of stock to invest, the expected return for the portfolio will be approximately the same as that value. It is not a surprising fact because there are so many bad quality stocks in the stock market that may lower the overall performance of the portfolio. Even one gets no loss in the random investment; he has already had a loss due to the opportunity cost of capital. Meantime, this result also indicates that the selection of good quality stock is very important step in the portfolio selection, which is often neglected by Markowitz's theory.

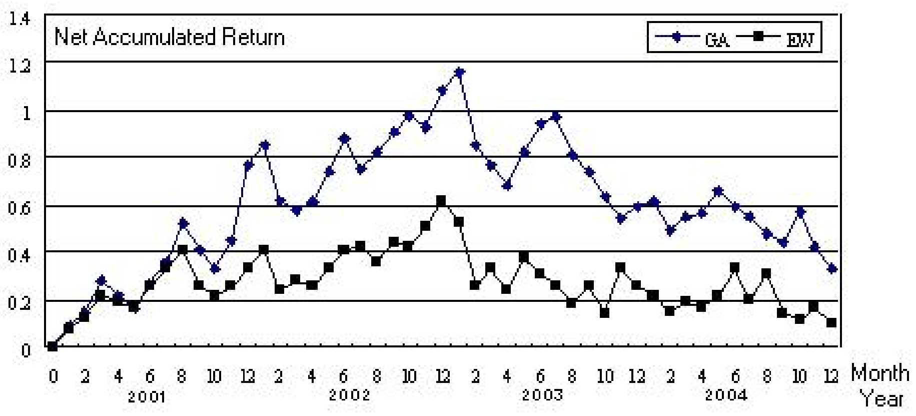


Fig. 3. The return comparison between optimal portfolio and equally weighted portfolio

B. The return comparison with different number of stocks

In this study, three portfolios with 10, 20 and 30 stocks are compared. The optimal asset allocation is performed by GA. Accordingly the results are shown in Fig. 4.

From Fig. 4, we can find that the portfolio performance decreases with the increase of the number of stock in the portfolio and the portfolio performance of the 10 stocks is the best in the testing. As earlier noted, portfolio management does not only focus on the expected return but also on risk minimization. The larger the number of stocks in the portfolio is, the more flexible for the portfolio to make the best composition to avoid risk. However, selecting good quality stocks is the prerequisite of obtaining a good portfolio. That is, although the portfolio with the large number of stocks can lower the risk to some extent, some bad quality stocks may include into the portfolio, which influences the portfolio performance. This result also demonstrates that if the investors select good quality stocks, the portfolio with the large number of stocks does not necessary outperform the portfolio with the small number of stocks. Therefore it is wise for investors to select a limit number of good quality stocks for portfolio optimization.

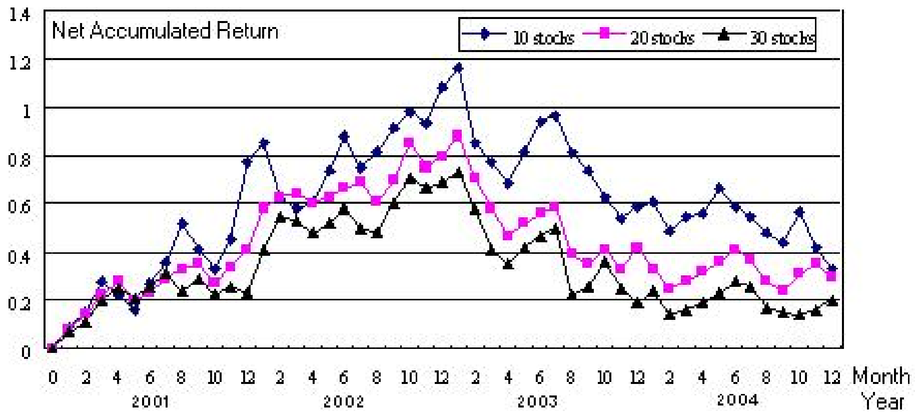


Fig. 4. The result comparison with different number of stocks

In addition, Fig. 4 also shows that the performance trend for different portfolios with different number of stocks is very similar except for the magnitude. Although a portfolio can reduce asymmetric risk, it can do little in the case where overall market has poor performance. For example, the market condition is good for the first two years and all the portfolios perform well, however, for the last two years, especially for 2004, the market trend reverses and that causes all the portfolios to have reversal trends too.

4 Conclusions

In this study, a two-stage genetic optimization algorithm is proposed to mine the most valuable portfolio. In the first stage, GA is used to rank the stock and select the good quality stock for portfolio optimization. In the second stage, optimal asset allocation for portfolio can be realized by GA. Simulation results demonstrate that the proposed two- stage genetic optimization algorithm is an effective portfolio optimization approach, which can mine the most valuable portfolio for investors. In addition, experiment results also find that (1) selecting some good quality stocks before portfolio asset allocation is very important; (2) the quantity of stocks in the portfolio may not necessary satisfy the principle of “the more, the better”, therefore a limit number of stock in the portfolio can generally improve the portfolio performance.

Acknowledgements

This work is partially supported by National Natural Science Foundation of China (NSFC No. 70221001); Chinese Academy of Sciences and Strategic Research Grant of City University of Hong Kong (SRG No. 7001677).

References

1. Markowitz, H.M.: Portfolio Selection. Journal of Finance 7 (1952) 77-91
2. Levin, A.U.: Stock Selection via Nonlinear Multi-factor Models. Advances in Neural Information Processing Systems (1995) 966-972
3. Chu, T.C. Tsao, C.T. Shiue, Y.R.: Application of Fuzzy Multiple Attribute Decision Making on Company Analysis for Stock Selection. Proceedings of Soft Computing in Intelligent Systems and Information Processing (1996) 509-514
4. Zargham, M.R., Sayeh, M.R.: A Web-Based Information System for Stock Selection and Evaluation. Proceedings of the First International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems (1999) 81-83
5. Fan, A., Palaniswami, M.: Stock Selection Using Support Vector Machines. Proceedings of International Joint Conference on Neural Networks 3 (2001) 1793-1798
6. Berger, A.J., Glover, F., Mulvey, J.M.: Solving Global Optimization Problems in Long-Term Financial Planning. Statistics and Operation Research Technical Report, Princeton University (1995)
7. Casas, C.A.: Tactical Asset Allocation: An Artificial Neural Network Based Model. Proceedings of International Joint Conference on Neural Networks, 3 (2001) 1811-1816
8. Chapados, N., Bengio, Y.: Cost Functions and Model Combination for VaR-based Asset Allocation Using Neural Networks. IEEE Transactions on Neural Networks 12 (2001) 890-906
9. Mulvey, J.M., Rosenhaum, D.P., Shetty, B.: Strategic Financial Risk Management and Operations Research. European Journal of Operational Research 97 (1997) 1-16
10. Holland, J. H.: Genetic Algorithms. Scientific American 267 (1992) 66-72
11. Goldberg, D.E.: Genetic Algorithm in Search, Optimization, and Machine Learning. Addison-Wesley, Reading, MA (1989)
12. Yu, L., Wang, S.Y., Lai, K.K.: An Integrated Data Preparation Scheme for Neural Network Data Analysis. IEEE Transactions on Knowledge and Data Engineering 18 (2006) 217-230

Image Reconstruction Using Genetic Algorithm in Electrical Impedance Tomography

Ho-Chan Kim, Chang-Jin Boo, and Min-Jae Kang

Faculty of Electrical and Electronic Engineering, Cheju National University, Jeju, Jeju-do, 690-756, Korea
{hckim, boo1004, minjk}@cheju.ac.kr

Abstract. In electrical impedance tomography (EIT), various image reconstruction algorithms have been used in order to compute the internal resistivity distribution of the unknown object with its electric potential data at the boundary. Mathematically the EIT image reconstruction algorithm is a nonlinear ill-posed inverse problem. This paper presents a genetic algorithm technique for the solution of the static EIT inverse problem. The computer simulation for the 32 channels synthetic data shows that the spatial resolution of reconstructed images in the proposed scheme is improved compared to that of the modified Newton–Raphson algorithm at the expense of increased computational burden.

1 Introduction

EIT plays an important role as a new monitoring tool for engineering applications such as biomedical imaging and process tomography, due to its relatively cheap electronic hardware requirements and nonintrusive measurement property [1]. In EIT, different current patterns are injected to the unknown object through electrodes and the corresponding voltages are measured on its boundary surface. The physical relationship between inner resistivity (or conductivity) and boundary surface voltage is governed by the nonlinear Laplace equation with appropriate boundary conditions, so that it is impossible to obtain the closed-form solution for the resistivity distribution. Hence, the internal resistivity distribution of the unknown object is computed using the boundary voltage data based on various reconstruction algorithms.

Yorkey et al. [2] developed a modified Newton-Raphson (mNR) algorithm for a static EIT image reconstruction and compared it with other existing algorithms such as backprojection, perturbation and double constraints methods. They concluded that the mNR reveals relatively good performance in terms of convergence rate and residual error compared to those of the other methods. However, in real situations, the mNR method is often failed to obtain satisfactory images from physical data due to large modeling error, poor signal to noise ratios (SNRs) and ill-posed characteristics.

The major difficulties in impedance imaging are in the nonlinearity of the problem itself and the poor sensitivity of the boundary voltages to the resistivity of the flow domain deep inside. Several researchers suggested various element or mesh grouping methods where they force all meshes belonging to certain groups to have the same resistivity values [3,4].

In this paper, we will discuss the image reconstruction based on a genetic algorithm approach in EIT. We have broken the procedure for obtaining the internal resistivity distribution into two steps. In the first step, each mesh is classified into three mesh groups: target, background, and temporary groups. In the second step, the values of these resistivities are determined using a genetic algorithm (GA) [5]. This two-step approach allows us to better constrain the inverse problem and subsequently achieve a higher spatial resolution.

2 Mathematical Model

The numerical algorithm used to convert the electrical measurements at the boundary to a resistivity distribution is described here. The algorithm consists of iteratively solving the forward problem and updating the resistivity distribution as dictated by the formulation of the inverse problem. The forward problem of EIT calculates boundary potentials with the given electrical resistivity distribution, and the inverse problem of EIT takes potential measurements at the boundary to update the resistivity distribution.

2.1 The Forward Model

When electrical currents I_l are injected into the object $\Omega \in R^2$ through electrodes e_l attached on the boundary $\partial\Omega$ and the resistivity distribution $\rho(x, y)$ is known over Ω , the corresponding induced electrical potential $u(x, y)$ can be determined uniquely from the following nonlinear Laplace equation [6] which can be derived from the Maxwell equation, Ohm's law, and the Neumann type boundary condition.

$$\nabla \cdot (\rho^{-1} \nabla u) = 0 \ \text{ in } \ \Omega \tag{1}$$

$$\begin{aligned} &\int_{e_l} \rho^{-1} \frac{\partial u}{\partial n} dS = I_l, \ l = 1, \cdots, L \\ &u + z_l \rho^{-1} \frac{\partial u}{\partial n} = U_l \ \text{ on } e_l, \ l = 1, \cdots, L \\ &\rho^{-1} \frac{\partial u}{\partial n} = 0 \ \text{ on } \partial\Omega \backslash \bigcup_{l=1}^{L} e_l \end{aligned} \tag{2}$$

where z_l is effective contact impedance between the l th electrode and the object, U_l is the measured potential at the l th electrode and n is outward unit normal. In addition, we have the following two conditions for the injected currents and measured voltages by taking into account the conservation of electrical charge and appropriate selection of ground electrode, respectively.

$$\sum_{l=1}^{L} I_l = 0, \ \sum_{l=1}^{L} U_l = 0 \tag{3}$$

The computation of the potential $u(x, y)$ for the given resistivity distribution $\rho(x, y)$ and boundary condition I_l is called the forward problem. The numerical solution for the forward problem can be obtained using the finite element method (FEM). In the FEM, the object area is discretized into small elements having a node at each corner.

It is assumed that the resistivity distribution is constant within an element. The potential at each node is calculated by discretizing (1) into $YU_C = I_C$, where U_C is the vector of boundary potential, I_C the vector of injected current patterns and the matrix Y is a functions of the unknown resistivities.

2.2 The Tikhonov Regularization Method

The inverse problem, also known as the image reconstruction, consists in reconstructing the resistivity distribution $\rho(x, y)$ from potential differences measured on the boundary of the object. The relatively simple situation depicted so far does not hold exactly in the real world. The methods used for solving the EIT problem search for an approximate solution, i.e., for a resistivity distribution minimizing some sort of residual involving the measured and calculated potential values. From a mathematical point of view, the EIT inverse problem consists in finding the coordinates of a point in a M -dimensional hyperspace, where M is the number of discrete elements whose union constitutes the tomographic section under consideration. To reconstruct the resistivity distribution inside the object, we have to solve the nonlinear ill-posed inverse problem. Regularization techniques are needed to weaken the ill-posedness and to obtain stable solutions. Generalized Tikhonov regularized version of the EIT inverse problem can be written in the form [6]

$$E(\rho) = \min_{\rho}\{\| U - V(\rho) \|^2 + \lambda \| R\rho \|^2\} \tag{4}$$

where $\rho \in R^N$ is the resistivity distribution. $V(\rho) \in R^{LK}$ is the vector of voltages obtained from the model with known ρ , $U \in R^{LK}$ are the measured voltages and R (λ) are the regularization matrix (parameter), respectively. L and K are the numbers of electrodes on the surface and injected current patterns, respectively. There are many approaches in the literature [7] to determine R and α, but the usual choice is to fix $R = I_M$ with the identity matrix and to adjust λ empirical.

Minimizing the objective function $E(\rho)$ gives an equation for the update of the resistivity vector

$$\begin{aligned} \rho_{k+1} &= \rho_k + \Delta\rho_{k+1} \\ \Delta\rho_{k+1} &= (H_k + \lambda I)^{-1}\{J_k^{\,T}(U - V(\rho_k)) - \lambda\rho_k\} \end{aligned} \tag{5}$$

where the partial derivative of E with respect to ρ has been approximately by a Taylor series expansion around ρ_k. The Jacobian J_k is a matrix composed of the derivative of the vector of predicted potentials with respect to the unknown resistivities. The Hessian H_k is the second derivative of the predicted potentials with respect to the resistivity and is approximated as the square of the Jacobian for computational efficiency. Since the objective function $E(\rho)$ is multimodal (i.e., it presents several local minima), the inversion procedure does not always converge to the true solution. The reconstruction algorithms are likely to be trapped in a local minimum and sometimes the best solution of a static EIT problem is rather unsatisfactory.

3 Image Reconstruction Based on GA Via Two-Step Approach

In some applications like visualization of two-component systems, we may assume that there are only two different representative resistivity values; one resistivity value for the background and the other for the target. In this paper, we will discuss the image reconstruction in EIT using two-step approach. We have broken the procedure for obtaining the internal resistivity distribution into two parts.

Step One – mNR Method and Mesh Grouping. In the first step, we adopted a mNR method as a basic image reconstruction algorithm. After a few initial mNR iterations performed without any grouping, we classify each mesh into one of three mesh groups: BGroup (or TGroup) is the mesh group with the resistivity value of the background (or target). RGroup is the group of meshes neither in BGroup nor in TGroup. All meshes in BGroup and in TGroup are forced to have the same but unknown resistivity value (ρ_{back} and ρ_{tar}), respectively. However, all meshes in RGroup can have different resistivity vaules ($\rho_{temp,i}(i=1,\ldots,n-2)$).

Let $s_i(i=1,\ldots,n)$ be the resistivity distribution after this rearrangement. Then, the typical shape of s_i becomes the curve shown in Fig. 1 during the reconstruction process. In Fig. 1, it is natural to assume that meshes in regions I and III belong to BGroup and TGroup, respectively. All meshes in region II can be classified into RGroup.

However, since we cannot always expect to get such a well-distinguished restivity distribution curve as shown in Fig. 1, it is useful to divide the regions and determine a typical resistivity value of each region. Let $\bar{\rho}_i(i=1,\ldots,3)$ be the representative resistivity value in each region and $k_i(i=1,2)$ be the boundary location between regions. Then, we can formulate the following optimization problem to determine $\bar{\rho}_i$ and k_i:

$$J(x)=\min_x\{\sum_{i=1}^{3}\sum_{j=k_{i-1}}^{k_i}(s_j-\bar{\rho}_i)^2\} \tag{6}$$

where $x=(\bar{\rho}_1,\bar{\rho}_2,\bar{\rho}_3,k_1,k_2)$, $k_0=1$, and $k_3=n$.

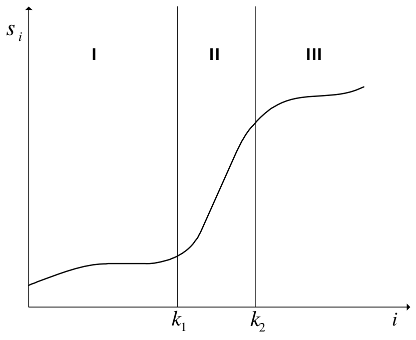


Fig. 1. Typical distribution of the sorted resistivity values during image reconstruction

We solve the problem in (6) using the GA and the solution provides one way of dividing regions [4].

Step Two – Image Reconstruction Based on the Genetic Algorithm. In the second step, a set (population) of EIT images is generated for the simplest implementation of GA in EIT. Each individual consists in a n-tuple of resistivity values (n is the number of elements discretizing the section under measurement), i.e., the EIT chromosome is a sequence of n resistivities. After mesh grouping, in this paper, we will determine the values of these resistivities using two GAs. The first GA searches for the optimal range of resistivities by generating and evolving a population of individuals whose chromosome consists of two real genes (ρ_{back} and ρ_{tar}), representing the BGroup and TGroup values. Furthermore, we will use ρ_{back} (or ρ_{tar}) as the minimum (or maximum) values of the unknown resistivity distribution. The second GA solves the EIT problem, searching for the resistivity distribution ($\rho_{temp,i}(i=1,\ldots,n-2)$) minimizing the reconstruction error. The computed resistivities is constrained between the minimum and maximum values obtained in the first GA.

We will iteratively reconstruct an image that fits best the measured voltages U_l at the l-th electrode. To do so, we will calculate at each iteration the pseudo voltages $V_l(\rho)$ that correspond to the present state of the reconstructed image. We assume that, by minimizing the difference between the measured voltages and the pseudo voltages, the reconstructed image will converge towards the sought-after original image.

A fitness value is computed for each individual. In the present case, the fitness function is the reciprocal of the reconstruction error, a function of the relative difference between the computed and measured potentials on the object boundary

$$f_c = \frac{L(L-1)}{2}\left[\sum_{i=1}^{L(L-1)/2}\left|\frac{U_i - V_i(\rho)}{U_i}\right|\right]^{-1} \tag{7}$$

where L is the number of electrodes on the surface.

4 Computer Simulation

The proposed algorithm has been tested by comparing its results for numerical simulations with those obtained by mNR method. For the current injection the trigonometric current patterns were used. For the forward calculations, the domain Ω was a unit disc and the mesh of 3104 triangular elements (M=3104) with 1681 nodes (N=1681) and 32 channels (L=32) was used as shown in Fig. 2(a). A different mesh system with 776 elements (M=776) and 453 nodes (N=453) was adopted for the inverse calculations as shown in Fig. 2(b). In this paper, under the assumption that the resistivity varies only in the radial direction within a cylindrical coordinate system [8], the results of the two inverse problem methods can be easily compared. The resistivity profile given to the finite element inverse solver varies from the center to the boundary of object and is divided into 9 radial elements ($\rho_1,\cdots,\rho_9$) in Fig. 2(b).

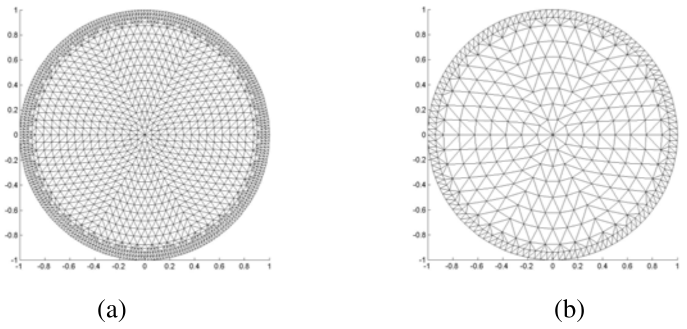


Fig. 2. Finite element mesh used in the calculation. (The resistivities of the elements within an annular ring are identical.) (a) mesh for forward solver, (b) mesh for inverse solver.

Synthetic boundary potentials were computed for idealized resistivity distributions using the finite element method described earlier. The boundary potentials were then used for inversion and the results were compared to the original resistivity profiles. The resistivity profile appearing in Fig. 3 has a step change at r/R=0.43. The inverted profile using mNR method matches the original profile very well near the boundary of the object at r/R=1 and the jump in resistivity was located successfully. However, the inverse method using mNR searches for a resistivity profile which is smooth, which explains the deviation near the center at r/R=0 and the boundary of target and background at r/R=0.43.

We started the mNR iteration without any mesh grouping with a homogeneous initial guess. In Table 1, we see that the mNR algorithm may roughly estimate the given true resistivities. Since the mNR have a large error at the boundary of target and background in Fig. 3, we can not obtain reconstructed images of high spatial resolution. This kind of poor convergence is a very typical problem in the NR-type algorithms. However, we can significantly improve the mNR's poor convergence by adopting the proposed GA via a two-step approach as follows.

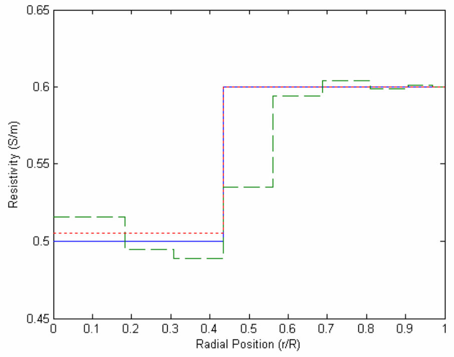


Fig. 3. True resistivities(solid line) and coumputed resistivities using mNR(dashed line) and GA(dotted line)

Table 1. True resistivities and computed resistivities using mNR and GA

	ρ_1	ρ_2	ρ_3	ρ_4	ρ_5	ρ_6	ρ_7	ρ_8	ρ_9
Real	0.5	0.5	0.5	0.6	0.6	0.6	0.6	0.6	0.6
mNR	0.516	0.495	0.489	0.535	0.594	0.604	0.599	0.601	0.600
GA	0.505	0.505	0.505	0.600	0.600	0.600	0.600	0.600	0.600

In the first step, we adopted a mNR method as a basic image reconstruction algorithm. After a few initial mNR iterations performed without any grouping, we classify each mesh into one of three mesh groups. After the mesh grouping in (6), we could obtained the following result that 2 meshes (ρ_2, ρ_3) and 5 meshes ($\rho_5, \rho_6, \rho_7, \rho_8, \rho_9$) among 9 are grouped to TGroup (ρ_{tar}) and BGroup (ρ_{back}) , respectively. The remainders of meshes (ρ_1, ρ_4) are grouped to RGroup. Hence, the number of unknowns is reduced to 4.

In the second step, after mesh grouping, we will determine the values of these resistivities using two GAs. The first GA searches for the optimal range of resistivities by generating and evolving a population of individuals whose chromosome consists of two real genes (ρ_{back} and ρ_{tar}) . $\bar{\rho}_1$ and $\bar{\rho}_3$ in (6) are used the initial value of ρ_{tar} and ρ_{back} in BGroup and TGroup, respectively. Table 2 shows the computed resistivities as a function of the population size at generation 200. The reconstructed errors at a given generation generally decrease when the population size is increased. Hence, even if error does not depend linearly on the population size due to the stochastic nature of GA's, 40 or 60-individual GA reconstruction gives a higher spatial resolution than a mNR method.

The second GA solves the EIT problem, searching for the resistivities of remainders (ρ_1, ρ_4) minimizing the reconstruction error. The computed resistivities in this second GA is constrained between the minimum and maximum values obtained in the first GA. In Fig. 3, the inverted profile using GA matches the original profile very well near the wall at r/R =1.0 as well as the center at r/R =0.0. Furthermore, the GA reconstruction is practically perfect for the jump of resistivity at r/R =0.43.

Table 2. True and computed resistivities using GA vs population size at generation 200

Popsize	ρ_{back}		ρ_{tar}	
	True	Computed	True	Computed
20	0.5	0.4898	0.6	0.6000
40	0.5	0.5051	0.6	0.6001
60	0.5	0.4998	0.6	0.6039

5 Conclusion

In this paper, an EIT image reconstruction method based on GA via two-step approach was presented to improve the spatial resolution. A technique based on two binary-coded GA's with the knowledge of mNR was developed for the solution of the EIT inverse problem. One GA calculates the resistivity values of target group and background group, and the other GA is used to search for the resistivities of remainders. Although GA is expensive in terms of computing time and resources, which is a weakness of the method that renders it presently unsuitable for real-time tomographic applications, the exploitation of *a priori* knowledge will produce very good reconstructions.

Acknowledgement

The part of researchers participating in this study are supported by the grant from "the 2nd phase BK21 project".

References

1. Webster, J.G.: Electrical Impedance Tomography, Adam Hilger (1990)
2. Yorkey, T.J., Webster, J.G., Tompkins, W.J.: Comparing Reconstruction Algorithms for Electrical Impedance Tomography. IEEE Trans. on Biomed. Eng.. 34 (1987) 843-852
3. Glidewell, M., Ng, K.T.: Anatomically Constrained Electrical Impedance Tomography for Anisotropic Bodies Via a Two-step Approach. IEEE Trans. on Med. Imag.. 14 (1995) 498-503
4. Cho, K.H., Kim, S., Lee, Y.J.: Impedance Imaging of Two-phase Flow Field with Mesh Grouping. Nuclear Engineering and Design. 204 (2001) 57-67
5. Olmi, R., Bini, M., Priori, S.: A Genetic Algorithm Approach to Image Reconstruction in Electrical Impedance Tomography. IEEE Trans. on Evolutionary Comput.. 4 (2000) 83-87
6. Vauhkonen, M.: Electrical Impedance Tomography and Priori Information, Kuopio Univerisity Publications Co., Natural and Environmental Sciences 62 (1997)
7. Cohen-Bacrie, C., Goussard, Y., Guardo, R.: Regularized Reconstruction in Electrical Impedance Tomography Using a Variance Uniformization Constraint. IEEE Trans. on Medical Imaging. 16 (1997) 170-179
8. Kim, H.C., Boo, C.J., Lee, Y.J.: Image Reconstruction using Simulated Annealing Algorithm in EIT. Int. J. of Control, Automation, and Systems. 3 (2005) 211-216

Mitigating Deception in Genetic Search Through Suitable Coding

S.K. Basu* and A.K. Bhatia**

Department of Computer Science
Banaras Hindu University
Varanasi-221005, India

Abstract. Formation of hamming cliff hampers the progress of genetic algorithm in seemingly deceptive problems. We demonstrate through an analysis of neighbourhood search capabilities of the mutation operator in genetic algorithm that the problem can somtimes be overcome through proper genetic coding. Experiments have been conducted on a 4-bit deceptive function and the pure-integer programming problem. The integer-coded genetic algorithm performs better and requires less time than the binary-coded genetic algorithm in these problems.

Keywords: Genetic algorithm, Deception, Neighbourhood search.

1 Introduction

Genetic algorithms (GAs) [6] are a class of adaptive search procedures derived from the Darwin's law of survival of the fittest. GAs start from a randomly generated population of individuals representing potential solutions of the candidate problem. These individuals are selected according to their fitness values. Genetic operators such as crossover and mutation alter the composition of the selected individuals and make them ready to be part of the population in later generations. GA marches towards the global optimum over a number of generations. Successful implementation of GA involves suitable genetic coding, genetic operators, and the values of genetic parameters such as population size, crossover probability, mutation probability and number of generations. Genetic algorithms have been used to find the global optimum in a number of hard optimization problems.

In a fully deceptive function, the string with the next higher fitness value has differing bit on each of the locus compared with the string with lower fitness value. It results in formation of hamming cliff and genetic algorithms get trapped at a sub-optimal solution. Other genetic codings have been used to solve hard problems because of the bottleneck of formation of hamming cliff in binary-coded genetic algorithms. Real-number-coded GA have been used for solving continuous function optimization problems [9]. Specialized codings based on problem structure have been devised for many

* Corresponding author. Email: swapankb@bhu.ac.in; Fax: +91-542-2368285

** Current address: National Bureau of Animal Genetic Resources, Karnal - 132001 (India) Email: avnish@lycos.com problems such as bin-packing problem, which equip the GA with neighbourhood search capabilities [5, 4].

In this paper, we analyze the neighbourhood search capabilities of mutation operator in deceptive functions using GA with binary and integer codings. We apply genetic algorithm with the two codings to solve 4-bit deceptive function and pure-integer programming problem.

A 4-bit deceptive function [8] is shown in the table 1 . The function is called fully deceptive because the string '0000' has a fitness value equal to 3 while the string '1111' has the next higher fitness value of 4. The GA gets stuck up at the suboptimal value with string '0000' and cannot escape from this trap.

Table 1. 4-bit deceptive function

Number of '1's in a 4-bit string	Fitness
0	3
1	2
2	1
3	0
4	4

We also take a real-world pure-integer programming problem [11], which is defined below.

$$max\ \{cx\ :\ Ax \leq b,\ x \in Z_+^n\} \tag{1}$$

where c is an n-component cost vector, A is (m, n) coefficient matrix, b is a m-component resource vector, x is an n-component integer vector, n is the number of variables and m is the number of constraints.

The paper is organized as: section 2 analyzes the neighbourhood search capabilities of genetic algorithm in the deceptive functions, section 3 describes the experimental setup, section 4 presents the results obtained, and section 5 contains our concluding remarks.

2 Genetic Coding and Neighbourhood Search

Traditionally, the crossover has been considered the major recombination genetic operator and the mutation has been given a background role of maintaining diversity in the population. In reality, the crossover recombines two (or more) parents to produce offsprings which are at a large hamming distance from the parents [12]. Thus, the operator helps genetic algorithm to jump over peaks in a multi-modal landscape. The mutation operator results in producing a chromosome at a very small hamming distance from the parents and is an instrument for neighbourhood search in genetic algorithms.

With binary coding, neighbourhood search is effective in solving the 4-bit deceptive problem if GA can reach the string '1111' from the string '0000', resulting in a better function value. Integer coding can also be used to solve the problem where each gene consists of an integer corresponding to the number of '1's and has the corresponding fitness value shown in table 1. The neighbourhood search is effective with the integer coding if we can reach the gene '4' from the gene '3' at a locus. The following analysis shows that the probability of getting a string with next higher fitness value with binary-coding is very small compared with the integer-coding in deceptive problems.

We consider a function with n variables, each in the range $[0,\ I-1]$, where $I = 2^k$.

With the binary coding, each variable can be represented as $k = \log_2(I)$ bits so that n variables are coded as a string of $n.k$ bits. The binary coding provides a chromosome size $l = n.k$. We assume a probability of mutation equal to $1/l$. Thus, we get the probability of mutation (p_m) equal to $\frac{1}{n.k}$.

With the integer coding, each gene consists of an integer in the range $[0,\ I-1]$. It provides a chromosome size $l = n$ and we get $p_m = \frac{1}{n}$. The mutation operator in this coding is implemented by updating a gene with a $uniform[0,\ I-1]$ integer with the given probability of mutation.

For neighbourhood search to be effective, we require a chromosome having one gene with the next higher fitness value while all the other genes remain unperturbed. We denote this probability as p_n. The p_n with binary coding (p_{nb}) and with integer coding (p_{ni}) are calculated below.

$$p_{nb} = \left(1 - \frac{1}{n.k}\right)^{(n-1)k} . \left(\frac{1}{n.k}\right)^{k}$$

$$p_{ni} = \left(1 - \frac{1}{n}\right)^{(n-1)} . \frac{1}{n} . \frac{1}{I}$$

As n becomes large, $\left(1 - \frac{1}{n.k}\right)^{(n-1)k} \simeq \left(1 - \frac{1}{n}\right)^{(n-1)} \simeq \frac{1}{e}$.

$$\begin{aligned} \text{Let } M &= \frac{p_{ni}}{p_{nb}} \\ &\simeq \frac{\frac{1}{n}.\frac{1}{I}}{\left(\frac{1}{n.k}\right)^{k}} \simeq \frac{\frac{1}{n}.\frac{1}{2^k}}{\frac{1}{n^k.k^k}} \\ &= n^{k-1}.\left(\frac{k}{2}\right)^{k} \end{aligned}$$

The factor M is the order of magnitude by which $p_{ni} > p_{nb}$. Large values of M for given values of n and k indicate that neighbourhood search is much more effective with integer coding compared to binary coding in fully deceptive problems.

3 Experimental Setup

3.1 Test Data

Three test instances for the 4-bit fully deceptive function are formed by concatenating 20, 50 and 125 copies of the four bits to make strings of length 80, 200 and 500 bits.

Test instances for the pure-integer programming problem are generated according to the procedure developed by Lin and Rardin [10] and used in [1]. The procedure generates problem instances of the form:

$$\text{Maximize } c_B x_B + c_N x_N$$
$$\text{subject to } Bx_B + Nx_N = b$$

with $x_B, x_N \geq 0$ and integral, B is (m, m) basic matrix, N is $(m, n-m)$ nonbasic matrix, x_B, c_B and b are m-vectors, x_N and c_N are $(n-m)$-vectors, n is the number of variables and m is the number of constraints. Elements of B, N, c_B, c_N and b are integers.

We discard the terms $c_N x_N$ and Nx_N to obtain the problem instances of the form (1). It generates the instances with the number of variables (n) equal to the number of constraints (m).

Further, without loss of generality, we generate the problem instances in a way so that all the elements of the coefficient matrix A and the resource vector b are positive. This facilitates calculation of the lower bounds (l_i) and upper bounds (u_i) for the variables (x_i) required for genetic coding. With this relaxation, we get the bounds as:
$l_i = 0, u_i = \lceil max\{b_j/a_{ji}\} \rceil,\ \ a_{ji} \neq 0$ and $i = 1 \cdots n\,; j = 1 \cdots m;\ a_{ji}$ is the $(j, i)^{th}$ element of the matrix A, b_j is the j^{th} element of the vector b.

Four test instances of sizes 20, 50, 100 and 200 have been generated using the above procedure.

3.2 Genetic Operators and Parameters

We solve the 4-bit deceptive function with the binary as well as the integer coding. The binary strings are constructed by concatenating the 4-bit strings corresponding to a fitness value shown in the table 1. The integer-coded strings are formed by concatenating integers in the range [0,4] corresponding to the number of bits for a fitness value. A locus can take values in this range and a gene has the function values according to table 1. Chromosome sizes with this coding are 20, 50 and 125 instead of 80, 200 and 500 with the binary representation. Initial population is generated with the traditional method, where each locus can have an allele from the alphabet with equal probability.

The pure-integer programming problem also utilizes both the codings. In the binary coding, a variable of the problem is coded as a group of bits and decoded as the integer value of this group of bits. A chromosome is formed as a string of these groups of bits arranged in the sequence of the variables $x_i,\ i = 1 \cdots n$. The number of bits required for coding a variable x_i is $\lceil \log_2(u_i - l_i + 1) \rceil$. Table 2 shows the chromosome sizes of the four pure-integer programming problem instances.

Genetic algorithm is unable to solve the pure-integer programming problem with the traditional method of initial population generation as all the individuals remain infeasible throughout the GA run. We use the genetic algorithm by taking the initial population with gene-induction approach [3]. The approach uses an initial population consisting of single-gene strings where a randomly selected locus has an allele from the alphabet and all the other genes are made '0'. The infeasible strings are replaced with the newly generated single-gene strings. '0010000000' and '0000300000' are examples of single-gene binary string and single-gene integer string of size ten, respectively.

Table 2. Number of variables and chromosomes size with binary coding in the pure-integer programming problem instances

Problem Instance	Problem Size (n)	Chromosome Size
1	20	94
2	50	242
3	100	529
4	200	1070

The population size is fixed at 60 for both the problems. Linear rank selection is utilized with the value 1.5 assigned to the best string in the population [7]. Single-point crossover is used with a crossover probability equal to 0.5. The probability of mutation is fixed at $1/l$ where l is the string length. The maximum number of generations used are 2000. We use elitism in the experiments.

4 Results

4.1 4-bit Deceptive Function

Table 3 shows the results obtained with both the binary-coded and the integer-coded GA on the three instances of the 4-bit deceptive function. All the results are summary of ten experiments.

Table 3. Result of binary-coded GA execution on 4-bit deceptive problem instances in 10 experiments. (W-worst, B-best, A-average, SD-standard deviation). n is the number of 4-bit substrings.

n	Optimum		Binary-coding	Integer-coding
20	80	W	67	80
		B	71	80
		A	68.8	80
		SD	1.3	0
50	200	W	167	200
		B	177	200
		A	168.8	200
		SD	2.9	0
125	500	W	404	499
		B	419	500
		A	411.0	500
		SD	4.2	0.3

Binary-coded GA cannot reach the global optimum due to trap in the local optimum when all the bits in a substring of 4-bits are ‘0’. The integer-coded GA can reach the global optimum in all the three instances. The average function values obtained are

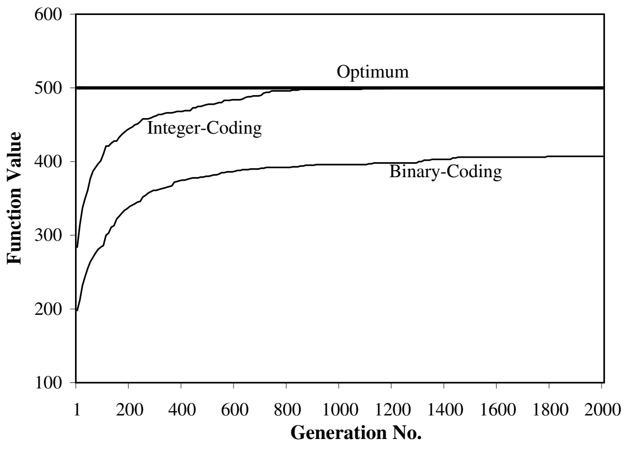


Fig. 1. Progress of the GA on 4-bit deceptive problem instance of size 125

411 and 500 with the binary-coding and the integer-coding respectively in the problem instance of size 125 substrings.

Figure 1 shows the progress of a single execution of GA with binary coding and integer coding on the 4-bit deceptive function instance of size 125. The integer-coded GA marches much faster towards the optimum function value and it can reach the global optimum. The binary-coded GA cannot reach the optimum. It stops moving towards the optimum in the last phase of the GA run.

4.2 Pure-Integer Programming Problem

Table 4 shows the results of both the binary-coded and integer-coded GAs on the pure-integer programming problem instances. The results are summary of ten experiments. Optimum value of the test instances have been found with Lp_solve [2].

Integer-coded GA provides better results compared with those reached by the binary-coded GA. The average function values reached are 14903 and 15347 with binary-coding and integer-coding respectively against the optimum of 16594 in the problem instance of size 200.

Figure 2 shows the progress of a single execution of GA with the binary coding as well as the integer coding on the pure-integer programming problem instance of size 200. GAs with both the codings start marching towards the optimum. Integer-coded GA takes over the binary-coded GA in the last phase of the GA execution.

Table 5 shows the computational time used by both the GAs for the two test problem instances. Integer-coded GA uses less time compared to the binary-coded GA in both the test problems. Integer-coded GA is computationally fast mainly due to small array size required to represent a chromosome in computer. In the 4-bit deceptive problem, a binary-coded GA requires array size of 80 integer variables to represent a chromosome for the problem size of 20 substrings. Integer-coded GA requires array size of 20 for the same instance.

Table 4. Result of GA execution on pure-integer programming problem instances (10 experiments). (W-worst, B-best, A-average, SD-standard deviation).

n	Optimum		Binary coding	Integer coding
20	722	W	696	710
		B	722	722
		A	716.6	718.4
		SD	7.7	4.5
50	4098	W	3836	3946
		B	3998	4078
		A	3909.8	4016.6
		SD	50.5	28.4
100	8770	W	8122	8418
		B	8436	8572
		A	8295.4	8507.4
		SD	77.9	49.3
200	16594	W	14740	15104
		B	15120	15704
		A	14903.3	15347.4
		SD	135.8	211.8

Table 5. Average computational time (seconds) used by binary-coded and integer-coded GAs on a PC (Pentium III 733 MHz, 64 MB RAM)

Test problem	n	Binary-coded GA	Integer-coded GA
4-bit deceptive	20	2.3	1.0
	50	5.1	2.1
	125	13.5	4.9
Pure-integer programming	20	9.2	4.9
	50	32.1	22.6
	100	111.0	82.9
	200	516.0	468.3

5 Conclusion

Genetic algorithms get trapped at the local optimum in deceptive functions due to formation of hamming cliff. The problem can be mitigated through proper coding, which empowers genetic algorithms with neighbourhood search capabilities necessary to reach the global optimum. Analysis of the neighbourhood search capabilities of genetic algorithm using mutation operator shows that the probability of getting chromosome having better fitness value in deceptive problems is much higher with integer coding compared with binary coding. The analysis has been corroborated through experiments conducted on the 4-bit deceptive function and the pure-integer programming problem. It is observed that the integer-coded GA performs better than the binary-coded GA and requires less time. The results lead us to the conclusion that deception in genetic search can also be mitigated through suitable genetic coding.

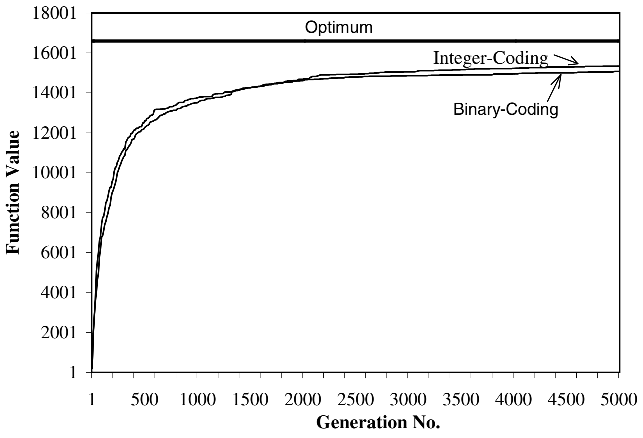


Fig. 2. Progress of the GA on pure-integer programming problem instance of size 200

References

[1] S. K. Basu and A. K. Bhatia. A naive genetic approach for non-stationary constrained problems. *Soft Computing*, 10(2):152–162, 2006.

[2] M. R. Berkelaar and J. Dirks. Lp_solve - a solver for linear programming problems with a callable subroutine library. ftp://ftp.es.ele.tue.nl/pub/lp_solve, 1999.

[3] A. K. Bhatia and S. K. Basu. Tackling 0/1 knapsack problem with gene induction. *Soft Computing*, 8(1):1–9, 2003.

[4] A. K. Bhatia and S. K. Basu. Packing bins using multi-chromosomal genetic representation and better-fit heuristic. In N. R. Pal et al., editor, *Proc. of the 11th International Conference on Neural Information Processing (ICONIP 2004), Calcutta, India, LNCS 3316*, pages 181–186, 2004.

[5] E. Falkenauer. *Genetic Algorithms and Grouping Problems*. John Wiley & Sons, 1998.

[6] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, 1989.

[7] P. J. B. Hancock. An empirical comparison of selection methods in evolutionary algorithms. In T. C. Fogarty, editor, *Lecture Notes in Computer Science No. 865*, pages 80–94, 1994.

[8] G. Harik, E. Cantú-Paz, D. E. Goldberg, and B. L. Miller. The gambler's ruin problem , genetic algorithms, and the sizing of populations. *Evolutionary Computation*, 7(3):231–253, 1999.

[9] F. Herrera, M. Lozano, and J. L. Verdegay. Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis. *Artificial Intelligence Review*, 12(4):265–319, 1998.

[10] B. W. Y. Lin and R. L. Rardin. Development of a parametric generating procedure for integer programming test problems. *Journal of the ACM*, 24(3):465–472, 1977.

[11] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, 1988.

[12] C. R. Reeves. Genetic algorithms and neighbourhood search. In T. C. Fogarty, editor, *Lecture Notes in Computer Science No. 865*, pages 115–130, 1994.

The Hybrid Genetic Algorithm for Blind Signal Separation

Wen-Jye Shyr

Department of Industrial Education and Technology,
National Changhua University of Education,
No. 1, Jin-De Road, Changhua 500, Taiwan, R.O.C.
Tel.: +886-4-7232105 ext.7244; Fax:+886-4-7211097
shyrwj@cc.ncue.edu.tw

Abstract. In this paper, a hybrid genetic algorithm for blind signal separation that extracts the individual unknown independent source signals out of given linear signal mixture is presented. The proposed method combines a genetic algorithm with local search and is called the hybrid genetic algorithm. The implemented separation method is based on evolutionary minimization of the separated signal cross-correlation. The convergence behaviour of the network is demonstrated by presenting experimental separating signal results. A computer simulation example is given to demonstrate the effectiveness of the proposed method. The hybrid genetic algorithm blind signal separation performance is better than the genetic algorithm at directly minimizing the Kullback-Leibler divergence. Eventually, it is hopeful that this optimization approach can be helpful for blind signal separation engineers as a simple, useful and reasonable alternative.

1 Introduction

The Blind Signal Separation (BSS) problem consists of recovering unknown signals or sources from their several observer mixtures. Typically, these mixtures are acquired by a number of sensors in which each sensor receives signals from all sources. Blind Source Separation [1] is the central part of Independent Component Analysis (ICA) [2], a developed signal processing technique for representing given noisy signal mixtures as a linear combination of statistically independent signals. This technique is important for many applications such as communication systems [3], speech enhancement and noise reduction [4] and speech recognition [5].

From the mathematical point of view, the solution to the BSS problem is a separating matrix which transforms the mixed signals into signals with a maximal degree of independence by estimating the original source signals [6,7]. This recognition, as well as the Signal to Interference Ratio (SIR) results [8], although promising for the field of speech recognition, showed a certain incompetence in performing efficiently due to the stochastic nature of the gradient descent optimization technique that was used. Furthermore, the convergence behavior of the above optimization methods depends on the step size choice and the initial separation filter coefficient values. The Genetic Algorithm (GA) [9] is a global optimization technique, which is able to find the global optimum solution without being trapped in local minima. In this optimization framework, a novel Blind Signal Separation method based on the minimization of the separated signal cross-correlation function was proposed in [10, a method proposed to separate a noise component, which directly minimizes the Kullback-Leibler divergence using a genetic algorithm [11]. These proposed genetic algorithms for BSS can find the global optimum solution without being trapped in local minima. However, there exist some disadvantages in genetic algorithms such as premature convergence and a long convergence time. To solve these problems, local improvement procedures are used as part of the evaluation of individuals. For many problems a well-developed, efficient search strategy exists for local optimization improvement. These local search strategies compliment the genetic algorithm global search strategy, yielding a more efficient overall search strategy. The idea of combining a genetic algorithm and local search heuristics for solving combinatorial optimization problems was proposed to improve the genetic algorithm search ability in [12]. High performance was reported. For the genetic algorithm, the stopping criteria set up uses the trial and error method. When comparing hybrid genetic algorithms with the genetic algorithm method, the hybrid genetic algorithm iteration automatically stops when the predefined criteria is reached.

2 Blind Signal Separation

The basic two-input two-output (TITO) Blind Signal Separation (BSS) network is shown in Fig. 1.

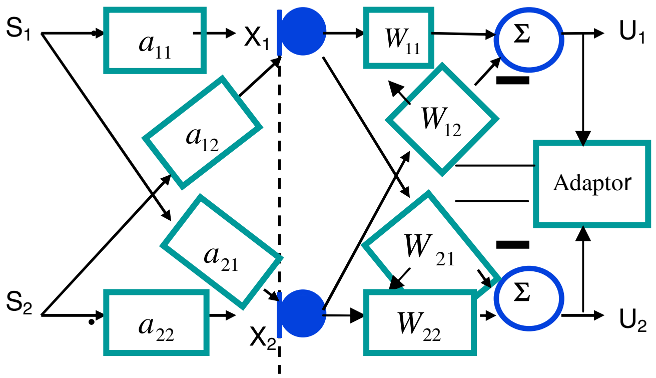


Fig. 1. Two Input Two Output (TITO) Blind Signal Separation Network

In the linear BSS problem, signals $s_i(t)$ for $i \in [1,2]$, are assumed which are considered to be zero meaned, mutually stochastic independent. These signals are acquired from a set of sensors, so $x_i(t)$ for $i \in [1,2]$ is obtained.

The BSS objective is to recover the original signals $S(t)$, without any prior knowledge of the mixing coefficients given the mixed signals $X(t)$.

Under these circumstances, a more general mixing scenario, known as the convolutive mixture, is used. To adequately express the mixing phase, the Finite Impulse Response (FIR) Matrix Algebra proposed by Lambert [13]. Using its notation, the convolved mixing case is expressed as the following form:

$$X(t) = \overline{A} \bullet S(t) \tag{1}$$

where $X(t) = [x_1(t),...,x_n(t)]^T$, $S(t) = [s_1(t),...,s_n(t)]^T$ and $\overline{A}$ is an unknown nxn non-singular matrix with each element being a FIR filter. For example, for $\overline{A}_{2x2}$, equation (1) gives as follows:

$$\overline{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \text{ and } \begin{array}{l} X_1(t) = a_{11}S_1(t) + a_{12}S_2(t) \\ X_2(t) = a_{21}S_1(t) + a_{22}S_2(t) \end{array} \tag{2}$$

Let $X_1(t)$ and $X_2(t)$ be the mixed signal of both $S_1(t)$ and $S_2(t)$. From Fig. 1, the network output $U_1(t)$ and $U_2(t)$ denotes that the separated signals can be expressed as:

$$\begin{array}{l} U_1(t) = W_{11}X_1(t) - W_{12}X_2(t) \\ U_2(t) = W_{22}X_2(t) - W_{21}X_1(t) \end{array}. \tag{3}$$

Using its notation, the source signal estimation is expressed in the form:

$$U(t) = W \bullet X(t) \tag{4}$$

where W is the separating matrix.

Substituting equation (2) into equation (3) yields

$$\begin{array}{l} U_1(t) = W_{11}a_{11}S_1(t) + W_{11}a_{12}S_2(t) - W_{12}a_{21}S_1(t) - W_{12}a_{22}S_2(t) \\ U_2(t) = W_{22}a_{21}S_1(t) + W_{22}a_{22}S_2(t) - W_{21}a_{11}S_1(t) - W_{21}a_{12}S_2(t) \end{array}. \tag{5}$$

Based on evolutionary minimization of the cross-correlation of the separated signals, the equation (6) is obtained as follows:

$$\begin{array}{l} E[U_1(t)U_2(t)] = E[W_{11}W_{22}a_{11}a_{21}S_1(t)S_1(t) - a_{11}{}^2W_{11}W_{21}S_1(t)S_1(t) \\ + W_{11}W_{22}a_{12}a_{22}S_2(t)S_2(t) - a_{12}{}^2W_{11}W_{21}S_2(t)S_2(t) - W_{12}W_{22}a_{21}{}^2S_1(t)S_1(t) \\ + W_{12}W_{21}a_{21}a_{11}S_1(t)S_1(t) - W_{12}W_{22}a_{22}{}^2S_2(t)S_2(t) + W_{12}W_{21}a_{22}a_{12}S_2(t)S_2(t)] \\ = R_1(0)[W_{11}W_{22}a_{11}a_{21} - a_{11}{}^2W_{11}W_{21} - W_{12}W_{22}a_{21}{}^2 + W_{12}W_{21}a_{21}a_{11}] + \\ R_2(0)[W_{11}W_{22}a_{12}a_{22} - a_{12}{}^2W_{11}W_{21} - W_{12}W_{22}a_{22}{}^2 + W_{12}W_{21}a_{22}a_{12}] \end{array} \tag{6}$$

If the inverse of A is find, then $W = A^{-1}$, thus

$$W = A^{-1} = \frac{1}{|A|}\begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix} \text{ and } \begin{bmatrix} W_{11} & -W_{12} \\ -W_{21} & W_{22} \end{bmatrix} = \frac{1}{|A|}\begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix} \tag{7}$$

Comparing each element in equation (7) yields

$$\begin{array}{ll} W_{11} = \frac{a_{22}}{|A|} \quad , & W_{12} = \frac{a_{12}}{|A|} \\ W_{21} = \frac{a_{21}}{|A|} \quad , & W_{22} = \frac{a_{11}}{|A|} \end{array} \tag{8}$$

From equations (7) and (8), equation (6) becomes

$$\begin{aligned}
E[U_1(t)U_2(t)] &= R_1(0)\left[\frac{a_{11}^{2}a_{22}a_{21}}{|A|^2}-\frac{a_{11}^{2}a_{22}a_{21}}{|A|^2}-\frac{a_{11}a_{12}a_{21}^{2}}{|A|^2}+\frac{a_{11}a_{12}a_{21}^{2}}{|A|^2}\right]+\\
&R_2(0)\left[\frac{a_{11}a_{12}a_{22}^{2}}{|A|^2}-\frac{a_{12}^{2}a_{21}a_{22}}{|A|^2}-\frac{a_{11}a_{12}a_{22}^{2}}{|A|^2}+\frac{a_{12}^{2}a_{21}a_{22}}{|A|^2}\right]\\
&=0
\end{aligned} \tag{9}$$

If equation (9) is equal to zero, then the output signals are independent of each other. Therefore, equation (9) minimizes the dependency among the output signals as much as possible. According to the above statement, it can be utilized to separate the mixed signals.

3 The Hybrid Genetic Algorithm

In the proposed blind signal separation method for a TITO network, the hybrid genetic algorithm is applied to minimize the output signal cross-correlation. The following evaluation (fitness) function can be introduced as follows:

$$\text{C=Xcorr}(U_1(t),\ U_2(t)) \text{ and fitness\_value=}\frac{1}{C*C} \tag{10}$$

With fitness function given in equation (10), a hybrid genetic algorithm is used to minimize the output signal cross-correlation. Genetic algorithms are used in many disciplines because of their efficient optimization capabilities. Local Search, also referred to as neighborhood search or hill-climber, is the basis of many heuristic methods. In this local search procedure the current solution (initially obtained by a constructive heuristic) is replaced by the neighboring solution that results in the greatest improvement in the evaluation function to be optimized. The process continues until a solution with no improving neighbor in a fixed generation number has been reached, i.e., until an optimum has been found. Its advantage can set up reasonable stopping criteria flexibly and avoid the shortcoming for the stopping criteria of the try and error of genetic algorithms.

The hybrid genetic algorithm combines the advantages of efficient heuristics incorporating domain knowledge and population-based search approaches for optimization problems. In this study, the usefulness of a hybrid genetic algorithm for global search and flexibly reasonable stopping criteria is shown. In this study, the usefulness of a hybrid genetic algorithm for global search and flexibly reasonable stopping criteria are shown.

In this section, the hybrid genetic algorithm implementation for blind signal separation is described.

Step 1. Initialization: A set of chromosomes is randomly generated. A chromosome is composed of genes. For this problem, the gene is w. So, the initial step is generating a collection of random matrix vectors. Define a vector and variable, to which the gradually optimized chromosome and its fitness are saved separately.

Their initial values are the first chromosome of the generated chromosome set and its fitness.

Step 2. Evaluation: For every chromosome, the fitness objective functions are calculated for evaluating its fitness. Check every chromosome's fitness step by step.

Step 3. Selection: At first, roulette wheel selection was used to select pairs of individuals for reproduction. The best fitness of the population always survives.

Step 4. Crossover: Pairs of parents are selected from these survivors. Single point crossover is selected to produce the next generation. The string of the element of matrix from the beginning of chromosome to the crossover point is copied from one parent; the rest are copied from the second parent.

Step 5. Mutation: Some useful genes are not generated in the initial step. This difficulty can be overcome using the mutation technique. The basic mutation operator randomly generates a number as the crossover position and then changes the value of this gene randomly.

Step 6. Local Search: A hill-climber in hybrid genetic algorithm, which used as local search. Apply the modified local search procedure in the current population [14].

Step a : Specify an initial solution x.

Step b : Examine a neighborhood solution y of the current solution x.

Step c : If y is a better solution than x, replace the current solution x with y and return to step b.

Step d : If the current solution x has no better than the chosen k front neighborhood solutions (i.e., if there is no worse solution among the examined k front neighborhood solutions of x), then end this procedure. Otherwise return to Step 2.

This algorithm is terminated if no better solution is found among k neighborhood solutions that are randomly selected from the neighborhood of the current solution. Therefore, if a very small value of k is used, the local search procedure may be terminated soon. However, it cannot be certain of good convergence. Conversely, if a large value of k is used, the local search procedure examines many solutions. But it will take long convergence time. In the modified local search procedure method, according to the experience, the value of k in the local search procedure in our hybrid genetic algorithm is setup.

Step 7. Termination Criteria: If the number of the current generation is equal to or larger than the predefined number of generations, end the algorithm. Otherwise return to Step 2.

4 Simulation Results

The following simulations were performed to evaluate the performance of the proposed method for blind signal separation for a TITO network.

Example 1

The source signals are illustrated in Fig. 2 and defined by

$$S = \begin{bmatrix} \sin(400t)\cos(30t) \\ sign[\sin(500t + 9\cos(40t))] \end{bmatrix}.$$

Each elements of mixing matrix $\overline{A}$ are chosen in [-1, 1] and shown as follows:

$$\overline{A} = \begin{bmatrix} 1.0 & 0.4 \\ 0.6 & 1.0 \end{bmatrix},$$

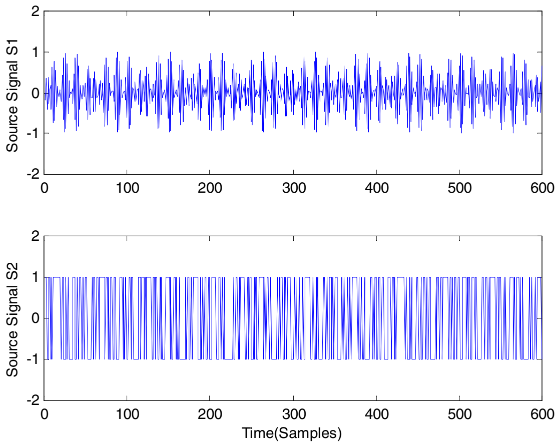


Fig. 2. The source signals S1 and S2

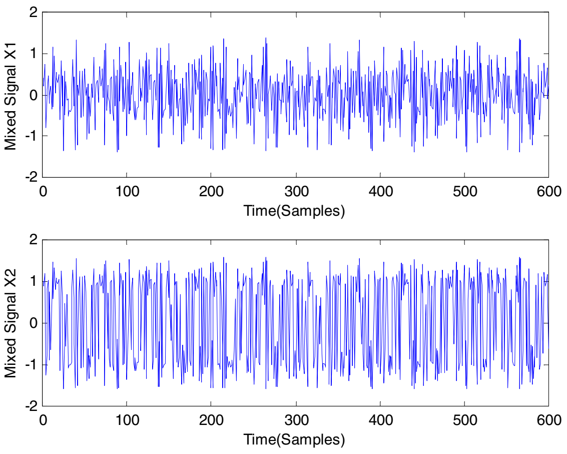


Fig. 3. The mixed signals X1 and X2

The defining parameters for the hybrid genetic algorithm are as follows: population size=20, probability of crossover =0.65, probability of mutation=0.01. The stopping criteria of hybrid genetic algorithm are no better solution among the examined 50 neighbor generations, and the number of the current generation is equal to or larger than 200 generations.

The mixed signals $x_i(t)$ are depicted in Fig. 3. The convergence behavior is illustrated in Fig. 4. From Fig 4, it can be shown that the hybrid genetic algorithms can minimization the cross correlation of the output signal. As the result of the simulation, the separating matrix W is shown as follows:

$$W = \begin{bmatrix} 1.000 & 0.3989 \\ 0.6059 & 1.0000 \end{bmatrix}.$$

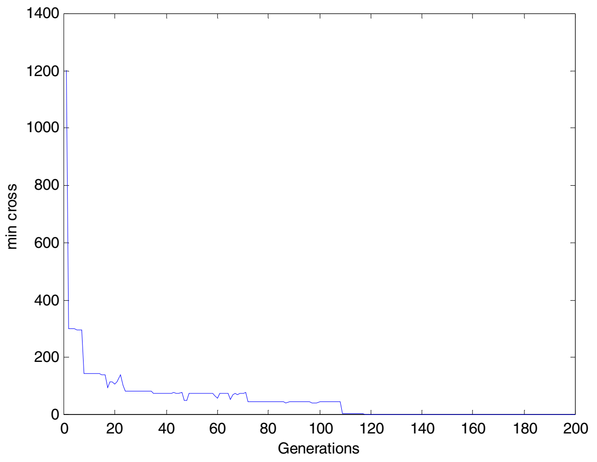


Fig. 4. Convergence of the cross-correlation (example1)

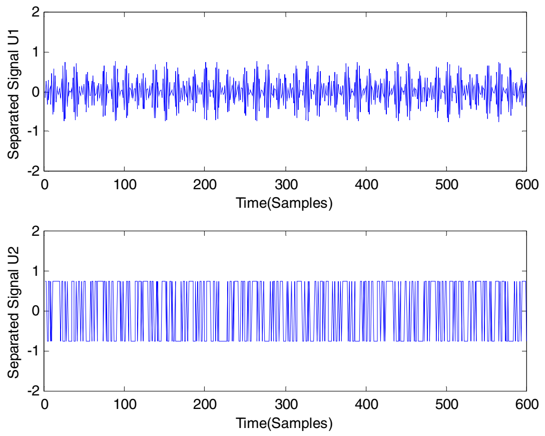


Fig. 5. The separated signals U_1 and U_2

The separated signals U are shown in Fig 5. Comparing source signals in Fig 2 with the separated signals in Fig 5, the best result is achieved for the separated signal, which corresponds at the same time to the best primary signal.

The blind signal separation using performance the hybrid genetic algorithm based on evolutionary minimization of the cross-correlation has proven more efficient than that based on the genetic algorithm, which directly minimize the Kullback-Leibler divergence [10].

Example 2

The defining parameters for the MA are as follows: population size=40, probability of crossover =0.7, probability of mutation=0.008. The convergence behavior is illustrated in Figure 6. From Figure 6, it can be shown that the MA can minimization the cross correlation of the output signal.

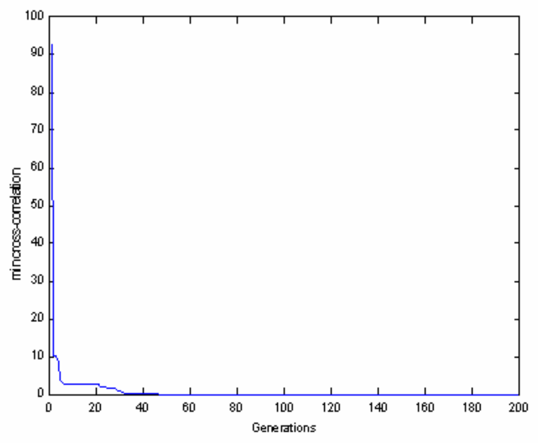


Fig. 6. Convergence of the cross-correlation (example 2)

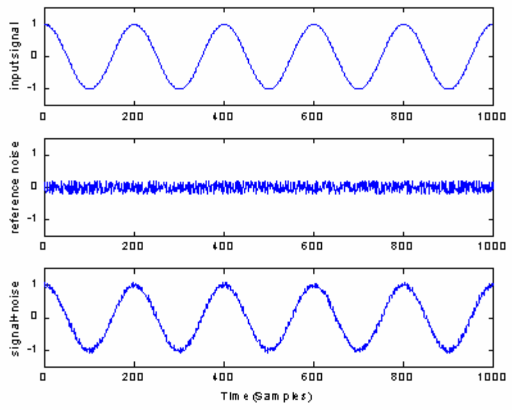


Fig. 7. Input signal, reference noise and signal with noise

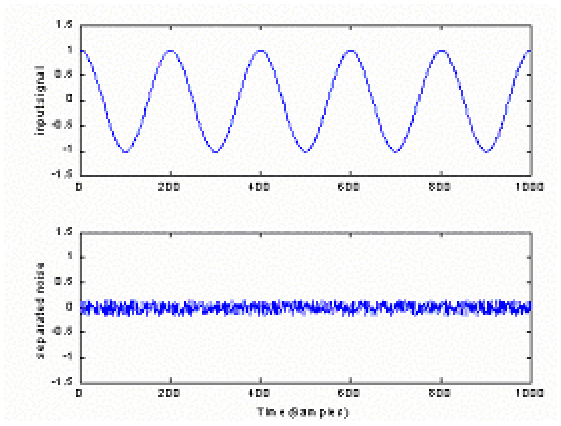


Fig. 8. Results of separated signal and separated noise

The mixing matrix with each element being a FIR filter are shown as follows:

$$\bar{A} = \begin{bmatrix} 1.0 & 0.4 \\ 0.6 & 1.0 \end{bmatrix}, \qquad W = \begin{bmatrix} 1.000 & 0.3891 \\ 0.6049 & 1.0000 \end{bmatrix}.$$

The input signal S_1, reference noise S_2 (a white noise with variance $\sigma^2 = 1/75$) and input signal with reference noise X_1 are shown in Fig 7. The separated signal U_1 and separated noise U_2 are shown in Fig 8. From Fig 7 to Fig 8, it is easy to see that the best result is achieved for the separated signal, which corresponds at the same time to the best primary signal.

5 Conclusion

This paper presented a hybrid genetic algorithm for blind signal separation based on the minimization of the output signal cross-correlation. The network convergence behaviour was demonstrated using simulation results. As the result of simulation, the blind signal separation performance based on the hybrid genetic algorithm verified that evolutionary minimization of the cross-correlation is better than that based on the genetic algorithm, which directly minimizes the Kullback-Leibler divergence. Eventually, it is hopeful that this optimization approach can be helpful for blind signal separation engineers as a simple, useful and reasonable alternative.

References

1. Comon, P., Jutten, C., and Herault, J.: Blind separation of sources, Part II: Problem statements, Signal Processing, 24(1991) 11-20
2. Comon, P.: Independent component analysis - A new concept, Signal Processing, 36(1994) 287-314

3. Fren, M., and Kammeyer, K.: Application of source separation slgorithms for mobile communication environment, the 1st International Conference Workshop on ICA & Signal Separation, Aussois, France, (1999) 431-436
4. Girolami, M.: Noise reduction and speech enhancement via temporal anti-hebbian learning, ICASSP, 2(1998) 1233-1236
5. Choi, S., and Cichocki, A.: Adaptive blind separation of speech signals: Cocktail party problem, International Conference of Speech Processing, Seoul, Korea, (1997) 6117-6221
6. Yen, K., and Zhao, Y.: Co-channel speech separation for robust automatic speech recognition, ICASSP, 2 (1997) 859-862
7. Buchner, H., Aichner, R., KellermannH, W.: A Generalization of a Class of Blind Source Separation Algorithms for Convolutive Mixtures, Proc. IEEE Int. Symposium on Independent Component Analysis and Blind Signal Separation, (2003) 945-950
8. Koutras, A., Dermatas, E., and Kokkinakis, G.: Blind signal separation and speech recognition in the frequency domain, the 6th International Conference on Electronics, Circuits and Systems, 1(1999) 427-430
9. Koutras, A., Dermatas, E., and Kokkinakis, G.: Recognizing simultaneous speech: A genetic algorithm approach, Euro-Speech, Budapest, 6(1999) 2551-2554
10. Chen, Y. C., Shyr, W. J., Su, T. J., and Wang, B. W.: Memetic algorithm for adaptive infinite impulse response filtering, Proceeding of the 2002 IEEE International Symposium on Intelligent Signal Processing and Communication Systems, (2002) 1-5
11. Freisleben, B., and Merz, P.: A genetic local search algorithm for solving symmetric and asymmetric travelling salesman problems, Proceedings of the IEEE International Conference on Evolutionary Computation, IEEE Press, (1996) 616-621
12. Burke, E. K., and Smith, A. J.: Hybrid evolutionary techniques for the maintenance scheduling problem, IEEE Transactions on Power Systems, 15(1)(2000) 122-128
13. Lambert, R.: Multi channel blind deconvolution: FIR matrix algebra and separation of multipath mixtures, Ph.D Thesis, University of Southern California, Dept.of Electrical Engineering, (1996)
14. Ishibuchi, H., and Murata, T.: Multi-objective genetic local search algorithm, Proceedings of the IEEE International Conference on Evolutionary Computation, IEEE Press, (1996) 119-124

Genetic Algorithm for Satellite Customer Assignment

S.S. Kim[1], H.J. Kim[2], V. Mani[3], and C.H. Kim[1]

[1] Department of Industrial Engineering
Kangwon National University, Chunchon 200-701, Korea
[2] Department of Control and Instrumentation Engineering
Kangwon National University, Chunchon 200-701, Korea
[3] Department of Aerospace Engineering,
Indian Institute of Science, Bangalore, India

Abstract. The problem of assigning customers to satellite channels is considered. Finding an optimal allocation of customers to satellite channels is a difficult combinatorial optimization problem and is shown to be NP-complete in an earlier study. We propose a genetic algorithm (GA) approach to search for the best/optimal assignment of customers to satellite channels. Various issues related to genetic algorithms such as solution representation, selection methods, genetic operators and repair of invalid solutions are presented. A comparison of this approach with the standard optimization method is presented to show the advantages of this approach in terms of computation time.

1 Introduction

The problem of satellite customer assignment is considered in [1], and an excellent discussion on bandwidth resource issues related to satellite-based communication is presented. An integer programming formulation is presented for the satellite customer assignment problem. It is shown in [1] that the optimal assignment of customers to channels has real and observable costs and benefits, both in terms of dollars and customer ratings. The basic idea in their study is to determine the optimal resource utilization. Efficient resource utilization is one problem that has been studied in satellite communication [2,3]. A detailed overview of the scheduling problems that arise in satellite communication is described in [4]. Satellite customer assignment problem has a lot of similarities with the well-known generalized assignment problem (GAP). The GAP is known to be NP-complete combinatorial optimization problem and has received a lot of attention in literature [5]. The GAP involves finding the minimum cost assignment of I jobs to J machines (agents) such that each job is exactly assigned to only one machine, subject to machine's available capacity. Another problem in this context well studied is flow shop scheduling [6].

Genetic Algorithm (GA) is perhaps the most well-known of all evolution based search technique. GA is a search algorithm based on natural selection that transforms a set of individuals within the population of solutions into a new set for the next generation using genetic operators such as crossover and mutation [7,8,9,10]. A survival of the fittest strategy is adopted to identify the best solutions and subsequently genetic operators are used to create new solutions for the next generation. This process is repeated generation after generation until a satisfactory solution is found. Genetic algorithms have been successfully used to obtain solutions for many combinatorial optimization problems. It is difficult to include all the applications. So we refer only applications that are closely related to our satellite customer assignment problem.

Genetic algorithm for a combinatorial optimization problem works as follows: The search space contains all the search nodes for the given problem. GA starts with an initial population of search nodes from the search space. Each search node has a fitness value assigned to it using the objective function. New search nodes are generated for next generation based on the fitness value and applying genetic operators to the current search nodes. These process is repeated for generation after generation until the algorithm converges.

We propose a genetic algorithm with a new solution representation for the satellite customer assignment problem. This new solution representation is a particular sequence of integers. The integers are the channel numbers to which a customer is assigned. We present genetic operators for this new solution representation and a repair algorithm, when the genetic operators produce an invalid solution. We show via numerical examples that this genetic algorithm approach can obtain near-optimal or optimal solution very fast, in terms of computation time, in comparison with standard optimization method.

2 Satellite Customer Assignment Problem

Problem Formulation: For ease of understanding, we follow the same notation used in an earlier study [1]. Let there be I customers to be assigned to one of the J channels. As in [1], we assume the following data is available:

- SBW_j : satellite bandwidth available in channel j,
- SP_j : satellite power available in channel j,
- CBW_i : bandwidth required by customer i, and
- CP_i : power required by customer i.

The decision variable is x_{ij}. The decision variable $x_{ij} = 1$ if customer i is assigned to channel j and $x_{ij} = 0$ otherwise. The satellite customer assignment problem is

$$Minimize \quad \sum_{j=1}^{J} \left| \frac{\sum_{i=1}^{I} CBW_i x_{ij}}{SBW_j} - \frac{\sum_{i=1}^{I} CP_i x_{ij}}{SP_j} \right| . \qquad (1)$$

The constraints are:

$$\sum_{i=1}^{I} CBW_i * x_{ij} \leq SBW_j, \quad j = 1, 2, ..., J \tag{2}$$

$$\sum_{i=1}^{I} CP_i * x_{ij} \leq SP_j, \quad j = 1, 2, ..., J \tag{3}$$

$$\sum_{j=1}^{J} x_{ij} = 1, \quad i = 1, 2, ..., I \tag{4}$$

$$x_{ij} = 0 \ or \ 1 \tag{5}$$

This objective function is used in the earlier study [1]. The objective function (1) minimizes the total deviation of fraction of bandwidth utilized from fraction of power utilized. The constraint (2) represents that the capacity restriction of available bandwidth, and the constraint (3) takes into account the capacity restrictions of available power. Constraint (4) ensures that each customer is assigned to only one channel. From the above, we see that, this satellite customer assignment problem is similar to the generalized assignment problem and hence it is NP-complete. Genetic algorithms have been used for such NP-complete combinatorial optimization problems [11,12,13,14]. Now we will illustrate the various steps involved in applying genetic algorithms to our satellite customer assignment problem.

Solution Representation: Solution representation is an important step in the genetic algorithm. In earlier studies of genetic algorithms a binary representation of the solution is used. For our problem, we use the solution representation as given in [11,12,13]. The solution representation for our problem is an I dimensional vector of integers. The integer values range from i to J, the number of channels available. One possible solution (assignment) for 5 customers and 3 channels in our representation is

$$\{2 \ \ 3 \ \ 1 \ \ 1 \ \ 2\}. \tag{6}$$

This solution representation takes care of the constraint (4). Also note that this representation does not take care of the constraints (2) and (3). A valid solution in our problem should satisfy the constraints (2) and (3). Hence, we must keep in mind the constraints (2) and (3), when we generate initial population and design genetic operators.

Consider an example with 5 customers (I=5) and 3 channels (J=3). Let one solution be {3 1 3 2 1}. The meaning of the representation is:

Customers 1 and 3 are assigned to channel 3.
Customers 2 and 5 are assigned to channel 1.
Customer 4 is assigned to channel 2.

This is equivalent to: $x_{13} = 1$, $x_{33} = 1$, $x_{21} = 1$, $x_{51} = 1$, and $x_{42} = 1$ and all other $x_{ij} = 0$ for $i = 1, ..., 5$ and $j = 1, ..., 3$. This solution $\{3\ 1\ 3\ 2\ 1\}$ is a valid solution only if the constraints (2) and (3) are satisfied. The advantage is that the solution representation is a vector of size I (number of customers), and constraint (4) is automatically satisfied.

Population Initialization: The first step in genetic algorithms is to create an initial population of solutions to the problem. All the solutions in the initial population should satisfy the constraints (2) and (3). In our study, the initial population is chosen completely at random. In order to satisfy the constraints (2) and (3), we have used a repair algorithm. By using this repair algorithm, we convert any solution in the population which violates the constraints (2) and (3) to a valid solution. The repair algorithm used in our study is given below.

Repair Algorithm: In an invalid solution either constraint (2) or constraint (3) or both are violated. The repair algorithm first find out the channel number for which the constraints are violated. It randomly removes one of the customer assigned to that channel. This removed customer is assigned to one of the other channels. We will show how our repair algorithm works with the following numerical example.

Numerical Example: Let the number of customers (I) be 5, and the number of channels (J) be 3. The channels are numbered from 0 to 2. Consider the customers with the following bandwidth requirements: $CBW_1 = 5$, $CBW_2 = 4$, $CBW_3 = 6$, $CBW_4 = 7$, and $CBW_5 = 3$. The satellite bandwidth available in channels are: $SBW_0 = 10$, $SBW_1 = 16$, and $SBW_2 = 14$. Consider the solution $\{2\ 0\ 0\ 2\ 0\}$. In this solution customers 2,3 and 5 are assigned to channel 0. This is an invalid solution because constraint (2) is violated for channel 0. Assigned bandwidth is $4+6+3 = 13$ and the available channel bandwidth is 10. Randomly select one of the customers assigned to this channel (say, customer 3 is randomly selected). Randomly select another channel and assign this customer 3 to that channel. The channel randomly selected is 1. Now the solution after the repair algorithm is $\{2\ 0\ 1\ 2\ 0\}$. In this example, we considered constraint (2) is violated. Repeat this repair algorithm to constraint (3) also if it is violated. In this manner we will always have a valid solution.

Selection Function: Selection methods use survival of the fittest idea. The selection of a solution from the population of solutions to produce new solutions for next generation plays an important role. In literature there are several selection schemes such as roulette wheel selection and its extensions, scaling techniques, tournament, elitist models and ranking methods are presented [7,8,9,10]. In our study we have used roulette selection method.

2.1 Genetic Operators

Genetic operators such as crossover and mutation provide basic search mechanism in genetic algorithms. We now describe the genetic operators used in our study.

Crossover Operator: The crossover operator is considered as the main search operator in genetic algorithms. The role of crossover operator is to produce new solutions that have some portions of both parent solutions. The crossover operator takes two solutions (parents) from the existing population and produces two new solutions. In our study, we have used the simplest form of crossover namely single point crossover.

Single Point Crossover: Let C_1 and C_2 are the two solutions selected for crossover operation. This operator first selects a random point k in the solution. Two new solutions are produced as:

$$\begin{aligned} C_1 &= \{0\ 1\ 1\ 2\ 0\}, \\ C_2 &= \{1\ 0\ 0\ 0\ 2\}. \end{aligned}$$

Let the crossover point be 3. The two new solutions produced are H_1 and H_2.

$$\begin{aligned} H_1 &= \{0\ 1\ 0\ 0\ 2\}, \\ H_2 &= \{1\ 0\ 1\ 2\ 0\}. \end{aligned}$$

It is possible in the above crossover method that some of the new solutions produced may not be a valid solution to the problem. In other words, the constraints (2) and (3) may not be satisfied. In that situation, we use the repair algorithm and obtain a valid solution. Many crossover methods are available in literature.

Mutation Operator: The mutation operator uses one solution to produce a new solution. Mutation operator is needed to ensure diversity in the population and to avoid premature convergence and local minima problems. The mutation operator used in our study is the following.

Let C_1 be the solution selected for mutation operation. This operator first selects a mutation point i randomly. The new solution H_1 is generated by changing the channel at the mutation point by another randomly generated channel.

$$C_1 = \{0\ 1\ 1\ 2\ 0\}.$$

Let the mutation point be 3. The new solution produced is H_1.

$$H_1 = \{0\ 1\ 2\ 2\ 0\}.$$

Here the channel at the mutation point is changed from 1 to 2. In this operation one customer from a channel is removed and assigned to a different channel. Here also if H_1 is an invalid solution, we use the repair algorithm to get a valid solution.

Fitness Function: Fitness is the driving force in genetic algorithms. The only information used in the execution of genetic algorithms is the observed values of fitness of the solutions in the population. The fitness function is the objective function. In our problem, the objective function minimizes the total deviation of fraction of bandwidth utilized from fraction of power utilized. The calculation of the fitness function is easy. The solution gives the assignment of customers to various channels. In other words x_{ij}s are given by the solution. We substitute the values of x_{ij} in the objective function and obtain the value. The value represents total deviation of fraction of bandwidth utilized from fraction of power utilized for this solution. Our problem is a minimization problem whereas the genetic algorithms will try to maximize the fitness. Hence the fitness for our problem is

$$F = - \sum_{j=1}^{J} \left| \frac{\sum_{i=1}^{I} CBW_i x_{ij}}{SBW_j} - \frac{\sum_{i=1}^{I} CP_i x_{ij}}{SP_j} \right| . \tag{7}$$

Termination Function: In genetic algorithm, in each generation, solutions are selected on the basis of their fitness and subject to genetic operations such as crossover and mutation. The evolution process is repeated until a termination criterion is satisfied. The most frequently used convergence criterion is the average fitness value from generation to generation. This algorithm stops if these average values are equal. Other convergence criteria are population convergence criterion: the algorithm stops when the solutions in the population are the same in two successive generations. Another criterion used for termination is a specified maximum number of generations. In our studies, we have used a specified maximum number of generations.

3 Simulation Results

The GA approach to search for the best assignment of customers to satellite channels is tested with some test problems. Two sets of test problems were generated. The first set of problems were generated with 5 customers and 3 channels. The second set of test problems were generated with 20 customers and 10 channels.

For the first set of problems (I=5, and J=3), parameters used in our study are given as in Table 1.

Table 1. Satellite Bandwidth (SBW_j), and Power (SP_j) for the channels

Problem	SBW_1	SBW_2	SBW_3	SP_1	SP_2	SP_3
1.1	30	35	40	40	45	50
1.2	9	11	9	21	17	11
1.3	10	16	14	15	31	14
1.4	20	13	11	33	21	15
1.5	11	17	19	28	14	17
1.6	18	11	19	21	21	21

The advantage with genetic algorithm approach is that it starts with random solutions to the problem and modify the solutions in successive generations, and the best solution is obtained. So in our study, we have used the genetic algorithm for each problem 5 times and the best assignment of customers to channels is taken as the solution.

To verify the assignment of customers to satellite channels obtained using genetic algorithms, we have solved the same problems (1.1-1.6) using the zero-one linear programming formulation given in [1]. The objective function in our problem is non-linear and hence the following approach is used in [1]. The problem is:

$$Minimize \ \sum_{j=1}^{J}(d_j^+ \ + \ d_j^-)$$

The constraints are:

$$\frac{\sum_{i=1}^{I} CBW_i * x_{ij}}{SBW_j} - \frac{\sum_{i=1}^{I} CP_i * x_{ij}}{SP_j} - (d_j^+ \ + \ d_j^-) \ = \ 0$$
$$\sum_{i=1}^{I} CBW_i * x_{ij} \ \leq \ SBW_j, \quad j = 1,2,...,J$$
$$\sum_{i=1}^{I} CP_i * x_{ij} \ \leq \ SP_j, \quad j = 1,2,...,J$$
$$\sum_{j=1}^{J} x_{ij} \ = \ 1, \quad x_{ij} \ = \ 0 \ or \ 1, \quad i = 1,2,...,I$$
$$d_j^+ \ , \ d_j^- \ \geq \ 0, \quad for \ (j = 1,...,J) \tag{8}$$

It is shown that the number of possible solutions to this problem is J^I [1]. For our problem with 5 customers ($I = 5$) and 3 channels ($J = 3$), the number of possible solutions is 243. Note that all the 243 solutions may not be feasible. If the number of channels and the number of customers increase then the number of possible solutions increases rapidly. We have solved the problems (1.1-1.6) formulated as zero-one linear programming using LINGO 4.0 solver. The best assignment obtained from genetic algorithm approach is the same as the optimal assignment obtained for all these problems. Here, we are able to obtain the optimal assignment because the search space is small. The computation time is almost same for both the genetic algorithm approach and the LINGO 4.0 solution.

Next, we consider a 20 customer (I=20) and 10 channel (J=10) problem. This is really a tough problem because the number of possible solutions are 10^{20}. Hence, the genetic algorithm approach will be very much useful here. The numerical values of satellite bandwidth available in channels (SBW) and the satellite power available (SP), used in our study are given in Table 2.

For the problem 2.1, the values of CBW_i for the customers from 1 to 20 are 5, 5, 5, 5, 4, 4, 3, 7, 6, 5, 6, 3, 6, 7, 4, 3, 6, 6, 4, and 4, respectively. The values of SP_i for customers from 1 to 20 are 5, 8, 9, 5, 7, 6, 8, 6, 7, 8, 6, 7, 9, 9, 7, 7, 5, 7, 5, and 9, respectively.

Table 2. Satellite Bandwidth (SBW_j), and Power (SP_j) for the channels

Problem	SBW_1	SBW_2	SBW_3	SBW_4	SBW_5	SBW_6	SBW_7	SBW_8	SBW_9	SBW_{10}
2.1	22	13	15	20	15	15	15	22	19	13
2.2	20	22	16	17	23	14	14	15	21	14
Problem	SP_1	SP_2	SP_3	SP_4	SP_5	SP_6	SP_7	SP_8	SP_9	SP_{10}
2.1	31	19	24	26	25	24	18	28	23	29
2.2	33	18	30	26	28	31	28	19	33	18

For the problem 2.2, the values of CBW_i for each customer are: 6, 7, 7, 7, 4, 7, 4, 6, 6, 4, 3, 3, 6, 3, 6, 6, 4, 3, 7, and 6, respectively. The values of SP_i are 8, 6, 9, 8, 6, 7, 6, 9, 8, 9, 9, 9, 8, 7, 5, 8, 6, 7, 7, and 7, respectively.

Analysis of the Results: The optimal solution obtained for problem 2.1, using LINGO is: {0 2 5 7 7 0 4 0 4 2 5 0 3 7 3 0 3 7 3 5}. The meaning of the assignment is:

- Customers 1, 6, 8, 12 and 16 are assigned to channel 0.
- Customers 2 and 10 are assigned to channel 2.
- Customers 13, 15, 17, and 19 are assigned to channel 3.
- Customers 7, and 9 are assigned to channel 4.
- Customers 3, 11, and 20 are assigned to channel 5.
- Customers 4, 5, 14, and 18 are assigned to channel 7.

Note that no customers are assigned to channels 1, 6, 8 and 9. The objective function value for this assignment is 0.0000. The computation time to obtain this assignment is 2 hours on Pentium machine (over Windows 2000).

Now consider the genetic algorithm approach. The assignment obtained at the end of GA is: {7 4 4 3 0 1 7 0 3 4 1 3 5 0 5 1 7 3 7 0}. Note that no customers are assigned to channels 2, 6, 8 and 9. The objective function value for this assignment is 0.003246. The computation time to obtain this assignment is 46 seconds.

3.1 How Good Is Genetic Algorithm Assignment?

To show that the customer assignment obtained from genetic algorithm is close to the optimal assignment, consider problem 2.1. First we consider the optimal assignment. For each channel, we evaluate the objective function:

- Customers 1, 6, 8, 12, and 16 are assigned to channel 0: The objective function for this assignment is (22/22)-(31/31)=0.
- Assign customers 2 and 10 to channel 2:(10/15)-(16/24)=0.
- Assign customers 13, 15, 17 and 19 to channel 3:(20/20)-(26/26)=0.
- Assign customers 7 and 9 to channel 4:(9/15)-(15/25)=0.
- Assign customers 3, 11 and 20 to channel 5:(15/15)-(24/24)=0.
- Assign customers 4, 5, 14 and 18 to channel 7:(22/22)-(28/28)=0.

Hence the objective function value is 0, which is optimal. Now, consider the assignment obtained from genetic algorithm. For each channel, similarly, we evaluate the objective function and found channel 7, for which the objective function value is not zero and is given as:

– Assign customers 1, 7, 17 and 19 to channel 7:(18/22)-(23/28)=0.0032467.

We see that only for channel 7, the objective function value is non-zero and is 0.0032467. From this assignment (or solution), if we try any crossover or mutation, the resulting solution will be either inferior or invalid. So, our GA will result in this assignment. Hence, we can say that this solution is very close to the optimal solution. But the computation time taken by GA to find a solution close to the optimal solution is just 46 seconds. The LINGO solver takes 2 hours to obtain the optimal solution. In terms of computation time, our GA approach gives a good solution to this problem.

We consider problem 2.2, and conduct a similar study between optimal assignment and assignment obtained from GA. The computation time for optimal solution using LINGO is 6 hours and we obtain a good solution in genetic algorithm in just 56 seconds.

It is known that the solution obtained from genetic algorithms can not be guaranteed to be optimal; i.e., no formal proof of optimality is available. But for a large class of difficult combinatorial optimization problems, it has been shown that the GA produces solution that are very close to the optimal solution, in less computation time. The GA parameters used in our simulation are: the population size is 20, the crossover probability is 0.8, and the mutation probability is 0.5.

4 Discussions and Conclusions

Discussions: Genetic algorithms have been successfully used for many scheduling problems. The performance of genetic algorithms depend on a number of factors such as: population size, initial population, type of genetic operators and the probability of genetic operations. The performance of genetic algorithms is compared with heuristics for the case of scheduling problems in parallel processors is presented in [15]. This study [15] will be very much useful for understanding the conditions under which a genetic algorithm performs best.

We can see similar assignment problems arise in mobile networks. They are known as channel-assignment problem for mobile cellular systems and are solved using graph-coloring algorithms, heuristics, and optimization methods. The objective function considered is driven by considerations of battery operations (which impacts the lifetime of satellite) [1].

Conclusions: Assigning customers to satellite channels is a difficult combinatorial optimization problem. Hence, a genetic algorithm approach is presented for this problem. In this approach, a new solution representation scheme is proposed. This new solution representation is an ordered structure of integer numbers. The integer numbers are the channel numbers to which a customer is assigned. We present genetic operators for this new solution representation. A repair algorithm is also presented to convert the invalid solutions to valid solution. Numerical examples are presented to show that this approach takes very little computation time to get very-near optimal assignment for this problem, in comparison with standard optimization techniques.

Acknowledgement. This work was in part supported by ITRC under the auspices of Ministry of Information and Communication, Korea.

References

1. C. H. Scoot, O. G. Skelton, and E. Rolland, "Tactical and strategic models for satellite customer assignment," *Journal of the Operational Research Society*, vol. 51. pp. 61-71, 2000.
2. H. Lee, D. H. Ahn, and S. Kim, "Optimal routing in non-geostationary satellite ATM networks with intersatellite link capacity constraints," *Journal of the Operational Research Society*, vol. 54. pp. 401-409, 2003.
3. M. Dell'Amico, and S. Martello, "Open shop, satellite communication and a theorem by Egervary (1931)," *Operations Research Letters*, vol. 18, pp. 209-211, 1996.
4. C. Prins, "An overview of scheduling problems arising in satellite communications," *Journal of the Operational Research Society*, vol. 45. pp. 611-623, 1994.
5. D. Cattrysse, and L. N. Van Wassenhove, "A survey of algorithms for the generalized assignment problem," *European Journal of Operational Research*, vol. 60, pp. 260-272, 1992.
6. D. C. Montgomery, and L. A. Johnson, *Operations Research in Production Planning Scheduling and Inventory Control*, John Wiely & Sons, Chichester, 1974.
7. H. J. Holland, *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor, 1975.
8. D. E. Goldberg, *Genetic algorithms in search, optimization and machine learning*, Addison-Wesley, New York, 1989.
9. L. David, *Handbook of Genetic Algorithms*, Van Nostrand Reingold, New York, 1991.
10. Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, AI Series, Springer-Verlag, New York, 1994.
11. P. C. Chu, and J. E. Beasley, "A genetic algorithm for the generalized assignment problem," *Computers and Operations Research*, vol. 24, pp. 17-23, 1997.
12. O. Etiler, B. Toklu, M. Atak, and J. Wilson, "A genetic algorithm for flow shop scheduling problems," *Journal of the Operational Research Society*, vol. 55. pp. 830-835, 2004.
13. M. D. Kidwell, and D. J. Cook, "Genetic algorithm for dynamic task scheduling," *Proceedings of the International Phoenix Conference on Computers and Communication*, pp. 61-67, 1994.
14. S. S. Kim, A. E. Smith, and J. H. Lee, "A memetic algorithm for channel assignment in wireless FDMA systems," *Computers and Operations Research*, In press, Corrected proof available online.
15. A. Y. Zomaya, C. Ward, and B. Macey, "Genetic scheduling for parallel processor systems: Comparative studies and performance issues," *IEEE Transactions on Parallel and Distributed Systems*, vol. 10, pp. 795-812, 1999.

A Look-Ahead Fuzzy Back Propagation Network for Lot Output Time Series Prediction in a Wafer Fab*

Toly Chen

Department of Industrial Engineering and Systems Management, Feng Chia University, 100, Wenhwa Road, Seatwen, Taichung City, Taiwan
tolychen@ms37.hinet.net
http://www.geocities.com/tinchihchen/

Abstract. Lot output time series is one of the most important time series data in a wafer fab (fabrication plant). Predicting the output time of every lot is therefore a critical task to the wafer fab. To further enhance the effectives and efficiency of wafer lot output time prediction, a look-ahead fuzzy back propagation network (FBPN) is constructed in this study with two advanced features: the future release plan of the fab is considered (look-ahead); expert opinions are incorporated. Production simulation is also applied in this study to generate test examples. According to experimental results, the prediction accuracy of the look-ahead FBPN was significantly better than those of four existing approaches: multiple-factor linear combination (MFLC), BPN, case-based reasoning (CBR), and FBPN without look-ahead, by achieving a 12%~37% (and an average of 19%) reduction in the root-mean-squared-error (RMSE) over the comparison basis – MFLC.

1 Introduction

Lot output time series is one of the most important time series data in a wafer fab. Predicting the output time for every lot in a wafer fab is a critical task not only to the fab itself, but also to its customers. After the output time of each lot in a wafer fab is accurately predicted, several managerial goals can be simultaneously achieved [5]. Predicting the output time of a wafer lot is equivalent to estimating the cycle (flow) time of the lot, because the former can be easily derived by adding the release time (a constant) to the latter. There are six major approaches commonly applied to predicting the output/cycle time of a wafer lot: multiple-factor linear combination (MFLC), production simulation (PS), back propagation networks (BPN), case based reasoning (CBR), fuzzy modeling methods, and hybrid approaches. Among the six approaches, MFLC is the easiest, quickest, and most prevalent in practical applications. The major disadvantage of MFLC is the lack of forecasting accuracy [5]. Conversely, huge amount of data and lengthy simulation time are two shortages of PS. Nevertheless, PS is the most accurate output time prediction approach if the related databases are continuingly updated to maintain enough validity, and often serves as a benchmark for

* This work was support by the National Science Council, R.O.C.

evaluating the effectiveness of another method. PS also tends to be preferred because it allows for computational experiments and subsequent analyses without any actual execution [3]. Considering both effectiveness and efficiency, Chang et al. [4] and Chang and Hsieh [2] both forecasted the output/cycle time of a wafer lot with a BPN having a single hidden layer. Compared with MFLC approaches, the average prediction accuracy measured with the root mean squared error (RMSE) was considerably improved with these BPNs. On the other hand, much less time and fewer data are required to generate an output time forecast with a BPN than with PS. Chang et al. [3] proposed a k-nearest-neighbors based case-based reasoning (CBR) approach which outperformed the BPN approach in forecasting accuracy. Chang et al. [4] modified the first step (i.e. partitioning the range of each input variable into several fuzzy intervals) of the fuzzy modeling method proposed by Wang and Mendel [15], called the WM method, with a simple genetic algorithm (GA) and proposed the evolving fuzzy rule (EFR) approach to predict the cycle time of a wafer lot. Their EFR approach outperformed CBR and BPN in prediction accuracy. Chen [5] constructed a fuzzy BPN (FBPN) that incorporated expert opinions in forming inputs to the FBPN. Chen's FBPN was a hybrid approach (fuzzy modeling and BPN) and surpassed the crisp BPN especially in the efficiency respect.

To further enhance the effectiveness and efficiency of wafer lot output time prediction, a look-ahead FBPN is constructed in this study with two advanced features:

1. The future release plan of the fab is considered (look-ahead).
2. Expert opinions are incorporated to enhance the network learning efficiency.

PS is also applied in this study to generate test examples. Using simulated data, the effectiveness of the look-ahead FBPN is shown and compared with those of four existing approaches, MFLC, BPN, CBR, and FBPN without look-ahead.

2 Methodology

The look-ahead FBPN is composed of two parts. Firstly, the future release plan of the fab is incorporated (look-ahead).

2.1 Incorporating the Future Release Plan (Look-Ahead)

All aforementioned traditional methods are based on the historical data of the fab. However, a lot of studies have shown that the performance of sequencing and scheduling in a fab relies heavily on the future release plan, which has been neglected in this field. In addition, the characteristic re-entrant production flows of a fab lead to the phenomenon that a lot that will be released in the future might appear in front of another lot that currently exists in the fab. For these reasons, to further improve the accuracy of wafer lot output time prediction, the future release plan of the fab has to be considered (look-ahead). There are many possible ways to incorporate the future release plan in predicting the output time of a wafer lot currently existing in the fab. In this study, the three nearest future discounted workloads on the lot's processing route (according to the future release plan) are proposed for this purpose:

1. *The 1st nearest future discounted workload* ($FDW^{(1)}$): the sum of the (processing time/release time)'s of the operations of the lots that will be released within time [now, now + T_1].
2. *The 2nd nearest future discounted workload* ($FDW^{(2)}$): the sum of the (processing time/release time)'s of the operations of the lots that will be released within time [now + T_1, now + T_1 + T_2].
3. *The 3rd nearest future discounted workload* ($FDW^{(3)}$): the sum of the (processing time/release time)'s of the operations of the lots that will be released within time [now + T_1 + T_2, now + T_1 + T_2 + T_3].

Note that only the operations performed on the machines on the lot's processing route are considered in calculating these future workloads, which then become three additional inputs to the FBPN.

Subsequently, a FBPN that incorporates expert opinions is applied for wafer lot output time prediction. The procedure for determining the parameters is described in the next section.

2.2 Output Time Prediction with FBPN That Incorporates Expert Opinions

The configuration of the FBPN is established as follows:

1. Inputs: nine parameters associated with the *n*-th example/lot including the average fab utilization (U_n) [5], the total queue length on the lot's processing route (Q_n) [2, 5, 14] or before bottlenecks (BQ_n) [5] or in the whole fab (FQ_n) [2, 5], the fab WIP (WIP_n) [2, 5], the latenesses ($D_n^{(i)}$) of the *i*-th recently completed lots [2, 5, 14], and the three nearest future discounted workloads on the lot's processing route ($FDW^{(1)}$, $FDW^{(2)}$, and $FDW^{(3)}$). These parameters have to be normalized so that their values fall within [0, 1]. Then some production execution/control experts are requested to express their beliefs (in linguistic terms) about the importance of each input parameter in predicting the cycle (output) time of a wafer lot. Linguistic assessments for an input parameter are converted into several pre-specified fuzzy numbers. The subjective importance of an input parameter is then obtained by averaging the corresponding fuzzy numbers of the linguistic replies for the input parameter by all experts. The subjective importance obtained for an input parameter is multiplied to the normalized value of the input parameter. After such a treatment, all inputs to the FBPN become triangular fuzzy numbers, and the fuzzy arithmetic for triangular fuzzy numbers is applied to deal with all calculations involved in training the FBPN.
2. Single hidden layer: Generally one or two hidden layers are more beneficial for the convergence property of the network.
3. Number of neurons in the hidden layer: the same as that in the input layer. Such a treatment has been adopted by many studies (e.g. [2, 5]).
4. Output: the (normalized) cycle time forecast of the example. The output is chosen according to the purpose. Besides, the same output has been adopted in many previous studies (e.g. [2-5]).
5. Transfer function: Delta rule.
6. Network learning rule: Back propagation (BP) rule.
7. Transformation function: Sigmoid function,

$$f(x) = 1/(1+e^{-x}). \tag{1}$$

8. Learning rate ( ): 0.01~1.0.
9. Batch learning.

The procedure for determining the parameter values is now described. A portion of the examples is fed as "training examples" into the FBPN to determine the parameter values. Two phases are involved at the training stage. At first, in the forward phase, inputs are multiplied with weights, summated, and transferred to the hidden layer. Then activated signals are outputted from the hidden layer as:

$$\tilde{h}_j = (h_{j1},\ h_{j2},\ h_{j3}) = 1/1+e^{-\tilde{n}_j^h}\ , \tag{2}$$

where

$$\tilde{n}_j^h = (n_{j1}^h,\ n_{j2}^h,\ n_{j3}^h) = \tilde{I}_j^h(-)\tilde{\theta}_j^h\ , \tag{3}$$

$$\tilde{I}_j^h = (I_{j1}^h,\ I_{j2}^h,\ I_{j3}^h) = \sum_{all\ i} \tilde{w}_{ij}^h(\times)\tilde{x}_{(i)}\ , \tag{4}$$

and (−) and (×) denote fuzzy subtraction and multiplication, respectively; $\tilde{h}_j$'s are also transferred to the output layer with the same procedure. Finally, the output of the FBPN is generated as:

$$\tilde{o} = (o_1,\ o_2,\ o_3) = 1/1+e^{-\tilde{n}^o}\ , \tag{5}$$

where

$$\tilde{n}^o = (n_1^o,\ n_2^o,\ n_3^o) = \tilde{I}^o(-)\tilde{\theta}^o\ , \tag{6}$$

$$\tilde{I}^o = (I_1^o, I_2^o, I_3^o) = \sum_{all\ j} \tilde{w}_j^o(\times)\tilde{h}_j\ . \tag{7}$$

To improve the practical applicability of the FBPN and to facilitate the comparisons with conventional techniques, the fuzzy-valued output $\tilde{o}$ is defuzzified according to the centroid-of-area (COA) formula:

$$o = \text{COA}(\tilde{o}) = (o_1 + 2o_2 + o_3)/4. \tag{8}$$

Then the defuzzified output o is applied to predict the actual cycle time a, for which the RMSE is calculated:

$$RMSE = \sqrt{\sum (o-a)^2 / \text{number of examples}}. \tag{9}$$

Subsequently in the backward phase, the deviation between o and a is propagated backward, and the error terms of neurons in the output and hidden layers can be calculated, respectively, as

$$\delta^o = o(1-o)(a-o)\,, \tag{10}$$

$$\tilde{\delta}_j^h = (\delta_{j1}^h,\ \delta_{j2}^h,\ \delta_{j3}^h) = \tilde{h}_j(\times)(1-\tilde{h}_j)(\times)\tilde{w}_j^o\delta^o\,. \tag{11}$$

Based on them, adjustments that should be made to the connection weights and thresholds can be obtained as

$$\Delta\tilde{w}_j^o = (\Delta w_{j1}^o,\ \Delta w_{j2}^o,\ \Delta w_{j3}^o) = \eta\delta^o\tilde{h}_j\,, \tag{12}$$

$$\Delta\tilde{w}_{ij}^h = (\Delta w_{ij1}^h,\ \Delta w_{ij2}^h,\ \Delta w_{ij3}^h) = \eta\tilde{\delta}_j^h(\times)\tilde{x}_i\,, \tag{13}$$

$$\Delta\theta^o = -\eta\delta^o\,, \tag{14}$$

$$\Delta\tilde{\theta}_j^h = (\Delta\theta_{j1}^h, \Delta\theta_{j2}^h, \Delta\theta_{j3}^h) = -\eta\tilde{\delta}_j^h\,. \tag{15}$$

Theoretically, network-learning stops when the RMSE falls below a pre-specified level, or the improvement in the RMSE becomes negligible with more epochs, or a large number of epochs have already been run. In addition, to avoid the accumulation of fuzziness during the training process, the lower and upper bounds of all fuzzy numbers in the FBPN will no longer be modified if Chen's index [5] converges to a minimal value. Then test examples are fed into the FBPN to evaluate the accuracy of the network that is also measured with the RMSE. Finally, the FBPN can be applied to predicting the cycle time of a new lot. When a new lot is released into the fab, the six parameters associated with the new lot are recorded and fed as inputs to the FBPN. After propagation, the network output determines the output time forecast of the new lot.

3 A Demonstrative Example from a Simulated Wafer Fab

In practical situations, the history data of each lot is only partially available in the factory. Further, some information of the previous lots such as Q_n, BQ_n, and FQ_n is not easy to collect on the shop floor. Therefore, a simulation model is often built to simulate the manufacturing process of a real wafer fabrication factory [1-7]. Then, such information can be derived from the shop floor status collected from the simulation model [3]. To generate a demonstrative example, a simulation program coded using Microsoft Visual Basic .NET is constructed to simulate a wafer fabrication environment with the following assumptions:

1. The distributions of the interarrival times of orders are exponential.
2. The distributions of the interarrival times of machine downs are exponential.
3. The distribution of the time required to repair a machine is deterministic.
4. The percentages of lots with different product types in the fab are predetermined. As a result, this study is only focused on fixed-product-mix cases. However, the product mix in the simulated fab does fluctuate and is only approximately fixed in the long term.

5. The percentages of lots with different priorities released into the fab are controlled.
6. The priority of a lot cannot be changed during fabrication.
7. Lots are sequenced on each machine first by their priorities, then by the first-in-first-out (FIFO) policy. Such a sequencing policy is a common practice in many foundry fabs.
8. A lot has equal chances to be processed on each alternative machine/head available at a step.
9. A lot cannot proceed to the next step until the fabrication on its every wafer has been finished. No preemption is allowed.

The basic configuration of the simulated wafer fab is the same as a real-world wafer fabrication factory which is located in the Science Park of Hsin-Chu, Taiwan, R.O.C. A trace report was generated every simulation run for verifying the simulation model. The simulated average cycle times have also been compared with the actual values to validate the simulation model. Assumptions (1)~(3), and (7)~(9) are commonly adopted in related studies (e.g. [2-5]), while assumptions (4)~(6) are made to simplify the situation. There are five products (labeled as A~E) in the simulated fab. A fixed product mix is assumed. The percentages of these products in the fab's product mix are assumed to be 35%, 24%, 17%, 15%, and 9%, respectively. The simulated fab has a monthly capacity of 20,000 pieces of wafers and is expected to be fully utilized (utilization = 100%). POs with normally distributed sizes (mean = 300 wafers; standard deviation = 50 wafers) arrive according to a Poisson process, and then the corresponding MOs are released for these POs a fixed time after. Based on these assumptions, the mean inter-release time of MOs into the fab can be obtained as (30.5 * 24) / (20000 / 300) = 11 hours. An MO is split into lots of a standard size of 24 wafers per lot. Lots of the same MO are released one by one every 11 / (300/24) = 0.85 hours. Three types of priorities (normal lot, hot lot, and super hot lot) are randomly assigned to lots. The percentages of lots with these priorities released into the fab are restricted to be approximately 60%, 30%, and 10%, respectively. Each product has 150~200 steps and 6~9 reentrances to the most bottleneck machine. The singular production characteristic "reentry" of the semiconductor industry is clearly reflected in the example. It also shows the difficulty for the production planning and scheduling people to provide an accurate due-date for the product with such a complicated routing. Totally 102 machines (including alternative machines) are provided to process single-wafer or batch operations in the fab. Thirty replicates of the simulation are successively run. The time required for each simulation replicate is about 12 minute on a PC with 512MB RAM and Athlon™ 64 Processor 3000+ CPU. A horizon of twenty-four months is simulated. The maximal cycle time is less than three months. Therefore, four months and an initial WIP status (obtained from a pilot simulation run) seemed to be sufficient to drive the simulation into a steady state. The statistical data were collected starting at the end of the fourth month. For each replicate, data of 30 lots are collected and classified by their product types and priorities. Totally, data of 900 lots can be collected as training and testing examples. Among them, 2/3 (600 lots, including all product types and priorities) are used to train the network, and the other 1/3 (300 lots) are reserved for testing. The three parameters in calculating the future discounted workloads are specified as: T_1 = one week; T_2 = 1.5 weeks; T_3 = 2 weeks.

3.1 Results and Discussions

To evaluate the effectiveness and efficiency of the look-ahead FBPN and to make some comparisons with four approaches – MFLC, BPN, CBR, and FBPN without look-ahead, all the five methods were applied to five test cases containing the data of full-size (24 wafers per lot) lots with different product types and priorities. The convergence condition was established as either the improvement in the RMSE becomes less than 0.001 with one more epoch, or 1000 epochs have already been run. The minimal RMSEs achieved by applying the five approaches to different cases were recorded and compared in Table 1. RMSE is adopted instead of mean absolute error (MAE) because the same measure has been adopted in the previous studies in this field (e.g. [2-5]), namely, to facilitate comparison. As noted in Chang and Liao [5], the *k*-nearest-neighbors based CBR approach should be fairly compared with a BPN trained with only randomly chosen *k* cases. MFLC was adopted as the comparison basis, and the percentage of improvement on the minimal RMSE by applying another approach is enclosed in parentheses following the performance measure. The optimal value of parameter *k* in the CBR approach was equal to the value that minimized the RMSE [5].

Table 1. Comparisons of the RMSEs of various approaches

RMSE	A(normal)	A(hot)	A(super hot)	B(normal)	B(hot)
MFLC	185.1	106.01	12.81	302.86	79.94
BPN	177.1(-4%)	102.27(-4%)	12.23(-5%)	286.93(-5%)	75.98(-5%)
FBPN	171.82(-7%)	89.5(-16%)	11.34(-11%)	286.14(-6%)	76.14(-5%)
CBR	172.44(-7%)	86.66(-18%)	11.59(-10%)	295.51(-2%)	78.85(-1%)
L/a FBPN	163.45(-12%)	85.6(-19%)	8.09(-37%)	264.8(-13%)	69.65(-13%)

According to experimental results, the following discussions are made:

1. From the effectiveness viewpoint, the prediction accuracy (measured with the RMSE) of the look-ahead FBPN was significantly better than those of the other approaches by achieving a 12%~37% (and an average of 19%) reduction in the RMSE over the comparison basis – MFLC. The average advantage over CBR was 12%.
2. The effect of incorporating the future release plan (look-ahead) is revealed by the fact that the prediction accuracy of the look-ahead FBPN was considerably better than that of FBPN without look-ahead in all cases with an average advantage of 10%.
3. As the lot priority increases, the superiority of the look-ahead FBPN over BPN and CBR becomes more evident.
4. In the respect of efficiency, after observing the fluctuations in the RMSEs during the learning processes of these two networks, the FBPN (with or without look-ahead) appeared to start with a considerably smaller value of the initial RMSE, and reached the minimum RMSE with much fewer epochs than the crisp BPN did (see Fig. 1).

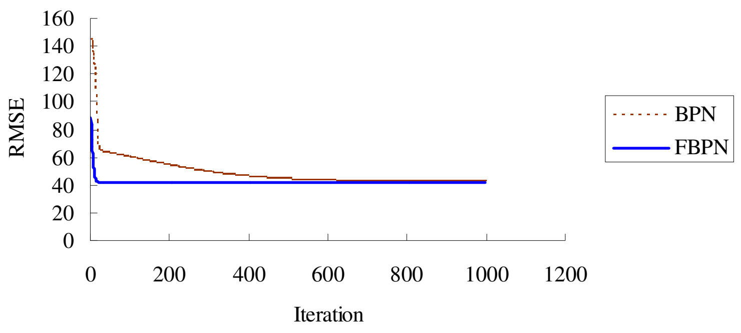


Fig. 1. Fluctuations in the RMSEs during the learning processes of BPN and FBPN

4 Conclusions and Directions for Future Research

To further enhance the effectiveness of wafer lot output time series prediction, a look-ahead FBPN is constructed in this study with two advanced features: the future release plan of the fab is considered (look-ahead); expert opinions are incorporated. For evaluating the effectiveness of the intelligent neural system and to make some comparisons with four approaches – MFLC, BPN, CBR, and FBPN without look-ahead, PS is applied in this study to generate test data. Then all the five methods are applied to five cases elicited from the test data. According to experimental results, the prediction accuracy of the look-ahead FBPN was significantly better than those of the other approaches by achieving a 12%~37% (and an average of 19%) reduction in the RMSE over the comparison basis – MFLC. The effect of incorporating the future release plan (look-ahead) is revealed with an average advantage of 10% over FBPN without look-ahead. In the respect of efficiency, the FBPN (with or without look-ahead) appeared to start with a considerably smaller value of the initial RMSE, and reached the minimum RMSE with much fewer epochs than the crisp BPN did.

However, to further evaluate the effectiveness and efficiency of the proposed look-ahead FBPN, it has to be applied to fab models of different scales, especially a full-scale actual wafer fab. In addition, the proposed look-ahead FBPN can also be applied to cases with changing product mixes or loosely controlled priority combinations, under which the cycle time variation is often very large. These constitute some directions for future research.

References

1. Barman, S.: The Impact of Priority Rule Combinations on Lateness and Tardiness. IIE Transactions 30 (1998) 495-504
2. Chang, P.-C., Hsieh, J.-C.: A Neural Networks Approach for Due-date Assignment in a Wafer Fabrication Factory. International Journal of Industrial Engineering 10(1) (2003) 55-61

3. Chang, P.-C., Hsieh, J.-C., Liao, T. W.: A Case-based Reasoning Approach for Due Date Assignment in a Wafer Fabrication Factory. In: Proceedings of the International Conference on Case-Based Reasoning (ICCBR 2001), Vancouver, British Columbia, Canada (2001)
4. Chang, P.-C., Hsieh, J.-C., Liao, T. W.: Evolving Fuzzy Rules for Due-date Assignment Problem in Semiconductor Manufacturing Factory. Journal of Intelligent Manufacturing 16 (2005) 549-557
5. Chen, T.: A Fuzzy Back Propagation Network for Output Time Prediction in a Wafer Fab. Journal of Applied Soft Computing 2/3F (2003) 211-222
6. Chung, S.-H., Yang, M.-H., Cheng, C.-M.: The Design of Due Date Assignment Model and the Determination of Flow Time Control Parameters for the Wafer Fabrication Factories. IEEE Transactions on Components, Packaging, and Manufacturing Technology – Part C 20(4) (1997) 278-287
7. Foster, W. R., Gollopy, F., Ungar, L. H.: Neural Network Forecasting of Short, Noisy Time Series. Computers in Chemical Engineering 16(4) (1992) 293-297
8. Goldberg, D. E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading, MA (1989)
9. Hung, Y.-F., Chang, C.-B.: Dispatching Rules Using Flow Time Predictions for Semiconductor Wafer Fabrications. In: Proceedings of the 5th Annual International Conference on Industrial Engineering Theory, Applications and Practice, Taiwan (2001)
10. Ishibuchi, H., Nozaki, K., Tanaka, H.: Distributed Representation of Fuzzy Rules and Its Application to Pattern Classification. Fuzzy Sets and Systems 52(1) (1992) 21-32
11. Lin, C.-Y.: Shop Floor Scheduling of Semiconductor Wafer Fabrication Using Real-time Feedback Control and Prediction. Ph.D. Dissertation, Engineering-Industrial Engineering and Operations Research, University of California at Berkeley (1996)
12. Piramuthu, S.: Theory and Methodology – Financial Credit-risk Evaluation with Neural and Neuralfuzzy Systems. European Journal of Operational Research 112 (1991) 310-321
13. Ragatz, G. L., Mabert, V. A.: A Simulation Analysis of Due Date Assignment. Journal of Operations Management 5 (1984) 27-39
14. Vig, M. M., Dooley, K. J.: Dynamic Rules for Due-date Assignment. International Journal of Production Research 29(7) (1991) 1361-1377
15. Wang, L.-X., Mendel, J. M.: Generating Fuzzy Rules by Learning from Examples. IEEE Transactions on Systems, Man, and Cybernetics 22(6) (1992) 1414-1427
16. Weeks, J. K.: A Simulation Study of Predictable Due-dates. Management Science 25 (1979) 363–373

Extraction of Fuzzy Features for Detecting Brain Activation from Functional MR Time-Series

Juan Zhou and Jagath C. Rajapakse

BioInformatics Research Center, School of Computer Engineering
Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798
{zhou0025, asjagath}@ntu.edu.sg

Abstract. We propose methods to extract fuzzy features from fMR time-series in order to detect brain activation. Five discriminating features are automatically extracted from fMRI using a sequence of temporal-sliding-windows. A fuzzy model based on these features is first developed by gradient method training on a set of initial training data and then incrementally updated. The resulting fuzzy activation maps are then combined to provide a measure of strength of activation for each voxel in human brain; a two-way thresholding scheme is introduced to determine actual activated voxels. The method is tested on both synthetic and real fMRI datasets for functional activation detection, illustrating that it is less vulnerable to correlated noise and is able to adapt to different hemodynamic response functions across subjects through incremental learning.

1 Introduction

Functional Magnetic Resonance Imaging (fMRI) is a noninvasive technique measuring functional activity of the brain *in vivo*, both spatially and temporally. The signal is based on the blood-oxygenation-level-dependent (BOLD) contrast, derived from the increase in blood oxygenation followed by neuronal activity, resulting in a larger magnetic resonance (MR) signal. The detection of the fMRI signal is not a trivial process as the BOLD signal change due to a typical experimental stimulation of the brain is very subtle, ranging from 1 to 5% on a 1.5 T scanner [1]. Furthermore, various noise and artifacts such as motion, electronic, physical, and physiological processes significantly confound the fMRI signal. Therefore, the techniques for fMRI analysis should be insensitive to the uncertainties and fuzziness introduced by these interference signals.

Two groups of methods have been used to detect activated voxels in fMRI data: hypothesis-driven and data-driven. Statistical Parametric Mapping (SPM) is the most widely used hypothesis-driven method for fMRI analysis, which assumes a linear regression model for the fMR signal with a specific noise structure. It is voxel-based and tests the hypothesis about fMR time-series by construction and assessment of spatially extended statistical processes based on Gaussian Random Field (GRF) theory [2]. However, it has become clear that there is a nonlinear relationship between the variation in the fMRI signal and the stimulus presentation [3]; and the hemodynamic response function (HRF) varies spatially and between subjects [4]. Moreover, the structure of noise in fMRI is not well understood and remains a contentious subject [5]. Thus, the validity of the statistical models depends on the extent to which the data satisfies the underlying assumptions.

In contrast, data-driven methods do not require any prior knowledge of the hemodynamic behaviors and are considered more powerful and relevant for fMRI studies in which unknown or complex differential responses are expected [6]. Generally, these data-driven methods can be divided into two groups: transformation-based and clustering-based. Principle component analysis (PCA) [7] and independent component analysis (ICA) belongs to the first group, which transform the original high-dimensional input space in order to separate functional responses and various noise sources from each other. ICA is an information theoretic approach which enables recovery of underlying signals or independent components from linear data mixtures. It has been applied to fMRI analysis both spatially [8] and temporally [9]. The second clustering group consists of fuzzy clustering [10] and self-organizing map [11], attempting to classify time signals of the brain into several patterns according to temporal similarity. For these data-driven methods, usually the contents of one class or component are interpreted as activations but how the signal is divided into classes is difficult to ascertain or comprehend. Certain class with a particular activation pattern has physiological interpretation but others are still unknown. Other fMRI analysis methods include specified-resolution wavelet analysis [12] and Bayesian modelling [4].

Our objective is to develop a new functional activation detection approach from fMRI data, which is less vulnerable to noise and HRF variability than hypothesis-driven methods and is more interpretable than previous data-driven method. Because of the complex, nonlinear, and imprecise nature and noise, accurately model fMRI signal using a conventional nonlinear mathematical approach is a difficult task with limited prior knowledge in this case. In contrast, fuzzy modeling is a practical way to model fMRI data of high uncertainty using limited available information. Moreover, methods based on raw fMRI data often give high computational complexity and high vulnerability to noise because of its large volume. The proposed feature extraction from time-series is able to decrease computational complexity and increase system ability to handling noise. In order to account for HRF variability across subjects, incremental learning is used to adapt the fuzzy model to individual subject in order to better identify activated regions. Here, we propose a novel fuzzy feature modelling (FFM) approach to detect activated voxels in human brain. It consists of (1) features based on temporal-sliding-windows (TSW) extracted from fMR time-series; (2) fuzzy feature modeling by incremental learning; (3) thresholding on fuzzy activation map. The details of our approach will be described in next section. In experiment and results section, the performance of this approach is illustrated with functional activation detection for both synthetic and real fMRI data.

2 Method

2.1 Extracting Features

Different voxels have different hemodynamic behaviors, for example, the signal magnitude at activated or deactivated states and hemodynamic response time to reach its peak could differ. The principle behind our feature extraction based on temporal-sliding-window (TSW) is that by shifting a set of windows over a time-series, the activated and deactivated voxels for each condition could have consistent discriminating patterns regardless of different shape, magnitude, or delay of hemodynamic response function. Let $F : \Omega \times \Theta \rightarrow Y$ be a functional time-series where $\Omega \subset N^3$ denotes the three-dimensional (3D) spatial domain of image voxels, $\Theta = \{1, 2...n\}$ indexes n number of 3D scans taken during the experiments. Let $Y = \{y_i^t : i \in \Omega, t \in \Theta, y_i^t \in Q\}$ is a 4D data where Q denotes range of image intensity and y_i^t represents the voxel intensity of voxel i at time t. Let Ω_B denote the set of brain voxels.

Here, we consider an experiment C, consisting of K conditions: $C = \{C_k : k = 1, 2...K\}$. Each condition C_k is presented, alternatively or randomly for P_k times in a single run while n 3D brain scans are taken, so each block of condition C_k is denoted by C_k^p, $p = 1, 2..P_k$. Block C_k^p lasts for a duration of length L_k^p and the beginning of C_k^p is denoted as B_k^p where $k = 1, 2...K$ and $p = 1, 2...P_k$. The above represents a general paradigm design, which applies to both block and event-related designs of fMRI experiments. Then, a sequence of Temporal-Sliding-Windows (TSW) for each condition C_k is constructed for from time-series F as follows:

1. For each condition C_k, create a sequence of P_k number of windows denoted by $W_k = \{W_k^p : p = 1, 2...P_k, k = 1, 2...K\}$; in other words, one window W_k^p for each condition block C_k^p. The length of window W_k^p equals to L_k^p and the initial starting point of window W_k^p is given by B_k^p.
2. For each condition C_k, shift the sequence of windows W_k temporally forward by a sliding time interval s simultaneously, resulting in a new set of windows denoted by $W(s) = \{W_k^p(s) : s = 0, 1...S, k = 1, 2...K, p = 1, 2...P_k\}$. Depending on different inter-scan time, the maximum sliding time interval S varies: $S = 32/RT$ (seconds) based on the fact that the total length of hemodynamic response function is approximately 32s. Thus, the starting and ending time of window $W_k^p(s)$ is $B_k^p + s$ and $B_k^p + s + L_k^p - 1$, denoted by $T1_{W_k^p(s)}$ and $T2_{W_k^p(s)}$ respectively for simplicity reason.
3. For each window $W_k^p(s)$ of condition block C_k for $s = 0, 1, ..S$, calculate the average intensity $A_k^p(s, i)$ of each voxel i as:

$$A_k^p(s, i) = \frac{\sum_{t=T1_{W_k^p(s)}}^{T2^*_{W_k^p(s)}} y_i^t}{T2^*_{W_k^p(s)}) - T1_{W_k^p(s)}} \tag{1}$$

where $T2^*_{W_k^p(s)} = \min(n, T2_{W_k^p(s)})$.

Thus, we can observe a curve $A_k^p(i) = \{A_k^p(s,i) : s = 0, 1...S\}$ for each voxel i in each condition block C_k^p, whose shape is highly discriminating between activated and non-activated voxels. We call it *Quasi-Hemodynamic Curve* (QHC) since it is similar to HRF in a general sense over the whole time-series. Five discriminating features $F_{f,p}^k(i)$, $f = 1, 2...5$, are extracted from this QHC for each voxel i in each condition block C_k^p as follows:

1. Area under curve ratio for QHC:

$$F_{1,p}^k(i) = \frac{\sum_{s=0}^{L_{k,p}^*} A_k^p(s,i)}{(\max_s A_k^p(s,i) - \min_s A_k^p(s,i)) \cdot L_{k,p}^*} \tag{2}$$

   where $L_{k,p}^* = \min(L_k^p, S)$.
2. Area difference ratio for QHC:

$$F_{2,p}^k(i) = \frac{\sum_{s=0}^{L_{k,p}^*} A_k^p(s,i)}{\sum_{s=L_{k,p}^*+1}^{S} A_k^p(s,i)} \tag{3}$$

3. Correlation between QHC $A_k^p(i)$ and the standard QHC SA_k^p:

$$F_{3,p}^k(i) = \frac{\sum_{s=0}^{L_{k,p}^*} (A_k^p(s,i) - \overline{A_k^p(s,i)})(SA_k^p - \overline{SA_k^p})}{\sqrt{\sum_{s=0}^{L_{k,p}^*} (A_k^p(s,i) - \overline{A_k^p(s,i)})^2 \sum_{s=0}^{L_{k,p}^*} (SA_k^p - \overline{SA_k^p})^2}} \tag{4}$$

   where $SA_k^p = \{SA_k^p(s) = -(s - L_{k,p}^*/2)^2 : s = 0, 1...L_{k,p}^*\}$.
4. Time ratio at peak amplitude for QHC:

$$F_{4,p}^k(i) = \arg_{s\in[0,L_{k,p}^*]} \max_s A_k^p(s,i)/L_{k,p}^* \tag{5}$$

5. Time ratio at lowest amplitude for QHC:

$$F_{5,p}^k(i) = \arg_{s\in[0,L_{k,p}^*]} \min_s A_k^p(s,i)/L_{k,p}^* \tag{6}$$

   Two curves have been normalized to $[0, 1]$ before correlation computation in feature 3 for easy comparison between voxels. Since the shape of QHC of an activated and nonactivated voxel are usually quite the opposite with each other, the five five features could be significantly discriminating.

The above five features of each condition block are sum up over all blocks as in equation 7, resulting in a robust and simple 5D feature space for fuzzy feature modeling in next stage because it is less vulnerable to noise or changes in HRFs. Note that we assume L_k^p is smaller than S, so for case like L_k^p is larger than S, we should use window length equal to 32 seconds while the maximum sliding time for each window $S_{W_k^p} = L_k^p$. The properties of the resulting curve is similar to QHC and same features could be extracted.

$$F_f^k(i) = \sum_{p=1}^{P_k} F_{f,p}^k(i)/P_k \tag{7}$$

2.2 Learning Fuzzy Activation Maps

Based on the feature space developed in the previous section, we use gradient technique to derive a fuzzy feature model. A new incremental learning scheme is proposed to extract useful knowledge not only from prior standard hemodynamic response function but also from fMRI data of each subject itself. This scheme is able to adapt to the variability of hemodynamic response function among subjects. The resulting fuzzy model is able to (1) calculate the strength of activation of each voxel to each condition and (2) provide a rule-base for activation pattern interpretation. For each condition C_k, one five-input $F_k(i) = \{F_f^k(i) : f = 1, 2...5\}$, single-output $Z_k(i)$ (activation strength) fuzzy model M_k is developed for all voxels $i \in \Omega_B$. The proposed incremental learning scheme to derive model M_k is as follows:

1. *Building initial training data*: For activation class, randomly select parameters of the canonical hemodynamic response function within a certain range to create a set of time-series, consisting of H different variations of original HRF, e.g. time and dispersion derivatives. Add all these time-series to the initial training set $O(1)$ with the desired output D as 1. For initial training time-series of non-activation class, time-series of constant ($[0, 1]$) amplitude are added to $O(1)$ with the desired output as 0. The initial test dataset is the time-series of all brain voxels in the target subject, denoted by $Y(1) = \{y_i^t : i \in \Omega_B, t \in \Theta\}$.
2. *Extracting features*: For incremental learning iteration r, extract features $F_k(O(r))$ from training data $O(r)$ as described in section 2.1.
3. *Training*: Apply Gradient Method (GM) to train the fuzzy model $M_k(r)$ based on features space $F_k(O(r))$ until convergence criterion is met. The objective of GM based training is to minimize the error between the predicted output value $Z_k(i)$ and the desired output value $D_k(i)$ for training data i defined as:
$$\epsilon = \frac{1}{2}[Z_k(i) - D_k(i)]^2 \tag{8}$$
Gaussian fuzzy membership functions are presumed for input features as in Eq. 9 and the mechanism of fuzzy model M_k employ a product t-norm for the input features (premise of the rule base) and a product implication as in Eq. 10. Delta output fuzzy membership function is used.
$$\mu_f^k(i) = \exp[-\frac{1}{2}(\frac{F_f^k(i) - \alpha_f}{\sigma_f})^2] \tag{9}$$
$$Z_k(i) = \frac{\sum_{c=1}^{2} b_c \prod_{f=1}^{5} \mu_f^k(i)}{\sum_{c=1}^{2} \prod_{f=1}^{5} \mu_f^k(i)} \tag{10}$$
where $c = 1$ and $c = 2$ denotes the classes of activated and non-activated voxels for condition k, respectively. Therefore, parameters of the fuzzy model include input center α_f, input variance σ_f and output center b_c, where $c \in \{1, 2\}$ and $f = 1, 2...5$. GM is able to tune these parameters of the fuzzy model and detailed parameter updating equations can be found in [13].

4. *Testing*: Feed the extracted features from testing data at current round to the resulting fuzzy model $M_k(r)$ in step 3 to determine output $Z_k(i)$.
5. *Updating training data*: Select the top H activated voxels and non-activated voxels in terms of high or low outputs $Z_k(i)$ and add into the training data set of next round $O(r+1)$. The desired output is again 1 for activation and 0 for non-activation respectively. Remove those points from the testing data set $Y(r)$ to form $Y(r+1)$. Repeat steps 2-5 till certain criteria is reached e.g. maximum learning round R is reached or no more suitable voxels can be added into training set.
6. *Final testing*: Apply the whole fMRI data $Y(1)$ to the final fuzzy model $M_k(R)$ once more to obtain finalized output $Z_k = \{Z_k(i) : i \in \Omega_B\}$.

2.3 Detecting Activation

In order to detect activation by thresholding, a half-triangular fuzzy membership function is designed to convert original activation strength map $Z_k(i)$ to a fuzzy activation map $Z_k^* = \{Z_k^*(i) : i \in \Omega_B, Z_k^*(i) \in [0,1]\}$ as in Eq.11 where 1 for activated and 0 for nonactiveted. This map represents the activation strength of each voxel in terms of fuzzy membership values, which enables a rather consistent and effective performance across different data by the following two-way thresholding method.

First, a simple thresholding is applied as in Eq. 12 based on two parameters ζ_1 and ζ_2 ($\zeta_1 > \zeta_2$) for addding high confident voxels to the activated set Υ_1 and the non-activated set Υ_2. Then, for undetermined voxels i where $Z_k^*(i) < \zeta_1$ & $Z_k^*(i) > \zeta_2$, an ordered list is formulated based on the value of $Z_k^*(i)$. A screen process begins simultaneously from the two ends of this list and assign voxels in the list to be activated or nonactivated according to Eq. 13.

$$Z_k^*(i) = \begin{cases} 1 & \text{if } Z_k(i) >= b_1 \\ (Z_k(i) - b_2)/(b_1 - b_2) & \text{if } Z_k(i) < b_1 \;\&\; Z_k(i) > b_2 \\ 0 & \text{if } Z_k(i) <= b_2 \end{cases} \tag{11}$$

$$\beta_k(i) = \begin{cases} 1 \text{ if } Z_k^*(i) >= \zeta_1 \\ 0 \text{ if } Z_k^*(i) <= \zeta_2 \end{cases} \tag{12}$$

$$\beta_k(i) = \begin{cases} 1 \text{ if } \chi(1,i,v) > \chi(0,i,v) \\ 0 \text{ if } \chi(1,i,v) < \chi(0,i,v) \end{cases} \tag{13}$$

where $\chi(c,i,v) = \prod_{j \in \varphi(i,v), j \in \Upsilon_c, \beta_k(j)=c} |Z_k^*(j) - Z_k^*(i)| \cdot distance(i,j)$; $\varphi(i,v)$ is the neighborhood of voxel i within a cube of size v^3. Each voxel i is removed from list and added into the Υ_1 or Υ_2 accordingly and the screen continues until there is no more voxels in the list. For other interested contrasts related to different conditions, fusion of the fuzzy activation maps Z_k^* is required to form a specific fuzzy activation map for that contrast; and the same thresholding procedure can be applied.

3 Experiments and Results

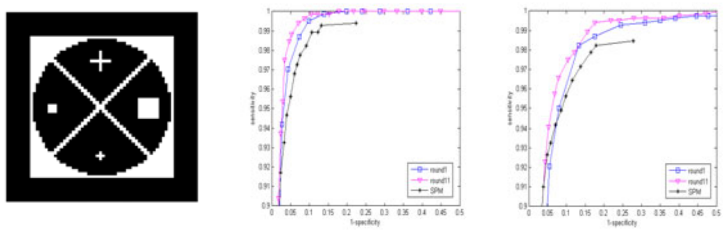
All simulations were done in MATLAB. Both synthetic and real fMRI data were used in experiments and a comparison between the results produced by our approach and Statistical Parametric Mapping (SPM2) is given.

3.1 Synthetic Data

A 2D synthetic functional data was simulated consisting six cycles (8 ON and 8 OFF, TR = 2s, n=96). The response of the activated pixels was generated by convolving a box-car time-series with a hemodynamic response function, which is a mixture of two gamma functions, while the inactive pixels remained unchanged as zero-amplitude. True activation pattern is shown Figure 1 a. Since the nature of noise in fMRI time-series are often correlated, synthetic image series with different levels of spatially correlated noises (the average of neighboring i.i.d. Gaussian noises) were tested. The SPM analysis followed the standard procedure in SPM2 [15] on the smoothed data (FWHM=6mm).

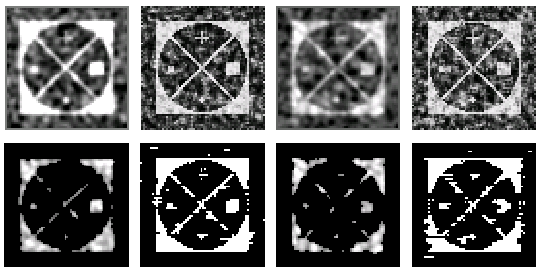
For the present (FFM) approach, $H = 20$, 40 time-series (20 for activated class and 20 for non-activated class) are used as initial training data set and 20 voxels are selected for each class at training data updating step. This small value of H recruits fewer confounding voxels for training purpose. It is observed that more iterations of incremental training in FFM approach can adapt to the data to a large extent, but overfitting and performance deterioration will occur when confounding voxels are added into training data. Also computational complexity will increase. Thus, 10 incremental learning iterations is used. For step 3 in section 2.2, initial conditions for fuzzy membership parameters are set to the same values: input center $\alpha_f = F_f$ of the first training sample; input variance $\sigma_f = 1$; output center $b_1 = 1$ for activated class and $b_2 = 0$ for non-activated class. Parameters controlling the updating steps for α, σ and b were all initialized to 1. The training stop when the average error is less than 10^{-3} or the absolute changes between rounds is less than 10^{-5}. In section 2.3, the neighborhood size for undetermined voxels is $v = 5$ and if no neighbors with known class was found, continue searching by increasing v by 1. However, the neighborhood was limited to 1/125 of the whole brain volume.

The performance of FFM is compared to SPM by plotting the ROC curves for functional activation detection. Figure 1 (b) & (c) shows the results by thresholding on SPMs using different significance levels and on the final fuzzy activation map Z_k^* using various ζ_1 and ζ_2 for both correlated noise level SNR=2.0 and 1.2. It is observed that FFM approach outperforms SPM for data at both noise levels and it is important to note that incremental training does improve the performance (comparing ROC of round 1 and round 11). Statistical parametric map and fuzzy activation map are compared in row 1 of Figure 2. The thresholded map produced by SPM (T-contrast with FDR $p < 0.05$) is compared to FFM ($\zeta_1 = 0.95$, $\zeta_2 = 0.3$) in row 2 of Figure 2. It is observed that our approach is able to discover more important and detailed signal pattern than SPM, especially in high correlated noise case.



(a) Truth (b) SNR=2 (c) SNR=1.2

Fig. 1. ROC curve for detecting activation on synthetic functional data



(a) (b) (c) (d)

Fig. 2. Detected activation for synthetic data having correlated noise by (a) SPM at SNR=2.0, (b) FFM at SNR=2.0, (c) SPM at SNR=1.2, and (d) FFM at SNR=1.2. Row 1 are unthresholded SPMs and fuzzy activation maps while row 2 are thresholded activation maps for each case.

3.2 Real Data

A set of real fMRI data obtained from experiments with 'visual stimulus' are analyzed. Please refer to [14] for further details about this data set. For SPM, all functional images were first corrected for movement artifacts, resampled and smoothed with FWHM=4.47mm 3D Gaussian filter. F-contrast is used for statistical analysis in SPM2 based on canonical hemodynamic function plus time and dispersion derivatives as basis function. Voxels with $p < 0.05$ corrected using family-wise-error (FWE) is determined to be activated. For our fuzzy feature modeling (FFM) approach, same parameters have been used as for synthetic data. Figure 3 (a) shows the similar activated regions of visual task on a single slice for both SPM and FFM approach. Since there are no ground truth, it is rather difficult to evaluate the performance, but still expected activation is found in visual cortex for both approach. Figure 3 (row 1 in (b) & (c)) illustrates typical Quasi-Hemodynamic Curve (QHC) for activated voxel and non-activated voxel in real fMRI data with visual task ($L = 4, S = 6$) respectively. Other voxels in the same class does not necessarily follow the exact same shape, but their

QHCs often comprise of similar discriminating features. QHC also have variability across subject, brain regions and task similar to HRF and it is related to the length of condition block L and maximum sliding time $S = 32s/RT$. Activated and nonactivated QHCs for real fMRI data on one subject performing motor task ($L = 10, S = 16$) in Figure 3 (row 2 in (b) & (c)) demonstrates this difference as compared to QHC for visual data. However, it is observed that activated and nonactivated class have quite the opposite QHC shape for both datasets and common discriminating features could still be discovered with some degrees of uncertainty.

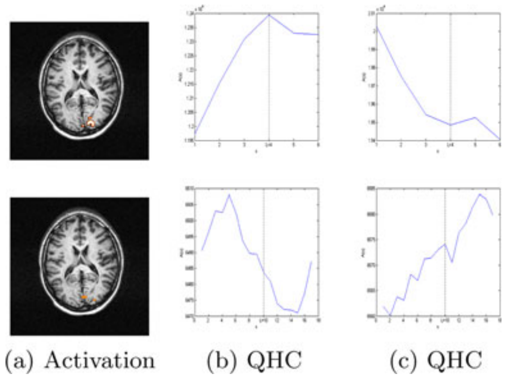


Fig. 3. Detected activation on selected axial slice by SPM (row 1 a) and by (FFM) (row 2 a) for visual task and QHC for visual (row 1) and motor (row 2) fMRI data: activated (b) and nonactivated (c)

4 Conclusion

By a novel fuzzy feature extraction method, we are able to convert 4D fMRI dataset into a much simpler and robust feature space to detect functional activation. A fuzzy feature model (FFM) is first built on limited prior knowledge as initial training set, and is further updated by incremental learning to account for different hemodynamic response functions across subjects. A general two-way thresholding scheme is proposed to obtain actual activation regions from resulting fuzzy activation map effectively. Experiments on both synthetic and real fMRI data shows that our FFM approach is less vulnerable to correlated noise and more sensitive to discover significant signals. Activated and nonactivated class for each condition is discovered simultaneously so no interpretation problem exists. Future work includes taking into account the spatial variability of hemodynamic response function within a single subject. Current algorithm can be easily extended to comparison between conditions and group study by fuzzy activation map fusion. Moreover, the resulting parameters in fuzzy models could give us meaningful relationship between input features and output class.

References

1. S. Ogawa, D. Tank, and R. Menon, "Intrinsic signal changes accompanying sensory simularion: Functional brain mapping with magnetic resonance imaging," *Proc. Nat. Acad. Sci.*, vol. 89, pp. 5951–5955, 1992.
2. K. J. Friston, A. P. Holmes, K. J. Worsley, J.-B. Poline, C. D. Frith, and R. S. J. Frackowiak, "Statistical parametric maps in functional imaging: A general linear approach," *Human Brain Mapping*, vol. 2, pp. 189–210, 1995.
3. A. L. Vazquez and D. C. Noll, "Nonlinear aspects of the bold response in functional mri," *Human Brain Mapping*, vol. 40, pp. 249–260, 1998.
4. M. W. Woolrich, M. Jenkinson, J. M. Brady, and S. M. Smith, "Fully bayesian spatio-temporal modeling of fmri data," *IEEE trans. on Med. Imag.*, vol. 24, no. 2, pp. 213–231, Feb. 2004.
5. E. Bullmore, C. Long, J. Fadili, G. Calvert, F. Zalaya, T. A. Carpenter, and M. Brammer, "Colored noise and computational inference in neuophysiological (fmri) time series analysis : Resampling mehtods in time and wavelet domains," *Human Brain Mapping*, vol. 12, pp. 61–78, 2001.
6. A. Meyer0Baese, A. Wismueller, and O. Lange, "Comparison of two exploratory data analysis methods for fmri: unsupervised clustering versus independent component analysis," *IEEE trans. of Infor. Techno. in Biomed.*, pp. 387–398, September, 2004.
7. A. H. Andersen, D. M. Gash, and M. J. Avison, "Principal component analysis of the dynamic response measured by fmri," *Mgn. Reson. Imag.*, vol. 17, no. 6, pp. 795–815, 1999.
8. M. J. McKeown, S. Makeig, G. G. Brown, T. P. Jung, S. S. Kindermann, A. J. Bell, and T. J. Sejnowski, "Analysis of fmri data by blind separation into independent spatial components," *Human Brain Mapping*, vol. 6, pp. 160–188, 1998.
9. B. Biswal and J. Ulmer, "Blind source separation of multiple signal sources of fmri data sets using independent component analysis," *Journal of Comput. Assist. Tomogr.*, vol. 23, no. 2, pp. 265–271, Mar-Apr 1999.
10. K. Chuang, M. Chiu, C. Lin, and J. Chen, "Model-free functional mri analysis using kohonen clustering neural network and fuzzy c-means," *IEEE Trans. Med. Imag.*, vol. 18, pp. 1117–1128, December 1999.
11. S. Ngan and X. Hu, "Analysis of functional magnetic resonance imaging data using self-organizing mapping with spatial connectivity," *Mgn. Reson. in Med.*, vol. 41, pp. 939–946, 1999.
12. G. A. Hossein-Zadeh, H. Soltanian-Zadeh, and B. A. Ardekani, "Multiresolution fmri activation detection using translation invariant wavelet transform and statistical analysis based on resampling," *IEEE Trans. on Med. Imag.*, vol. 22, no. 3, pp. 302–314, March 2003.
13. T. J. Ross, "Fuzzy logic with engineering applications," *John Wiley & Sons Inc. 2nd edition*, pp. 213–241, 2004.
14. J. C. Rajapakse, F. Kruggel, J. M. Maisog, and D. Y. von Cramon, "Modeling Hemodynamic Response for Analysis of Functional MRI Time-Series," *Human Brain Mapping*, vol. 6, pp.283–300,1998.
15. Wellcome Department of Imaging Neuroscience, "Statistical Parametric Mapping," *http://www.fil.ion.ucl.ac.uk/spm/*, 2004.

An Advanced Design Methodology of Fuzzy Set-Based Polynomial Neural Networks with the Aid of Symbolic Gene Type Genetic Algorithms and Information Granulation

Seok-Beom Roh, Hyung-Soo Hwang, and Tae-Chon Ahn

Department of Electrical Electronic and Information Engineering, Wonkwang University, 344-2, Shinyong-Dong, Iksan, Chon-Buk, 570-749, South Korea
{nado, hshwang, tcahn}@wonkwang.ac.kr

Abstract. In this paper, we propose a new design methodology that adopts Information Granulation to the structure of fuzzy-neural networks called Fuzzy Set–based Polynomial Neural Networks (FSPNN). We find the optimal structure of the proposed model with the aid of symbolic genetic algorithms which has symbolic gene type chromosomes. We are able to find information related to real system with Information Granulation through numerical data. Information Granules obtained from Information Granulation help us understand real system without the field expert. In Information Granulation, we use conventional Hard C-Means Clustering algorithm and proposed procedure that handle the apex of clusters using 'Union' and 'Intersection' operation. We use genetic algorithm to find optimal structure of the proposed networks. The proposed networks are based on GMDH algorithm that makes whole networks dynamically. In other words, FSPNN is built dynamically with symbolic genetic algorithms. Symbolic gene type has better characteristic than binary coding GAs from the size of solution space's point of view. . Symbolic genetic algorithms are capable of reducing the solution space more than conventional genetic algorithms with binary genetype chromosomes. The performance of genetically optimized FSPNN (gFSPNN) with aid of symbolic genetic algorithms is quantified through experimentation where we use a number of modeling benchmarks data which are already experimented with in fuzzy or neurofuzzy modeling.

1 Introduction

In recent, a great deal of attention has been directed towards usage of Computational Intelligence such as fuzzy sets, neural networks, and evolutionary optimization towards system modeling. A lot of researchers on system modeling have been interested in the multitude of challenging and conflicting objectives such as compactness, approximation ability, generalization capability and so on which they wish to satisfy. Fuzzy sets emphasize the aspect of linguistic transparency of models and a role of a model designer whose prior knowledge about the system may be very helpful in facilitating all identification pursuits. In addition, to build models with substantial approximation capabilities, there should be a need for advanced tools.

As one of the representative and sophisticated design approaches comes a family of fuzzy polynomial neuron (FPN)-based self organizing neural networks (abbreviated as FPNN or SOPNN and treated as a new category of neuro-fuzzy networks) [1], [2], [3], [4]. The design procedure of the FPNNs exhibits some tendency to produce overly complex networks as well as comes with a repetitive computation load caused by the trial and error method being a part of the development process. The latter is in essence inherited from the original GMDH algorithm that requires some repetitive parameter adjustment to be completed by the designer.

In this study, in addressing the above problems coming with the conventional SOPNN (especially, FPN-based SOPNN called "FPNN") [1], [2], [3], [4] as well as the GMDH algorithm, we introduce a new genetic design approach as well as a new FSPN structure treated as a FPN within the FPNN. Bearing this new design in mind, we will be referring to such networks as genetically optimized FPNN with fuzzy set-based PNs ("gFPNN" for brief). The determination of the optimal values of the parameters available within an individual FSPN (viz. the number of input variables, the order of the polynomial corresponding to the type of fuzzy inference method, the number of membership functions(MFs) and a collection of the specific subset of input variables) leads to a structurally and parametrically optimized network. The network is directly contrasted with several existing neurofuzzy models reported in the literature.

2 The Architecture and Development of Fuzzy Set-Based Polynomial Neural Networks (FSPNN)

The FSPN encapsulates a family of nonlinear "if-then" rules. When put together, FSPNs results in a self-organizing Fuzzy Set-based Polynomial Neural Networks (FSPNN). Each rule reads in the form.

$$\begin{aligned}&\text{if } x_p \text{ is } \boldsymbol{A_k} \text{ then } z \text{ is } P_{pk}(x_i, x_j, \boldsymbol{a}_{pk})\\&\text{if } x_q \text{ is } \boldsymbol{B_k} \text{ then } z \text{ is } P_{qk}(x_i, x_j, \boldsymbol{a}_{qk})\end{aligned} \tag{1}$$

where $\boldsymbol{a}_{qk}$ is a vector of the parameters of the conclusion part of the rule while $P(x_i, x_j, \boldsymbol{a})$ denoted the regression polynomial forming the consequence part of the fuzzy rule. The activation levels of the rules contribute to the output of the FSPN being computed as a weighted average of the individual condition parts (functional transformations) P_K. (note that the index of the rule, namely "k" is a shorthand notation for the two indices of fuzzy sets used in the rule (1), that is $K=(l,k)$).

$$\begin{aligned}z &= \sum_{l=1}^{\text{total inputs}}\left(\sum_{k=1}^{\text{total\_rules related to input } l}\mu_{(l,k)}P_{(l,k)}(x_i, x_j, \boldsymbol{a}_{(l,k)}) \Big/ \sum_{k=1}^{\text{total\_rules related to input } l}\mu_{(l,k)}\right)\\ &= \sum_{l=1}^{\text{total inputs}}\left(\sum_{k=1}^{\text{rules related to input } l}\tilde{\mu}_{(l,k)}P_{(l,k)}(x_i, x_j, \boldsymbol{a}_{(l,k)})\right)\end{aligned} \tag{2}$$

In the above expression, we use an abbreviated notation to describe an activation level of the "k"th rule to be in the form

$$\tilde{\mu}_{(l,k)} = \frac{\mu_{(l,k)}}{\sum\limits_{k=1}^{\textit{total rule related to input } l}\mu_{(l,k)}} \tag{3}$$

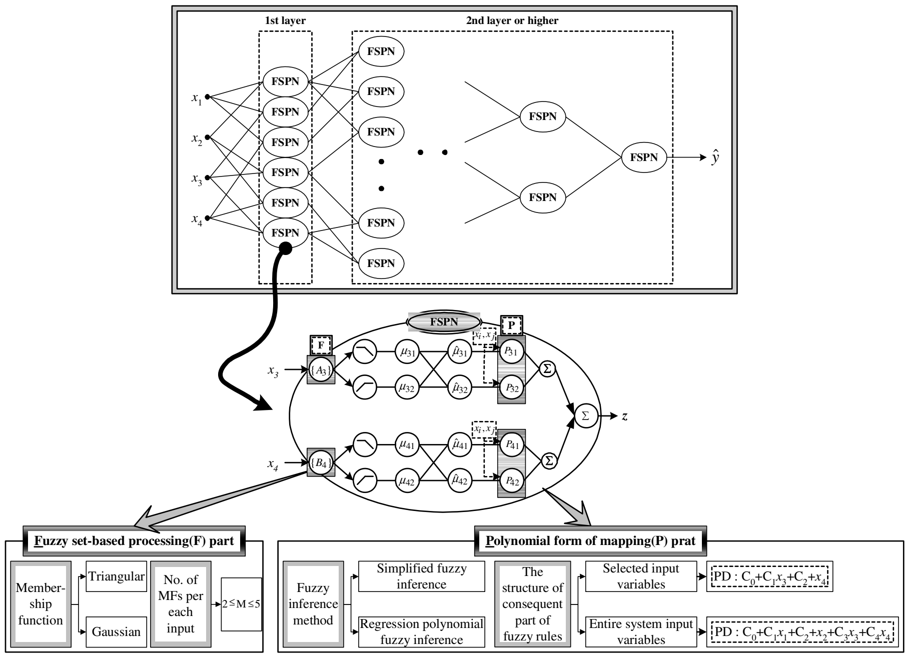


Fig. 1. A general topology of the FSPN based FPNN along with the structure of the generic FSPN module (F: fuzzy set-based processing part, P: the polynomial form of mapping)

Table 1. Different forms of the regression polynomials forming the consequence part of the fuzzy rules

Order of the polynomial \ No. of inputs	1	2	3
0 (Type 1)	**Constant**	**Constant**	**Constant**
1 (Type 2)	**Linear**	**Bilinear**	**Trilinear**
2 (Type 3)	**Quadratic**	**Biquadratic-1**	**Triquadratic-1**
2 (Type 4)		**Biquadratic-2**	**Triquadratic-2**

1: Basic type, 2: Modified type

3 Information Granulation Through Hard C-Means Clustering Algorithm

Information granules are defined informally as linked collections of objects (data points, in particular) drawn together by the criteria of indistinguishability, similarity or functionality [9]. Granulation of information is a procedure to extract meaningful concepts from numeric data and an inherent activity of human being carried out with intend of better understanding of the problem. We granulate information into some classes with the aid of Hard C-means clustering algorithm, which deals with the conventional crisp sets.

3.1 Definition of the Premise and Consequent Part of Fuzzy Rules Using Information Granulation

We assume that given a set of data X={$\mathbf{x}_1$,$\mathbf{x}_2$,…,$\mathbf{x}_n$} related to a certain application, there are some clusters which are capable of being found through HCM. The center point and the membership elements represent each cluster. The set of membership elements is crisp. To construct a fuzzy mode, we should transform the crisp set into the fuzzy set. The center point means the apex of the membership function of the fuzzy set. Let us consider building the consequent part of fuzzy rule. We can think of each cluster as a sub-model composing the overall system. The fuzzy rules of Information Granulation-based FSPN are as followings.

$$\begin{aligned}&\text{if } x_p \text{ is } \boldsymbol{A}^*_k \text{ then } z-m_{pk} = P_{pk}((x_i-v^i_{pk}),(x_j-v^j_{pk}),a_{pk})\\ &\text{if } x_q \text{ is } \boldsymbol{B}^*_k \text{ then } z-m_{qk} = P_{qk}((x_i-v^i_{qk}),(x_j-v^j_{qk}),a_{qk})\end{aligned} \tag{4}$$

Where, $\boldsymbol{A}^*_k$ and $\boldsymbol{B}^*_k$ mean the fuzzy set, the apex of which is defined as the center point of information granule (cluster) and m_{pk} is the center point related to the output variable on cluster$_{pk}$, v^i_{pk} is the center point related to the *i-th* input variable on cluster$_{pk}$ and $\boldsymbol{a}_{qk}$ is a vector of the parameters of the conclusion part of the rule while $P((x_i-v^i),(x_j-v^j),a)$ denoted the regression polynomial forming the consequence part of the fuzzy rule which uses several types of high-order polynomials (linear, quadratic, and modified quadratic) besides the constant function forming the simplest version of the consequence; refer to Table 1. If we are given m inputs and one output system and the consequent part of fuzzy rules is linear, the overall procedure of modification of the generic fuzzy rules is as followings. The given inputs are $\mathbf{X}$=[$\mathbf{x}_1$ $\mathbf{x}_2$ … $\mathbf{x}_m$] related to a certain application, where $\mathbf{x}_k$ =[x_{k1} … x_{kn}]^T, n is the number of data and m is the number of variables and the output is $\mathbf{Y}$=[y_1 y_2 … y_n]^T.

Step 1) build the universe set

Universe set U={{x_{11}, x_{12}, …, x_{1m}, y_1}, {x_{21}, x_{22}, …, x_{2m}, y_2}, …, {x_{n1}, x_{n2}, …, x_{nm}, y_n}}

Step 2) build m reference data pairs composed of [$\mathbf{x}_1$;Y], [$\mathbf{x}_2$;Y], and [$\mathbf{x}_m$;Y].

Step 3) classify the universe set U into ***l*** clusters such as c_{i1}, c_{i2}, …, c_{il} (subsets) by using HCM according to the reference data pair [$\mathbf{x}_i$;Y]. Where c_{ij} means the *j-th* cluster (subset) according to the reference data pair [$\mathbf{x}_i$;Y].

Step 4) construct the premise part of the fuzzy rules related to the *i-th* input variable (x_i) using the directly obtained center points from HCM.

Step 5) construct the consequent part of the fuzzy rules related to the *i-th* input variable (x_i). On this step, we need the center points related to all input variables. We should obtain the other center points through the indirect method as followings.

Sub-step1) make a matrix as equation (5) according to the clustered subsets

$$A^i_j = \left[\begin{array}{cccc|c} x_{21} & x_{22} & \cdots & x_{2m} & y_2 \\ x_{51} & x_{52} & \cdots & x_{5m} & y_5 \\ x_{k1} & x_{k2} & \cdots & x_{km} & y_k \\ \vdots & \vdots & \cdots & \vdots & \vdots \end{array}\right] \tag{5}$$

Where, $\{x_{k1}, x_{k2}, \ldots, x_{km}, y_k\} \in c_{ij}$ and A^i_j means the membership matrix of ***j-th*** subset related to the ***i-th*** input variable.

Sub-step2) take an arithmetic mean of each column on A^i_j. The mean of each column is the additional center point of subset c_{ij}. The arithmetic means of column is equation (6)

$$center\ points = \left[v^1_{ij} \quad v^2_{ij} \quad \cdots \quad v^m_{ij} \mid m_{ij} \right] \tag{6}$$

Step 6) if i is m then terminate, otherwise, set i=i+1 and return step 3.

4 Genetic Optimization of FSPNN with Aid of Symbolic Gene-Type Genetic Algorithms

Let us briefly recall that GAs is a stochastic search technique based on the principles of evolution, natural selection, and genetic recombination by simulating a process of "survival of the fittest" in a population of potential solutions (individuals) to the given problem. GAs are aimed at the global exploration of a solution space. They help pursue potentially fruitful search paths while examining randomly selected points in order to reduce the likelihood of being trapped in possible local minima. The main features of genetic algorithms concern individuals viewed as strings, population-based optimization (where the search is realized through the genotype space), and stochastic search mechanisms (selection and crossover). The conventional genetic algorithms use several binary gene type chromosomes. However, symbolic gene type genetic algorithms use symbolic gene type chromosomes not binary gene type chromosomes. We are able to reduce the solution space with aid of symbolic gene type genetic algorithms. That is the important advantage of symbolic gene type genetic algorithms. In order to enhance the learning of the FPNN, we use GAs to complete the structural optimization of the network by optimally selecting such parameters as the number of input variables (nodes), the order of polynomial, and input variables within a FSPN.

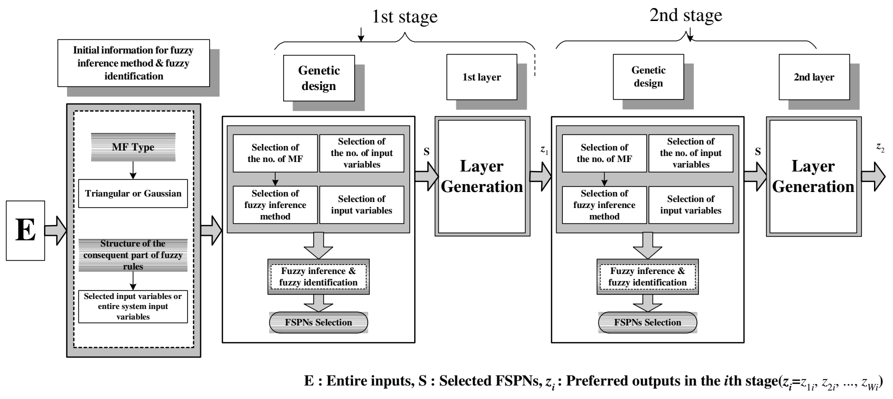

1st stage
2nd stage
Initial information for fuzzy inference method & fuzzy identification
Genetic design
1st layer
Genetic design
2nd layer
MF Type
Triangular or Gaussian
Structure of the consequent part of fuzzy rules
Selected input variables or entire system input variables
E
Selection of the no. of MF
Selection of the no. of input variables
Selection of fuzzy inference method
Selection of input variables
Fuzzy inference & fuzzy identification
FSPNs Selection
S
Layer Generation
z1
S
Layer Generation
z2
E : Entire inputs, S : Selected FSPNs, zi : Preferred outputs in the ith stage(zi=z1i, z2i, ..., zWi)

Fig. 2. Overall genetically-driven structural optimization process of FSPNN

In this study, GA uses the serial method of binary type, roulette-wheel used in the selection process, one-point crossover in the crossover operation, and a binary inversion (complementation) operation in the mutation operator. To retain the best individual and carry it over to the next generation, we use elitist strategy [3]. The overall genetically-driven structural optimization process of FPNN is visualized in Fig. 2.

5 The Design Procedure of Genetically Optimized FSPNN (gFSPNN)

The framework of the design procedure of the genetically optimized Fuzzy Polynomial Neural Networks (FPNN) with fuzzy set-based PNs (FSPN) comprises the following steps

[Step 1] *Determine system's input variables*
[Step 2] *Form training and testing data*
[Step 3] *specify initial design parameters*

- Fuzzy inference method
- Type of membership function : Triangular or Gaussian-like MFs
- Number of MFs allocated to each input of a node
- Structure of the consequence part of the fuzzy rules

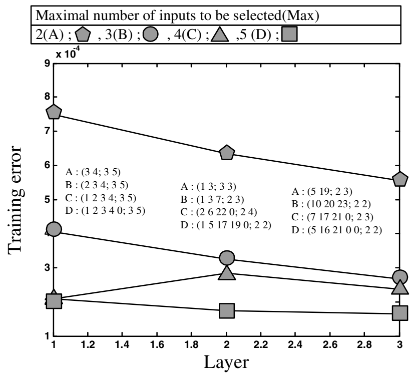
[Step 4] *Decide upon the FSPN structure through the use of the genetic design*
[Step 5] *Carry out fuzzy-set based fuzzy inference and coefficient parameters estimation for fuzzy identification in the selected node(FSPN)*
[Step 6] *Select nodes (FSPNs) with the best predictive capability and construct their corresponding layer*
[Step 7] *Check the termination criterion*
[Step 8] *Determine new input variables for the next layer*

6 Experimental Studies

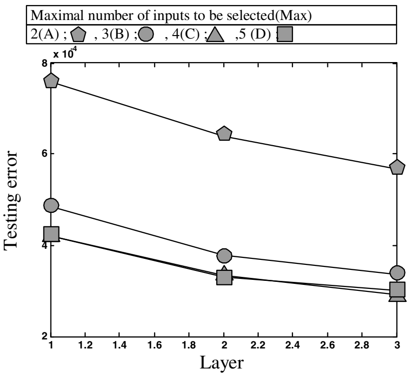
We demonstrate how the IG-gFSPNN can be utilized to predict future values of a chaotic Mackey-Glass time series. This time series is used as a benchmark in fuzzy and neurofuzzy modeling. The performance of the network is also contrasted with some other models existing in the literature [5-7]. The time series is generated by the chaotic Mackey-Glass differential delay equation. To come up with a quantitative evaluation of the network, we use the standard RMSE performance index.

Fig. 3 depicts the performance index of each layer of gFPNN with Type T* according to the increase of maximal number of inputs to be selected.

Fig. 4 illustrates the different optimization process between gFSPNN and the proposed IG-gFSPNN by visualizing the values of the performance index obtained in successive generations of GA when using Type T*.

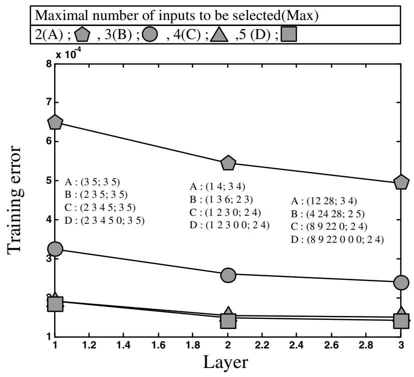


(a-1) Training error

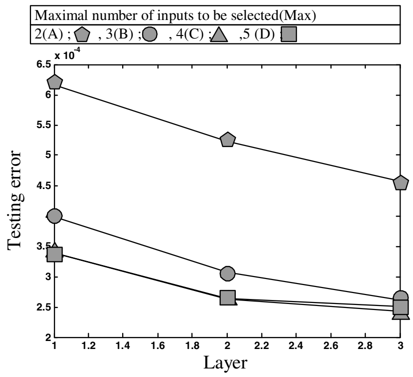


(a-2) Testing error

(a) Triangular MF



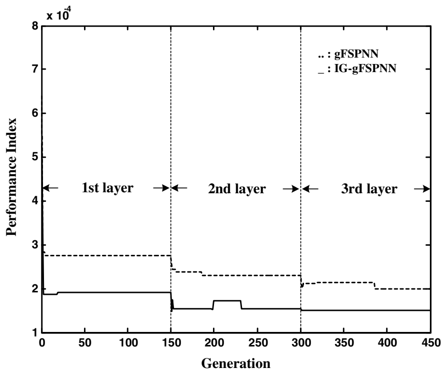
(b-1) Training error



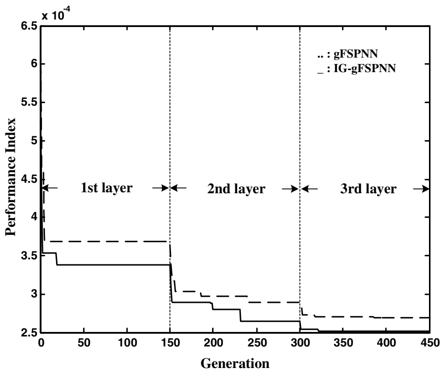
(b-2) Testing error

(b) Gaussian-like MF

Fig. 3. Performance index of IG-gFSPNN (with Type T*) with respect to the increase of number of layers



(a) Training error



(b) Testing error

Fig. 4. The optimization process quantified by the values of the performance index (in case of using Gaussian MF with Max=5 and Type T*)

Table 2 summarizes a comparative analysis of the performance of the network with other models.

Table 2. Comparative analysis of the performance of the network; considered are models reported in the literature

Model				Performance index		
				PI	PI_s	EPI_s
Wang's model[5]				0.044		
				0.013		
				0.010		
ANFIS[6]					0.0016	0.0015
FNN model[7]					0.014	0.009
Proposed IG-gFSPNN	Type T*	Triangular (2nd layer)	Max=5		1.72e-4	3.30e-4
		Gaussian (2nd layer)	Max=5		1.48e-4	2.61e-4

7 Concluding Remarks

In this study, we have surveyed the new structure and meaning of fuzzy rules and investigated the GA-based design procedure of Fuzzy Polynomial Neural Networks (FPNN) along with its architectural considerations. The whole system is divided into some sub-systems that are classified according to the characteristics named information granules. Each information granule seems to be a representative of the related sub-systems. A new fuzzy rule with information granule describes a sub-system as a stand-alone system. A fuzzy system with some new fuzzy rules depicts the whole system as a combination of some stand-alone sub-system.

The GA-based design procedure applied at each stage (layer) of the FSPNN leads to the selection of the preferred nodes (or FSPNs) with optimal local characteristics (such as the number of input variables, the order of the consequent polynomial of fuzzy rules, and input variables) available within FSPNN. The comprehensive experimental studies involving well-known datasets quantify a superb performance of the network in comparison to the existing fuzzy and neuro-fuzzy models.

Acknowledgement. This paper was supported by Wonkwang University in 2005.

References

1. Oh, S.K., Pedrycz, W.: The design of self-organizing Polynomial Neural Networks. Information Science. **141** (2002) 237-258
2. Michalewicz Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag Berlin Heidelberg (1996)
3. Goldberg, D.E.: Genetic algorithms in Search, Optimization, and Machine Learning. Addison-Wesley (1989)
4. Oh, S.-K., Pedrycz, W.: Fuzzy Polynomial Neuron-Based Self-Organizing Neural Networks. Int. J. of General Systems. **32**(3) (2003) 237-250

5. Wang, L. X., Mendel, J. M.: Generating fuzzy rules from numerical data with applications. IEEE Trans. Systems, Man, Cybern. **22**(6) (1992) 1414-1427
6. Jang, J. S. R.: ANFIS: Adaptive-Network-Based Fuzzy Inference System. IEEE Trans. System, Man, and Cybern. **23**(3) (1993) 665-685
7. Oh, S.-K., Pedrycz W., Ahn, T.-C., Self-organizing neural networks with fuzzy polynomial neurons. Applied Soft Computing. **2**(1F) (2002) 1-10
8. Maguire, L. P., Roche, B., McGinnity, T. M., McDaid, L. J.: Predicting a chaotic time series using a fuzzy neural network. Information Sciences. **112** (1998) 125-136
9. Zadeh, L. A.: Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic. Fuzzy Sets System. **90** (1997) 111-117

A Hybrid Self-learning Approach for Generating Fuzzy Inference Systems

Yi Zhou[1] and Meng Joo Er[2]

School of Electrical and Electronic Engineering
Nanyang Technological University
Nanyang Ave, Singapore, 639798
[1]yizhou@pmail.ntu.edu.sg,[2]emjer@ntu.edu.sg

Abstract. In this paper, a novel hybrid self-learning approach termed Enhanced Dynamic Self-Generated Fuzzy Q-Learning (EDSGFQL) for automatically generating a Fuzzy Inference System (FIS) is presented. In the EDSGFQL approach, the structure of an FIS is generated via Reinforcement Learning (RL) while the centers of Membership Functions (MFs) are updated by an extended Self Organizing Map (SOM) algorithm. The proposed EDSGFQL methodology can automatically create, delete and adjust fuzzy rules without any *priori* knowledge. In the EDSGFQL approach, fuzzy rules of an FIS are regarded as agents of the entire system and all of the rules are recruited, adjusted and terminated according to their contributions and participation. At the mean time, the centers of MFs are adjusted to move to the real centers in the sense of feature representation by the extended SOM approach. Comparative studies on a wall-following task by a mobile robot have been done for the proposed EDSGFQL approach and other current methodologies and the demonstration results show that the proposed EDSGFQL approach is superior.

1 Introduction

Generation of Fuzzy Inference Systems (FISs) is always an attractive area for research. However, the conventional approaches for designing FISs are subjective, which require significant human's efforts. Other than time consuming, the subjective approaches may not be successful if the system is too complex or uncertain. Therefore, many researchers have been seeking automatic methods for generating the FIS [1]- [5]. A paradigm of acquiring the parameters of fuzzy rules was proposed in [1] and a self-identified structure was achieved by a Supervised Learning (SL) approach termed Generalized Dynamic Fuzzy Neural Networks (GDFNN) in [2]. However, the training data are not always available especially when a human being has little knowledge about the system or the system is uncertain. In those situations, Unsupervised Learning (UL) and Reinforcement Learning (RL) are preferred over SL as UL and RL are learning processes that do not need any supervisor to tell the learner what action to take. A number of researchers have applied RL to train the consequent parts of an FIS [3]- [5]. However, the preconditioning parts of the FIS are either predefined as in [3] or through the ε-completeness and the squared TD error criteria in [4] or through the "aligned clustering" in [5]. The TD error criterion proposed in [4] generates new rules when the squared TD error is too big. However, this criterion is not applicable for RL methods which are not based on TD, and the thresholds of this TD error criterion are hard to define. An "aligned clustering algorithm" which is similar to the ε-completeness criterion in [4] has been adopted by the author of [5] in online Clustering and Q-value based Genetic Algorithm learning schemes for Fuzzy system design (CQGAF) to generate a fuzzy structure. However, both DFQL and CQGAF cannot delete fuzzy rules once they are generated even when the rules become redundant. To reduce the computational cost and the training time, dormant or unnecessary rules should be deleted. Thus, a scheme termed Dynamic Self-Generated Fuzzy Q-learning (DSGFQL) which is capable of automatically generating and pruning fuzzy rules as well as fine tuning the premise parameters of an FIS by RL was proposed in [6]. Based on the key idea of the DSGFQL algorithm, an enhanced approach termed Enhanced Dynamic Self-Generated Fuzzy Q-Learning (EDSGFQL) is proposed in this paper. In this novel EDSGFQL method, an extended Self Organizing Map (SOM) approach is adopted to adjust the centers of the EBF neurons. The main reason of choosing the SOM is that the EBF networks are local approximation and the centers of local units (EBF neurons) are adjusted to move to the real centers in the sense of feature representation [7].

By virtue of the capability of generating and pruning fuzzy rules as well as adjusting the Membership Functions (MFs) by the EDSGFQL methodology, experimental results and comparative studies on a wall-following task by a mobile robot demonstrate that the proposed EDSGFQL approach is superior.

2 Self-Generated Fuzzy Inference Systems by EDSGFQL

2.1 Architecture of the EDSGFQL System

The architecture of the EDSGFQL system is based on extended EBF neural networks, which are functionally equivalent to TSK fuzzy systems [1]. The neural networks structure of the EDSGFQL system is depicted in Fig. 1. Layer one is an input layer and layer two is a fuzzification layer which evaluates the MFs of the input variables. The MF is chosen as a Gaussian function and each input variable x_i $(i = 1, 2, ..., N)$ has L MFs. Layer three is a rule layer. Normalization takes place in layer 4 and nodes of layer five define output variables. For details of the architecture, readers may refer to [4].

2.2 Sharing Mechanism for Newly Generated Rules

Similarly to that in [4] and [5], a new rule will be created if the ε-completeness criterion fails. For details of the ε-completeness criterion, readers may refer to [2]

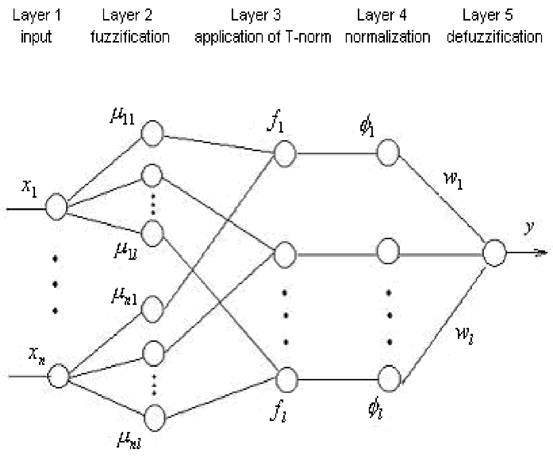


Fig. 1. Structure of the EDSGFQL System

and [4]. In order to reduce the training time, the initial Q-values for the newly generated fuzzy rule are set by the nearest neighbors. For instance, if rules R_m and R_n are the two nearest neighbors to the newly generated rule R_i, then

$$q(i,j) = \frac{q(m,j) * f_m(i) + q(n,j) * f_n(i)}{f_m(i) + f_n(i)}, \quad j = 1, 2, \cdots M. \tag{1}$$

where $f_m(i)$ and $f_n(i)$ stand for the firing strengths of the center of the newly generated ith rule R_i to rules R_m and R_n.

2.3 Generating an FIS Via RL

The objective of RL is to maximize the expected reward value. Therefore, the reward value function becomes a natural criterion for judging the performance of the RL agents. The reward is to be checked periodically, e.g. 1000 training steps or an episode, i.e.

$$Avg\_Reward = \frac{\sum_{t=k}^{t=k+T} r(t)}{T} \tag{2}$$

At the mean time, a reward sharing mechanism which is proposed in [8] is used here to share the reinforcement with each local rule according to its contributions. The local reward for each fuzzy agent is given, as in [8], as follows:

$$r_{local}^{j}(t) = \phi_j r(t) \quad j = 1, 2, ..., L \tag{3}$$

The local reward offers a direct evaluation of contributions of fuzzy agents. The higher the local reward is obtained, the better contributions are offered by the fuzzy agent. If the FIS passes the ε-completeness criterion but the global average reward is less than a threshold k_g, it means that the input space is well partitioned but the overall performance needs to be improved. To resolve this problem, the weights of some good agents should be increased which means that the system will be modified by promoting the agent with the best performance. In this case, the width of the jth fuzzy rule's MF (the one with the best local reward) will be increased as follows

$$\sigma_{ij} = \kappa\sigma_{ij}, \qquad i = 1, 2...N \tag{4}$$

where κ is slightly larger than 1.

On the other hand, a fuzzy agent should be punished if the local reward is too bad. If local reward of a fuzzy agent is less than a certain threshold k_{lh}, the individual contributions are unsatisfactory and that fuzzy rule will be deleted.

Moreover, if the local reward is larger than k_{lh}, but smaller than a light threshold k_{ll}, a punishment will be given to the agent by decreasing its width of the MF as follows:

$$\sigma_{ik} = \tau\sigma_{ik}, \qquad i = 1, 2...N \tag{5}$$

where τ is a positive value less than 1.

Besides the reward evaluation, firing strength should also be considered for system evaluation as it is a measurement for participation. If a fuzzy agent has very low firing strength during the entire episode or a long period of time recently, it means that this rule is unnecessary for the system. As more fuzzy rules mean more computation and longer training time, the rule whose mean firing strength over a long period of time is less than a threshold k_f should be deleted.

Remark: If a fuzzy rule keeps failing the light local reward check, its firing strength will be reduced by decreasing the width of its MF. When the width is reduced to a certain level, it will fail the firing strength criterion and will be deleted.

At the early stage of training, each fuzzy agent needs time to adjust its own performance. For a training system, it is natural to set the demanding requirement small at the beginning and increase it later when the training becomes stable. Thus, the idea of gradualism learning in [2] and [9] which uses coarse learning in the early training stage and fine learning in the later stage is adopted here. For details, reader may refer to [2] and [6].

2.4 Extended Self Organizing Map

The development of the Self Organizing Map (SOM) is inspired by the research in neurobiology. During the training of the EDSGFQL algorithm, online SOM is performed for each incoming training pattern. Extended from the original SOM in [10], the concept of fuzzy region is applied to take the place of lattice point. In the original SOM, the distances between the input data and the center points are considered. The neuron whose center is with the smallest distance with the input is regarded as the best matching neuron. The shortcoming of this method is that it does not consider the width or the size of the fuzzy region. To circumvent this, the M-distance or firing strength of the input data is considered for deciding the winning neuron. The fuzzy agent who is with the smallest M-distance or with the biggest firing strength is appointed as the best matching agent. Obviously, the extended method is more suitable as it considers each fuzzy agent as a region instead of a point. In other words, the extended SOM algorithm considers not only the center position but also the width of the fuzzy region.

Similar to as the original SOM, the winning fuzzy agent and its topological neighbors are updated so that they move closer to the current input $X(k)$ in the input space. The update rules for the centers are as follows:

$$C_j(t+1) = C_j(t) + \beta(t)\phi_\nu(j)h_{\nu j}(t)[X(t) - C_j(t)], \quad j = 1, 2, ..., L \tag{6}$$

where ν denotes the winning neuron, $\beta(t)$ is the adaptation coefficient and $\phi_\nu(j)$ is the firing strength of the center of the jth rule R_j to the winning neuron ν. The term $h_{\nu j}(t)$ is the neighborhood kernel centered on the winner unit and is given by

$$h_{\nu j}(t) = exp(-\frac{||C_\nu - C_j||^2}{2\sigma^2(t)}) \tag{7}$$

Here, $\beta(t)$ and $h_{\nu j}(t)$ decrease monotonically with t. Therefore, as an enhancement from the DSGFQL algorithm, the centers of MFs are updated according to the extended SOM algorithm for the EDSGFQL approach.

The flowchart of the EDSGFQL algorithm is presented in Fig. 2.

3 Simulation Studies

In order to apply the EDSGFQL algorithm and compare them with other related methodologies, the approaches have been applied to control a Khepera mobile robot of [11] for a wall-following task.

The Khepera mobile robot is cylindrical in shape, with 55 mm in diameter and 30 mm in height. It is a light robot with only 70g. The robot is equipped with two dc motors coupled with incremental sensors, eight analogue Infra-Red (IR) proximity sensors and an on-board power supply which is the same as that in [4] which is shown in Fig. 3. Each IR sensor is composed of an emitter and an independent receiver. The dedicated electronic interface uses multipliers, sample holds and operational amplifiers. This allows absolute ambient light and estimation, by reflection, of the relative position of an object to the robot to be measured. By this estimation, the distance between the robot and the obstacle can be derived.

Similar to the experiment in [4], the task is to design a controller for wall-following, while the robot is only moving in clockwise direction at a fixed speed. Four input variables, which are the values of sensors S_i ($i = 0, 1, 2, 3$), are considered and all values are normalized. The output of the controller is the steering angle of the robot. In this experiment, the set of discrete actions is

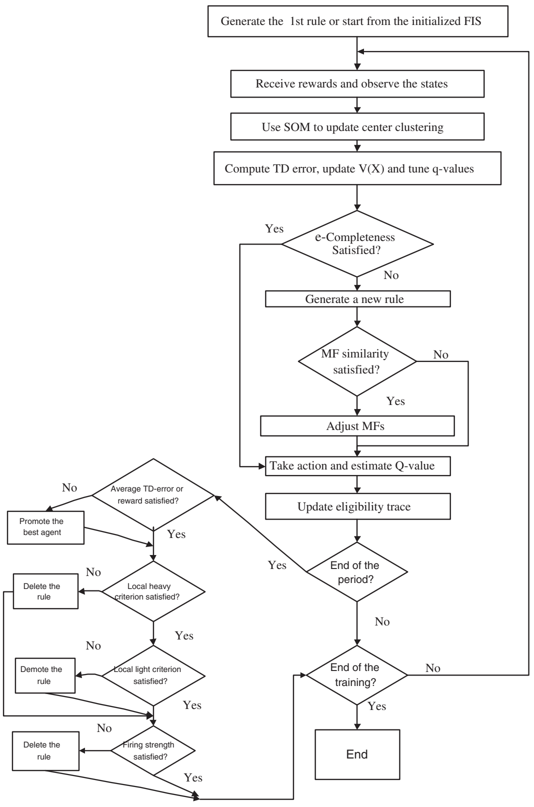

Generate the 1st rule or start from the initialized FIS
Receive rewards and observe the states
Use SOM to update center clustering
Compute TD error, update V(X) and tune q-values
e-Completeness Satisfied?
Yes
No
Generate a new rule
MF similarity satisfied?
No
Yes
Adjust MFs
Take action and estimate Q-value
Update eligibility trace
End of the period?
Yes
No
End of the training?
No
Yes
End
Average TD-error or reward satisfied?
No
Yes
Promote the best agent
Local heavy criterion satisfied?
No
Yes
Delete the rule
Local light criterion satisfied?
No
Yes
Demote the rule
Firing strength satisfied?
No
Yes
Delete the rule

Fig. 2. Flowchart of the learning algorithm for EDSGFQL

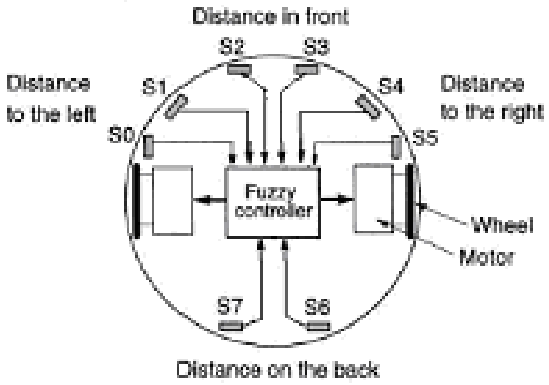


Fig. 3. Position and orientation of sensors on the Khepera II

$A = [-30, -25, -20, -15, -10, -5, 0, 5, 10, 15, 20, 25, 30]$ and continuous actions will be generated by fuzzy approaches.

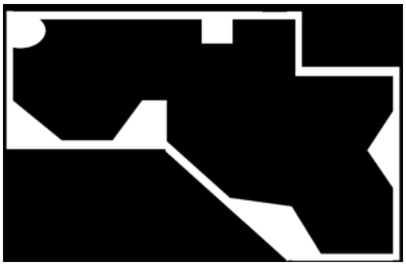


Fig. 4. Training environment

The aim of the wall-following task is to control the mobil robot to follow a wall while keeping a distance from the wall in the range of $[d_{\_}, d_{+}]$. The same reward function as what is used in [4] is adopted:

$$r = \begin{cases} 0.1, & if(d_{\_} < d < d_{+}) \ and \ (U \in [-8^o, +8^o]) \\ -3.0, & if(d \leq d_{\_}) \ or \ (d_{+} \leq d) \\ 0.0, & otherwise. \end{cases} \qquad (8)$$

Here, $d_{\_} = 0.15$ and $d_{+} = 0.85$ which are normalized values, U is the output steering angle and all the settings are the same as the settings used in [4].

The simulation version of the Khepera robot reported in [12] is used for comparison studies. A complicated environment which is shown in Fig. 4 is adopted. In [4], it was shown that the DFQL algorithm is superior to the basic fuzzy controller, fixed FQL [3] and continuous-action Q-learning [13]. Hence, it is sufficient for us to compare the EDSGFQL approach with the DFQL of [4] and CQGAF of [5] and DSGFQL in this work. The performances of these different approaches are evaluated at every episode of 1000 control steps. Three criteria, namely *number of failures*, *rewards* and *number of rules*, are considered for measuring the performance. The first two criteria are to measure how well the task has been performed and the last one is to measure how much computational cost has been spent.

For the DFQL, the same setting as in [4] has been adopted, i.e. initial Q-value, $k_q = 3.0$, exploration rate, $S_p = 0.001$; discounted factor, $\gamma = 0.95$; trace-decay factor, $\lambda = 0.7$; learning rate, $\alpha = 0.05$; ε-completeness, $\varepsilon = 0.5$; similarity of MF, $k_{mf} = 0.3$; TD error factor $K = 50$; TD error criterion, k_e=1 and set of discrete actions, $A = [-30, -25, -20, -15, -10, -5, 0, 5, 10, 15, 20, 25, 30]$. The CQGAF proposed in [5] utilizes Genetic Algorithms (GA) to obtain an optimal solution for the action set and employs the aligned clustering algorithm for structure identification of the FIS. As this paper focuses on self-generation of the FIS structure, the same action set as that for the DFQL is applied for the CQGAF, which does not apply the GA approach to train the action set. The aligned clustering algorithm in [5] is adopted and the consequent part of FIS is trained through normal FQL which does not apply the GA in the CQGAF. As the aligned clustering algorithm is similar to the ε-completeness criterion in the DFQL but described in another way, all parameters for the aligned clustering algorithm are set the same as those listed above for the DFQL. For the DSGFQL, the global reward thresholds are $k_g^{max} = -0.05$ and $k_g^{min} = -0.45$; heavy local thresholds are $k_{lh}^{max} = -0.30$ and $k_{lh}^{min} = -0.10$; light local reward thresholds are $k_{ll}^{max} = -0.20$ and $k_{ll}^{min} = 0$; the firing strength thresholds are $k_f = 0.0002$ and $K_r = 20$, $\kappa = 1.05$ and $\tau = 0.95$. The adaption coefficient is β=0.05 and σ=1 for the EDSGFQL approach.

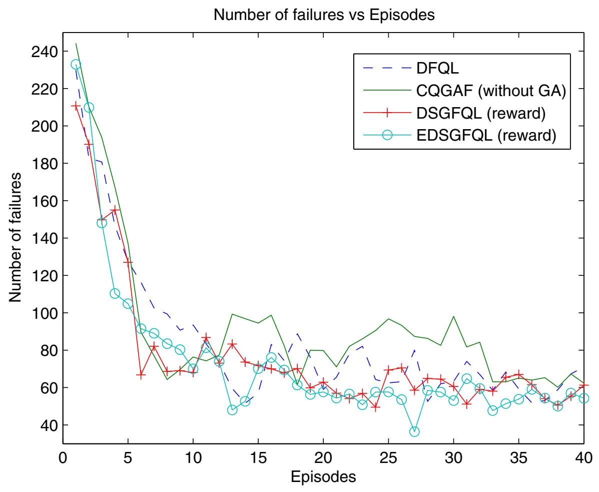


Fig. 5. Comparison of performances of DFQL, CQGAF, DSGFQL and EDSGFQL: Number of failures vs number of episodes

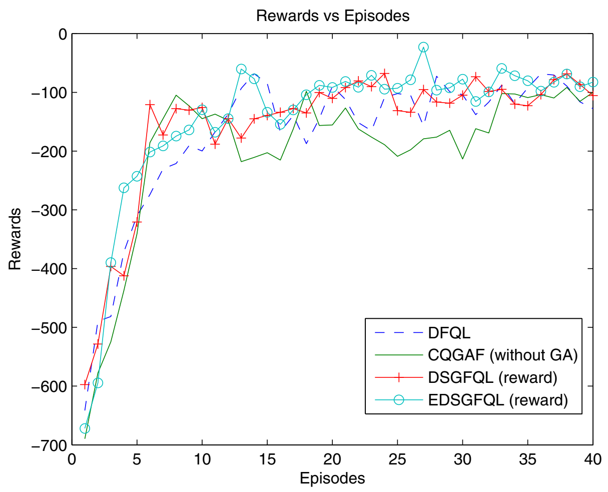


Fig. 6. Comparison of performances of DFQL, CQGAF and DSGFQL and EDSGFQL: Reward value vs number of episodes

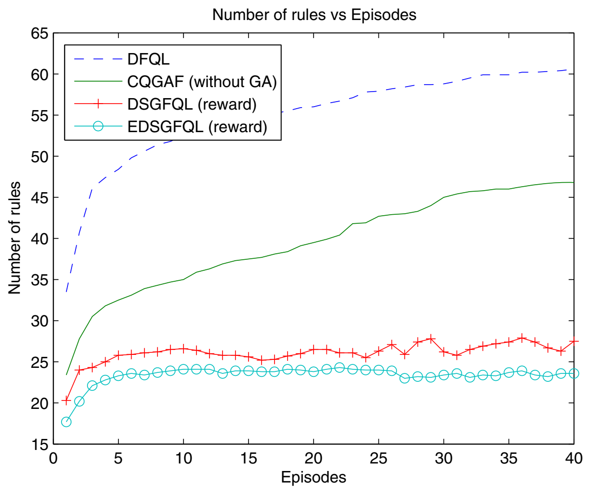


Fig. 7. Comparison of performances of DFQL, CQGAF and DSGFQL and EDSGFQL: Number of fuzzy rules vs number of episodes

In this work, comparison studies among the DFQL, the CQGAF, the original DSGFQL and the EDSGFQL algorithms are carried out. The performances of these four approaches have been measured and the mean values for 40 episodes over 10 runs have been presented in Figs. 5-7.

From Figs. 5-7, it can be concluded that the EDSGFQL algorithm is superior to the DFQL, CQGAF without the GA approach and the DSGFQL as the EDSGFQL methods achieve similar or even better performances than those approaches but with much smaller number of rules. The superiority of the EDSGFQL algorithm is due to the pruning capability as well as the self-modification of the fuzzy MFs. By updating the centers of MFs through the extended SOM approach, the original DSGFQL is enhanced.

4 Conclusions

In this paper, a dynamic self-generated FIS methodology has been proposed. Compared with conventional subjective approaches to generate an FIS, the proposed EDSGFQL approach can automatically generate an FIS without any *priori* knowledge. Compared with recent self-generation approaches, the EDSGFQL algorithm can self-generate and delete fuzzy rules. Normal fuzzy RL is actually using the continuity of fuzzy logic to convert discrete states and actions to continues ones. On the other hand, the EDSGFQL approach proposed employs the idea of RL to generate the structure and adjust the antecedent parameters of an FIS. Compared with conventional FQL approaches which apply FIS to enhance Q-learning, the key feature of the EDSGFQL methodology is that they adopt the RL approach to train the the structure and premises of an FIS. Therefore, not only the consequent parts of the FIS are trained by RL, but also the entire structure and parameters are self-generated through RL. Moreover, by utilizing the SOM to update the centers of MFs, the original DSGFQL algorithm is enhanced. By virtue of structure identification and parameter modifications, simulation studies demonstrate that the EDSGFQL approach is superior to the original DSGFQL which is superior to the DFQL and CQGAF in generating an FIS.

References

1. J. S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Trans. System, Man and Cybernetics*, vol. 23, pp. 665-684, 1993.
2. S. Wu, M. J. Er and Y. Gao,"A fast approach for automatic generation of fuzzy rules by generailized dynamic fuzzy neural networks," *IEEE Trans. Fuzzy Systems*, vol. 9, pp. 578-594, 2001.
3. L. Jouffe, "Fuzzy inference system learning by reinforcement methods," *IEEE Trans. Systems, Man, and Cybernetics, Part C*, vol. 28, pp. 338-355, 1998.
4. M. J. Er, and C. Deng, "Online tuning of fuzzy inference systems using dynamic fuzzy Q-learning," *IEEE Trans on Systems, Man and Cybernetics, Part B*, vol. 34, no. 3, pp. 1478-1489, June 2004.
5. C. F. Juang, "Combination of online clustering and Q-value based GA for reinforcement fuzzy systems," *IEEE Trans on Fuzzy Systems*, vol. 13, no. 3, pp. 289-302, June 2005.
6. M. J. Er and Y. Zhou, "Dynamic self-generated fuzzy systems for reinforcement learning," *in Proc. International Conference on Computational Intelligence for Modelling Control and Automation, CIMCA'2005, Vienna, Austria, pp. 193-198, 2005.*
7. M. J. Er, Z. Li, H. Cai and Q. Chen, "Adaptive noise cancellation using enhanced dynamic fuzzy neural networks," *IEEE Trans. Fuzzy Systems*, vol. 13, no. 3, pp. 331-342, 2005.
8. P. Ritthiprava, T. Maneewarn, D. Laowattana, and J. Wyatt, "A modified approach to fuzzy-Q learning for mobile robots," *in Proc. IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, pp.2350 - 2356, 2004.
9. K. B. Cho and B. H. Wang, "Radial basis function based adaptive fuzzy systems and their applications to system identification and prediction," *Fuzzy Set Systems*, vol. 6, pp. 12-32, 1998.
10. T. Kohonen, "Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, vol. 43, pp. 59-69, 1982.
11. S. A. K-Team, Khepera 2 user manual, Switzerland, 2002.
12. T. Nilsson. [Online]. Avaiable: http://www.kiks.f2s.com
13. J. D. R. Millan, D. Posenato, and E. Dedieu, "Continuous-action Q-learning," *Machine Learning*, vol. 49, no. 2-3, pp.247-265, 2002.

A Fuzzy Clustering Algorithm for Symbolic Interval Data Based on a Single Adaptive Euclidean Distance

Francisco de A.T. de Carvalho

Centro de Informatica - CIn/UFPE
Av. Prof. Luiz Freire, s/n - Cidade Universitaria
CEP: 0740-540 Recife-PE, Brazil
fatc@cin.ufpe.br

Abstract. The recording of symbolic interval data has become a common practice with the recent advances in database technologies. This paper presents a fuzzy *c*-means clustering algorithm for symbolic interval data. This method furnishes a partition of the input data and a corresponding prototype (a vector of intervals) for each class by optimizing an adequacy criterion which is based on a suitable single adaptive Euclidean distance between vectors of intervals. Experiments with real and synthetic symbolic interval data sets showed the usefulness of the proposed method.

1 Introduction

Cluster analysis have been widely used in numerous fields including pattern recognition, data mining and image processing. Their aim is to organize a set of items into clusters such that items within a given cluster have a high degree of similarity, whereas items belonging to different clusters have high degree of dissimilarity [10].

Partitioning methods can be divided in hard clustering and fuzzy clustering. Hard clustering furnishes a hard partition where each object of the data set is assigned to one and only one cluster. Fuzzy clustering [1] generates a fuzzy partition that furnishes a degree of membership of each pattern in a given cluster. In many real applications fuzzy clustering outperforms hard clustering methods particularly when the classes are not well separated.

The partitioning dynamic cluster algorithms [6] are iterative two steps relocation hard clustering algorithms involving at each iteration the construction of the clusters and the identification of a suitable representative or prototype (means, factorial axes, probability laws, groups of elements, etc.) of each cluster by locally optimizing an adequacy criterion between the clusters and their corresponding prototypes. This optimization process begins from a set of prototypes or an initial partition and interactively applies an *allocation step* (the prototypes are fixed) in order to assign the patterns to the clusters according to their proximity to the prototypes. This is followed by a *representation step* (the partition is fixed) where the prototypes are updated according to the assignment of the patterns in the allocation step, until achieving the convergence of the algorithm, when the adequacy criterion reaches a stationary value.

The adaptive dynamic clustering algorithm [7] also optimize a criterion based on a measure of fitting between the clusters and their prototypes, but there are distances to compare clusters and their prototypes that change at each iteration. These distances are not determined once and for all, and moreover, they can be different from one cluster to another. The advantage of these adaptive distances is that the clustering algorithm is able to recognize clusters of different shapes and sizes. The initialization, the allocation step and the stopping criterion are nearly the same in the adaptive and non-adaptive dynamic clustering algorithms. The main difference between these algorithms occurs in the representation step which has two stages in the adaptive case: a first stage, where the partition and the distances are fixed and the prototypes are updated, is followed by a second one, where the partition and their corresponding prototypes are fixed and the distances are updated.

Often, objects to be clustered are represented as a vector of quantitative features. However, the recording of interval data has become a common practice in real world applications and nowadays this kind of data is often used to describe objects. Symbolic Data Analysis (SDA) is an area related to multivariate analysis, data mining and pattern recognition, which has provided suitable data analysis methods for managing objects described as a vector of intervals [2].

Concerning dynamical cluster algorithms for symbolic interval data, SDA has provided suitable tools. [8] have presented an ad-hoc fuzzy c-means algorithm to cluster data on the basis of different types of symbolic variables. [12] introduced an algorithm considering context dependent proximity functions and [3] proposed an algorithm using an adequacy criterion based on Hausdorff distances. [11] presented a dynamic cluster algorithm for symbolic interval data based on L_1 Minkowsky distances. [5] proposed an algorithm using an adequacy criterion based on adaptive Hausdorff distances for each cluster. More recently, [4] have introduced (adaptive and non-adaptive) fuzzy c-means clustering algorithm to symbolic interval data.

This paper introduces a fuzzy c-means clustering algorithm for symbolic interval data based on a suitable single adaptive Euclidean distance. This method furnishes a partition of the input data and a corresponding prototype (a vector of intervals) for each class by optimizing an adequacy criterion which is based on a single adaptive Euclidean distance between vectors of intervals. In order to show the usefulness of this method, synthetic interval data sets ranging from different degree of difficulty to be clustered and an application with a real data set were considered. The evaluation of the clustering results is based on an external validity index.

This paper is organized as follow. Section 2 presents the previous fuzzy c-means clustering methods for symbolic inteval data based on (adaptive and non-adaptive) Euclidean distances and it is introduced the model based on a single adaptive Euclidean distance. In section 3 it is presented the evaluation of this method in comparison with previous fuzzy c-means clustering methods having adequacy criterion based on Euclidean (non-adaptive and adaptive for each cluster) distances. The accuracy of the results furnished by these clustering methods is assessed by the corrected Rand index [9] considering synthetic interval data sets in the framework of a Monte Carlo experience and an applications with a real data set. Finally, section 4 gives the conclusions and final remarks.

2 Fuzzy Clustering Algorithms for Symbolic Interval Data

In this section we recall some previous fuzzy clustering methods for symbolic interval data and we introduce the model based on a single adaptive Euclidean distance. Let Ω be a set of n objects indexed by k and described by p interval variables indexed by j. An *interval variable* X [2] is a correspondence defined from Ω in $\Re$ such that for each $i \in \Omega, X(k) = [a, b] \in \Im$, where $\Im$ is the set of closed intervals defined from $\Re$. Each object k is represented as a vector of intervals $\mathbf{x}_k = (x_k^1, \cdots, x_k^p)$, where $x_k^j = [a_k^j, b_k^j] \in \Im = \{[a, b] : a, b \in \Re,\ a \leq b\}$. A prototype $\mathbf{g}_i$ of cluster P_i is also represented as a vector of intervals $\mathbf{g}_i = (g_i^1, \cdots, g_i^p)$, where $g_i^j = [\alpha_i^j, \beta_i^j] \in \Im$.

2.1 Fuzzy c-Means Clustering Method for Symbolic Interval Data

Here we present a version of the fuzzy c-means clustering method for symbolic interval data (labeled as FCMNAD). This methode has been introduced by [4].

As in the standard fuzzy c-means algorithm [1], the FCMNAD clustering method for symbolic interval data is a non-hierarchical clustering method the aim of which is to furnish a fuzzy partition of a set of patterns in c clusters $\{P_1, \ldots, P_c\}$ and a corresponding set of prototypes $\{\mathbf{g}_1, \ldots, \mathbf{g}_c\}$ such that a criterion $J1$ measuring the fitting between the clusters and their representatives (prototypes) is locally minimized. This criterion $J1$ is based on a non-adaptive Euclidean distance and it is defined as:

$$J1 = \sum_{i=1}^{c} \sum_{k=1}^{n} (u_{ik})^m \phi(\mathbf{x}_k, \mathbf{g}_i) = \sum_{i=1}^{c} \sum_{k=1}^{n} (u_{ik})^m \sum_{j=1}^{p} \left[(a_k^j - \alpha_i^j)^2 + (b_k^j - \beta_i^j)^2 \right] \quad (1)$$

where

$$\phi(\mathbf{x}_k, \mathbf{g}_i) = \sum_{j=1}^{p} \left[(a_k^j - \alpha_i^j)^2 + (b_k^j - \beta_i^j)^2 \right] \quad (2)$$

is the square of an Euclidean distance measuring the dissimilarity between a pair of vectors of intervals, $\mathbf{x}_k = (x_k^1, \ldots, x_k^p)$ is a vector of intervals describing the $k - th$ item, $\mathbf{g}_i = (g_i^1, \ldots, g_i^p)$ is a vector of intervals describing the prototype of class P_i, u_{ik} is the membership degree of pattern k in cluster P_i and $m \in]1, +\infty[$ is a parameter that controls the fuzziness of membership for each pattern k.

As in the standard fuzzy c-means algorithm [1], this algorithm sets an initial membership degree for each item k in each cluster P_i and alternates a *representation step* and an *allocation step* until convergence when the criterion $J1$ reaches a stationary value representing a local minimum.

Representation Step: Definition of the Best Prototypes. In the representation step, the membership degree u_{ik} of each item k in cluster P_i is fixed and the prototype $\mathbf{g}_i = (g_i^1, \ldots, g_i^p)$ of class P_i $(i = 1, \ldots, c)$, which minimizes the clustering criterion $J1$, has the bounds of the interval $g_i^j = [\alpha_i^j, \beta_i^j]$ $(j = 1, \ldots, p)$ updated according to the following expression:

$$\alpha_i^j = \frac{\sum_{k=1}^n (u_{ik})^m a_k^j}{\sum_{k=1}^n (u_{ik})^m} \text{ and } \beta_i^j = \frac{\sum_{k=1}^n (u_{ik})^m b_k^j}{\sum_{k=1}^n (u_{ik})^m} \tag{3}$$

Allocation Step: Definition of the Best Fuzzy Partition. In the allocation step, each prototype $\mathbf{g}_i$ of class P_i $(i = 1, \ldots, c)$ is fixed and the membership degree u_{ik} $(k = 1, \ldots, n)$ of each item k in each cluster P_i, minimizing the clustering criterion $J1$ under $u_{ik} \geq 0$ and $\sum_{i=1}^c u_{ik} = 1$, is updated according to the following expression:

$$u_{ik} = \left[\sum_{h=1}^{c} \left(\frac{\sum_{j=1}^p \left[(a_k^j - \alpha_i^j)^2 + (b_k^j - \beta_i^j)^2 \right]}{\sum_{j=1}^p \left[(a_k^j - \alpha_h^j)^2 + (b_k^j - \beta_h^j)^2 \right]} \right)^{\frac{1}{m-1}} \right]^{-1} \tag{4}$$

2.2 Fuzzy *c*-Means Clustering Method for Symbolic Interval Data Based on a Single Adaptive Euclidean Distance

This section presents a fuzzy c-means clustering method for symbolic interval data based on a single adaptive Euclidean distance (labeled as SFCMAD). The main idea of these methods is that there is a distance to compare clusters and their representatives (prototypes) that changes at each iteration but that is the same for all clusters.

This adaptive method looks for a fuzzy partition of a set of items in c clusters $\{P_1, \ldots, P_c\}$, the corresponding c prototypes $\{\mathbf{g}_1, \ldots, \mathbf{g}_c\}$ and the square of a single adaptive Euclidean distance defined on vectors of intervals which is the same for all the clusters such that a criterion $J2$ measuring the fitting between the clusters and their representatives (prototypes) is locally minimized. This criterion $J2$ is based on a single adaptive Euclidean distance and it is defined as:

$$J2 = \sum_{i=1}^{c} \sum_{k=1}^{n} (u_{ik})^m \varphi(\mathbf{x}_k, \mathbf{g}_i) = \sum_{i=1}^{c} \sum_{k=1}^{n} (u_{ik})^m \sum_{j=1}^{p} \lambda^j \left[(a_k^j - \alpha_i^j)^2 + (b_k^j - \beta_i^j)^2 \right] \tag{5}$$

where $\mathbf{x}_k, \mathbf{g}_i$, u_{ik} and m are defined as before and

$$\varphi(\mathbf{x}_k, \mathbf{g}_i) = \sum_{j=1}^{p} \lambda^j \left[(a_k^j - \alpha_i^j)^2 + (b_k^j - \beta_i^j)^2 \right] \tag{6}$$

is a single adaptive Euclidean distance measuring the dissimilarity between an object $\mathbf{x}_k$ $(k = 1, \ldots, n)$ and a cluster prototype $\mathbf{g}_i (i = 1, \ldots, c)$, parameterized by the weight vector $\boldsymbol{\lambda} = (\lambda^1, \ldots, \lambda^p)$, which changes at each iteration but is the same for all clusters.

The algorithm starts from an initial membership degree for each pattern k in each cluster P_i and alternates a representation step and an allocation step until the convergence of the algorithm when the criterion $J2$ reaches a stationary value representing a local minimum. The representation step has now two stages.

Representation Step: Definition of the Best Prototypes. In the first stage, the membership degree u_{ik} of each pattern k on cluster P_i and and the weight vector $\boldsymbol{\lambda}$ are fixed.

Proposition 1. *The prototype $\boldsymbol{g}_i = (g_i^1, \ldots, g_i^p)$ of cluster $P_i\,(i = 1, \ldots, c)$, which minimizes the clustering criterion $J2$, has the bounds of the interval $g_i^j = [\alpha_i^j, \beta_i^j]\,(j = 1, \ldots, p)$ updated according to equation (3).*

Representation Step: Definition of the Best Distance. In the second stage, the membership degree u_{ik} of each pattern k in cluster P_i and the prototypes $\mathbf{g}_i$ of class $P_i\,(i = 1, \ldots, c)$ are fixed.

Proposition 2. *The vector of weights $\boldsymbol{\lambda} = (\lambda^1, \ldots, \lambda^p)$, which minimizes the clustering criterion $J2$ under $\lambda^j > 0\,(j = 1, \ldots, p)$ and $\prod_{j=1}^p \lambda^j = 1$, is updated according to the following expression:*

$$\lambda^j = \frac{\left\{\prod_{h=1}^p \left[\sum_{i=1}^c \sum_{k=1}^n (u_{ik})^m \left((a_k^h - \alpha_i^h)^2 + (b_k^h - \beta_i^h)^2\right)\right]\right\}^{\frac{1}{p}}}{\sum_{i=1}^c \sum_{k=1}^n (u_{ik})^m \left[(a_k^j - \alpha_i^j)^2 + (b_k^j - \beta_i^j)^2\right]} \tag{7}$$

Allocation Step: Definition of the Best Fuzzy Partition. In the allocation step, the prototypes $\mathbf{g}_i$ of class $P_i\,(i = 1, \ldots, c)$ and the vector of weights $\boldsymbol{\lambda} = (\lambda^1, \ldots, \lambda^p)$ $(i = 1, \ldots, c)$ are fixed.

Proposition 3. *The membership degree $u_{ik}\,(k = 1, \ldots, n)$ of each item k in each cluster P_i, minimizing the clustering criterion $J2$ under $u_{ik} \geq 0$ and $\sum_{i=1}^c u_{ik} = 1 (i = 1, \ldots, c)$, is updated according to the following expression:*

$$u_{ik} = \left[\sum_{h=1}^c \left(\frac{\sum_{j=1}^p \lambda^j \left[(a_k^j - \alpha_i^j)^2 + (b_k^j - \beta_i^j)^2\right]}{\sum_{j=1}^p \lambda^j \left[(a_k^j - \alpha_h^j)^2 + (b_k^j - \beta_h^j)^2\right]}\right)^{\frac{1}{m-1}}\right]^{-1} \tag{8}$$

2.3 Fuzzy c-Means Clustering Method for Symbolic Interval Data Based on an Adaptive Euclidean Distance for Each Cluster

Here we present a fuzzy c-means clustering method for symbolic interval data based on an adaptive Euclidean distance for each cluster (labeled as FCMADC).

This method has been introduced by [4]. The main idea of these methods is that there is a different distance associated to each cluster to compare clusters and their representatives (prototypes) that changes at each iteration, i.e., the distance is not determined once for all, furthermore is different from one cluster to another. Again, the advantage of these adaptive distances is that the clustering algorithm is able to find clusters of different shapes and sizes.

The FCMADC adaptive method looks for a fuzzy partition of a set of items in c clusters $\{P_1, \ldots, P_c\}$, the corresponding c prototypes $\{\mathbf{g}_1, \ldots, \mathbf{g}_c\}$ and the square of an adaptive Euclidean distance defined on vectors of intervals which is different for each class such that a criterion $J3$ measuring the fitting between the clusters and their representatives (prototypes) is locally minimized. This criterion $J3$ is based on an adaptive Euclidean distance for each cluster and it is defined as:

$$J3 = \sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^m \psi_i(\mathbf{x}_k, \mathbf{g}_i) = \sum_{i=1}^{c}\sum_{k=1}^{n}(u_{ik})^m \sum_{j=1}^{p}\lambda_i^j \left[(a_k^j - \alpha_i^j)^2 + (b_k^j - \beta_i^j)^2\right] \tag{9}$$

where $\mathbf{x}_k, \mathbf{g}_i$, u_{ik} and m are defined as before and

$$\psi_i(\mathbf{x}_k, \mathbf{g}_i) = \sum_{j=1}^{p}\lambda_i^j \left[(a_k^j - \alpha_i^j)^2 + (b_k^j - \beta_i^j)^2\right] \tag{10}$$

is now the square of an adaptive Euclidean distance measuring the dissimilarity between an object $\mathbf{x}_k\,(k = 1, \ldots, n)$ and a cluster prototype $\mathbf{g}_i(i = 1, \ldots, c)$, defined for each class and parameterized by the vectors of weights $\boldsymbol{\lambda}_i = (\lambda_i^1, \ldots, \lambda_i^p)\,(i = 1, \ldots, c)$, which change at each iteration.

The algorithm starts from an initial membership degree for each pattern k in each cluster P_i and alternates a representation step and an allocation step until the convergence of the algorithm when the criterion $J3$ reaches a stationary value representing a local minimum. The representation step has also two stages.

Representation Step: Definition of the Best Prototypes. In the first stage, the membership degree u_{ik} of each pattern k on cluster P_i and the vectors of weights $\boldsymbol{\lambda}_i = (\lambda_i^1, \ldots, \lambda_i^p)\,(i = 1, \ldots, c)$, are fixed and the prototype $\mathbf{g}_i = (g_i^1, \ldots, g_i^p)$ of class $P_i\,(i = 1, \ldots, c)$, which minimizes the clustering criterion $J3$, has the bounds of the interval $g_i^j = [\alpha_i^j, \beta_i^j]\,(j = 1, \ldots, p)$ updated according to equation (3).

Representation Step: Definition of the Best Distances. In the second stage, the membership degree u_{ik} of each pattern k in cluster P_i and the prototypes $\mathbf{g}_i$ of class $P_i\,(i = 1, \ldots, c)$ are fixed and the vectors of weights $\boldsymbol{\lambda}_i = (\lambda_i^1, \ldots, \lambda_i^p)\,(i = 1, \ldots, c)$, which minimizes the clustering criterion $J3$ under $\lambda_i^j > 0\,(j = 1, \ldots, p)$ and $\prod_{j=1}^{p}\lambda_i^j = 1$, is updated according to the following expression:

$$\lambda_i^j = \frac{\left\{\prod_{h=1}^{p}\left[\sum_{k=1}^{n}(u_{ik})^m \left((a_k^h - \alpha_i^h)^2 + (b_k^h - \beta_i^h)^2\right)\right]\right\}^{\frac{1}{p}}}{\sum_{k=1}^{n}(u_{ik})^m \left[(a_k^j - \alpha_i^j)^2 + (b_k^j - \beta_i^j)^2\right]} \tag{11}$$

Allocation Step: Definition of the Best Partition. In the allocation step, the prototypes $\mathbf{g}_i$ of class $P_i\,(i = 1,\ldots,c)$ and the vectors of weights $\boldsymbol{\lambda}_i = (\lambda_i^1,\ldots,\lambda_i^p)\,(i = 1,\ldots,c)$ are fixed and the membership degree $u_{ik}\,(k = 1,\ldots,n)$ of each item k in each cluster P_i, minimizing the clustering criterion $J3$ under $u_{ik} \geq 0$ and $\sum_{i=1}^{c} u_{ik} = 1(i = 1,\ldots,c)$, is updated according to the following expression:

$$u_{ik} = \left[\sum_{h=1}^{c}\left(\frac{\sum_{j=1}^{p}\lambda_i^j\left[(a_k^j-\alpha_i^j)^2+(b_k^j-\beta_i^j)^2\right]}{\sum_{j=1}^{p}\lambda_h^j\left[(a_k^j-\alpha_h^j)^2+(b_k^j-\beta_h^j)^2\right]}\right)^{\frac{1}{m-1}}\right]^{-1} \tag{12}$$

3 Experimental Results

To show the usefulness of these methods, experiments with synthetic symbolic interval data sets of different degrees of clustering difficulty (clusters of different shapes and sizes, linearly non-separable clusters, etc) and an application with a real data set are considered.

3.1 Synthetic Data Sets

In each experiment, we considered two standard quantitative data sets in $\Re^2$. Each data set has 450 points scattered among four classes of unequal sizes and ellipses shapes: two classes of size 150 each and two classes of sizes 50 and 100. Each class in these quantitative data sets were drawn according to a bi-variate normal distribution.

We consider two different configurations for the standard quantitative data sets: 1) data drawn according to a bi-variate normal distribution where the class covariance matrices are unequal and 2) data drawn according to a bi-variate normal distribution where the class covariance matrices are almost the same.

Each data point (z_1, z_2) of each one of these synthetic quantitative data sets is a seed of a vector of intervals (rectangle): $([z_1 - \gamma_1/2, z_1 + \gamma_1/2], [z_2 - \gamma_2/2, z_2 + \gamma_2/2])$. These parameters γ_1, γ_2 are randomly selected from the same predefined interval. The intervals considered in this paper are: $[1, 10], [1, 20], [1, 30]$ and $[1, 40]$.

Symbolic interval data set 1 were constructed from standard data drawn according to the following parameters (configuration 1):

a) Class 1: $\mu_1 = 28$, $\mu_2 = 23$, $\sigma_1^2 = 144$, $\sigma_2^2 = 16$ and $\rho_{12} = 0.8$;
b) Class 2: $\mu_1 = 62$, $\mu_2 = 30$, $\sigma_1^2 = 81$, $\sigma_2^2 = 49$ and $\rho_{12} = 0.7$;
c) Class 3: $\mu_1 = 50$, $\mu_2 = 15$, $\sigma_1^2 = 49$, $\sigma_2^2 = 81$ and $\rho_{12} = 0.6$;
d) Class 4: $\mu_1 = 57$, $\mu_2 = 48$, $\sigma_1^2 = 16$, $\sigma_2^2 = 144$ and $\rho_{12} = 0.9$;

Symbolic interval data set 2 were constructed from standard data drawn according to the following parameters (configuration 2):

a) Class 1: $\mu_1 = 28$, $\mu_2 = 23$, $\sigma_1^2 = 100$, $\sigma_2^2 = 9$ and $\rho_{12} = 0.7$;
b) Class 2: $\mu_1 = 62$, $\mu_2 = 30$, $\sigma_1^2 = 81$, $\sigma_2^2 = 16$ and $\rho_{12} = 0.8$;

c) Class 3: $\mu_1 = 50$, $\mu_2 = 15$, $\sigma_1^2 = 100$, $\sigma_2^2 = 16$ and $\rho_{12} = 0.7$;
d) Class 4: $\mu_1 = 57$, $\mu_2 = 37$, $\sigma_1^2 = 81$, $\sigma_2^2 = 9$ and $\rho_\rho 12 = 0.8$;

It is expected, for example, that the SFCMAD clustering method performs well if the data are drawn considering configuration 2.

The evaluation of these clustering methods was performed in the framework of a Monte Carlo experience: 100 replications are considered for each interval data set, as well as for each predefined interval. In each replication a clustering method is run (until the convergence to a stationary value of the adequacy criterion) 50 times and the best result, according to the corresponding criterion, is selected.

In order to compare the clustering results furnished by the adaptive and non-adaptive fuzzy clustering methods for interval data presented in this paper an external index, the corrected Rand index (CR), will be considered [9]. The CR index measures the similarity between an a priori hard partition and a hard partition obtained from the fuzzy partition furnished by the fuzzy clustering algorithm. A hard partition $P = \{P_1, \ldots, P_c\}$ is obtained from a fuzzy partition defining the class $P_i \, (i = 1, \ldots, c)$ as: $P_i = \{k \in \{1, \ldots, n\} : u_{ik} \geq u_{jk}, \forall j \in \{1, \ldots, c\}\}$. CR takes its values from the interval [-1,1], where the value 1 indicates perfect agreement between partitions, whereas values near 0 (or negatives) correspond to cluster agreement found by chance.

Table 1 shows the values of the average and standard deviation of CR index according to the different methods and data configurations.

Table 1. Comparison between the clustering methods for interval data sets 1 and 2

Range of values	Interval Data Set 1			Interval Data Set 2		
of γ_i $i = 1, 2$	FCMNAD	SFCMAD	FCMADC	FCMNAD	SFCMAD	FCMADC
$\gamma_i \in [1, 10]$	0.689 (0.110)	0.683 (0.108)	0.777 (0.108)	0.478 (0.099)	0.819 (0.136)	0.833 (0.117)
$\gamma_i \in [1, 20]$	0.687 (0.037)	0.693 (0.032)	0.755 (0.040)	0.455 (0.063)	0.771 (0.039)	0.772 (0.046)
$\gamma_i \in [1, 30]$	0.668 (0.033)	0.661 (0.033)	0.691 (0.046)	0.429 (0.056)	0.644 (0.049)	0.632 (0.058)
$\gamma_i \in [1, 40]$	0.620 (0.035)	0.623 (0.032)	628 (0.035)	0.395 (0.035)	0.548 (0.043)	0.505 (0.063)

As expected, in data configuration 1 (the class covariance matrices are unequal) the method based on an adaptive distance for each cluster (FCMADC) outperforms the method based on a single adaptive distance (SFCMAD). For this configuration, the method based on a non-adaptive distance (FCMNAD) presented a similar performance to SFCMAD method.

Data configuration 2 presents class covariance matrices that are almost the same. As expected, in this case the method based on a single adaptive distance (SFCMAD) outperform the method based on an adaptive distance for each cluster (FCMADC). The method based on a non-adaptive distance (FCMNAD) has the worst performance.

In conclusion, for these data configurations, the methods based on adaptive distances outperform the FCMNAD method. Concerning the adaptive methods, their performance depend on the intra-cluster structure: the method based on a single adaptive distance performs well when the *a priori* classes have similar dispersions whereas the method based on an adaptive distance for each cluster performs well when the *a priori* classes have dissimilar dispersions.

3.2 Applications with a Real Data Set

A data set with 33 car models described by 8 interval variables is used in this application. These car models are grouped in four *a priori* classes of unequal sizes: *Utilitarian* (size 10), *Berlina* (size 8), *Sporting* (size 7) and *Luxury* (size 8). The symbolic interval variables are: *Price, Engine Capacity, Top Speed, Acceleration, Step, Length, Width* and *Height.*

For this data set, each clustering method was run until the convergence to a stationary value of the adequacy criterion Jl $(l = 1, 2, 3)$ 60 times and the best result, according to the corresponding adequacy criterion, is selected. Moreover, m was set equal to 2.

FCMNAD, SFCMAD and FCMADC clustering algorithms have been applied to this data set. The 4-cluster hard partition obtained with these clustering methods were compared with the 4-cluster hard partition known *a priori.* The comparison index used is the corrected Rand index CR which is calculated for the best result. The CR indices were 0.254, 0.500 and 0.526, respectively, for these clustering methods. In conclusion, for this interval data set, FCMADC method presents the best performance whereas the FCMNAD method presents the worst one.

4 Conclusions

In this paper, a fuzzy c-means clustering method for symbolic interval data is introduced. This method furnish a partition of the input data and a corresponding prototype for each class by optimizing an adequacy criterion which is based on a single adaptive Euclidean distance between vectors of intervals.

The evaluation of this method in comparison with previous fuzzy c-means clustering methods having adequacy criterion based on (non-adaptive and adaptive for each cluster) Euclidean distances have been carried out. The accuracy of the results furnished by these clustering methods were assessed by the corrected Rand index considering synthetic interval data sets in the framework of a Monte Carlo experience and an application with a real data set. Concerning the average CR index for synthetic and real symbolic interval data sets, the methods with adaptive distances outperform the method with non-adaptive distance. Regarding the adaptive methods, their performance depend on the intra-cluster structure: the method based on a single adaptive distance performs well when the *a priori* classes have a similar dispersion whereas the method based on an adaptive distance for each cluster performs well when the *a priori* classes have a dissimilar dispersion.

Acknowledgments: The author would like to thank CNPq (Brazilian Agency) for its financial support.

References

1. J. C. Bezdek.: Pattern recognition with fuzzy objective function algorithms. New York: Plenum Press (1981)
2. Bock, H.H. and Diday, E.: Analysis of Symbolic Data: Exploratory Methods for Extracting Statistical Information from Complex Data. Springer, Berlin, (2000)
3. Chavent, M. and Lechevallier, Y.: Dynamical Clustering Algorithm of Interval Data: Optimization of an Adequacy Criterion Based on Hausdorff Distance. In: Sokolowsky and H.H. Bock Eds., K. Jaguja, A. (eds) Classification, Clustering and Data Analysis (IFCS2002). Springer, Berlin, (2002) 53–59
4. De Carvalho, F.A.T, and Cavalvanti Junior, N.L.: Fuzzy clustering algoritms for symbolic interval data based on L_2 Norm. Proceedings of the 2006 IEEE International Conference on Fuzzy Systems (to appear).
5. De Carvalho, F.A.T, Souza, R.M.C.R., Chavent, M. and Lechevallier, Y.: Adaptive Hausdorff distances and dynamic clustering of symbolic data. Pattern Recognition Letters, **27** (3) (2006) 167–179
6. Diday, E. and Simon, J.C.: Clustering analysis. In: K.S. Fu (ed) Digital Pattern Clasification. Springer, Berlin et al, (1976) 47–94
7. Diday, E. and Govaert, G.: Classification Automatique avec Distances Adaptatives. R.A.I.R.O. Informatique Computer Science, **11** (4) (1977) 329–349
8. El-Sonbaty Y. and Ismail, M.A.: Fuzzy Clustering for Symbolic Data IEEE Transactions on Fuzzy Systems, **6** (1998) 195–204.
9. Hubert, L. and Arabie, P.: Comparing Partitions. Journal of Classification, **2** (1985) 193–218
10. Jain, A.K., Murty, M.N. and Flynn, P.J.: Data Clustering: A Review. ACM Computing Surveys, **31**, (3) (1999), 264–323
11. Souza, R.M.C.R. and De Carvalho, F. A. T.: Clustering of interval data based on city-block distances. Pattern Recognition Letters, **25** (3) (2004) 353–365
12. Verde, R., De Carvalho, F.A.T. and Lechevallier, Y.: A dynamical clustering algorithm for symbolic data. In: Diday, E., Lechevallier, Y. (eds) Tutorial on Symbolic Data Analysis (Gfkl2001), (2001) 59–72

Approximation Accuracy of Table Look-Up Scheme for Fuzzy-Neural Networks with Bell Membership Function

Weimin Ma[1,2]

[1] School of Economics and Management
Beijing University of Aeronautics & Astronautics, Beijing, 100083, P.R. China
[2] School of Economics and Management, Xi'an Institute of Technology
Xi'an, Shaanxi Province, 710032, P.R. China
mawm@buaa.edu.cn

Abstract. Although many works have been done in recent years for the designing Fuzzy-Neural Networks (FNN) from input-output data, the results concerning how to analyze the performance of some methods from a rigorous mathematical point of view are somewhat few. In this paper, the approximation bound for the Table Look-up Scheme with the Bell Membership Function is established. The detailed formulas of the error bound between the nonlinear function to be approximated and the FNN system designed based on the input-output data are derived.

1 Introduction

FNN systems are hybrid systems that combine the theories of fuzzy logic and neural networks. Designing the FNN system based on the input-output data is a very important problem [1,2,3]. Some universal approximation capabilities for a broad range of neural network topologies have been established by T. P. Chen [4]. More results concerning the approximation of Neural Network can be found in [5,6,7].

In paper [2], an approach so called Table Lookup Scheme was introduced for training of Fuzzy Logic System. The paper [3] studied the relevant approximation accuracy of the table lookup scheme with Triangular Membership Function and Gaussian Membership Function. In this paper, taking advantage of some similar techniques of paper [3], we obtain the upper bound for the approximation by using the table lookup scheme with Bell Membership Function.

2 Table Lookup Scheme with Bell Membership Function

Before introducing the main results, we firstly introduce some basic knowledge on the designing FNN systems with table lookup scheme, which was proposed in [2] and also can be found in [3]. Given the input-out data pairs

$$(x_0^q, y_0^q), \quad q = 1, 2, \cdots \tag{1}$$

where $x_0^q \in U = [\alpha_1, \beta_1] \times \ldots \times [\alpha_n, \beta_n] \subset R^n$ and $y_0^q \in U_y = [\alpha_y, \beta_y] \subset R$. If the data are assumed to be generated by an unknown nonlinear function $y = f(x)$, the table lookup scheme in [2] can help us to design a FNN system to approximate the function $f(x)$. For convenience to illustrate the main result of this paper, we describe this method in a brief way as follows.

Step 1. To begin with define fuzzy sets to cover the input and output spaces. Define $N_i (i = 1, 2, \cdots, n)$ fuzzy sets $A_i^{j_i} (j_i = 1, 2, \cdots, N_i)$ on each $[\alpha_i, \beta_i]$ with the following bell membership functions:

$$\mu_{A_i^{j_i}}(x_i) = \mu_{A_i^{j_i}}(x_i; e_i^{j_i}) = \frac{1}{1 + \frac{(x_i - e_i^{j_i})^2}{\sigma^2}} \tag{2}$$

where $j_i = 1, 2, \cdots, N_i$ and $\alpha_i = e_i^1 < e_i^2 < \cdots < e_i^{N_i} = \beta_i$. Similarly, define N_y fuzzy sets $B^l (l = 1, 2, \cdots, N_y)$ on $[\alpha_i, \beta_i]$ with centers at $\alpha_y = e_y^1 < e_y^2 < \cdots < e_i^{N_y} = \beta_y$.

Step 2. Generate one fuzzy IF-THEN rule from one input-output pair. For each input variable $x_i (i = 1, 2, \cdots, n)$, determine the fuzzy set $A_i^{j_i^p}$ such that

$$\mu_{A_i^{j_i^p}}(x_{0i}^p) \geq \mu_{A_i^{j_i}}(x_{0i}^p), \qquad j_i = 1, 2, \cdots, N_i. \tag{3}$$

Similarly, determine $\mu_{B^{l_p}}$ such that $\mu_{B^{l_p}}(y_0^p) \geq \mu_{B^l}(y_0^p)$ for $l = 1, 2, \cdots, N_y$. Finally, obtain a fuzzy IF-THEN rule as

$$Ru^{(P)}: \text{ IF } x_1 \text{ is } A_1^{j_1^p} \text{and} \cdots \text{and } x_n \text{ is } A_n^{j_n^p} \text{ THEN } y \text{ is } B^{lp}. \tag{4}$$

Step 3. Assigned a degree to each rule generated in Step 2. If there exist conflicting rules, i.e., rules with the same IF parts but different THEN parts, then assign a degree to each generated rule in Step 2 as follows:

$$D(Ru^{(P)}) = \prod_{i=1}^{n} \mu_{A_i^{j_i^p}}(x_i^p) \cdot \mu_{B^{l_p}}(y^p) \tag{5}$$

Keep only one rule from a conflicting group that has the maximum degree.

Step 4. Create the fuzzy rule base as follows:

$$Ru^{(j)}: \text{ IF } x_1 \text{ is } A_1^{j_1} \text{and} \cdots \text{and } x_n \text{ is } A_n^{j_n} \text{ THEN } y \text{ is } B^{(j)}. \tag{6}$$

where $j = (j_1, \cdots, j_n) \in J$ and J is the index set of the fuzzy rule base. For convenience of discussing of the following text, let $\|j \in J\| = N$.

Step 5. Construct the fuzzy system as

$$\hat{f}(x) = \frac{\sum\limits_{j \in J} \left[\overline{y}^{(j)} \cdot \left(\frac{1}{1 + \frac{(x - e^j)^2}{\sigma^2}} \right) \right]}{\sum\limits_{j \in J} \left[\frac{1}{1 + \frac{(x - e^j)^2}{\sigma^2}} \right]} = \frac{\sum\limits_{j \in J} \left[\overline{y}^{(j)} \cdot \prod\limits_{i=1}^{n} \left(\frac{1}{1 + \frac{(x_i - e_i^{j_i})^2}{\sigma^2}} \right) \right]}{\sum\limits_{j \in J} \prod\limits_{i=1}^{n} \left[\frac{1}{1 + \frac{(x_i - e_i^{j_i})^2}{\sigma^2}} \right]} \tag{7}$$

where $\overline{y}^{(j)}$ is the center of $B^{(j)}$.

The above FNN system is constructed by using the membership functions that achieve the lagest values at the sampling points of the input-output pairs, as detailed in [2]. This paper concentrates on the approximation bound for this method with bell membership function.

3 Approximation Bound with Bell Membership Function

The following theorem gives the approximation bound of FNN system $\hat{f}(x)$ of (7) which is constructed by using the clustering method with bell membership function.

Theorem 1. *Let $f(x)$ be a continuous function on U that generates the input-output pairs in (1). Then the following approximation property holds for the FNN system $\hat{f}(x)$ of (7):*

$$|f(x) - \hat{f}(x)| \leq \frac{h_y}{2} + \sum_{i=1}^{n} \left\{ \left\| \frac{\partial f}{\partial x_i} \right\|_\infty \cdot \left[\frac{h_i}{2} + d_x + 2^{n-1} \cdot N \cdot \left(\sigma + \frac{d_x^2}{\sigma} \right) \right] \right\} \tag{8}$$

where the infinite norm $\|\cdot\|_\infty$ is defined as $\|d(x)\|_\infty = \sup_{x \in U} |d(x)|$ and,

$$\begin{aligned} h_i &= \max_{1 \leq j_i \leq N_i - 1} |e_i^{j_i+1} - e_i^{j_i}| \\ h_y &= \max_{1 \leq l \leq N_y - 1} |e_i^{l+1} - e_i^{l}| \\ d_x &= \min_j |x - e^j| \end{aligned} \tag{9}$$

Proof. From (7) we have

$$\begin{aligned} |f(x) - \hat{f}(x)| &\leq \frac{\sum\limits_{j \in J} \left[|f(x) - \overline{y}^{(j)}| \cdot \left(\frac{1}{1 + \frac{(x - e^j)^2}{\sigma^2}} \right) \right]}{\sum\limits_{j \in J} \left[\frac{1}{1 + \frac{(x - e^j)^2}{\sigma^2}} \right]} \\ &= \frac{\sum\limits_{j \in J} \left[|f(x) - \overline{y}^{(j)}| \cdot \prod\limits_{i=1}^{n} \left(\frac{1}{1 + \frac{(x_i - e_i^{j_i})^2}{\sigma^2}} \right) \right]}{\sum\limits_{j \in J} \prod\limits_{i=1}^{n} \left[\frac{1}{1 + \frac{(x_i - e_i^{j_i})^2}{\sigma^2}} \right]} \end{aligned} \tag{10}$$

Paper [3] obtain the following result when the relevant approximation bound for the clustering method with triangular membership function was discussed:

$$|f(x) - \overline{y}^{(j)}| \leq \sum_{i=1}^{n} \left\| \frac{\partial f}{\partial x_i} \right\|_\infty \cdot \left(|x_i - e_i^{j_i}| + \frac{h_i}{2} \right) + \frac{h_y}{2} \tag{11}$$

Combining the (10) and (11), we have

$$|f(x)-\hat{f}(x)| \leq \frac{\sum\limits_{j\in J}\left\{\left[\sum\limits_{i=1}^{n}\left\|\frac{\partial f}{\partial x_i}\right\|_{\infty}\cdot\left(|x_i-e_i^{j_i}|+\frac{h_i}{2}\right)+\frac{h_y}{2}\right]\cdot\prod\limits_{i=1}^{n}\left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right]\right\}}{\sum\limits_{j\in J}\prod\limits_{i=1}^{n}\left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right]}$$
$$\leq \frac{h_y}{2}+\sum_{i=1}^{n}\left\{\left\|\frac{\partial f}{\partial x_i}\right\|_{\infty}\left[\frac{h_i}{2}+\frac{\sum\limits_{j\in J}|x_i-e_i^{j_i}|\cdot\prod\limits_{i=1}^{n}\left(\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right)}{\sum\limits_{j\in J}\prod\limits_{i=1}^{n}\left(\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right)}\right]\right\} \tag{12}$$

Now, we just focus on analyzing the term

$$\frac{\sum\limits_{j\in J}|x_i-e_i^{j_i}|\cdot\prod\limits_{i=1}^{n}\left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right]}{\sum\limits_{j\in J}\prod\limits_{i=1}^{n}\left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right]}$$

on the right-hand side of (12). We only consider the case $i = 1$ and the proof remains the same for $i = 1, 2, \ldots, n$.

Employing the similar method in [3], given any point $x = (x_1, x_2, \ldots, x_n) \in U$, we divide space U in to 2^n areas and define some sets concerning the table lookup center as follows:

$$\begin{aligned}
U_1^x &= \{\overline{x} \in U : \overline{x}_1 - x_1 \geq 0, \ldots, \overline{x}_n - x_n \geq 0\}\\
U_2^x &= \{\overline{x} \in U : \overline{x}_1 - x_1 \geq 0, \ldots, \overline{x}_n - x_n < 0\}\\
&\cdots\\
U_{2^n-1}^x &= \{\overline{x} \in U : \overline{x}_1 - x_1 < 0, \ldots, \overline{x}_n - x_n \geq 0\}\\
U_{2^n}^x &= \{\overline{x} \in U : \overline{x}_1 - x_1 < 0, \ldots, \overline{x}_n - x_n < 0\}
\end{aligned} \tag{13}$$

And define some sets concerning the cluster centers

$$\begin{aligned}
V^x &= \{\overline{x} \in U : |\overline{x}_1 - x_1| < d_x\},\\
\overline{V}^x &= \{\overline{x} \in U : |\overline{x}_1 - x_1| \geq d_x\},\\
V_m^x &= \overline{V}^x \cap U_m^x, \quad (m = 1, \ldots, 2^n).
\end{aligned} \tag{14}$$

Apparently, there are two cases need to consider.

Case 1: $x_c^l \in V^x$, which indicates $|x_{c,1}^l - x_1| < d_x$, we have

$$\frac{\sum\limits_{j\in J} |x_i - e_i^{j_i}| \cdot \prod\limits_{i=1}^{n} \left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right]}{\sum\limits_{j\in J} \prod\limits_{i=1}^{n} \left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right]} \le \frac{d^x \cdot \sum\limits_{j\in J} \prod\limits_{i=1}^{n} \left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right]}{\sum\limits_{j\in J} \prod\limits_{i=1}^{n} \left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right]} \le d^x \tag{15}$$

Case 2: $e^j \in \overline{V}^x = \bigcup_{m=1}^{2^n} V_m^x$. We only consider the case $e^j \in V_1^x$. For the cases $e^j \in V_2^x, \dots, V_{2^n}^x$, the same result can be obtained. From (12), (13) and (14), we have

$$\begin{aligned}
&\sum_{e^j \in V_1^x} \left\{ |x_1 - e_1^{j_1}| \cdot \prod_{i=1}^{n} \left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right]\right\} \\
&\quad = \sum_{e^j \in V_1^x} \left\{ |x_1 - e_1^{j_1}| \cdot \left[\frac{1}{1+\frac{(x_1-e_1^{j_1})^2}{\sigma^2}}\right] \cdot \prod_{i=2}^{n} \left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right]\right\} \\
&\quad \le \sum_{e^j \in V_1^x} \left[\frac{\sigma}{2} \cdot \prod_{j=2}^{n} \left(\frac{1}{1+\frac{|0|^2}{\sigma^2}}\right)\right] \\
&\quad = \frac{\sigma}{2} \cdot \sum_{e^j \in V_1^x} 1 \\
&\quad \le \frac{N \cdot \sigma}{2}
\end{aligned} \tag{16}$$

The second inequality of (16) holds for the following reason

$$\begin{aligned}
|x_1 - e_1^{j_1}| \cdot \left[\frac{1}{1+\frac{(x_1-e_1^{j_1})^2}{\sigma^2}}\right] &= \frac{|x_1 - e_1^{j_1}|}{1+\frac{(x_1-e_1^{j_1})^2}{\sigma^2}} \\
\le \frac{1}{\frac{1}{|x_1-e_1^{j_1}|} + \frac{|x_1-e_1^{j_1}|}{\sigma^2}} &\le \frac{1}{2\sqrt{\frac{1}{|x_1-e_1^{j_1}|}} \cdot \sqrt{\frac{|x_1-e_1^{j_1}|}{\sigma^2}}} = \frac{\sigma}{2}
\end{aligned} \tag{17}$$

Considering cases $e^j \in V_2^x, \dots, V_{2^n}^x$, we have

$$\sum_{e^j \in \overline{V}^x} \left\{ |x_1 - e_1^{j_1}| \cdot \prod_{i=1}^{n} \left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right]\right\} \le \frac{2^n \cdot N \cdot \sigma}{2} = 2^{n-1} \cdot N \cdot \sigma \tag{18}$$

On the other hand, it follows from (9) that

$$\sum_{j\in J} \prod_{i=1}^{n} \left[\frac{1}{1+\frac{(x_i-e_i^{j_i})^2}{\sigma^2}}\right] = \sum_{j\in J} \left[\frac{1}{1+\frac{(x-e^j)^2}{\sigma^2}}\right] \ge \frac{1}{1+\frac{(d^x)^2}{\sigma^2}} \tag{19}$$

From (18) and (19), we have

$$\frac{\sum_{e^j \in \overline{V}^x} \left\{ |x_1 - e_1^{j_1}| \cdot \prod_{i=1}^{n} \left[\frac{1}{1+\frac{(x_i - e_i^{j_i})^2}{\sigma^2}} \right] \right\}}{\sum_{j \in J} \prod_{i=1}^{n} \left[\frac{1}{1+\frac{(x_i - e_i^{j_i})^2}{\sigma^2}} \right]} \leq 2^{n-1} \cdot N \cdot (\sigma + \frac{d_x^2}{\sigma}) \tag{20}$$

Combining (14), (15) and (20), it can be shown that

$$\frac{\sum_{j \in J} |x_i - e_i^{j_i}| \cdot \prod_{i=1}^{n} \left[\frac{1}{1+\frac{(x_i - e_i^{j_i})^2}{\sigma^2}} \right]}{\sum_{j \in J} \prod_{i=1}^{n} \left[\frac{1}{1+\frac{(x_i - e_i^{j_i})^2}{\sigma^2}} \right]} \leq d_x + 2^{n-1} \cdot N \cdot (\sigma + \frac{d_x^2}{\sigma}) \tag{21}$$

From (12) and (21), we get the desired result. □

4 Concluding Remarks

In this paper, an upper bound, $\frac{h_y}{2} + \sum_{i=1}^{n} \left\{ \left\| \frac{\partial f}{\partial x_i} \right\|_\infty \left[\frac{h_i}{2} + d_x + 2^{n-1} N \left(\sigma + \frac{d_x^2}{\sigma} \right) \right] \right\}$, concerning the approximation bound of table lookup scheme with Bell Membership Function is proved in a rigorous mathematical way. The techniques employed in the proof of the theorem are expected to be used to obtain or improve other approximation bound of other methods of FNN.

Acknowledgements. The work was supported by the National Natural Science Foundation of China (70401006, 70521001, 70501015).

References

1. Wang, L.X., Mendel, J.M., W.: Fuzzy Basis Functions, Universal Approximation, and orthogonal least squares learning. IEEE Trans. Neural Network. **3** (1992) 807–814
2. Wang, L.X.: Generating Fuzzy Rules by Learning from Examples. IEEE Trans Syst., Man, Cybern.,**22** (1992) 1414–1427
3. Wang, L.X. and Chen, W.: Approximation Accuracy of Some Neuro-Fuzzy Approches. IEEE Trans. Fuzzy Systems. **8** (2000) 470–478
4. Chen, T.P., Chen, H.: Approximation Capability to Functions of Several Variables, Nonlinear Functions, and Operators by Radial Function Neural Networks. IEEE Trans. Neural Networks. **6** (1995) 904–910
5. Maiorov, V., Meir, R.S.: Approximation Bounds for Smooth Functions in $C(R^d)$ by Neural and Mixture Networks. IEEE Trans. Neural Networks. **9** (1998) 969–978
6. Burger, M., Neubauer, A.: Error Bounds for Approximation with Neurnal Networks. J. Approx. Theory. **112** (2001) 235–250
7. Ma, W.M., Chen, G.Q.: Improvement on the Approximation Bound for Fuzzy-Neural Networks Clustering Method with Gaussian Membership Function, Proceedings of the First International Conference on Advanced Data Mining and Applications (ADMA2005). LNAI **3584**(2005), 225–231

Prototype-Based Threshold Rules

Marcin Blachnik[1] and Włodzisław Duch[2,3]

[1] Division of Computer Methods, Department of Electrotechnology and Metallurgy, The Silesian University of Technology, Krasińskiego 8, 40-019 Katowice, Poland
marcin.blachnik@polsl.pl
[2] Department of Informatics, Nicolaus Copernicus University, Grudziądzka 5, Toruń, Poland
[3] School of Computer Engineering, Nanyang Technological University, Singapore
Google: Duch

Abstract. Understanding data is usually done extracting fuzzy or crisp logical rules using neurofuzzy systems, decision trees and other approaches. Prototype-based rules are an interesting alternative providing in many cases simpler, more accurate and more comprehensible description of the data. Algorithm for generation of threshold prototype-based rules are described and a comparison with neurofuzzy systems on a number of datasets provided. Results show that systems for data understanding generating prototypes deserve at least the same attention as that enjoyed by the neurofuzzy systems.

1 Introduction

Data mining and knowledge discovery algorithms are focused on understanding of data structures, still one of most important challenges facing computational intelligence. Data understanding requires extraction of crisp or fuzzy logical rules. For some datasets surprisingly accurate and simple logical rules may be generated [1], but in some cases sets of logical rules may be too large or too complicated to be useful. Crisp rules partition the input space into hyperboxes and thus even relatively simple tasks that require oblique decision borders may lead to complicated sets of rules. All major data mining software suits use as their important component decision trees for crisp logical rule extraction. C4.5 [3] and CART [4] are the most popular algorithms used in many packages, but there are many others, for example SSV trees [5] used in the Ghostminer package [6]. Inductive machine learning algorithms are rarely used in data mining systems. Fuzzy rules (F-rules) are more flexible and have been used in many successful real word applications reported in literature [7]. Unfortunately this group of methods is also limited; usually they are less transparent then crisp logical rules (C-rules) and may be difficult to understand, therefore they are rarely found in data mining software suits. F-rules work well with continuous numerical attributes but the real word applications often require analysis of mixed data, including symbolic or nominal attributes, which are not supported directly by fuzzy rules.

An alternative approach to data understanding, based on similarity [8] rather than logic, extracts rules based on prototypes (P-rules). People making decisions rarely use logic, but most often use their memory to recall similar cases and outcomes of previous decisions. In similarity-based learning framework (SBL) two major types of P-rules have been defined [9]: nearest neighbor rules, where classification decisions are based on rules assigning query vectors to the same class that majority of the closest prototypes belong to, and the threshold rules, where each prototype has an associated distance threshold which defines subspace around the prototype with associated class label.

One way to define prototypes threshold rules is by using heterogeneous decision trees (HDT) [10], a classical decision tree algorithm that is extended to use new attributes based on distances between training vectors. This algorithm has found some of the most accurate and simplest descriptions of several datasets. Another approach to threshold rules is based on a Prototype Threshold Decision List (PTDL), where linear list of ordered rules is created. In this paper the PTDL algorithm is introduced and compared with HDT. The next section describes how P-rules support different types of attributes, the third section presents threshold rules decision list algorithm, in the fourth section results of numerical experiments are presented, and the last section contains conclusions and discussion.

2 Heterogeneous Distance Functions

Real word datasets often contain different types of features, creating serious difficulties for large group of computational intelligence algorithms, including most methods based on fuzzy rules [7][11][13]. P-rules solve the problem of combination of continuous, discrete, symbolic and nominal attributes using heterogeneous distance function (HDF).

HDFs introduced by Wilson and Martinez [14] allow for calculation of distance for all types of attributes. Several types of HDF have been introduced, based on an assumption that distances are additive:

$$D(\mathbf{x},\mathbf{r})^{\alpha} = \sum_{i=1}^{n} d(x_i, r_i)^{\alpha} \tag{1}$$

where $D(\mathbf{x},\mathbf{r})$ is the distance between two vectors and $d(x_i,r_i)$ is the distance calculated for a single dimension. In the SBL framework [8] HDF allow for treating different types of features in a natural way. For real-valued or ordered discrete features Minkovski's distances (2) are used and for symbolic features probabilistic distance functions (3) are used, for example:

$$D_{Mink}(\mathbf{x},\mathbf{r})^{\alpha} = \sum_{i=1}^{n} |x_i - r_i|^{\alpha} \tag{2}$$

$$D_{VMD}(\mathbf{x},\mathbf{r})^{\alpha} = \sum_{i=1}^{n}\sum_{j=1}^{C} \left|p(c_j \mid x_i) - p(c_j \mid r_i)\right|^{\alpha} \tag{3}$$

where **x** and **r** are respectively data and reference vectors, n is the number of features, C is the number of classes, and α is the value of exponent (α=2 for Euclidean functions). $p(c_j \mid x_i)$ and $p(c_j \mid r_i)$ are calculated as

$$p\left(c_j|x_i\right) = \frac{Nx_{ij}}{Nx_i} \tag{4}$$

where Nx_i is the number of instances in the training set that have value x for attribute i, and Nx_{ij} is the some as Nx_i but for class j.

This types of distance functions are additive, so the overall distance function can be calculated as a sum of contributions from both types of distance measures, depending on the attribute types (4):

$$D(\mathbf{x},\mathbf{r})^{\alpha} = D_{Mink}(\mathbf{x_a},\mathbf{r_a})^{\alpha} + D_{VDM}(\mathbf{x_b},\mathbf{r_b})^{\alpha} \tag{5}$$

where $\mathbf{x_a}$ and $\mathbf{r_a}$ are subsets of continuous attributes of vectors $\mathbf{x}$ and $\mathbf{r}$, and $\mathbf{x_b}$ and $\mathbf{r_b}$ are subsets of their symbolic features. Features should be normalized to assure that each distance component has the some or comparable influence.

In P-rules α parameter have significant influence on the shape of decision borders. Changing α value from 1 to ∞ different shapes of hypersurfaces of constant value are obtained. For α equal 1 rhomboidal shape is obtained, for $\alpha=2$ spherical, higher α values lead to rectangular shapes, and for $\alpha=\infty$ lines of constant distance reach a square shape. This aspect of P-rules can allow for smooth transition to crisp logical rules if it is necessary. Also fuzzy rules can be extracted from datasets in this way, as discussed in [15].

3 Threshold Rules

P-rules based on distances from prototypes create tessellation of the input space, with most distance functions leading to convex polytope cells with hyperplane faces. Some cells are infinite, and the use of only two prototypes $\mathbf{r}_1$, $\mathbf{r}_2$ is equivalent to the linear discrimination, defining single hyperplane perpendicular to the line that joins them. Threshold based rules are not based on competition for the closest prototype, but simply assign all vectors $\mathbf{x}$ with $D(\mathbf{x},\mathbf{r})<\theta$ to the same class as the prototype $\mathbf{r}$. The effect is somehow similar to the use of basis expansion networks with localized functions, such as RBFs with Gaussian functions. However, the emphasis here is not on approximation but on data understanding, generation of a small number of simple rules with distance functions based only on relevant features.

A constructive algorithm is recommended, creating first quite general P-rules, and then more detailed rules and possible exceptions until the whole input space is covered. Two strategies to solve the problem of adding new rules developed so far are based on:

– Heterogeneous Decision Trees;
– ordered prototype threshold decision list.

3.1 Heterogeneous Decision Trees

Standard decision trees, such as the C4.5 [3], CART [4] or SSV trees [5], use only one type of test to split the data, $T(x_i < \theta)$, dividing the range of feature values x_i into two or more branches. Heterogeneous decision trees (HDT), introduced in [10] use at least two qualitatively different types of tests. Adding the second test $\mathrm{T}(D(\mathbf{x},\mathbf{r}_k) < \theta_k)$ based on similarity to prototypes provides localized decision borders to the hyperplanes contributed by the standard tests. In the simplest case all training vectors are initially taken as prototypes, using the square $[N \times N]$ distance matrix $D(\mathbf{r}_i,\mathbf{r}_j)$, where N is the number of input vectors. Then prototype vectors are consecutively removed and accuracy checked, until a small number of prototypes is left and accuracy starts to degrade. Similarities may be calculated either using Euclidean distances or Gaussian kernel functions.

Combination of hyperplanes obtained from binary splits of features with spherical decision borders from distance based threshold tests is quite powerful and may lead to interesting rules, although the search for the prototype by elimination of the training vectors is a rather costly procedure, with complexity of $O(N^2)$. This approach applied to the Wisconsin Breast Cancer data generated a single distance based P-rule with 97.3% accuracy, providing the simplest and most accurate description of this data found so far [10].

3.2 Prototype Threshold Decision List Algorithm

Heterogeneous classification trees used for extraction of prototype threshold rules create hierarchical sets of rules. An alternative is given by a covering algorithm that creates ordered list of rules that may overlap, called here Prototype Threshold Decision List (PTDL). This algorithm is based on similar criteria like HDT algorithms, however individual rules are stored in an ordered list, starting from the most general rule to the most detailed. Because they are overlapping this list of rules should be applied in an order, beginning from the most specific (and least reliable), and if its conditions are not fulfilled the next more general rule should be checked. In the end if none of the rules may be applied, the output label is assigned to the *else* condition, covering all the remaining vectors (Fig. 1).

PTDL searches among all training vectors for a prototype that maximizes appropriate decision tree criterion, like separability (SSV), the Gini index (CART) or Information Gain (C4.5). Each prototype and threshold define particular rule, splitting the data into vectors in the subspace covered by this prototype with selected threshold (vectors that fall inside of the rule borders), and the remaining vectors that fall outside of this subspace (outside of the rule). For multiclass problems these two types of prototype threshold rules should be explicitly distinguished: inside rules with $D(\mathbf{x},\mathbf{r}_k) < \theta_k$, and outside rules with $D(\mathbf{x},\mathbf{r}_k) \geq \theta_k$, where each rule is defined for one particular class. In the first case prototype $\mathbf{r_k}$ belong to the subspace, and in the second case it does not belong to the subspace defined by the prototype and threshold. For two class problems distinguishing between these two types of rules is not important, however for the multiclass problems it has significant meaning, increasing generalization and model simplicity.

The sketch of the PTDL code is presented in Fig. 2, where for simplicity two class problem is described. In the first step `CreateList` function calculates distances or similarities between all training vectors, storing them in the square matrix `D` of size

NxN where N is the number of training vectors. Then search for all possible splits of each training vector that may increase criterion value is performed (for loop). Only splits between pairs of neighboring vectors in each column of matrix D belonging to different classes are considered, because only such situation guarantees maximization of the criterion function. The middle points between these pairs of vectors are taken as thresholds. All parameters: the criterion value (C_Crit), threshold (C_Threshol) , and rule consequence – class labels are calculated by the function CalcCriterion, which returns column vectors with appropriate parameter values for currently analyzed i-th training vector .

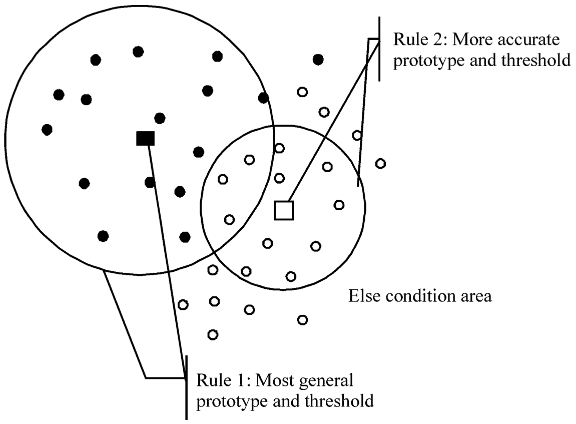

Rule 2: More accurate prototype and threshold
Else condition area
Rule 1: Most general prototype and threshold

Fig. 1. Example of threshold rules: Rule 1 – most general; Rule 2 – more accurate; and Else area in the remaining subspace

The best among *N* training vectors with appropriate threshold maximizing particular criterion function is stored in the rule set list. When a new rule is accepted all training vectors are classified with the current set of rules to mark all vectors that are incorrectly classified and should be used to search for further rules. The PTDL algorithm stops if the maximum number of rules is reached, or when all vectors are correctly classified.

This straightforward covering algorithm does not assure good generalization. To remedy its weakness optimal number of rules is found using internal crossvalidation on the training data (as it is done in the SSV trees [5][10]). Using *k*-fold crossvalidation test for each fold a new decision list is created. In the end at each level of the list appropriate criterion is checked (Gini has been used here, but information gain, balanced accuracy, separability or other criteria may be optimized), and the optimal number of rules that maximize the desired criterion is selected. Rule extraction algorithms frequently generate quite different sets of rules, suffering from high variance of solutions. To avoid such situation the difference between accuracy and standard deviation in crossvalidation calculations is optimized, selecting highly accurate low variance solutions.

```
function [P,PLab,TH] = CreateList(T,TLab,MaxRules)
1. input:
       T – training set
       TLab – labels of training vectors
       MaxRules – maximum number of rules

2. output:
       P – set of prototypes
       PLab–set of labels associated with each prototype P
       TH – set of thresholds for appropriate prototypes

3. var
       N – Number of training vectors
       D – distance matrix NxN
       RulN – number of created rules
       splits - list of possible splits where evaluation
       of the criterion is calculated
       C_Crit - vector of criterion values calculated for
                each split
       C_Threshold – appropriate threshold for each split
       C_PLab – class label for current split
       CurLab – Class labels predicted by set of rules,
               initially all vectors are wrong classified
       MXcrit – maximum criterion value for i-th prototype
       idx – index of best split

begin
4. D = dist(T,T);// distances between training vectors;
5. RulN=1;
6. while (RulN < MaxRuls) or ErrorsN == 0
7.      for I = 1:N  // considered each training vector
                      as prototype
8.      splits = FindPossibleSplits(D,Lab); //find all
                                     possible thresholds
9.     [C_Crit,C_Threshold,C_PLab]=
       CalcCriterion(Dat,LabT,splits,CurLab); //For each
       threshold calculate criterion value
10.    [MXcrit,idx]=max(C_Crit);//Find max. criterion value
11.       if MXcrit > bestCrit
12.          bestCrit = MXcrit;
13.          P(RulN) = T(i);
14.          TH(RulN) = C_Threshold(idx);
15.          PLab = C_PLab(idx);
16.          RulN = RulN+1;
17.       end;
18.     end
19.    CurLab = ApplyRules(D,Lab,P,PLab,TH);
20.  endwhile;
21. end;
```

Fig. 2. The PTDL algorithm code

4 Experiments with Real Datasets

To compare results obtained with PTDL, HDT and other well established methods WEKA software was used with two popular rule extraction methods: C4.5 decision tree and the Decision Table algorithm, as implemented in WEKA [19]. Results obtained with the neurofuzzy rule extraction system NefClass [12][17] are also given for comparison. The NefClass calculations were carefully optimized changing the number and the type of membership functions to obtain the best solution (the difference between accuracy and standard deviation).

4.1 Datasets

For tests six different datasets were used, all from the UCI machine learning database repository [16], except for Lancet data obtained from the authors of [16]. Each data represents a two class problem with mixed type of attributes. A summary of these datasets follows:

Appendicitis – small dataset with 7 attributes and 106 cases, 85 from the first class and 21 from the second class. From this dataset 2 most relevant features were selected using SSV tree and all tests were performed for these two features.

Cleveland Heart Disease (Cleveland) – 5 continuous attributes and 8 discrete, 303 vectors describing healthy and sick persons; 6 cases with missing values were removed, leaving 297 vectors.

Ionosphere – two different types of radar signals reflected from ionosphere; 351 vectors with 34 attributes.

Lancet dataset – 692 breast cancer cases, 235 malignant, 457 benign, characterized by age plus 10 binary indicators obtained from fine-needle aspirates of breast lumps, with the final diagnosis confirmed by biopsy.

Pima Indians Diabetes (Diabetes) – 768 cases describing results of tests for diabetes, with 500 healthy and 268 cases sick people, 8 features.

Wisconsin Breast Cancer (Wisconsin) – well known breast cancer data from a Wisconsin hospital, with 241 cases of malignant, and 458 of benign tumors, each case described by 9 discrete features.

4.2 Classification Results

10-fold stratified crossvalidation calculations on each dataset were performed using 5 algorithms that generate crisp and fuzzy rules, providing estimates of their generalization. Mean accuracy obtained on test partitions is presented in Table 1. The best results obtained for each dataset are marked as bold.

From Table 1 it is evident that the accuracy of the PTDL algorithm is almost always among the best among algorithms tested, creating a small number of rules and achieving in most cases best results. The Appendicitis dataset is very small and although NefClass has produced slightly better result it has used much larger number of fuzzy rules. For the Cleveland dataset only three P-rules were created by PTDL, reaching significantly higher accuracy than other systems. In the diabetes case all rule-based results are relatively poor, while MLP or SVM results on this dataset reach 77.5±2.5%, close to the simple linear discrimination analysis (LDA) reported in [20]. Therefore a single P-rule based on the shortest distance to two prototypes is sufficient in this case instead of the threshold based rules. PTDL did surprisingly well on the Ionosphere data, but HDT has an advantage here, achieving almost the same accuracy with only 3 rules. Insignificant differences are found on the Lancet data, with an exception of C4.5 rules that are less accurate, with PTDL using just 3 P-rules and NefClass 85 F-rules. Also on the Wisconsin dataset only two P-rules were used to reach the highest accuracy with the lowest standard deviation.

Table 1. Classification results, accuracy (Acc) and standard deviation (Std) in %, the number of rules estimates the complexity of the model

	C4.5			Decision Table			NefClass			PTDL (Gini)			HDT (Gini)		
10 x CV	Acc	Std	Rules	Acc	Std	Rules	Acc	Std	Rules	Acc	Std	Rules	Acc	Std	Rules
Appendicitis	85.82	8,51	3	82,00	11,65	2	**87.73**	**8.6**	**33**	85.77	8.6	5	83.78	9,0	3
Cleveland	76.77	7,17	17	82.09	9,14	8	82.82	6.8	6	**84.21**	**5.1**	**3**	80.83	6,1	5
Diabetes	74,48	4,42	20	**74,87**	**5,16**	**32**	73.83	2,3	5	70.43	3.5	8	71.74	4.1	2
Ionosphere	**94.94**	**2.5**	**9**	93,06	3,66	23	72,67	6.7	9	93.45	3.1	15	93.15	2,9	3
Lancet	92,29	4,62	18	90,33	4,42	22	94.51	2.6	85	93,94	2,5	3	**94,51**	**2,1**	**4**
Wisconsin	94,58	2,87	11	95.75	1,65	20	94.86	2,6	6	**97.66**	**1.4**	**2**	96.93	1.85	1

5 Conclusions and Future Works

The prototype threshold decision list (PTDL) rule extraction algorithm presented in this paper is a simple method that creates a small number of accurate P-rules. Results obtained on several benchmark datasets are quite encouraging, even though only one criterion (Gini) has been considered so far and the simplest heterogeneous distance functions have been used. In a few cases these results are significantly better comparing to crisp rules obtained with C4.5 decision trees or decision table, or F-rules generated by the NefClass, a leading neurofuzzy algorithm. In some cases P-rules based on nearest neighbors rather than thresholds should lead to better results. As show in [15] prototype-based rules may be converted directly into fuzzy rules, therefore algorithms generating P-rules provide and interesting and little explored alternative to the neurofuzzy approaches.

The PTDL algorithm has the following advantages:

- it supports all attribute types;
- different types of rules may be generated, depending on the desired requirements: C-rules, P-rules or F-rules;
- it is simple to program and provides flexible decision borders;
- various distance functions may be used to improve generalization;
- small number of accurate and comprehensible rules are generated.

These properties make PTDL algorithm a very interesting and promising tool for data analysis. It can be further extended by adding various feature selection techniques. In PTDL the output of each rule is binary and each rule may operate on a different, independent, locally relevant subset of attributes. This is not quite true for the nearest neighbor type of P-rules where common feature space is required for pairs of prototypes, although different pairs may operate in different subspaces.

Unfortunately the PTDL algorithm has some limitations. Its computational complexity is relatively high, requiring $O(N^2)$ operations for N training vectors to calculate all distances between the training vectors. All distance-based algorithms have $O(N^2)$ complexity and are thus much slower than simple decision trees and therefore quite costly to use on datasets with very large number of vectors. However, initial clusterization or a similar technique will significantly reduce the effort [21]. For example, joint information obtained from the whole dataset and clustered prototypes may be used, in the first step selecting best prototypes from all those obtained after clusterization, and then making a local search among training vectors close to the selected prototype to tune the rules. A combination of P-rules in both threshold and the nearest neighbor style may lead to the best of both worlds: localized decision regions combined with the hyperplanes, that sometimes are necessary for high accuracy (as in the case of Pima Indian Diabetes data). If a small number of features is used to evaluate similarity P-rules have simpler interpretation (the case is more similar to a given prototype than to any other) than combinations of features used in definition of hyperplanes.

These and other improvements of the PTDL algorithm will be explored in the near future. However, it is already clear that P-rules deserve at least as much attention as that enjoyed by the neurofuzzy systems.

Acknowledgement. Both authors are grateful for the support by the Polish Committee for Scientific Research, research grant 2005-2007.

References

[1] W. Duch, R. Setiono, J.M. Zurada, Computational intelligence methods for understanding of data. Proc. of the IEEE ,Vol. 92(5), pp. 771-805, 2004.

[2] The handbook of data mining, Ed. Nong Ye, Lawrence Erlbaum Associates, London 2003.

[3] J.R. Quinlan, C4.5: Programs for machine learning. Morgan Kaufman, CA, 1993.

[4] L. Breiman, J.H. Friedman, R.A. Oslhen, and C.J. Stone, Classification and Regression Trees. Belmont, CA: Wadsworth International Group, 1984.

[5] K. Grąbczewski and W. Duch, The separability of split value criterion. 5th Conference on Neural Network and Soft Computing, Polish Neural Network Society, Zakopane, Poland, 2000, pp. 201-208.

[6] N. Jankowski, K. Grąbczewski, W. Duch, A. Naud and R. Adamczak, Ghostminer data mining software, http://www.fqspl.com.pl/ghostminer/

[7] W. Pedrycz, Fuzzy set technology in knowledge discovery, Fuzzy Sets and Systems Vol. 98, pp. 279-290, 1998.

[8] W. Duch, Similarity based methods: a general framework for classification, approximation and association. Control and Cybernetics Vol. 29(4), pp. 937-968, 2000.

[9] W. Duch and K. Grudziński, Prototype based rules - a new way to understand the data. Proc. of the International Joint Conference on Neural Networks (IJCNN) 2001, Washington D.C, USA, pp. 1858-1863.

[10] K. Grąbczewski and W. Duch, Heterogenous forests of decision trees. Springer Lecture Notes in Comp. Science, Vol. 2415, pp. 504-509, 2002.
[11] B. Kosko, Neural Networks and Fuzzy Systems. Prentice Hall, 1992.
[12] D. Nauck, F. Klawonn and R. Kruse, Foundations on Neuro-Fuzzy Systems. J. Wiley, New York, 1997.
[13] S.K. Pal and S. Mitra, Neuro-Fuzzy Pattern Recognition. J. Wiley, New York, 1999.
[14] D.R. Wilson, T.R. Martinez, Improved Heterogeneous Distance Functions, Journal of Artificial Intelligence Research, Vol. 6, pp. 1-34, 1997.
[15] W. Duch and M. Blachnik, Fuzzy rule-based system derived from similarity to prototypes, Lecture Notes in Computer Science, Vol. 3316, pp. 912-917, 2004.
[16] C.J. Mertz and P.M. Murphy, UCI repository of machine learning databases, http://www.ics.uci.edu/pub/machine-learning-databases.
[17] D. Nauck and U. Nauck, http://fuzzy.cs.uni-magdeburg.de/nefclass/nefclass.html
[18] A.J. Walker, S.S. Cross, and R.F. Harrison, Visualization of biomedical datasets by use of growing cell structure networks: a novel diagnostic classification technique. Lancet, Vol. 354, pp. 1518-1522, 1999.
[19] I.H. Witten and E. Frank, Data Mining: Practical machine learning tools and techniques, 2nd Ed, Morgan Kaufmann, San Francisco, 2005.
[20] D. Michie, D.J. Spiegelhalter, and C.C. Taylor (eds), Machine Learning, Neural and Statistical Classification. Ellis Horwood, 1994.
[21] G. Shakhnarovish, T. Darrell, P. Indyk (eds), Nearest-Neighbor Methods in Learning and Vision, MIT Press, 2005.

A Fuzzy LMS Neural Network Method for Evaluation of Importance of Indices in MADM

Feng Kong and Hongyan Liu

School of Business Administration, North China Electric Power University, Baoding 071003, P.R. China
wenfeng3596@hotmail.com

Abstract. A fuzzy LMS (least-mean-square algorithm) neural network evaluation model, with fuzzy triangular numbers as inputs, is set up to compare the importance of different indices. The model can determine attribute or index weights (importance) automatically so that they are more objectively and accurately distributed. The model also has a strong self-learning ability so that calculations are greatly reduced and simplified. Further, decision maker's specific preferences for uncertainty, i.e., risk-averse, risk-loving or risk-neutral, are considered in the evaluation model. Hence, our method can give objective results while taking into decision maker's subjective intensions. Meanwhile, it is simple. A numerical example is given to illustrate the method.

1 Introduction

Multi-attribute decision making involves selecting an alternative that most satisfies the objectives after examining and comparing the multiple attributes of the alternatives. However, it is often difficult to express all the attributes in crisp numbers. Therefore, we have to describe some attributes with linguistic words such as "very important", or "equally important", and so on. Hence, fuzzy theory was introduced into multi-attribute decision making to form the branch of fuzzy multi-attribute decision making [1].

Multi-attribute decision making are widely applied in a number of fields relating to economic evaluation, project evaluation, performance evaluation, investment decisions, etc. It has long been an important topic in decision making. In multi-attribute decision making problems, the determination of weights plays an important role. At present, weight determination methods could be roughly divided into two categories: subjective and objective weight determination methods. Subjective methods can fully reflect the intensions of decision makers, so decision results are more likely to be in agreement with decision makers desires. However, they tend to be, sometimes, too subjective. Objective methods often do not take the decision maker's intensions into account, so they can yield objective results at the expense of fully reflecting decision maker's intensions. In multi-attribute decision making, the decision maker's different preferences for uncertainty,i.e., risk-loving, risk-averse or risk-neutral, will have an effect on the decision results [1].Therefore, how to combine both subjective and objective information into decision making to make results both objective and able to reflect decision maker's subjective intensions is of both theoretical and practical importance [2, 3, 4].

Fuzzy decision making is a mathematical method to deal with decision making under fuzzy environments [5, 6, 7]. Prevalent fuzzy decision methods, such as multi-objective fuzzy decision making, analytical weighting decision analysis, fuzzy priority ratio approximation method, consensus opinion ranking, etc, usually strive to order the alternatives under fuzzy environments, or select the optimal alternative, with certain fuzzy restrictions, from a universe of discourse. The huge number of calculations needed makes these methods very complex. Hence, we put forward a fuzzy neural network model which uses triangular fuzzy numbers as inputs. The feature of our model is that it trains and assigns the weights of attributes automatically so that weights are allocated more objectively and accurately. This model has a strong self-learning ability. At the same time, this model, being an objective method, can also reflect the influence of the decision-maker's subjective preferences for uncertainty on decision results.

We give a brief introduction to the fuzzy theory in section 2. In section 3, a fuzzy LMS (least-mean-square algorithm) neural network evaluation model, with fuzzy triangular numbers as inputs, is set up to solve fuzzy multi-attribute decision making (MADM) problems. A numerical simulation is shown in section 4. Section 5 concludes.

2 Neural Network Model with Fuzzy Inputs

This paper uses normalized triangular fuzzy numbers as inputs to the neural network model [1, 2] for MADM.

2.1 Neurons with Triangular Fuzzy Numbers as Inputs

Denote the membership function of triangular fuzzy numbers $x_j = (x_{j1}, x_{j2}, x_{j3})$ by $\mu(x_j)$, then, there is,

$$\mu(x_j) = \begin{cases} \dfrac{x_j - x_{j1}}{x_{j2} - x_{j1}}, & x_{j1} \le x_j < x_{j2}; \\ \dfrac{x_{j3} - x_j}{x_{j3} - x_{j2}}, & x_{j2} \le x_j \le x_{j3}; \\ 0, & \text{else.} \end{cases} \tag{1}$$

Specifically, when $x_{j1} = x_{j2} = x_{j3}$, the fuzzy numbers become crisp numbers.

Linguistic terms can be transformed into triangular fuzzy numbers according to certain rules [8, 9, 10].

In order to combine the decision maker's subjective preferences for uncertainty into this method, and at the same time make the output unaffected after inputting the above-mentioned neurons into crisp numbers, we set up, based on weighted fuzzy Hamming distance, the following output function:

$$z_j = \beta_j \int [\mu(x_j^L) - \mu(0^L)] dx + (1-\beta_j) \int [\mu(x_j^R) - \mu(0^R)] dx \tag{2}$$

$$\mu(x_j^L) = \sup_{x_j^L = y+z,\, z \le 0} \mu(y),$$

$$\mu(x_j^R) = \sup_{x_j^R = y+z,\, z \ge 0} \mu(y).$$

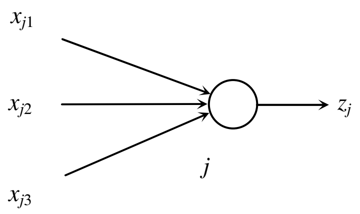


Fig. 1. Fuzzy input neurons, *j*, with triangular fuzzy numbers as inputs

where β_j represents the coefficient of the decision maker's preference for uncertainty, or the decision maker's uncertainty preference weight for the *j*-th attribute.

2.2 Fuzzy LMS Neural Networks with Triangular Fuzzy Numbers as Inputs

Fuzzy LMS neural network method for fuzzy MADM was developed from the LMS neural network [4]. It is a three-leveled network, with the input level being composed of initial uncertain signal sources, the second level being the input revision level which adjusts inputs after taking into account the decision-maker's specific preferences for uncertainty, and the third level being the output level.

Suppose a multi-attribute decision making problem has *M* fuzzy attributes (indeices) and *m* crisp attributes, then there are (*M*+*m*) input neurons and one output neuron, as is shown in Fig. 2.

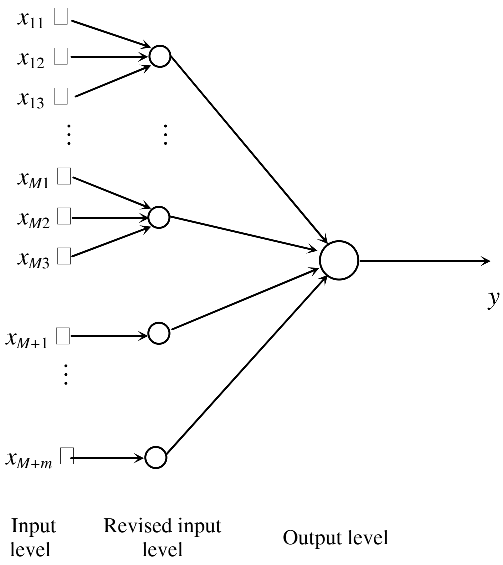


Fig. 2. Fuzzy LMS neural network

After the adjustment of the input sample X_k, the output in the network becomes:

$$y_k = \sum_{i=1}^{N} w_i X_k \tag{3}$$

where w_i represent the weights of the output level, X_k represent the input revision level which adjusts inputs after taking into account the decision-maker's specific preferences for uncertainty.

2.3 Leaning Algorithm for Fuzzy LMS Neural Networks

We adopt the monitored centre-selection algorithm. The specific learning procedures are as follows.

Define the objective function to be:

$$E = \frac{1}{2} e_k^2 = \frac{1}{2}(d - y)^2 \tag{4}$$

where d_k represent the expected output of network samples.

$$y = Z(n)^{\mathrm{T}} W(n)$$

The learning of the network is in fact the solving of freedom parameters, w_i, β_j, to minimize the objective function.

For the weights of the output level, w_i, there is,

$$\frac{\partial E(n)}{\partial w_i(n)} = e(n) \frac{\partial e(n)}{\partial W} = -e(n) Z(n) \tag{5}$$

$$w_i(n+1) = w_i(n) - \eta_1 \frac{\partial E(n)}{\partial w_i(n)} \tag{6}$$

For the weights of the input levels, there is

$$\frac{\partial E(n)}{\partial \beta_j(n)} = -e(n) Z(n) \frac{\partial Z(n)}{\partial \beta_j(n)} = -e(n) Z(n) S_j(n) \tag{8}$$

where $S_j = \int [\mu(x_{kj}^L) - \mu(0^L)] dx - \int [\mu(x_j^R) - \mu(0^R)] dx$, and

$$\beta_j(n+1) = \beta_j(n) - \eta_2 \frac{\partial E(n)}{\partial \beta_j(n)} \tag{9}$$

with η_1, η_2 being the learning rate.

3 Fuzzy LMS Neural Network MADM Method

Our fuzzy LMS neural network multi-attribute decision making model is fairly straightforward. We only need to input the attribute scales of the alternatives and the evaluation results into our model and train them. In order to take into account the decision-maker's specific preferences, the positive and negative ideal solutions are introduced into our fuzzy LMS model.

If there are *K* alternatives, with *M* fuzzy attributes and *m* crisp attributes, then, the decision matrix is:

$$C = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1,M+m} \\ c_{21} & c_{22} & \cdots & c_{2,M+m} \\ \vdots & \vdots & \cdots & \vdots \\ c_{K1} & c_{K2} & \cdots & c_{K,M+m} \end{pmatrix}.$$

Decisions are made according to *n*-2 alternatives or input samples, whose evaluation results have already been obtained.

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1,M+m} \\ a_{21} & a_{22} & \cdots & a_{2,M+m} \\ \vdots & \vdots & \cdots & \vdots \\ a_{N-2,1} & a_{N-2,2} & \cdots & a_{N-2,M+m} \end{pmatrix}.$$

The corresponding evaluation results, or output samples, are:

$$D = (d_1, d_2, ..., d_{N-2}).$$

For crisp attributes, the attribute scales of the ideal and negative ideal solutions respectively are (Benefit type scales are used as examples to illustrate):

$$c_j^+ = \max_i \{a_{ij}\}; c_j^- = \min_i \{a_{ij}\} \tag{10}$$

For fuzzy attributes, the attribute scales of the ideal and negative ideal solutions respectively are (Benefit type scales are used as examples to illustrate):

$$c_j^+ = \sup \max_i \{a_{ij}\}; c_j^- = \inf \min_i \{a_{ij}\} \tag{11}$$

These attribute scales can also be given by the decision maker, according to her subjective judges. Let the expected output of the positive and negative ideal solutions be 0.95 and 0.05 respectively.

Input the above data into the network and begin training, we can get the final weights of the index w_j and β_j, with w_j being the importance of each index.

4 Numerical Simulations

A firm now has to decide which product to develop. Suppose there are already 15 similar products in the market, whose attribute indices and overall market performances are shown in Table 1.

In Table 1, the former 3 are crisp attributes and the latter 5 are fuzzy attributes. For the fuzzy attributes, we could transform them into triangular fuzzy numbers according to Table 2.

Table 1. Attribute scales and overall market performances of similar products prevalent in the market

	Production cost ($)	Operational cost ($)	Noise (db)	Function	Maintenance	Reliability	Flexibility	Safety	Overall performances
1	42	64	35	VG	RB	VG	RB	RG	0.78
2	20	52	70	A	RB	RG	G	A	0.56
3	35	47	65	A	RG	G	G	RG	0.73
4	40	30	40	G	G	G	RG	G	0.92
5	30	55	55	RG	G	G	A	VG	0.80
6	63	75	79	G	RG	RB	G	RG	0.57
7	64	40	40	RG	RG	VG	G	RG	0.67
8	84	40	54	RG	VG	A	VG	RG	0.82
9	38	70	88	RG	G	RG	G	G	0.46
10	75	41	50	A	RG	G	RG	VG	0.69
11	49	68	79	G	A	G	G	VG	0.56
12	44	35	90	RG	G	A	VG	A	0.63
13	80	31	46	G	RG	A	RG	A	0.74
14	41	45	42	VG	RG	VG	A	VG	0.87
15	57	68	53	RG	A	RG	A	G	0.58

Table 2. Transformation rules for fuzzy linguistic words

Order	Linguistic words	Corresponding triangular fuzzy numbers
1	Very good (VG)	(0.85,0.95,1.00)
2	Good (G)	(0.70,0.80,0.90)
3	Relatively good (RG)	(0.55,0.65,0.75)
4	Average (A)	(0.40,0.50,0.60)
5	Relatively bad (RB)	(0.25,0.35,0.45)
6	Bad (B)	(0.10,0.20,0.30)
7	Very bad (VB)	(0.00,0.05,0.15)

Then select the ideal and negative ideal solutions (see Table 3). After normalization, input the 17 samples into the fuzzy neural network.

Having the network trained, input the data in Table 4 into the network, we will get the weights and the values of β_j (see Table 4).

Negative weights show that the less the value of this index, the more satisfaction the index gives the decision maker. The decision maker is risk averse if β_j is greater than 0.5.

Table 3. The ideal and the negative ideal solutions

	Ideal solution	Negative ideal solution
Production cost ($)	20	85
Operational cost ($)	30	75
Noise (dB)	19	70
Performance	(1.0,1.0,1.0)	(0.4,0.4,0.4)
Maintenance	(0.9,0.9,0.9)	(0.25,0.25,0.25)
Reliability	(1.0,1.0,1.0)	(0.4,0.4,0.4)
Flexibility	(1.0,1.0,1.0)	(0.4,0.4,0.4)
Safety	(1.0,1.0,1.0)	(0.4,0.4,0.4)
Overall performances	0.95	0.05

Table 4. Values of weights and β_j

Indices	Production cost ($)	Operational cost ($)	Noise (db)	Function	Maintenance	Reliability	Flexibility	Safety
Importance	-0.035	-0.111	-0.164	0.220	0.230	0.240	0.133	0.154
β_j	--	--	--	0.571	0. 642	0.621	0.513	0.782

5 Conclusion

Most multi-attribute decision making methods are either too objective or too subjective to give decision results that are both scientific and in agreement with the decision maker's intensions. Many of the methods also have the shortcoming of complexity as huge amount of calculations are often needed. In this paper an LMS neural network model was set up with fuzzy triangular numbers as inputs to solve multi-attribute decision making problems. The model can determine the weights of attributes automatically so that weights are more objectively and accurately distributed. The model also has a great self-learning ability so that calculations are greatly reduced and simplified. Further, decision maker's specific preferences for uncertainty, i.e., risk-averse, risk-loving or risk-neutral, are considered in the determination of weights. Hence, this method can give objective results after taking into decision maker's subjective intensions.

Acknowledgement

This paper is supported by the doctorial teachers' foundation of North China Electric Power University.

References

1. Li, R.J.: Fuzzy Multi-attribute Decision Making Theory and Its applications. Beijing: Science Press (2002).
2. Song, R.: Multi-attribute Decision Making Method Based on Wavelet Neural Networks. Computer Engineering and Applications, 35 (2000) 46-48.
3. Qiu, C, Liu, Y. :Multi-attribute Decision Making Based on Artificial Neural Network. Journal of Beijing Science and Engineering University, 20 (2000) 65-68.
4. Hdgan, M. T., Demuth, H. B., Beale, M.: Neural Network Design. PWS Publishing Company (1996).
5. Song, G., Zou, P.: Weight Determination in Multi-attribute Decision Making. Systems Engineering, 19 (2001) 84-89.
6. Fuller, R., Carlsson, C.: Fuzzy Multiple Criteria Decision Making: Recent Development. Fuzzy sets and Fuzzy systems, 78 (1996) 139-153.
7. L, Jie, Hou, Z.: Weight Determination Method Based on a Combination of AHP, Delphi method and Neural Network. Systems Engineering Theories and Practices, 20 (2001) 59-63.
8. Delgado, M., Herreral, F.: Combining numerical and linguistic information in group decision making. Journal of information sciences, 107 (1998) 177-197.
9. Herrera, F., Herrera E.: Linguistic decision analysis: steps for solving decision problems under linguistic information. Fuzzy Sets and Systems, 115 (2000) 67-82.
10. Cheng, C.-H., Lin, Y. : Evaluating the best main battle tank using fuzzy decision theory with linguistic criteria evaluation. European Journal of Operational Research, 142 (2002) 174-186.

Fuzzy RBF Neural Network Model for Multiple Attribute Decision Making

Feng Kong and Hongyan Liu

School of Business Administration, North China Electric Power University, Baoding 071003, P.R. China
Wen_feng3596@sina.com

Abstract. This paper studies how to compare and select one best alternative, from the new alternatives, according to historical or current ones. Previous methods not only need a lot of data but also are complex. So, we put forward an RBF neural network method that not only has the advantages of common neural network methods, but also need much less samples and are straightforward. The number of neurons at the hidden level is easily determined. This model can determine attribute weights automatically so that weights are more objectively and accurately distributed. Further, decision maker's specific preferences for uncertainty, i.e., risk-averse, risk-loving or risk-neutral, are considered in the determination of weights. Hence, our method can give objective results while taking into decision maker's subjective intensions. A numerical example is given to illustrate the method.

1 Introduction

Fuzzy multi-attribute decision making (MADM) is widely applied in both social and economic environments. Researches on this subject centers essentially around the following aspects[1-5].

Selection and normalization of indices. Up to now, researches on this subject are fairly rich and developed.

Comparison and ranking of fuzzy numbers. There are many researches on this subject. However, most of them do not take into account decision makers' subjective features so as to give results in agreement with their intesions. Only a few of them, such as reference, which gave a method to compare the fuzzy numbers based on decision makers' different preferences for uncertainty.

Determination of the weights of the multiple attributes. There are also many methods in this aspect, for example, linear weighting, TOPSIS, etc, but different method can often achieve quite contradictory results.

This paper studies how to compare and select one best alternative from the new ones according to historical or current alternatives. Previous methods not only need a lot of data but also are complex. So, we put forward an RBF neural network method that not only has the advantages of common neural network methods, but also need much less samples and are straightforward.

In this paper, a neural network model with triangular fuzzy numbers is given in section 2. In section 3, we set up the fuzzy RBF nueral network model for MADM. A numerical example is given to illustrate the method in section 4. Section 5 concludes.

2 Neural Network Model with Fuzzy Inputs

In this paper, we use standardized triangular fuzzy numbers as input neurons of the fuzzy RBF neural network [1, 2]. See Fig 1.

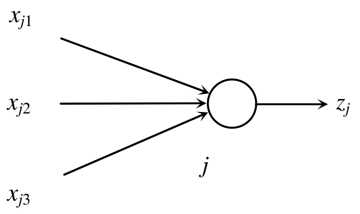


Fig. 1. Fuzzy input neurons, *j*, with triangular fuzzy numbers as inputs

2.1 Neurons with Triangular Fuzzy Numbers as Inputs

Denote the membership function of triangular fuzzy numbers $x_j = (x_{j1}, x_{j2}, x_{j3})$ by $\mu(x_j)$, then, there is,

$$\mu(x_j) = \begin{cases} \dfrac{x_j - x_{j1}}{x_{j2} - x_{j1}}, & x_{j1} \le x_j < x_{j2}; \\ \dfrac{x_{j3} - x_j}{x_{j3} - x_{j2}}, & x_{j2} \le x_j \le x_{j3}; \\ 0, & \text{else.} \end{cases} \tag{1}$$

Specifically, when $x_{j1} = x_{j2} = x_{j3}$, the fuzzy numbers become crisp numbers.

Linguistic terms can be transformed into triangular fuzzy numbers according to certain rules [8, 9, 10]. See Table 1.

In order to combine the decision maker's subjective preferences for uncertainty into this method, and at the same time make the output unaffected after inputting the above-mentioned neurons into crisp numbers, we set up, based on weighted fuzzy Hamming distance, the following output function:

$$z_j = \beta_j \int [\mu(x_j^L) - \mu(0^L)]dx + (1-\beta_j)\int [\mu(x_j^R) - \mu(0^R)]dx \tag{2}$$

$$\mu(x_j^L) = \sup_{x_j^L = y+z, z \le 0} \mu(y),$$

$$\mu(x_j^R) = \sup_{x_j^R = y+z, z \ge 0} \mu(y).$$

where β_j represent the coefficient of the decision maker's preference for uncertainty, or the decision maker's uncertainty preference weight for the *j*-th attribute.

Table 1. Transformation rules for fuzzy linguistic words

Order	Linguistic words	Corresponding triangular fuzzy numbers
1	Very good (VG)	(0.85,0.95,1.00)
2	Good (G)	(0.70,0.80,0.90)
3	Relatively good (RG)	(0.55,0.65,0.75)
4	Average (A)	(0.40,0.50,0.60)
5	Relatively bad (RB)	(0.25,0.35,0.45)
6	Bad (B)	(0.10,0.20,0.30)
7	Very bad (VB)	(0.00,0.05,0.15)

2.2 Fuzzy RBF Neural Networks with Triangular Fuzzy Numbers as Inputs

Fuzzy RBF neural network method for fuzzy MADM was developed from the RBF network put forward by Powell [4]. It is a four-leveled network, with the input level being composed of initial uncertain signal sources, the second level being the input revision level which adjusts inputs after taking into account the decision-maker's specific preferences for uncertainty, the third level being hidden levels whose numbers are determined according to the specific problems under consideration and whose transfer functions are nonnegative, nonlinear and central Radial-symmetric, and the fourth level being the output level.

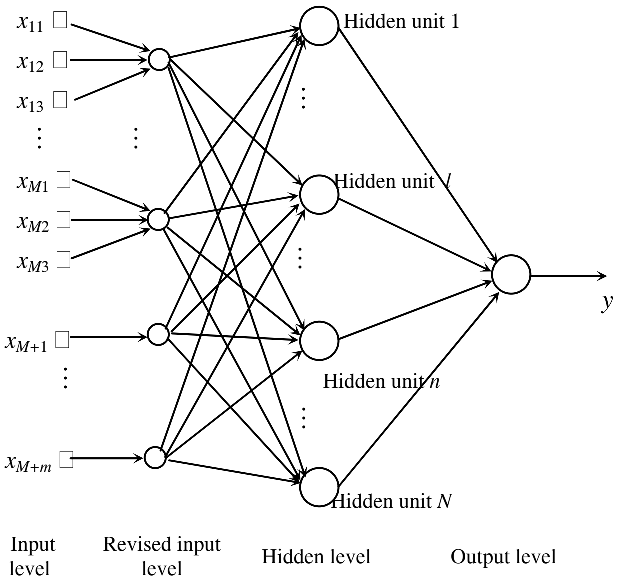


Fig. 2. Fuzzy RBF neural network

Suppose a multiple attribute decision making problems has M fuzzy attributes and m crisp attributes, then there are (M+m) input neurons, and one output neuron. Further suppose there are N hidden units, or N training samples, in the hidden levels of the network, as is shown in Fig. 2.

After the adjustment of the input sample X_k, the output in the network becomes:

$$y_k = \sum_{i=1}^{N} w_i \varphi(X_k, Z_i) \tag{3}$$

where w_i represent the weights of the output level, $\varphi(X_k, X_i)$ represent the incentive output functions of the hidden levels, which generally are Gauss functions:

$$\varphi(X_k, Z_i) = \exp\left(-\frac{1}{2\sigma_i^2} \sum_{p=1}^{M+m} (Z_{kp} - Z_{ip}) \right) \tag{4}$$

where Z_i=(z$_{i1}$, z$_{i2}$,…, z$_{i,M+m}$) represent the centre of the Gauss functions, and σ_i^2 represents the variance.

2.3 Leaning Algorithm for Fuzzy RBF Neural Networks

We adopt the monitored centre-selection algorithm. The specific learning procedures are:

Define the objective function to be:

$$E = \frac{1}{2}\sum_{k=1}^{N} e_k^2 = \frac{1}{2}\sum_{k=1}^{N} \left[d_k - y_k\right]^2 \tag{5}$$

where d_k represent the expected output of network samples.

The learning of the network is in fact the solving of freedom parameters, t_i, w_i, Σ_i^{-1}, and β_j to minimize the objective function.

For the weights of the output level, w_i, there is,

$$\frac{\partial E(n)}{\partial w_i(n)} = \sum_{k=1}^{N} e_k(n) \varphi(X_k, Z_i) \tag{6}$$

$$w_i(n+1) = w_i(n) - \eta_1 \frac{\partial E(n)}{\partial w_i(n)} \tag{7}$$

For the centers of the hidden levels, t_i, there is

$$\frac{\partial E(n)}{\partial t_i(n)} = 2w_i(n) \sum_{k=1}^{N} e_k(n) \varphi'(X_k, t_i(n)) \Sigma_i^{-1}(n)(X_k - t_i(n)) \tag{8}$$

where Σ_i^{-1} are the freedom parameters of the hidden levels related to the variance of Gauss function, $\Sigma_i^{-1} = -\frac{1}{2\sigma_{ii}^2}$. So,

$$t_i(n+1) = t_i(n) - \eta_2 \frac{\partial E(n)}{\partial t_i(n)} \tag{9}$$

For the freedom parameters of the hidden levels, Σ_i^{-1}, there is,

$$\frac{\partial E(n)}{\partial \Sigma_i^{-1}(n)} = -w_i(n) \sum_{k=1}^{N} e_k(n) \varphi'(X_k, t_i(n)) Q_{ki}(n) \tag{10}$$

where $Q_{ki}(n) = (X_k - t_i(n))(X_k - t_i(n))^{\mathrm{T}}$, and

$$\Sigma_i^{-1}(n+1) = \Sigma_i^{-1}(n) - \eta_3 \frac{\partial E(n)}{\partial \Sigma_i^{-1}(n)} \tag{11}$$

For the weights of the input levels, there is

$$\frac{\partial E(n)}{\partial \beta_j(n)} = -2 \sum_{k=1}^{N} e_k(n) \varphi'(X_k, t_i(n)) \Sigma_i^{-1}(n)(X_k - t_i(n)) S_{kj} \tag{12}$$

where $S_{kj} = \int [\mu(x_{kj}^L) - \mu(0^L)]dx - \int [\mu(x_{kj}^R) - \mu(0^R)]dx$, and

$$\beta_j(n+1) = \beta_j(n) - \eta_4 \frac{\partial E(n)}{\partial \beta_j(n)} \tag{13}$$

with $\eta_1, \eta_2, \eta_3, \eta_4$ being the learning rate.

3 Fuzzy RBF Neural Network Model for MADM

Our fuzzy RBF neural network multiple attribute decision making model is fairly straightforward. We only need to input the attribute scales of the alternatives and the evaluation results into our model and train them. In order to take into account the decision-maker's specific preferences, the positive and negative ideal solutions are introduced into our fuzzy RBF model.

We are to select one best alternative from *P* new alternatives. Decisions are to be made according to the evaluation results of some alternatives similar to the *P* new ones. Suppose we have already had *K* (*K*=*n*-2) similar alternatives, with *M* fuzzy attributes and *m* crisp attributes, and their evaluation result. Then, the data matrix, *C*, of the *K* alternatives are:

$$C = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1,M+m} \\ c_{21} & c_{22} & \cdots & c_{2,M+m} \\ \vdots & \vdots & \cdots & \vdots \\ c_{K1} & c_{K2} & \cdots & c_{K,M+m} \end{pmatrix}.$$

The normalized data matrix, A, of the N-2 samples is:

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1,M+m} \\ a_{21} & a_{22} & \cdots & a_{2,M+m} \\ \vdots & \vdots & \cdots & \vdots \\ a_{N-2,1} & a_{N-2,2} & \cdots & a_{N-2,M+m} \end{pmatrix}$$

The K=n-2 alternatives are used as input samples, whose evaluation results have been obtained. The corresponding evaluation results, or output samples, are:

$$D = (d_1, d_2, ..., d_{N-2}) .$$

Then determine the ideal solutions and the negative ideal solutions.

For crisp attributes, the attribute scales of the ideal and negative ideal solutions respectively are (Benefit type scales are used as examples to illustrate):

$$\begin{aligned} c_j^+ &\geq \max_i \{c_{ij}, a_{ij}\} \\ c_j^- &\leq \min_i \{c_{ij}, a_{ij}\} \end{aligned} \tag{14}$$

For fuzzy attributes, the attribute scales of the ideal and negative ideal solutions respectively are (Benefit type scales are used as examples to illustrate):

$$\begin{aligned} c_j^+ &\geq \sup \max_i \{c_{ij}, a_{ij}\} \\ c_j^- &\leq \inf \min_i \{c_{ij}, a_{ij}\} \end{aligned} \tag{15}$$

These attribute scales can also be given by the decision maker, according to her subjective judges. Let the expected output of the positive and negative ideal solutions be 1.0 and 0 respectively.

Input the N samples into the fuzzy RBF neural network. Having the network trained, input the data of the new alternatives into the network, we will get the ranking outputs of the P new alternatives.

4 Numerical Simulations

A firm has four new product development alternatives, A_1, A_2, A_3, and A_4. The four alternatives are evaluated from eight aspects, namely, production cost, operational cost, performance, noise, maintenance, reliability, flexibility and safety. The firm decides to select an alternative according to the overall market performances of similar products in the market.

Suppose there are already 15 similar products in the market, whose attribute index scales and overall market performances are shown in Table 2.

In Table 2, the former 3 attributes are crisp attributes and the latter 5 are fuzzy attributes.

For the fuzzy attributes, we could transform them into triangular fuzzy numbers according to Table 1.

Then select the ideal and negative ideal solutions (see Table 3). After normalization of the 17 samples, input the 17 samples into the fuzzy neural network.

The normalization equation is (Cost type scales are used as examples to illustrate):

$$a_{ij} = \frac{c_j^- - c_{ij}}{c_j^- - c_j^+} \tag{16}$$

where, c_j^+ and c_j^- are the *j*-th attribute scale of ideal solution and negative ideal solution respectively.

Table 2. Attribute scaless and overall market performances of similar products prevalent in the market

	Production cost ($)	Operational cost ($)	Noise (db)	Function	Maintenance	Reliability	Flexibility	Safety	Overall performances
1	42	64	35	VG	RB	VG	RB	RG	0.78
2	20	52	70	A	RB	RG	G	A	0.56
3	35	47	65	A	RG	G	G	RG	0.73
4	40	50	40	G	G	G	RG	G	0.92
5	30	55	55	RG	G	G	A	VG	0.80
6	63	65	79	G	RG	VB	G	RG	0.57
7	64	40	40	RG	RG	VG	G	RG	0.87
8	84	60	54	RG	VG	A	VG	RG	0.82
9	38	40	88	RG	G	RG	G	G	0.76
10	75	41	50	A	RG	G	RG	VG	0.69
11	49	68	79	G	A	G	G	VG	0.76
12	44	35	90	RG	G	A	VG	A	0.73
13	80	31	46	G	RG	A	RG	A	0.74
14	41	45	42	VG	RG	VG	A	VG	0.87
15	57	68	53	RG	A	RG	A	G	0.58

Table 3. The ideal and the negative ideal solutions

	Ideal solution	Negative ideal solution
Production cost ($)	20	48
Operational cost ($)	35	65
Noise (dB)	19	70
Performance	(1.0,1.0,1.0)	(0.4,0.4,0.4)
Maintenance	(0.9,0.9,0.9)	(0.25,0.25,0.25)
Reliability	(1.0,1.0,1.0)	(0.4,0.4,0.4)
Flexibility	(1.0,1.0,1.0)	(0.4,0.4,0.4)
Safety	(1.0,1.0,1.0)	(0.4,0.4,0.4)
Overall performances	1.00	0.00

Having the network trained, input the data in Table 4 into the network, we will get the outputs (see Table 4).

Table 4. Alternative indexes values of the product being developed

Attribute	Type of product being developed			
	A_1	A_2	A_3	A_4
Production cost ($)	45	25	35	48
Operational cost ($)	35	50	45	65
Noise (db)	25	60	50	19
Performance	G	RG	A	RG
Maintenance	A	A	G	RB
Reliability	G	G	A	RG
Flexibility	G	A	VG	A
Safety	RG	VG	RG	G
Overall performances	0.83	0.72	0.75	0.71

The ordering of the alternatives are: $A_1 \succ A_3 \succ A_2 \succ A_4$.

5 Conclusion

Most multiple attribute decision making methods are either too objective or too subjective to give decision results that are both scientific and in agreement with the decision maker's intensions. Many of the methods also have the shortcoming of complexity as huge amount of calculations are often needed.

This paper sets up a fuzzy RBF neural network model with fuzzy triangular numbers (or fuzzy linguistic words) as inputs to solve multiple attribute decision making problems. The model can determine the weights of attributes automatically so that weights are more objectively and accurately distributed. The model also has a great self-learning ability so that calculations are greatly reduced and simplified. Further, decision maker's specific preferences for uncertainty, i.e., risk-averse, risk-loving or risk-neutral, are considered in the determination of weights. Hence, this method can still give objective results even after taking into decision maker's subjective intensions.

Acknowledgement

This paper is supported by the doctorial teachers' foundation of North China Electric Power University.

References

1. Li, R.-J.: Fuzzy Multi-attribute Decision Making Theory and Its applications. Beijing: Science Press (2002).
2. Song, R.: Multi-attribute Decision Making Method Based on Wavelet Neural Networks. Computer Engineering and Applications, Vol. 35 (2000) 46-48.

3. Qiu, C, Liu, Y. :Multi-attribute Decision Making Based on Artificial Neural Network. Journal of Beijing Science and Engineering University, Vol. 20 (2000) 65-68.
4. Hdgan, M. T., Demuth, H. B., Beale, M.: Neural Network Design. PWS Publishing Company (1996).
5. Song, G., Zou, P.: Weight Determination in Multi-attribute Decision Making. Systems Engineering, 19 (2001) 84-89.
6. Fuller, R., Carlsson, C.: Fuzzy Multiple Criteria Decision Making: Recent Development. Fuzzy sets and Fuzzy systems, 78 (1996) 139-153.
7. L, Jie, Hou, Z.: Weight Determination Method Based on a Combination of AHP, Delphi method and Neural Network. Systems Engineering Theories and Practices, 20 (2001) 59-63.
8. Delgado, M., Herreral, F.: Combining numerical and linguistic information in group decision making. Journal of information sciences, 107 (1998) 177-197.
9. Herrera, F., Herrera E.: Linguistic decision analysis: steps for solving decision problems under linguistic information. Fuzzy Sets and Systems, 115 (2000) 67-82.
10. Cheng, C.-H., Lin, Y. : Evaluating the best main battle tank using fuzzy decision theory with linguistic criteria evaluation. European Journal of Operational Research, 142 (2002) 174-186.

A Study on Decision Model of Bottleneck Capacity Expansion with Fuzzy Demand

Bo He[1], Chao Yang[1], Mingming Ren[1], and Yunfeng Ma[2]

[1] School of Management, Huazhong University of Science & Technology, Wuhan, 430074 China
heboo@163.com
[2] School of Management, Wuhan University of Science & Technology, Wuhan, 430081 China

Abstract. After the network has been constructed, with the increasing demand, the network must be faced with the capacity expansion problem. In this paper, a mathematic model is formulated to solve the bottleneck capacity expansion problem of network with fuzzy demand. A linear program model with fuzzy coefficient is put forward. We present a decomposition algorithm to solve the model. The results show the decomposition algorithm can improve the solving speed greatly. So, we can minimize the expansion cost and provide evidence for the decision maker to make reasonable and effective decision.

1 Introduction

There are a lot of networks in real-world such as transportation networks, power networks, water/gas supply networks and logistics networks. With the development of socio-economy, the old networks can not meet the increasing demands, so it may be faced with the capacity expansion problem. For example, one path in power network need to be expanded, but what decides the capacity of the path is its bottleneck capacity, that is the minimum capacity of the segment of the path, so we call it the bottleneck capacity expansion problem. Yang Chao studied the bottleneck capacity expansion problem with budget constraints and put forward a strong polynomial algorithm for a special case [1,2,3]. Wu Yun studied the bottleneck capacity expansion problem with budget constraints based on the Yang's model, in which the unit expansion cost is stochastic. He devised a mixed intelligent algorithm using stochastic programming [4]. Usually, with the development of socio-economy, the needs which the network should meet may be difficult to forecast. In many cases, the needs are manifested in uncertain forms such as stochastic and fuzzy. Although, the stochastic model can deal with uncertainty better, it needs the known probability distribution. So, it has limitation. There are plenty of applications in virtue of fuzzy theory to describe demands [5,6,7]. In this paper, we put forward a new problem that we have a set of facilities which have different prices and expansion capabilities to expand the bottleneck capacity of the network. We want to decide which facility portfolio be employed to expand the bottleneck capacity of the network and minimize the expansion cost.

For example, there are a communication network need be expanding. We have cablesoptical fibers and multiplexers to expand it. These facilities have different prices and expansion capabilities. Which facility portfolio should we choose to meet the increasing needs so that the investment will be minimized. The paper is organized as follows. In Section 2 we define the problem and we propose a model for the problem. Section 3 descibes the decomposition algorithm to solve the model. In section 4, we give an example and the computational experiences are also presented.

2 Model

2.1 Parameter

Let $G(A, E, C)$ be a network, in which $A = \{a_1, a_2, \cdots a_m\}$ is a vertex set and $E = \{e_1, e_2, \cdots, e_n\}$ is an arc set. $C = \{c_1, c_2, \cdots c_n\}$ is initial capacity set of corresponding arcs. In this network, there are K paths, denoted as $p_k\,(k = 1, 2, \cdots K)$. There is a set of facilities denoted as $t\,(t = 1, \cdots, T)$ to expand the arcs in the network. Facility t's price is f_t and expansion capability is d_t. For simplicity, we only consider the fixed cost of facilities. Let $\tilde{D}_k$ be the fuzzy demand of path $p_k\,(k = 1, 2, \cdots K)$.

2.2 Fuzzy Demand

A fuzzy number is a normal convex fuzzy set. For the fuzzy number $\tilde{N}$, its membership function can be expressed as $\mu_{\tilde{N}} = \begin{cases} L(x), l \le x \le m \\ R(x), m \le x \le r \end{cases}$ in which $L(x)$ is a continuous increasing function, $0 \le L(x) \le 1$.$R(x)$ is a continuous decreasing function, $0 \le R(x) \le 1$. If both $L(x)$ and $R(x)$ are linear function, then $\tilde{N}$ is called triangle fuzzy number and denoted as $\tilde{N}(l, m, r)$. Generally, fuzzy demand is described as triangle fuzzy number. That is to say, the most possible value of fuzzy demand $\tilde{N}(l, m, r)$ is m, the optimistic value of that is r and pessimistic value is l. The membership function $\mu_{\tilde{N}}$ is the probability measure of fuzzy demand. If the demand is m, then the membership value is 1 which denote the most possible case. If the demand is r or l, then the membership value is 0 which denote the most impossible case.

2.3 Model

Now, we will expand some paths in the network so to let the expanded paths satisfying demands. At the same time, we should decide which facility portfolio be employed so that the investment will be minimized. Since what decides the capacity of a path is its segment (arc) that has the minimum capacity of all the segments in the path, for the path $p_k\,(k = 1, 2, \cdots K)$, the capacity before expanding is defined $Cap(p_k) = \min\limits_{e \in p_k} c(e)$. The model $P1$ is formulated as follows:

$$\min \sum_{i=1}^{n} \sum_{t=1}^{T} f_t x_{it} \tag{1}$$

$$s.t.Cap\left(p_k\right) = \min_{e \in p_k} w\left(e\right) \geq \tilde{D}_k \quad k = 1, 2, \cdots K \tag{2}$$

$$0 \leq w_i\left(e_i\right) - c_i\left(e_i\right) \leq \sum_{t=1}^{T} d_t x_{it} \quad i = 1, 2, \cdots n \tag{3}$$

$$x_{it} = 0, 1, 2, \cdots \quad i = 1, 2, \cdots, n \quad t = 1, 2, \cdots T \tag{4}$$

In the above model,x_{it} is a decision variable that indicates the number of facility t employed in arc i. $W = \{w_1, w_2, \cdots, w_n\}$ is a vector that denotes the corresponding arc's capacity after expanded. $\tilde{D}_k\left(l_k, m_k, r_k\right)$ is a triangle fuzzy number.

3 Decomposition Algorithm

The above model is a linear programming model with fuzzy coefficients. There are several approaches for this kind of problems [8]. In this paper, we employ the method based on fuzzy number sorting criteria to transform the fuzzy inequality in the model to non fuzzy inequality, so that an equivalent deterministic model $P2$ is formulated as follows:

$$\min \sum_{i=1}^{n} \sum_{t=1}^{T} f_t x_{it} \tag{5}$$

$$s.t.Cap\left(p_k\right) = \min_{e \in p_k} w\left(e\right) \geq l_k \quad k = 1, 2, \cdots K \tag{6}$$

$$Cap\left(p_k\right) = \min_{e \in p_k} w\left(e\right) \geq m_k \quad k = 1, 2, \cdots K \tag{7}$$

$$Cap\left(p_k\right) = \min_{e \in p_k} w\left(e\right) \geq r_k \quad k = 1, 2, \cdots K \tag{8}$$

$$0 \leq w_i\left(e_i\right) - c_i\left(e_i\right) \leq \sum_{t=1}^{T} d_t x_{it} \quad i = 1, 2, \cdots n \tag{9}$$

$$x_{it} = 0, 1, 2, \cdots \quad i = 1, 2, \cdots, n \quad t = 1, 2, \cdots T \tag{10}$$

Remove formula (6) and (7) from the above model, the feasible zone will not be changed. So the model $P2$ can be reformulated to model $P3$ as follows:

$$\min \sum_{i=1}^{n} \sum_{t=1}^{T} f_t x_{it} \tag{11}$$

$$s.t.Cap\left(p_k\right) = \min_{e \in p_k} w\left(e\right) \geq r_k \quad k = 1, 2, \cdots K \tag{12}$$

$$0 \leq w_i\left(e_i\right) - c_i\left(e_i\right) \leq \sum_{t=1}^{T} d_t x_{it} \quad i = 1, 2, \cdots n \tag{13}$$

$$x_{it} = 0, 1, 2, \cdots \quad i = 1, 2, \cdots, nt = 1, 2, \cdots T \tag{14}$$

If we remove formula (12) from model $P3$, it will be transformed into a knapsack problem. Since the knapsack problem was proved NP-complete problem and our model's computational complexity is not less than knapsack problem, so our problem is also NP-complete problem. Although Model $P3$ is formulated as integer programming which can be solved by branch and bound method, as to median and large-size problem, the computation time will become intolerable. In order to reduce computation time, a decomposition algorithm is brought forward according to the problem's characteristics to convert model $P3$ into a two-phase decision model. For formula (12),$Cap\left(p_k\right) = \min\limits_{e \in p_k} w\left(e\right) \geq r_k$, that is to expand the capacity of arcs on path p_k to r_k when the capacity of arcs are less than demand r_k. If the capacity of arcs after expanded is larger than r_k, and what decides the capacity of a path is its arc that has the minimum capacity, then doing this will increase the value of objective function, namely the cost, thus we can't get the optimum solution. Bearing in mind of this, the first phase of our decomposition algorithm is to determine the arcs need to be expanded, denoted $E' \subseteq E$. Usually, the arcs need to be expanded are far less than the arcs in network. Therefore, the number of constraints (13) is greatly reduced. The second phase is to solve the sub-problem of knapsack problem. We give the specific steps of our decomposition algorithm as follows:

Step 1: initialize, let $k = 1, q = 0$, set $E' = \Phi$

Step 2: search$e \in P_k$ and $c\left(e\right) < r_k$, if $e \notin E'$, put e into E', let $q = q + 1$; if all arcs have been searched, go into step 3

Step 3: let $k = k + 1$, if $k \leq K$, turn to step 2; otherwise, go into step 4

Step 4: solve the following sub- problem of knapsack problem

$$z_i = \min \sum_{t=1}^{T} f_t x_{it} \tag{15}$$

$$s.t. 0 \leq w_i\left(e_i\right) - c_i\left(e_i\right) \leq \sum_{t=1}^{T} d_t x_{it} \quad e \in E' i = 1, 2, \cdots q \tag{16}$$

Step 5: sum up$z_i \left(i = 1, \cdots, q\right)$, then we obtain the gross expanding cost.

4 Example and Computational Experience

Consider the following network shown in figure 1. There are 36 arcs on the network and their initial capacities are shown in table 1. There are 9 paths need to be expanded which are shown in table 2. We have 6 facilities shown in table 3 which have different price and expanding capability. Without losing generality, here we do not consider units.

Table 1. Initial capacity of every arc

Arc e_i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Capacity c_i	21	28	24	19	30	36	15	16	17	25	24	29	18	36	37	18	22	28
Arc e_i	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
Capacity c_i	27	31	39	48	41	28	24	37	34	35	29	26	38	36	20	27	28	24

Table 2. The arcs on every path and fuzzy demand of every path

Path p_k	1	2	3	4	5	6	7	8	9
Links on path	3,7,12,21	7,8,15	5,14,22,28,32	18,19,16	16,22,27	15,23,34,33	20,25,28,36	4,12,21,26,30	1,5,15,23
Fuzzy demand	(24,26,28)	(17,18,20)	(33,34,36)	(23,24,25)	(36,39,40)	(35,37,38)	(24,26,27)	(32,34,35)	(28,30,32)

Table 3. Facility's price and expanding capability

Facility	1	2	3	4	5	6
Price	200	240	320	280	400	350
Expanding capability	2	3	4	3	5	4

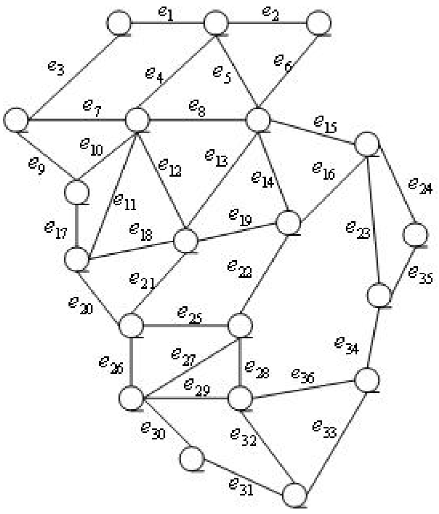


Fig. 1. A network to be expanded

Table 4. Results

Arcs need expanding	**Capacity need**	**Facility employed and its number**	**Expanding cost**
3	4	3(1)	320
7	18	2(3),3(1),5(1)	1440
8	4	3(1)	320
5	8	2(1),5(1)	640
28	1	1(1)	200
16	29	2(5),3(1),5(2)	2320
27	6	2(2)	480
15	1	1(1)	200
34	11	2(1),3(2)	880
33	18	2(3),3(1),5(1)	1440
25	3	2(1)	240
36	3	2(1)	240
4	16	3(4)	1280
12	6	2(2)	480
30	9	3(1),5(1)	720
1	11	2(1),3(2)	880

We solved the example by programming in MATLAB7.0. The results are shown in table 4. The first figure of the 3rd column refers to the facility employed and the figure in bracket refers to its number. The minimum cost is 12080.

The computational experiments described in the following section were designed to evaluate the performance of our solution procedure with respective to a serials of test problems. It was coded in matlab7.0 and run on a computer with

Table 5. Comparison of decomposition algorithm and LP+BB

|N|[a]	**|K|**[b]	**|T|**[c]	**LP+BB**	**Decomposition algorithm**
40	10	8	2:04	0:28
45	12	8	3:21	0:36
50	14	8	5:28	0:47
55	16	8	7:49	0:58
60	18	8	9:08	1:05
65	20	12	12:17	1:07
70	22	12	15:51	1:16
75	24	12	19:22	1:36
80	26	12	24:37	1:48
85	28	12	33:49	2:08

[a] number of arcs
[b] number of paths need expanding
[c] number of facilities

Intel Celeron 2.1G Processor and 256MB memory. Ten problem sets were generated randomly but systematically to capture a wide range of problem structures. The numbers of arcs and paths needing expanded vary from 40 to 85 and from 10 to 28, respectively. The number of facility was fixed to either 8 or 12. The initial capacity of every arc was generated from Normal distribution $N(30, 10)$. Expansion costs was generated from uniform distribution $U(2, 4)$. We compared the solutions obtained by our decomposition algorithm with that obtained by linear programming with branch and bound. The LP+BB are generated by the commercial package LINGO. Since both approaches can obtain the optimal solution, we only compared the computational time. The results are presented in table 5. From our test problems, We can see the decomposition algorithm is outperformed to LP+BB. As the problem's size increasing, the decomposition algorithm can greatly improve the computational speed.

5 Conclusions

1. In this paper, a mathematic model is developed to solve the network bottleneck capacity expansion problem with fuzzy demand. A linear program model with fuzzy coefficient is put forward. We present a decomposition algorithm to solve the model. The results show the decomposition algorithm can improve the solving speed greatly.
2. In our model, we only consider the fixed cost of facility and didn't consider the variable cost. In different settings, the variable cost has relation with different factors. Therefore, the only thing we should do is to modify the above model a little.
3. We employ the method based on fuzzy number sorting criteria to transform the fuzzy inequality to non fuzzy inequality in this paper. In fact, it is a most

conservative strategy. Although we can use tolerance approach, the capacity expansion is a tactic decision. So, it is appropriate to use our method.

4. We only consider the capacity expansion of arcs in this paper. In fact, we also need to consider the capacity expansion of nodes. As regard to this, we can transform the node-based capacity expansion into arc-based capacity expansion as the following figure 2. In fig 2, we want to expand node A's capacity, we split the node A into two nodes A1 and A2 linked by arc a, then we expand capacity of arc a. This is equivalent to expand capacity of node A.

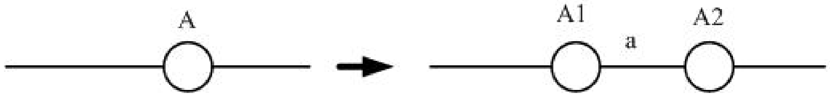


Fig. 2. node-based capacity expansion to arc-based capacity expansion

Acknowledgements

The authors are grateful to the two anonymous referees for their helpful comments which greatly helped improve this paper. This study is partially sponsored by a National Natural Science Foundation of China under Grant number 70471042.

References

1. Yang, C., Zhang, J. A Constrained Maximum Capacity Paths Problem on Network. International Journal of Computer and Mathematics. **70**(1998) 19–33.
2. Zhang J, Yang C, Lin Y. A Class of Bottleneck Expansion Problems. Computer &Operation Research. **28**(2001)505–519.
3. Yang, C., Zhu, Y. Capacity Expansion of Network System. Journal of Huazhong University of Science and Technology. **19**(2001)102–104
4. Wu, yun., Zhou, Jian., Yang, Jun. Dependent-Chance Programming Model for Stochastic Network Bottleneck Capacity Expansion. Chinese Journal of Management Science,**12**(2004) 113–117.
5. Tang, Jia, fu., Wang, Ding., Wei., Xu, Bao, don. Fuzzy modeling approach to aggregate production planning with multi product. Journal of Management Sciences in China. **6**(2003)44–50.
6. Ge, Jian., Li, Yan., Feng., Xia, Guo, Ping. Research on global supply chain production planning under uncertain environment. Computer Integrated Manufacturing Systems. **11**(2005)1120–1126.
7. Tang, J., Wang, D., Fung, R. Fuzzy Formulation for Multi-product Aggregate Production Planning. Production planning and control. **11**(2000)670–676
8. Tang, J, Wang, D., Fung, R. Understanding of Fuzzy Optimization: Theory and Methods. Journal of System Science and Complexity. **17**(2004)1–20

Workpiece Recognition by the Combination of Multiple Simplified Fuzzy ARTMAP

Zhanhui Yuan, Gang Wang, and Jihua Yang

Mechanical Engineering School, Tianjin University
92 WeiJin Road, NanKai Division, Tianjin City, P.R. China
ngia@163.com

Abstract. Simplified fuzzy ARTMAP(SFAM) is a simplification of fuzzy ARTMAP(FAM) in reducing architectural redundancy and computational overhead. The performance of individual SFAM depends on the ordering of training sample presentation. A multiple classifier combination scheme is proposed in order to overcome the problem. The sum rule voting algorithm combines the results from several SFAM's and generates reliable and accurate recognition conclusion. A confidence vector is assigned to each SFAM. The confidence element value can be dynamically adjusted according to the historical achievements. Experiments of recognizing mechanical workpieces have been conducted to verify the proposed method. The experimental results have shown that the fusion approach can achieve reliable recognition.

Keywords: ARTMAP, Neural network, workpiece recognition.

1 Introduction

FAM is an neural network which can be used as classifier. FAM is able to receive analog data and recognize accurately. But the structure of FAM is too complicated. SFAM is a modification to the conventional FAM. SFAM reserves the ability of FAM to overcome the stability-plasticity dilemma. The supervise learning method of SFAM can incorporate additional classes at any time. The SFAM needs less computation efforts than FAM.

One drawback of SFAM is that the network performance is affected by the ordering of training sample presentation[1]. Good initial formation of the cluster prototypes is important to SFAM. Recent years, some researchers have proposed fusion techniques to overcome this problem. Jervls et al [2] combines SFAM and Baye's classifier for reliable recognition. Loo[3] proposed a method in which effective probabilistic plurality voting technique is used to produce outputs from multiple SFAM's. The accuracy rate is monotonously improved with increased number of SFAM network .

A mechanical workpiece recognition scheme is proposed based on multiple classifier combination. The effective recognition is obtained by using sum rule voting algorithm according to the outputs from several SFAMs. The image features are Orthogonal Fourier-Mellin Memonts (OFMMs) which are invariant to shift, rotation, and scale. Experiments have been conducted to recognize mechanical workpieces.

2 SFAM for Workpiece Recognition

As shown in Fig.1, there are four fields in SFAM: F_0 , F_1, F_2 and F_M. The nodes of F_0 represent a current input vector. F_1 field receives both bottom-up of input from F_0 and top-down of input from F_2. F_2 represents the active code or category. F_M is map field to generate recognition class. The inputs to the network are in the complement code form $I=(a,\bar{a})$. A vector W_j of adaptive weights are associated with a F_2 category node j(j=1,…, N). For the input vector I and node j in the F_2 field, the choice function $T_j(I)$ is defined as:

$$T_j(I)=\frac{|I\wedge w_j|}{\alpha+|w_j|} \tag{1}$$

where $\wedge$ and $|\bullet|$ are AND and norm operators in fuzzy theory, respectively.
$\alpha(\alpha>0)$ is choice parameter.

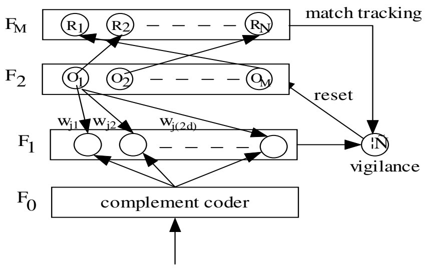


Fig. 1. The structure of SFAM

The category choice operation is performed to search the node in F_2 field for satisfying the following condition.

$$T_J=\max\{T_i: i=1,\ldots, M\} \tag{2}$$

F_2 field resonance occurs if the match function value of chosen category J meets the vigilance criterion.

$$|I\wedge W_j| / |I| \geqslant \rho \tag{3}$$

where $\rho$is the vigilance parameter.

Map field F_M is activated when one of the F_2 categories becomes active. When the Jth F_2 node is chosen , the F_m output vector X_m obeys,

$$x_m = w_J^M \tag{4}$$

where X_m is weight connecting the Jth F_2 node and F_m nodes.

In the supervised learning procedure, match tracking is triggered by a mismatch at the map field F_m if

$$X_m \neq y \tag{5}$$

where y denotes designed class vector.

Match tracking increases vigilance parameterρuntil it is slightly larger than the F_2 match value $|I \wedge W_j| / |I|$. As result, the Jth F_2 node doesn't satisfy the resonant condition (3). New search in F_2 field is performed until a node satisfies the matching criterions in F_2 and F_m. If no such node exists, an uncommitted node is assigned.

In the unsupervised learning procedure, the weight vector W_j is updated whenever F_2 field resonance occurs.

$$w_j^{(i+1)} = \beta(I \wedge w_j^i) + (1-\beta)w_j^i \tag{6}$$

where βis learning rate. β=1 for fast learning.
i denotes the ith learning iteration

3 Sum Rule Voting Algorithm for Workpiece Recognition

Multiple classifier combination can improve the recognition performance. The voting can be unweighted or weighted. The structure of mechanical workpiece recognition by multiple classifiers is shown in Fig.2. Several SFAMs are voters which receive visual features of workpieces and generate recognition results. Since the individual SFAM is sensitive to the sequence of training sample presentation, each SFAM is off-line trained with different orders. On-line training is also possible but the input vectors are applied to the network once only in the order in which they are generated. The voting algorithm combines the information provided by the SFAMs and produces final conclusion.

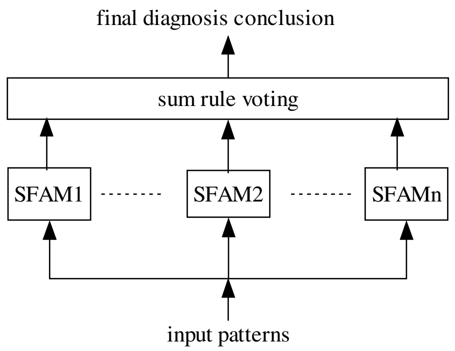


Fig. 2. The architecture of multiple classifier combination

For each SFAM, the output of the network is a class recognition vector with M element S_j(j=1,…, M) which can be 0 or 1. Since each SFAM has its own experience of training, the performance of each SFAM is different each other. We assign a confidence vector to each SFAM. The ith element of jth SFAM confidence vector is defined,

$$C_{ji} = 1 + \frac{S^t_{ji}}{S^0_{ji}} \tag{7}$$

Where S^t_{ji} represents the times of successful recognition.
S^0_{ji} represents the overall times of recognition.

In initial stage, as no recognition is performed, the value of C_{ji} equals to 1. If in the recognition history, no false recognition happens, the value of C_{ji} will approach to 2. The confidence value C_{ji} is dynamically adjusted according to their historical achievements.

The voting algorithm combines the outputs from the multiple SFAMs and generates a conclusion vector R. The ith element of vector R can be calculated according to sum rule voting mechanism.

$$R_i = \sum_{j=1}^{1} C_{ji} \cdot S_j \quad \text{(i=1,..., M)} \tag{8}$$

The final voting result is the element with maximum value,

$$R_I = \max\{R_i : i = 1, \cdots M\} \tag{9}$$

4 Experiments of Workpiece Recognition

4.1 Feature Extraction

The features for workpiece recognition are OFMM's which is based on a set of radial polynomials defined in a polar coordinate system (r,θ).

$$\phi_{n,m} = \frac{1}{2\pi a_0} \int_0^{2\pi} \int_0^1 f(r,\theta) Q_n(r) \exp(-jm\theta) r dr d\theta \tag{10}$$

where f(r,θ) is the image. M=0,±1, ±2, … is the circular harmonic order.
a_0 is a normalization constant.

$Q_n(r)$ is a polynomial in r of degree n. The set of $Q_n(r)$ is orthogonal over the range $0 \le r \le 1$.

$$|\phi_{nm}| \text{ with } a_{ns} = (-1)^{n+s} \frac{(n+s+1)!}{(n-s)!s!(s+1)!} \tag{11}$$

The radical polynomials with low degree are independent of the circular harmonic order m. The OFMMs are invariant to rotation changes in the image.

If we assign a frame to the center of the image. All the moments calculated in this frame are shift invariant. The scale and intensity factors can be obtained by using the low order FMMs. The normalized OFMMs based on the factors are scale and intensity invariant.

The properties of OFMMs being invariant to shift, rotation ,scale and intensity are very useful in pattern recognition, especially in our application case of mechanical workpiece recognition.

4.2 The Computation of OFFMs

The computation procedure of OFMMs consists of four steps

Step 1: Determing the center of the image by the first order geometric moments.

$$\bar{x} = \frac{m_{10}}{m_{00}}, \bar{y} = \frac{m_{01}}{m_{00}} \tag{12}$$

where $m_{pq} = \sum_x \sum_y x^p y^q f(x,y)$ $(\bar{x}, \bar{y})$ is the coordinate value of the image center .

Step 2: Assigning a frame to the image center and calculating the FMMs in this frame.
Step 3: Calculating the scale and intensity factors by low order FMM's and normalizing the FMMs by the scale and intensity factors.
Step 4: Normalizing OFFMs from normalized FMM.

The modulus of the OFMMs, $|\phi_{nm}|$, is shift, rotation ,scale and intensity invariant.

4.3 Experimental Results

The mechanical workpieces for recognition experiments are nuts, bolts, cushions and spindles with different positions, orientations and dimensions. A CDD camera is used to acquire the images of workpieces. The image processing and recognition computation are performed in a Pentium 4 PC with image card PCI-OK2 plugged into one of the expansion slots. Forty workpieces and twenty OFMMs (n=0,1,2,3 and m=0,1,2,3,4) for each workpiece are used to train the SFAMs off-line. The sum rule voting algorithm combines the six SFAMs and generates recognition conclusion.

The original images of some mechanical workpieces are shown in Fig.3. Table 1 lists the OFMMs feature values of one typical workpiece. Table 2 shows the recognition errors by multiple SFAMs. For comparison, the recognition errors by conventional ART2 are listed in Table 3. Nearly 10 percent improvement of recognition accuracy is obtained .

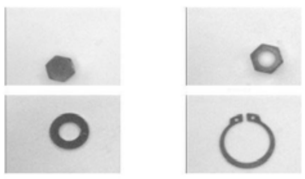


Fig. 3. Four kinds of mechanical parts

Table 1. The OFMMs feature values of hexangular screw

Φ_{nm}	m=0	m=1	m=2	m=3	m=4
n=0	3. 5299e+010	1. 1090e+009	8. 1499e+009	1. 7834e+009	5. 1311e+009
n=1	3. 8800e+018	4. 4369e+009	4. 0074e+017	1. 7021e+017	1. 2996e+018
n=2	4. 0653e+026	3. 0445e+026	1. 2721e+026	1. 1031e+026	9. 1379e+025
n=3	4. 2952e+034	1. 8643e+033	1. 8817e+034	1. 9027e+033	1. 8027e+034

Table 2. Recognition errors by multiple SFAMs

Transformation Method	Bolt	Round backup plate	Nut	Circlip
Diaplacement	2%	3%	4%	5%
Enlargement	3%	4%	3%	3%
Rotate	4%		5%	5%
Intensity	4%	4%	3%	2%

Table 3. Recognition errors by conventional ART2

Transformation method	Bolt	Round backup plate	Nut	Circlip
Diaplacement	12%	14%	13%	13%
Enlargement	12%	13%	13%	14%
Rotate	12%		13%	12%
Intensity	15%	13%	16%	15%

5 Conclusion

A multiple classifier combination scheme is proposed to recognize the mechanical workpieces. Each SFAM of the classifier group is trained with different sample order.

Sum rule voting algorithm assigns a confidence vector to each SFAM and combines the results from the SFAMs. Experimental results have demonstrated the improvements of recognition accuracy.

References

1. Carpenter.G.A ,Grossberg.S,et al: Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps, IEEE Trans. Neural Networks, Vol.3, (1992) 698-712.
2. Jervis.B.W, Garcia.T and Giahnakis.E.P: Probabilistic simplified fuzzy ARTMAP, IEEE Proc. Fuzzy Set, Vol.146, no.4, (1999) 165-169.
3. Loo.C.K and Rao.M.V.C: Accurate and reliable diagnosis and classifier using probabilistic ensemble simplified fuzzy ARTMAP, IEEE Trans. Knowledge and Data Engineering, Vol.17, no.11, (2005) 1589-1593.
4. Sheng.Y and Chen.L: Orthogonal Fourier-Mellin moments for invariant pattern recognition, Optical Society of America, Vol.11, no.6, (1994) 1748-1757.
5. Waxman.A.M, Scibert.M, Bernardon.A.M: Neural system for automatic target learning and recognition, Lincoln Lab.J., Vol.6, (1993) 77-166
6. Grossberg.S, Mingolla.E: Neural dynamics of form perception: Boundary completion, illusory figures, and neon color spreading, Psych. Rev., vol.92, (1985) 173-211.
7. Fahlman.S.E: faster-learning variations on back-propagations: An empirical study, Proc 1988 Connectionist Models Summer School, (1989) 38-51.
8. Frey.P.W,Slate.D.J: Letter recognition using Hooland-style adaptive classifiers, Machine Learning,vol.6,(1991) 161-182.
9. Slazberg.S.L: A nearest hyperrectangle learning method , Machine Learning, vol.6, (1991) 251-276.

Stability of Periodic Solution in Fuzzy BAM Neural Networks with Finite Distributed Delays

Tingwen Huang

Texas A&M University at Qatar, Doha, P.O. Box 5825, Qatar
tingwen.huang@qatar.tamu.edu

Abstract. In this paper, we investigate fuzzy bidirectional associative memory (BAM) neural networks with finite distributed delays. Easily verifiable sufficient conditions for global exponential periodicity of fuzzy BAM neural networks with finite distributed delays are obtained.

1 Introduction

After Kosko [1,2] proposed bidirectional associative memory neural networks:

$$\begin{aligned}\frac{dx_i(t)}{dt} &= -x_i(t) + \sum_{j=1}^{n} a_{ij} f_j(y_j(t)) + I_i(t), \quad i = 1, \cdots, m;\\ \frac{dy_j(t)}{dt} &= -y_j(t) + \sum_{i=1}^{m} b_{ji} f_{n+i}(x_i(t)) + I_{m+j}(t), \quad j = 1, \cdots, n \qquad (1)\end{aligned}$$

many researchers [3-5,10] have studied the dynamics of BAM neural networks with or without delays, including stability and periodic solutions. In this paper, we would like to integrate fuzzy operations into BAM neural networks and maintain local connectedness among cells. Speaking of fuzzy operations, Yang el al. [13-15] first combined those operations with cellular neural networks and investigated the stability of fuzzy cellular neural networks (FCNNs). So far researchers have found that FCNNs are useful in image processing, and some results have been reported on stability and periodicity of FCNNs [7-9, 18]. However, to the best of our knowledge, few results reported on periodicity of FCNNs or fuzzy BAM neural networks with finite distributed delays. It is believed that the conclusions made from finite distributed delays could provide us insight into their counterparts with unbounded delays.

In this paper, we investigate the exponential periodicity for the following fuzzy BAM neural networks:

$$\begin{aligned}\frac{dx_i(t)}{dt} = -d_i x_i(t) + \bigwedge_{j=1}^{n} \int_0^{\tau} \alpha_{ji}(s) f_j(y_j(t-s))ds + \bigwedge_{j=1}^{n} T_{ji}\mu_j(t)\\ \bigvee_{j=1}^{n} \int_0^{\tau} \beta_{ji}(s) f_j(y_j(t-s))ds + \bigvee_{j=1}^{n} H_{ji}\mu_j(t) + I_i(t), \quad i = 1, \cdots, m;\end{aligned}$$

$$\frac{dy_j(t)}{dt} = -d_j y_j(t) + \bigwedge_{i=1}^{m} \int_0^{\tau} \zeta_{ij}(s) f_{n+i}(x_i(t-s))ds + \bigwedge_{i=1}^{m} T_{ij}\mu_{n+i}(t)$$
$$\bigvee_{i=1}^{m} \int_0^{\tau} \eta_{ij}(s) f_{n+i}(x_i(t-s))ds + \bigvee_{i=1}^{m} H_{ij}\mu_{n+i}(t) + I_{m+j}(t), j = 1, \cdots, n; \tag{2}$$

where $c_i > 0$, $d_j > 0$, for $i = 1, \cdots, m$, $j = 1, 2, \cdots, n$; $\alpha_{ij}(s)$ & $\zeta_{ij}(s)$, $\beta_{ij}(s)$ & $\eta_{ij}(s)$ which denote elements of fuzzy feedback MIN templates, fuzzy feedback MAX templates, are continuous in the interval $[0, \tau]$, where τ is a constant; T_{ij} and H_{ij} are fuzzy feed-forward MIN template and fuzzy feed-forward MAX template, respectively; $\bigwedge$ and $\bigvee$ denote the fuzzy AND and fuzzy OR operation, respectively; x_i, y_j are activations of the ith neuron and the jth neuron, respectively; for $k = 1, \cdots, m+n$, μ_k, I_k, denote inputs, bias of the ith neuron, respectively; function f_k are activation functions; moreover, functions $\mu_k(t)$, $I_k(t)$ are continuously periodic functions with period ω, i.e. $\mu_k(t) = \mu_k(t+\omega)$, $I_k(t) = I_k(t+\omega)$, $\quad t \in R$, for $k = 1, \cdots, m+n$.

The initial conditions associated with (1) are the following

$$\begin{aligned} x_i(s) &= \varphi_i(s), \quad -\tau \le s \le 0, \quad i = 1, \cdots, m; \\ y_j(s) &= \varphi_{m+j}(s), \quad -\tau \le s \le 0, \quad j = 1, 2, \cdots, n. \end{aligned} \tag{3}$$

In this paper, we assume that

H: f_i is a bounded function defined on R and satisfies Lipschitz condition

$$|f_i(x) - f_i(y)| \le l_i |x-y|, \quad i = 1, \cdots, m+n, \tag{4}$$

for any $x, y \in R$.

Let's define that for any $\omega \in R^{m+n}$, $||\omega|| = \max_{1 \le k \le m+n} |\omega_k|$;

Let $C_\tau = C([-\tau, 0], R^{m+n})$ be the Banach space of all continuous functions mapping from $[-\tau, 0]$ to R^{m+n} with norm defined as follows: for each $\varphi \in C_\tau$, $||\varphi||_\tau = \sup_{-\tau \le t \le 0} \max_{1 \le i \le m+n} |\varphi_i(t)|$. Now we define the exponentially periodic solution of Fuzzy BAM neural networks.

To be convenient, for given $\varphi \in C_\tau$, let $z(t,\varphi) = (x_1(t,\varphi), \cdots, x_{m+n}(t,\varphi))$= $(x_1(t,\varphi), \cdots, x_m(t,\varphi), y_1(t,\varphi), \cdots, y_n(t,\varphi))^T \in R^{m+n}$ represent the solution of system (1) with initial condition: $z(t) = \varphi(t)$ when $-\tau \le t \le 0$.

Definition 1. *The solution $z(t,\varphi^*)$ is exponentially periodic if $z(t,\varphi^*)$ is periodic and there exist positive constants M, α such that any solution $z(t,\phi)$ satisfies*

$$||z(t,\phi) - z(t,\varphi^*)|| \le M||\phi - \varphi^*||_\tau e^{-\alpha t}, \quad t \ge 0. \tag{5}$$

Here, we present a lemma which we will use in the proof of the main theorem.

Lemma 1. *([13]). For any $a_{ij} \in R$, $x_j, y_j \in R$, $i, j = 1, \cdots, n$, we have the following estimations,*

$$|\bigwedge_{j=1}^{n} a_{ij}x_j - \bigwedge_{j=1}^{n} a_{ij}y_j| \le \sum_{1 \le j \le n} (|a_{ij}| \cdot |x_j - y_j|) \tag{6}$$

and

$$|\bigvee_{j=1}^{n} a_{ij}x_j - \bigvee_{j=1}^{n} a_{ij}y_j| \leq \sum_{1\leq j\leq n} (|a_{ij}|\cdot|x_j-y_j|) \tag{7}$$

2 Main Results

In this section, sufficient conditions to guarantee exponentially periodic solution of fuzzy BAM neural networks are obtained.

First, we would like to present the main result.

Theorem 1. *Fuzzy BAM neural networks (1) is exponentially periodic if there exist* $\lambda_1,\cdots,\lambda_{m+n}$ *such that the following two inequalities hold:*

$$\begin{aligned}
&\lambda_i(2d_i - \sum_{j=1}^{n} l_j \int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)ds) > l_{n+i}\sum_{j=1}^{n}\lambda_{m+j}\int_0^{\tau}(|\zeta_{ij}(s)|+|\eta_{ij}(s)|)ds,\\
&\lambda_{m+j}(2c_j - \sum_{i=1}^{m} l_i \int_0^{\tau}(|\zeta_{ij}(s)|+|\eta_{ij}(s)|)ds) > l_j\sum_{i=1}^{m}\lambda_i\int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)ds
\end{aligned} \tag{8}$$

where $i=1,\cdots,m,\ j=1,2,\cdots,n.$

Proof. First let us prove the following claim.
Claim: Let $z(t,\varphi)=(x_1(t,\varphi),\cdots,x_m(t,\varphi),y_1(t,\varphi),\cdots,y_n(t,\varphi))^T, z(t,\phi)=(x_1(t,\phi),\cdots,x_m(t,\phi),y_1(t,\phi),\cdots,y_n(t,\phi))^T$ be the solutions to system (1). Under the conditions of this theorem, there are positive constants M and ε which depend on λ_i, $i=1,\cdots,m+n$ and known coefficients of (1) such that

$$||z(t,\phi)-z(t,\varphi)|| \leq M||\phi-\varphi||_{\tau}e^{-\varepsilon t}, \quad t\geq 0. \tag{9}$$

Since $z(t,\phi)$, $z(t,\varphi)$ are the solutions to system (1), we have

$$\begin{aligned}
\frac{dx_i(t,\varphi)}{dt} &= -d_ix_i(t,\varphi) + \bigwedge_{j=1}^{n}\int_0^{\tau}\alpha_{ji}(s)f_j(y_j(t-s,\varphi))ds + \bigwedge_{j=1}^{n}T_{ji}\mu_j(t)\\
&\quad \bigvee_{j=1}^{n}\int_0^{\tau}\beta_{ji}(s)f_j(y_j(t-s,\varphi))ds + \bigvee_{j=1}^{n}H_{ji}\mu_j(t) + I_i(t), \quad i=1,\cdots,m;\\
\frac{dx_i(t,\phi)}{dt} &= -d_ix_i(t,\phi) + \bigwedge_{j=1}^{n}\int_0^{\tau}\alpha_{ji}(s)f_j(y_j(t-s,\phi))ds + \bigwedge_{j=1}^{n}T_{ji}\mu_j(t)\\
&\quad \bigvee_{j=1}^{n}\int_0^{\tau}\beta_{ji}(s)f_j(y_j(t-s,\phi))ds + \bigvee_{j=1}^{n}H_{ji}\mu_j(t) + I_i(t), \quad i=1,\cdots,m;
\end{aligned} \tag{10}$$

Thus, for $i = 1, \cdots, m$, we have

$$\begin{aligned}\frac{dx_i(t,\varphi)}{dt} - \frac{dx_i(t,\phi)}{dt} &= -d_i(x_i(t,\varphi) - x_i(t,\phi)) \\ &\quad + \bigwedge_{j=1}^{n} \int_0^{\tau} \alpha_{ji}(s)(f_j(y_j(t-s,\varphi)) - f_j(y_j(t-s,\phi)))ds \\ &\quad + \bigvee_{j=1}^{n} \int_0^{\tau} \beta_{ji}(s)(f_j(y_j(t-s,\varphi)) - f_j(y_j(t-s,\phi)))ds \end{aligned} \tag{11}$$

Similarly, for $j = 1, \cdots, n$, we have

$$\begin{aligned}\frac{dy_j(t,\varphi)}{dt} - \frac{dy_j(t,\phi)}{dt} &= -d_j(y_j(t,\varphi) - y_j(t,\phi)) \\ &\quad + \bigwedge_{i=1}^{m} \int_0^{\tau} \zeta_{ij}(s)(f_{n+i}(x_i(t-s,\varphi)) - f_{n+i}(x_i(t-s,\phi)))ds \\ &\quad + \bigvee_{i=1}^{m} \int_0^{\tau} \eta_{ij}(s)(f_{n+i}(x_i(t-s,\varphi)) - f_{n+i}(x_i(t-s,\phi)))ds \end{aligned} \tag{12}$$

By the condition (8), there exists a positive number ε such that

$$\begin{aligned}\lambda_i(2d_i - \varepsilon - \sum_{j=1}^{n} l_j \int_0^{\tau} (|\alpha_{ji}(s)| + |\beta_{ji}(s)|)ds) &> l_{n+i} \sum_{j=1}^{n} \lambda_{m+j} \int_0^{\tau} e^{\varepsilon s}(|\zeta_{ij}(s)| \\ &\qquad + |\eta_{ij}(s)|)ds, \\ \lambda_{m+j}(2c_j - \varepsilon - \sum_{i=1}^{m} l_i \int_0^{\tau} (|\zeta_{ij}(s)| + |\eta_{ij}(s)|)ds) &> l_j \sum_{i=1}^{m} \lambda_i \int_0^{\tau} e^{\varepsilon s}(|\alpha_{ji}(s)| \\ &\qquad + |\beta_{ji}(s)|)ds \end{aligned} \tag{13}$$

Now let us define the Lyapunov functional $V(z(t,\varphi), z(t,\phi)) = V_1(z(t,\varphi), z(t,\phi)) + V_2(z(t,\varphi), z(t,\phi))$, where

$$\begin{aligned} V_1(z(t,\varphi), z(t,\phi)) &= \sum_{i=1}^{m} \lambda_i \{(x_i(t,\varphi) - x_i(t,\phi))^2 e^{\varepsilon t} + \sum_{j=1}^{n} l_j \int_0^{\tau} (|\alpha_{ji}(s)| + |\beta_{ji}(s)|) \\ &\qquad \int_{t-s}^{t} (y_j(r,\varphi) - y_j(r,\phi))^2 e^{\varepsilon(r+s)} drds\} \\ V_2(z(t,\varphi), z(t,\phi)) &= \sum_{j=1}^{n} \lambda_{m+j} \{(y_j(t,\varphi) - y_j(t,\phi))^2 e^{\varepsilon t} \\ &\qquad + \sum_{i=1}^{m} l_i \int_0^{\tau} (|\zeta_{ij}(s)| + |\eta_{ij}(s)|) \\ &\qquad \int_{t-s}^{t} (x_i(r,\varphi) - (x_i(r,\phi))^2 e^{\varepsilon(r+s)} drds\} \end{aligned} \tag{14}$$

Take the derivative of V_1, using Lemma 1 and inequality $2ab \le a^2 + b^2$ in the following process, we have

$$\begin{aligned}
&\frac{d}{dt}V_1(z(t,\varphi),z(t,\phi)) = \sum_{i=1}^{m}\lambda_i\{\varepsilon e^{\varepsilon t}(x_i(t,\varphi)-x_i(t,\phi))^2\\
&+2e^{\varepsilon t}(x_i(t,\varphi)-x_i(t,\phi))(\dot{x}_i(t,\varphi)-\dot{x}_i(t,\phi))\\
&+\sum_{j=1}^{n} l_j\int_0^{\tau} e^{\varepsilon t}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)[e^{\varepsilon s}(y_j(t,\varphi)-y_j(t,\phi))^2\\
&-(y_j(t-s,\varphi)-y_j(t-s,\phi))^2ds]\}\\
=\;& e^{\varepsilon t}\sum_{i=1}^{m}\lambda_i\{\varepsilon(x_i(t,\varphi)-x_i(t,\phi))^2-2d_i(x_i(t,\varphi)-x_i(t,\phi))^2\\
&+2(x_i(t,\varphi)-x_i(t,\phi))\bigwedge_{j=1}^{n}\int_0^{\tau}\alpha_{ji}(s)(f_j(y_j(t-s,\varphi))-f_j(y_j(t-s,\phi)))ds\\
&+2(x_i(t,\varphi)-x_i(t,\phi))\bigvee_{j=1}^{n}\int_0^{\tau}\beta_{ji}(s)(f_j(y_j(t-s,\varphi))-f_j(y_j(t-s,\phi)))ds\\
&+\sum_{j=1}^{n} l_j\int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)[(y_j(t,\varphi)-y_j(t,\phi))^2e^{\varepsilon s}\\
&-(y_j(t-s,\varphi)-y_j(t-s,\phi))^2]ds\}\\
\le\;& e^{\varepsilon t}\sum_{i=1}^{m}\lambda_i\{(\varepsilon-2d_i)(x_i(t,\varphi)-x_i(t,\phi))^2\\
&+2|x_i(t,\varphi)-x_i(t,\phi)|\sum_{j=1}^{n}\int_0^{\tau} l_j|\alpha_{ji}(s)||y_j(t-s,\varphi)-y_j(t-s,\phi)|ds\\
&+2|x_i(t,\varphi)-x_i(t,\phi)|\sum_{j=1}^{n}\int_0^{\tau} l_j|\beta_{ji}(s)||y_j(t-s,\varphi)-y_j(t-s,\phi)|ds\\
&+\sum_{j=1}^{n} l_j\int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)[(y_j(t,\varphi)-y_j(t,\phi))^2e^{\varepsilon s}\\
&-(y_j(t-s,\varphi)-y_j(t-s,\phi))^2]ds\}\\
\le\;& e^{\varepsilon t}\sum_{i=1}^{m}\lambda_i\{(\varepsilon-2d_i)(x_i(t,\varphi)-x_i(t,\phi))^2\\
&+\sum_{j=1}^{n}\int_0^{\tau} l_j|\alpha_{ji}(s)|((x_i(t,\varphi)-x_i(t,\phi))^2+(y_j(t-s,\varphi)-y_j(t-s,\phi))^2)ds\\
&+\sum_{j=1}^{n}\int_0^{\tau} l_j|\beta_{ji}(s)|((x_i(t,\varphi)-x_i(t,\phi))^2+(y_j(t-s,\varphi)-y_j(t-s,\phi))^2)ds
\end{aligned}$$

$$+\sum_{j=1}^{n} l_j \int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)[(y_j(t,\varphi)-y_j(t,\phi))^2 e^{\varepsilon s}$$
$$-(y_j(t-s,\varphi)-y_j(t-s,\phi))^2]ds\}$$
$$= e^{\varepsilon t}\sum_{i=1}^{m}\lambda_i\{[\varepsilon-2d_i+\sum_{j=1}^{n} l_j \int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)ds](x_i(t,\varphi)-x_i(t,\phi))^2$$
$$+\sum_{j=1}^{n} l_j (y_j(t,\varphi)-y_j(t,\phi))^2 \int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)e^{\varepsilon s}ds\} \tag{15}$$

Similarly, we can obtain that

$$\frac{d}{dt}V_2(z(t,\varphi),z(t,\phi))$$
$$\le e^{\varepsilon t}\sum_{j=1}^{n}\lambda_{m+j}\{[\varepsilon-2c_j+\sum_{i=1}^{m} l_{n+i} \int_0^{\tau}(|\zeta_{ij}(s)|+|\eta_{ij}(s)|)ds](y_i(t,\varphi)-y_i(t,\phi))^2$$
$$+\sum_{i=1}^{m} l_{n+i} (x_i(t,\varphi)-x_i(t,\phi))^2 \int_0^{\tau}(|\zeta_{ij}(s)|+|\eta_{ij}(s)|)e^{\varepsilon s}ds\} \tag{16}$$

By inequalities (15), (16) and (13), we have

$$\frac{d}{dt}V(z(t,\varphi),z(t,\phi)) = \frac{d}{dt}V_1(z(t,\varphi),z(t,\phi)) + \frac{d}{dt}V_2(z(t,\varphi),z(t,\phi))$$
$$\le e^{\varepsilon t}\sum_{i=1}^{m}\lambda_i\{[\varepsilon-2d_i+\sum_{j=1}^{n} l_j \int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)ds](x_i(t,\varphi)-x_i(t,\phi))^2$$
$$+\sum_{j=1}^{n} l_j (y_j(t,\varphi)-y_j(t,\phi))^2 \int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)e^{\varepsilon s}ds\}$$
$$+e^{\varepsilon t}\sum_{j=1}^{n}\lambda_{m+j}\{[\varepsilon-2c_j+\sum_{i=1}^{m} l_{n+i} \int_0^{\tau}(|\zeta_{ij}(s)|+|\eta_{ij}(s)|)ds](y_i(t,\varphi)-y_i(t,\phi))^2$$
$$+\sum_{i=1}^{m} l_{n+i} (x_i(t,\varphi)-x_i(t,\phi))^2 \int_0^{\tau}(|\zeta_{ij}(s)|+|\eta_{ij}(s)|)e^{\varepsilon s}ds\}$$
$$= e^{\varepsilon t}\sum_{i=1}^{m}\lambda_i[\varepsilon-2d_i+\sum_{j=1}^{n} l_j \int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)ds$$
$$+l_{n+i}\sum_{j=1}^{n}\lambda_{m+j}\int_0^{\tau} e^{\varepsilon s}(|\zeta_{ij}(s)|+|\eta_{ij}(s)|)ds](x_i(t,\varphi)-x_i(t,\phi))^2$$
$$+e^{\varepsilon t}\sum_{j=1}^{n}\lambda_{m+j}[\varepsilon-2c_j+\sum_{i=1}^{m} l_{n+i} \int_0^{\tau}(|\zeta_{ij}(s)|+|\eta_{ij}(s)|)ds$$
$$+\sum_{j=1}^{n} l_j \int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|)e^{\varepsilon s}ds](y_i(t,\varphi)-y_i(t,\phi))^2$$
$$\le 0 \tag{17}$$

Thus, $V(z(t,\varphi),z(t,\phi)) \leq V(z(0,\varphi),z(0,\phi))$. For $k=1,\cdots,m+n$, we have

$$\begin{aligned}
e^{\varepsilon t}|z_k(t,\varphi)-z_k(t,\phi)|^2 &\leq \frac{e^{\varepsilon t}}{\min_{1\leq l\leq m+n}\{\lambda_l\}}\{\sum_{i=1}^{m}\lambda_i(x_i(t,\varphi)-x_i(t,\phi))^2 \\
&\quad +\sum_{j=1}^{n}\lambda_{m+j}(y_j(t,\varphi)-y_j(t,\phi))^2\} \\
&\leq V_1(z(t,\varphi),z(t,\phi))+V_2(z(t,\varphi),z(t,\phi)) \\
&\leq V_1(z(0,\varphi),z(0,\phi))+V_2(z(0,\varphi),z(0,\phi)) \\
&= \sum_{i=1}^{m}\lambda_i\{(x_i(0,\varphi)-x_i(0,\phi))^2+\sum_{j=1}^{n}l_j\int_0^{\tau}(|\alpha_{ji}(s)| \\
&\qquad +|\beta_{ji}(s)|) \\
&\quad \int_{-s}^{0}(y_j(r,\varphi)-y_j(r,\phi))^2e^{\varepsilon(r+s)}drds\} \\
&\quad +\sum_{j=1}^{n}\lambda_{m+j}\{(y_j(0,\varphi)-y_j(0,\phi))^2+\sum_{i=1}^{m}l_i\int_0^{\tau}(|\zeta_{ij}(s)| \\
&\qquad +|\eta_{ij}(s)|) \\
&\quad \int_{-s}^{0}(x_i(r,\varphi)-(x_i(r,\phi))^2e^{\varepsilon(r+s)}drds\} \\
&= \sum_{i=1}^{m}\lambda_i\{(\varphi_i(0)-\phi_i(0))^2+\sum_{j=1}^{n}l_j\int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|) \\
&\quad \int_{-s}^{0}(\varphi_{m+j}(r)-\phi_{m+j}(r))^2e^{\varepsilon(r+s)}drds\} \\
&\quad +\sum_{j=1}^{n}\lambda_{m+j}\{(\varphi_{m+j}(0)-\phi_{m+j}(0))^2+\sum_{i=1}^{m}l_i\int_0^{\tau}(|\zeta_{ij}(s)| \\
&\qquad +|\eta_{ij}(s)|) \\
&\quad \int_{-s}^{0}(\varphi_i(r)-\phi_i(r))^2e^{\varepsilon(r+s)}drds\} \\
&\leq ||\varphi-\phi||_{\tau}^2[\sum_{i=1}^{m}\lambda_i\{1+\sum_{j=1}^{n}l_j\int_0^{\tau}(|\alpha_{ji}(s)|+|\beta_{ji}(s)|) \\
&\qquad \int_{-s}^{0}e^{\varepsilon(r+s)}drds\} \\
&\quad +\sum_{j=1}^{n}\lambda_{m+j}\{1+\sum_{i=1}^{m}l_i\int_0^{\tau}(|\zeta_{ij}(s)|+|\eta_{ij}(s)|) \\
&\qquad \int_{-s}^{0}e^{\varepsilon(r+s)}drds\}] \qquad (18)
\end{aligned}$$

From the above inequalities, we have

$$|z_k(t,\phi) - z_k(t,\varphi^*)| \leq M||\phi - \varphi^*||_\tau e^{-\frac{\varepsilon}{2}t}, \quad t \geq 0, \quad k = 1, \cdots, m+n. \quad (19)$$

where $M = \sum_{i=1}^{m} \lambda_i \{1 + \sum_{j=1}^{n} l_j \int_0^\tau (|\alpha_{ji}(s)| + |\beta_{ji}(s)|) \int_{-s}^{0} e^{\varepsilon(r+s)} drds\}$ $+ \sum_{j=1}^{n} \lambda_{m+j} \{1 + \sum_{i=1}^{m} l_i \int_0^\tau (|\zeta_{ij}(s)| + |\eta_{ij}(s)|) \int_{-s}^{0} e^{\varepsilon(r+s)} drds\}$. So far, we have completed the proof of the Claim. By the Claim, to complete this theorem, we just need to prove that there exists a $\omega-$periodic solution for model (2).

Let $z_t(\varphi)(\vartheta) = z(t,\varphi)(\vartheta) = z(t+\vartheta,\varphi)$, for $\vartheta \in [-\tau, 0]$ and $z(t+\vartheta,\varphi)$ is a solution of model (2) with initial solution of (3). Now, we define a Poincare mapping $P : C \to C$ by $P\varphi = z_\omega(\varphi)$. By Claim, there are positives M and ε,

$$||z_t(\phi) - z_t(\varphi^*)||_\tau \leq M||\phi - \varphi^*||_\tau e^{-\frac{\varepsilon}{2}(t-\tau)}, \quad t \geq 0. \quad (20)$$

Choose a positive big integer l, such that $Me^{-\frac{\varepsilon}{2}(\omega l - \tau)} \leq \frac{1}{3}$.

Thus, we have $||P^l\phi - P^l\varphi||_\tau \leq \frac{1}{3}||\phi - \varphi||_\tau$. It implies that P^l is a contraction mapping, so there exists a unique fixed point $\varphi^* \in C$ such that $P^l\varphi^* = \varphi^*$. Note that $P^l(P\varphi^*) = P(P^l\varphi^*) = P\varphi^*$, so $P\varphi^*$ is a fixed point of mapping P^l. By the uniqueness of the fixed point of the contraction map, $P\varphi^* = \varphi^*$, i.e., φ^* is a fixed point of P. Thus, $z_{t+\omega}(\varphi^*) = z_t(z_\omega(\varphi^*)) = z_t(\varphi^*)$, i.e, $z(t+\omega,\varphi^*) = z(t,\varphi^*)$, for any $t > 0$. This proves that there exists one ω-periodic solution, and from the Claim, it is clear that all solutions converge exponentially to $z(t,\varphi^*)$ as $t \to \infty$. So far, we have completed the proof of the theorem.

3 Conclusion

In this paper, we discuss the existence and exponential stability of the equilibrium of FCNN with finite distributed delays. Sufficient condition set up here by using Lyapunov functional is easily verifiable.

Acknowledgments

The author is grateful for the support of Texas A&M University at Qatar.

References

1. Kosko, B.: Bidirectinal Associative Memories. IEEE. Trans. Syst. Man and Cybernetics, SMC-18 (1988) 49-60
2. Kosko, B.: Adaptive Bidirectinal Associative Memories. Appl. Optics 26 (1987) 4947-4960
3. Cao, J., Jiang Q.: An analysis of periodic solutions of bi-directional associative memory networks with time-varying delays. Physics Letters A 330(2004)203-213
4. Chen, A., Huang, L., Liu Z., Cao, J.: Periodic Bidirectional Associative Memory Neural Networks with Distributed Delays. Journal of Mathematical Analysis and Applications 317(2006)80-102

5. Chen, A., Huang, L., Cao, J.: Existence and Stability of Almost Periodic Solution for BAM Neural Networks with Delays. Applied Mathematics and Computation, 137(2003)177-193.
6. Horn, R.A., Johnson, C.R.: Topics in Matrix Analysis, Cambridge University Press, Cambridge (1999)
7. Huang, T.,Zhang, L.: Exponential Stability of Fuzzy Cellular Neural Networks. In: Wang, J., Liao X., Yi Z. (eds.): Advances in Neural Networks. Lecture Notes in Computer Science, Vol. 3496. Springer-Verlag, Berlin Heidelberg, New York (2005)168-173
8. Huang, T.: Exponential Stability of Delayed Fuzzy Cellular Neural Networks with Diffusion, to appear in Chaos, Solitons & Fractrals.
9. Huang, T.: Exponential Stability of Fuzzy Cellular Neural Networks with Unbounded Distributed Delay, Physics Letters A 351(2006)48-52
10. Li, C., Liao, X., Zhang, R.: Delay-dependent Exponential Stability Analysis of Bi-directional Associative Memory Neural Networks with Time Delay: an LMI approach. Chaos, Solitons & Fractals, 24(2005)1119-1134
11. Li, C., Liao, X., Zhang, R., Prasad A: Global Robust Exponential Stability Analysis for Interval Neural Networks with Time-varying Delays. Chaos, Solitons & Fractals 25(2005)751-757
12. Liao, X.F., Wong, K, Li, C: Global Exponential Stability for a Class of Generalized Neural Networks with Distributed Delays. Nonlinear Analysis: Real World Applications 5(2004)527-547
13. Yang, T., Yang, L.B., Wu, C.W., Chua, L.O.: Fuzzy Cellular Neural Networks: Theory. In Proc. of IEEE International Workshop on Cellular Neural networks and Applications (1996)181-186
14. Yang, T. , Yang, L.B., Wu, C.W., Chua, L.O.: Fuzzy Cellular Neural Networks: Applications. In Proc. of IEEE International Workshop on Cellular Neural Networks and Applications (1996)225-230
15. Yang, T., Yang, L.B.: The Global Stability of Fuzzy Cellular Neural Network. Circuits and Systems I: Fundamental Theory and Applications 43(1996)880-883
16. Yang, X., Liao, X., Evans, D., Tnag, Y.: Existence and Stability of Periodic Solution in Impulsive Hopfield Neural Networks with Finite Distributed Delays. Physics Letters A 343(2005)108-116
17. Yang, X., Li C., Liao X., Evans D., Megson G.: Global Exponential Periodicity of a Class of Bidirectional Associative Memory Networks with Finite Distributed Delays. Applied Mathematics and Computation 171(2005)108-121
18. Yuan, K., Cao J., Deng, J.: Exponential stability and periodic solutions of fuzzy cellular neural networks with time-varying delays, to appear in Neurocomputing

Design Methodology of Optimized IG_gHSOFPNN and Its Application to pH Neutralization Process

Ho-Sung Park[1], Kyung-Won Jang[1], Sung-Kwun Oh[2], and Tae-Chon Ahn[1]

[1] School of Electrical Electronic and Information Engineering, Wonkwang University, 344-2, Shinyong-Dong, Iksan, Chon-Buk, 570-749, South Korea
{neuron, jjang, tcahn}@wonkwang.ac.kr
[2] Department of Electrical Engineering, The University of Suwon, San 2-2 Wau-ri, Bongdam-eup, Hwaseong-si, Gyeonggi-do, 445-743, South Korea
ohsk@suwon.ac.kr

Abstract. In this paper, we propose design methodology of optimized Information granulation based genetically optimized Hybrid Self-Organizing Fuzzy Polynomial Neural Networks (IG_gHSOFPNN) by evolutionary optimization. The augmented IG_gHSOFPNN results in a structurally optimized structure and comes with a higher level of flexibility in comparison to the one we encounter in the conventional HSOFPNN. The GA-based design procedure being applied at each layer of IG_gHSOFPNN leads to the selection of preferred nodes (FPNs or PNs) available within the HSOFPNN. The obtained results demonstrate superiority of the proposed networks over the existing fuzzy and neural models.

1 Introduction

GMDH-type algorithms have been extensively used since the mid-1970's for prediction and modeling complex nonlinear processes [1]. While providing with a systematic design procedure, GMDH comes with some drawbacks. To alleviate the problems associated with the GMDH, Self-Organizing Polynomial Neural Networks (SOPNN)[2] Self-Organizing Fuzzy Polynomial Neural Networks (SOFPNN)[3], and Hybrid Self-Organizing Fuzzy Polynomial Neural Networks (HSOFPNN)[4] introduced. In this paper, to address the above design problems coming with the development of conventional self-organizing neural networks, in particular, HSOFPNN, we introduce the IG_gHSOFPNN with the aid of the information granulation [6] as well as the genetic algorithm [5]. The determination of the optimal values of the parameters available within an individual PN(viz. the number of input variables, the order of the polynomial and the collection of preferred nodes) and FPN(viz. the number of input variables, the order of the polynomial, the collection of preferred nodes, the number of MFs for each input variable, and the selection of MFs) leads to a structurally and parametrically optimized network.

2 The Architecture and Development of HSOFPNN

2.1 Fuzzy Polynomial Neuron : FPN

This neuron, regarded as a generic type of the processing unit, dwells on the concept of fuzzy sets. When arranged together, FPNs build the first layer of the

HSOFPNN. The FPN consists of two basic functional modules. The first one, labeled by **F**, is a collection of fuzzy sets that form an interface between the input numeric variables and the processing part realized by the neuron. The second module (denoted here by **P**) concentrates on the function – based nonlinear (polynomial) processing.

In other words, FPN realizes a family of multiple-input single-output rules. Each rule, reads in the form:

$$\text{If } x_p \text{ is } \boldsymbol{A}_l \text{ and } x_q \text{ is } \boldsymbol{B}_k \text{ then } z \text{ is } P_{lk}(x_i, x_j, \boldsymbol{a}_{lk}) \tag{1}$$

The activation levels of the rules contribute to the output of the FPN being computed as a weighted average of the individual condition parts (functional transformations) P_K.

$$z = \frac{\sum_{K=1}^{all\ rules} \mu_K P_K(x_i, x_j, \boldsymbol{a}_K)}{\sum_{K=1}^{all\ rules} \mu_K} = \sum_{K=1}^{all\ rules} \tilde{\mu}_K P_K(x_i, x_j, \boldsymbol{a}_K), \quad \tilde{\mu}_K = \mu_K \Big/ \sum_{L=1}^{all\ rules} \mu_L \tag{2}$$

Table 1. Different forms of the regression polynomials building a FPN and PN

No. of inputs / Order of the polynomial			1	2	3
Order	Type				
	FPN	PN			
0	Type 1		Constant	Constant	Constant
1	Type 2	Type 1	Linear	Bilinear	Trilinear
2	Type 3	Type 2	Quadratic	Biquadratic-1	Triquadratic-1
	Type 4	Type 3		Biquadratic-2	Triquadratic-2

1: Basic type, 2: Modified type

2.2 Polynomial Neuron : PN

The SOPNN algorithm in the PN based layer of HSOFPNN is based on the GMDH method and utilizes a class of polynomials such as linear, quadratic, modified quadratic, etc. to describe basic processing realized there.

The input-output relationship for the above data realized by the SOPNN algorithm can be described in the following manner:

$$y = f(x_1, x_2, \ldots, x_N) \tag{3}$$

Where, $x_1, x_2, \cdots, x_N$ denote the outputs of the 1st layer of FPN nodes(the inputs of the 2nd layer(PN nodes)).

$$\hat{y} = c_0 + \sum_{i=1}^{N} c_i x_i + \sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij} x_i x_j + \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N} c_{ijk} x_i x_j x_k + \cdots \tag{4}$$

3 Optimization of HSOFPNN by Information Granulation and Genetic Algorithms

3.1 Optimization of HSOFPNN by Information Granulation

3.1.1 Definition of the Premise Part of Fuzzy Rules Using IG

We assume that given a set of data $X=\{x_1, x_2, \ldots, x_n\}$ related to a certain application, there are some clusters revealed by the HCM[7]. Each cluster is represented by its center and all elements, which belong to it. Each membership function in the premise part of the rule is assigned to be complementary with neighboring ones.

3.1.2 Restructure of the Consequence Part of Fuzzy Rules Using IG

Here, let us concentrate on building the consequent part of the fuzzy rule. Each cluster (and the resulting rule) can be regarded as a sub-model of the overall system. The premise parts of the fuzzy rules help quantify how much overlap occurs between the individual sub-models. The consequent part of the fuzzy rule is a polynomial with independent variables for which the center point on this cluster (that is the sub-model) is mapped onto the point of origin. Therefore, all data belonging to the cluster are mapped into new coordinates. This is done by subtracting the value of the center point from all data belonging to the corresponding cluster.

3.2 Optimization of HSOFPNN by Genetic Algorithm

GAs has been theoretically and empirically demonstrated to provide robust search capabilities in complex spaces thus offering a valid solution strategy to problems requiring efficient and effective searching. It is eventually instructive to highlight the main features that tell GA apart from some other optimization methods: (1) GA operates on the codes of the variables, but not the variables themselves. (2) GA searches optimal points starting from a group (population) of points in the search space (potential solutions), rather than a single point. (3) GA's search is directed only by some fitness function whose form could be quite complex; we do not require it need to be differentiable.

In this study, for the optimization of the HSOFPNN model, GA uses the serial method of binary type, roulette-wheel used in the selection process, one-point crossover in the crossover operation, and a binary inversion (complementation) operation in the mutation operator. To retain the best individual and carry it over to the next generation, we use elitist strategy [5].

4 The Algorithm and Design Procedure of IG_gHSOFPNN

Overall, the framework of the design procedure of the IG_gHSOFPNN architecture comprises the following steps.

[Step 1] *Determine system's input variables.*

[Step 2] *Form training and testing data.*

The input-output data set $(\boldsymbol{x}_i, y_i)=(x_{1i}, x_{2i}, \ldots, x_{ni}, y_i)$, $i=1, 2, \ldots, N$ is divided into two parts, that is, a training and testing dataset.

[Step 3] *Decision of axis of MFs by Information granulation*
As mentioned in '3.1.1 Definition of the premise part of fuzzy rules using IG', we obtained the new axis of MFs by information granulation.

[Step 4] *Decide initial information for constructing the HSOFPNN structure.*
a) Initial specification of the fuzzy inference method and the fuzzy identification
b) Initial specification for decision of HSOFPNN structure

[Step 5] *Decide a structure of the PN and FPN based layer of HSOFPNN using genetic design.*
This concerns the selection of the number of input variables, the polynomial order, the input variables, the number of membership functions, and the selection of membership functions to be assigned at each node of the corresponding layer.

In nodes (PN and FPNs) of each layer of HSOFPNN, we adhere to the notation of Fig. 3.

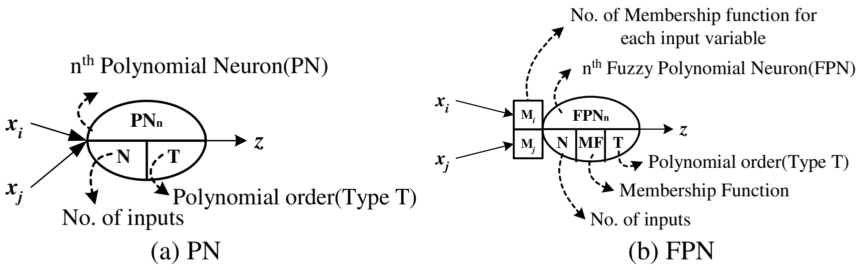


Fig. 3. Formation of each PN or FPN in HSOFPNN architecture

[Step 6] *Estimate the coefficient parameters of the polynomial in the selected node (PN or FPN).*
[Step 6-1] *In case of a PN (PN-based layer)*
The vector of coefficients $\mathbf{C}_i$ is derived by minimizing the mean squared error between y_i and z_{mi}.

$$E = \frac{1}{N_{tr}} \sum_{i=0}^{N_{tr}} (y_i - z_{mi})^2 \tag{5}$$

Using the training data subset, this gives rise to the set of linear equations

$$\mathbf{Y}=\mathbf{X}_i\mathbf{C}_i \tag{6}$$

Evidently, the coefficients of the PN of nodes in each layer are expressed in the form

$$y=f(x_1, x_2, \ldots, x_N) \quad \mathbf{C}_i=(\mathbf{X}_i^T\mathbf{X}_i)^{-1}\mathbf{X}_i^T\mathbf{Y} \tag{7}$$

[Step 6-2] *In case of a FPN (FPN-based layer)*
i) Simplified inference
The consequence part of the simplified inference mechanism is a constant. Using information granulation, the new rules read in the form

$$R^n : If\ x_1\ is\ A_{n1}\ and\ \cdots\ and\ x_k\ is\ A_{nk}\ then\ y_n - M_n = a_{n0} \tag{8}$$

$$\hat{y} = \frac{\sum_{j=1}^{n} \mu_{ji} y_i}{\sum_{i=1}^{n} \mu_{ji}} = \frac{\sum_{j=1}^{n} \mu_{ji}(a_{j0} + M)}{\sum_{i=1}^{n} \mu_{ji}} = \sum_{j=1}^{n} \hat{\mu}_{ji}(a_{j0} + M_j) \tag{9}$$

$$\mu_{ji} = A_{j1}(x_{1i}) \wedge \cdots \wedge A_{jk}(x_{ki}) \tag{10}$$

The consequence parameters (a_{j0}) are produced by the standard least squares method.
ii) Regression polynomial inference
The use of the regression polynomial inference method gives rise to the expression.

$$\begin{aligned} &R^n : If\ x_1\ is\ A_{n1}\ and \cdots and\ x_k\ is\ A_{nk}\ then \\ &\quad y_n - M_n = f_n\{(x_1 - v_{n1}), (x_2 - v_{n2}), \cdots, (x_k - v_{nk})\} \end{aligned} \tag{11}$$

$$\begin{aligned} \hat{y}_i &= \frac{\sum_{j=1}^{n} \mu_{ji} y_i}{\sum_{j=1}^{n} \mu_{ji}} = \frac{\sum_{j=1}^{n} \mu_{ji}\{a_{j0} + a_{j1}(x_{1i} - v_{j1}) + \cdots + a_{jk}(x_{ki} - v_{jk}) + M_j\}}{\sum_{i=1}^{n} \mu_{ji}} \\ &= \sum_{j=1}^{n} \hat{\mu}_{ji}\{a_{j0} + a_{j1}(x_{1i} - v_{j1}) + \cdots + a_{jk}(x_{ki} - v_{jk}) + M_j\} \end{aligned} \tag{12}$$

The coefficients of consequence part of fuzzy rules obtained by least square method(LSE) as like a simplified inference.

[Step 7] *Select nodes (PNs or FPNs) with the best predictive capability and construct their corresponding layer.*
All nodes of this layer of the IG_gHSOFPNN are constructed genetically. To evaluate the performance of PNs or FPNs constructed using the training dataset, the testing dataset is used. Based on this performance index, we calculate the fitness function. The fitness function reads as

$$F(fitness\ Function) = \frac{1}{1 + EPI} \tag{13}$$

where *EPI* denotes the performance index for the testing data (or validation data).

[Step 8] *Check the termination criterion.*
As far as the depth of the network is concerned, the generation process is stopped at a depth of less than three layers. This size of the network has been experimentally found to build a sound compromise between the high accuracy of the resulting model and its complexity as well as generalization abilities.
In this study, we use a measure (performance indexes) that is the Mean Squared Error (MSE)

[Step 9] *Determine new input variables for the next layer.*
The outputs of the preserved nodes (z_{1i}, z_{2i}, …, z_{Wi}) serves as new inputs to the next layer (x_{1j}, x_{2j}, …, x_{Wj})(j=i+1). This is captured by the expression

$$x_{1j} = z_{1i},\ x_{2j} = z_{2i},\ \ldots,\ x_{Wj} = z_{Wi} \tag{14}$$

The IG_gHSOFPNN algorithm is carried out by repeating steps 4-9 of the algorithm.

5 Simulation Study

To demonstrate the high modeling accuracy of the proposed model, we apply it to a highly nonlinear of pH neutralization of a weak acid and a strong based [8], [9], [10]. pH is the measurement of the acidity or alkalinity of a solution containing a proportion of water. We consider 500 pairs of the original input-output data. Total data are used as learning set. Table 2 summarizes the list of parameters used in the genetic optimization of the networks.

Table 2. Computational overhead and a list of parameters of the GAs and the HSOFPNN

	Parameters	1st layer	2nd layer	3rd layer
GAs	Maximum generation	100	100	100
	Total population size	150	150	150
	Selected population size	30	30	30
	Crossover rate	0.65	0.65	0.65
	Mutation rate	0.1	0.1	0.1
	String length	3+3+30+5+1	3+3+30+5	3+3+30+5
HSOF PNN	Maximal no. of inputs to be selected(Max)	$1 \le l \le$ Max(2~3)	$1 \le l \le$ Max(2~3)	$1 \le l \le$ Max(2~3)
	Polynomial type (Type T) of the consequent part of fuzzy rules	1≤T_F≤4	1≤T_P≤3	1≤T_P≤3
	Membership Function (MF) type	Triangular Gaussian		
	No. of MFs per each input(M)	2 or 3		

l, T_F, T_P : integer, T_F : Type of SOFPNN, T_P : Type of SOPNN.

Table 3 summarizes the results: According to the information of Table 2, the selected input variables (Node), the selected polynomial type (T), the selected no. of MFs (M), and its corresponding performance index (PI) was shown when the genetic optimization for each layer was carried out.

Table 3. Performance index of IG_gHSOFPNN for nonlinear function process

Max	1st layer				2nd layer			3rd layer		
	Node(M)	T	MF	PI	Node	T	PI	Node	T	PI
	(a) In case of selected input									
2	5(3) 6(3)	3	T	0.036102	18 19	2	0.030198	15 30	2	0.025244
3	1(2) 2(3) 3(3)	3	G	0.000148	7 10 27	2	0.000137	7 11 20	2	0.000133
	(b) In case of entire system input									
2	3(3) 6(2)	3	T	0.000132	16 22	2	0.000131	4 20	2	0.000130
3	3(3) 6(2) 0	3	T	0.000132	14 18 23	3	0.000130	11 26 30	2	0.000130

In case of entire system input, the result for network in the 3rd layer is obtained when using Max=2 with Type 2 polynomials (quadratic functions) and 2 node at input (node numbers are 4, 20); this network comes with the value of PI=0.000130.

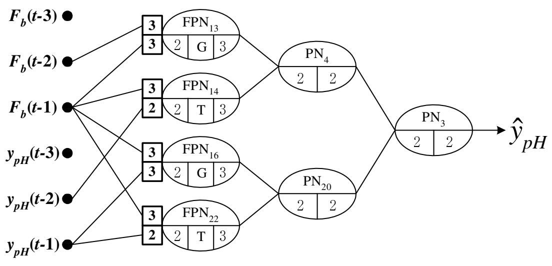
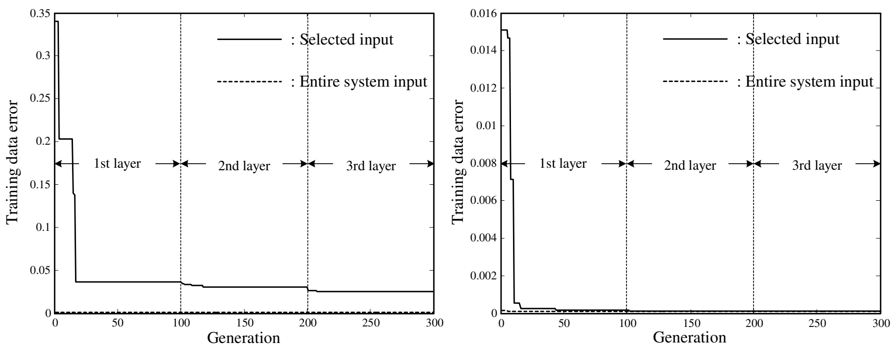


Fig. 4. Optimal IG_gHSOFPNN architecture

Fig. 5 illustrates differences in learning observed between selected input and entire system input by visualizing the values of the performance index obtained in successive generations of GA when using Max=2 and Max=3.



(a) Max=2 (b) Max=3

Fig. 5. Optimal procedure by IG and GAs

Table 4. Comparative analysis of the performance of the network; considered are models reported in the literature

Model			Performance Index
Nie's model [11]	USOCPN		0.230
	SSOCPN		0.012
SOPNN [2]	Basic SOPNN (15th layer)	Case 1	0.0015
		Case 2	0.0052
	Modified SOPNN (10th layer)	Case 1	0.0039
		Case 2	0.0124
Our model	Selected input	3rd layer (Max=2)	0.025244
		3rd layer (Max=3)	0.000133
	Entire system input	3rd layer (Max=2)	0.000130
		3rd layer (Max=3)	0.000130

Table 4 includes a comparative analysis of the performance of the proposed network with other models.

6 Concluding Remarks

In this paper, we have introduced and investigated a class of Information Granulation based genetically optimized Hybrid Self-Organizing Fuzzy Polynomial Neural Networks (IG_gHSOFPNN) driven to genetic optimization and information granulation regarded as a modeling vehicle for nonlinear and complex systems.

The GA-based design procedure applied at each stage (layer) of the HSOFPNN driven to information granulation leads to the selection of the preferred nodes (or FPNs and PNs) with optimal local. These options contribute to the flexibility of the resulting architecture of the network.

Through the proposed framework of genetic optimization we can efficiently search for the optimal network architecture (being both structurally and parametrically optimized) and this design facet becomes crucial in improving the overall performance of the resulting model.

Acknowledgement. This work was supported by the Korea Research Foundation Grant funded by the Korean Government(MOEHRD)(KRF-2005-041-D00713)

References

1. Ivakhnenko, A.G.: Polynomial theory of complex systems. IEEE Trans. on Systems, Man and Cybernetics. SMC-**1** (1971) 364-378
2. Oh, S.K., Pedrycz, W.: The design of self-organizing Polynomial Neural Networks. Information Science. **141** (2002) 237-258
3. Park, H.S., Park, K.J., Lee, D.Y, Oh, S.K.: Advanced Self-Organizing Neural Networks Based on Competitive Fuzzy Polynomial Neurons. Transactions of The Korean Institute of Electrical Engineers. **53D** (2004) 135-144
4. Oh, S.K., Pedrycz, W., Park, H.S.: Multi-layer hybrid fuzzy polynomial neural networks: a design in the framework of computational intelligence. Neurocomputing. **64** (2005) 397-431
5. Jong, D.K.A.: Are Genetic Algorithms Function Optimizers?. Parallel Problem Solving from Nature 2, Manner, R. and Manderick, B. eds., North-Holland, Amsterdam (1992)
6. Zadeh, L.A., et al.: Fuzzy Sets and Applications: Selected Paper. Wiley, New York (1987)
7. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. New York. Plenum (1981)
8. Hall, R.C., Seberg, D.E.: Modeling and Self-Tuning Control of a Multivariable pH Neutrallization Process. Proc. ACC. (1989) 1822-1827
9. McAvoy, T.J.: Time optimal and Ziegler-Nichols control. Ind.Eng.Chem.Process Des. Develop. **11** (1972) 71-78
10. Pajunen, G.A.: Comparison of linear and nonlinear adaptive control of a pH-process. IEEE Control Systems Maganize. (1987) 39-44
11. Nie, J., Loh, A.P., Hang, C.C.: Modeling pH neutralization processes using fuzzy-neurla approaches. Fuzzy Sets and Systems. **78** (1999) 5-22

Neuro-fuzzy Modeling and Fuzzy Rule Extraction Applied to Conflict Management

Thando Tettey and Tshilidzi Marwala

University of the Witwatersrand
School of Electrical and Information Engineering
Private Bag 3 Wits 2050,
Johannesburg, South Africa
{t.tettey, t.marwala}@ee.wits.ac.za

Abstract. This paper outlines all the computational methods which have been applied to the conflict management. A survey of all the pertinent literature relating to conflict management is also presented. The paper then introduces the Takagi-Sugeno fuzzy model for the analysis of interstate conflict. It is found that using interstate variables as inputs, the Takagi-Sugeno fuzzy model is able to forecast conflict cases with an accuracy of 80.36%. Furthermore, it found that the fuzzy model offers high levels of transparency in the form of fuzzy rules. It is then shown how these rules can be translated in order to validate the fuzzy model. The Takagi-Sugeno model is found to be suitable for interstate modeling as it demonstrates good forecasting ability while offering a transparent interpretation of the modeled rules.

1 Introduction

With the frequency at which wars are occurring, it has become imperative that more research effort be directed towards conflict management. The focus of this study over the years has been centered on finding improved approaches to conflict forecasting at the same time not neglecting causal intepretation of interstate interactions [1]. Understanding the reasons why countries go to war is just as significant as forecasting the onset of war because it proposes the steps that can be taken to avoid conflict. Therefore a successful interstate conflict tool is one which is able to forecast dispute outcomes fairly accurately at the same time allowing for an intuitive causal intepretation of interstate interactions.

International conflict has been studied using mainly techniques found in the fields of Statistics and Computational intelligence. These techniques have been applied on quantitative measures which have been collected over the years. It is widely accepted that improvements in conflict forecasting can mainly be acheived in two ways [1]. The first improvement that can be made is with the data and measures of interstate interactions [2] and secondly by improving the forecasting of interstate conflict involves finding models which approximate interstate interactions better.

In this paper a survey of the work performed on international conflict is presented. The paper focuses mainly on the techniques applied to analyse quantitative descriptions of interstate interactions. An outline of the methods together with their shortcomings is discussed. Finally, neuro-fuzzy modeling is presented as an alternative technique which addresses the current shortcomings that exist in international conflict studies.

2 Background and Literature Survey

Militarised interstate dispute (MID) is defined as a set of interactions between or among states that can result in the actual use, display or threat of using military force in an explicit way [3]. Projects such as the Correlates of War (COW) facilitate the collection, dissemination and use of accurate and reliable quantitative data in international relations [4]. The collected data, called interstate variables, are used to study the conditions associated with MID. The measures used in MID studies are *People Oriented Government, Dependency, Capability, Alliance, Contiguity, Distance* and *Major power*. Any set of measures describing a particular context has a dispute outcome attached to it. The dispute outcome is either a peace or conflict situation.

Statistical methods such as logit and probit have been used in the analysis of interstate variables. However, it has been found that these methods have several shortcomings [1]. Some of the problems associated with the use of logit and probit is that they require the use of *a priori* knowledge usually obtained from the analyst. A problem then arises when the analyst pushes their data analyses extremely hard in search of effects they believe exist but are difficult to discover [1]. The consequence of this is that the results vary from researcher to researcher and are therefore not exactly repeatable. The other problem, as one might expect, is that conflict cases occur far less frequently than peace cases. Interstate conflict is therefore a rare event and the processes which drive it are likely to be different from those found elsewhere. This has led quantitative researchers to conclude that the relationship between the interstate variables and dispute outcomes is highly nonlinear and highly correlated [1]. The conclusion is further confirmed by the studies performed by Lagazio and Russet [5]. This means that statistical techniques, linear-normal models in particular, would perform poorly at modeling the relationship between interstate disputes and their outcomes.

The neural network has also been applied to interstate conflict modeling and forecasting. The neural network was first introducted by Schrodt [6] in 1995 and by Zeng [7] in 1999 as a method of analysing conflict without the need for the researcher to incorporate qualitative *a priori* knowlege or make assumptions about the problem space. The neural network was presented as a function approximator which is able to model highly nonlinear and interdependant relationships. However the neural network itself suffers similar problems to statistical methods in that a model selection technique must be considered. In recent studies, Beck *et al* [1] make use of a neural network, which is trained using the Bayesian framework outlined in [8]. The Bayesian training of neural networks involves the use of Bayesian framework to identify the optimal weights and biases in a neural network model. It is found that the use of neural networks yields results expressed in the form of classification accuracy. This interpretation of the results is found to be unambigious compared to previous methods. However, the resulting neural network model is regarded as a black box due to the fact that it does not provide a way of obtaining a causal interpretion of dispute outcomes. The weights extracted from the neural network offer no understanding as to why countries go to war.

In [9], Marwala and Lagazio propose the use of Automatic Relevance Detection (ARD) as a means to making the neural network more transparent. The result of ARD reveals that the importance of the interstate variable in predicting dispute outcomes is as follows (listed in decreasing relavance): *Peoples Oriented Government, Capability, Dependancy, Allies, Contiguity, Distance* and *Major power*. From this work on neural networks we can conclude that neural network models have a fairly strong forecasting ability but only a limited amount of knowledge can be extracted.

In [10], Habtemariam *et al* introduce support vector machines (SVMs) to the study of conflict management. It is found that SVMs offer an improved forecasting ability over neural networks. However, a sensitivity analysis which aims to determine the influence of each variable on a dispute outcome reveals that results obtained from neural networks are much more intuitive. Therefore, while SVMs offer better forecasting ability they lack the ability to give an intuitive causal interpretation of the results.

As stated earlier on in the paper, the main focus of studies in international conflict has been on the ability of a model to accurately forecast dispute outcomes while at the same time allow the analyst to extract knowledge from the model. In the next section a Neuro-fuzzy model is proposed as a method of modeling interstate interaction which has a fairly accurate forecasting ability and at the same time offers intuitive causal explainations of disputes obtained from a fuzzy-rule extraction process.

3 Fuzzy Systems and Neuro-fuzzy Modeling

Fuzzy logic concepts provide a method of modelling imprecise models of reasoning, such as common sense reasoning, for uncertain and complex processes. Fuzzy set theory resembles human reasoning in its use of approximate information and uncertainty to generate decisions. In fuzzy systems, the evaluation of the output is performed by a computing framework called the *fuzzy inference system*. The fuzzy inference system is a computing framework that maps fuzzy or crisp inputs to the output - which is usually a fuzzy set [11]. The inference system performs a composition of the inputs using fuzzy set theory, fuzzy *if-then* rules and fuzzy reasoning to arrive at the output. More specifically, the fuzzy inference involves the fuzzification of the input variables, evaluation of rules, aggregation of the rule outputs and finally the defuzzification of the result. The are two popular fuzzy models: the Mamdani model and the Takagi-Sugeno (TS) model [12].

A fuzzy rule-based system can be viewed as a layered network similar to Radial basis function (RBF) artificial neural networks [12]. When training RBF networks several parameters of the kernel have to be optimised. Similarly when setting up a Fuzzy rule-based system we are required to optimise parameters such as membership functions and consequent parameters. In order to optimise these parameters, the neuro-fuzzy system relies on training algorithms inherited from artificial neural networks such as gradient descent [12]. There are two approaches to training neuro-fuzzy models [12]:

1. Fuzzy rules may be extracted from expert knowledge and used to create an initial model. The parameters of the model can then be fine tuned using data collected from the operational system being modelled.
2. The number of rules can be determined from collected numerial data using a model selection technique. The parameters of the model are also optimised using the existing data. The Takagi-Sugeno model is mostpopular when it comes to data-driven identification and has been proven to be a universal approximator [11].

The major motivation for using a neuro-fuzzy model in this work is that it is considered by many to be a 'gray-box' [12]. Unlike neural networks, once the neuro-fuzzy model has been optimised it is possible to extract the fuzzy rules and perhaps interpret the obtained results in an intuitive and qualitative manner. With neural networks e.g. the multilayer perceptron, it is not possible to intuitively explain input-output relationships using the weights of the network. In this study the neuro-fuzzy hybrid system is used to model the relationship between interstate (input) variables and the outcome of a dispute (output) from MID data.

4 Conflict Forecasting Using a Neuro-fuzzy Model

In our study, the TS neuro-fuzzy model has been optimised to map the relationship between the input variables (interstate variables) and the output i.e. the dispute outcome. The neuro-fuzzy model is trained using a balanced set of 500 peace cases and 500 conflict cases. The remaining 26845 peace instances and 392 conflict instance are then used for testing the forecasting ability of the model. The model selection process involves selecting the optimum number of rules of the TS model. This is done by creating models with rules ranging from 2 to 7 and evaluating them using 5 fold cross-validation. It is found that the optimum model consists of 2 fuzzy rules. The forecasting ability of the neuro-fuzzy model is then tested using the test examples. The prediction ability of the neuro-fuzzy model is evaluated based on how well it is able to predict both conflict and peace outcomes. The receiver operating characteristic (ROC) curve in fig 1 is used to illustrate the results.

As the output of the neuro-fuzzy is expressed as a decision value, an optimum threshold needs to be determined as the decision point which allows the overall

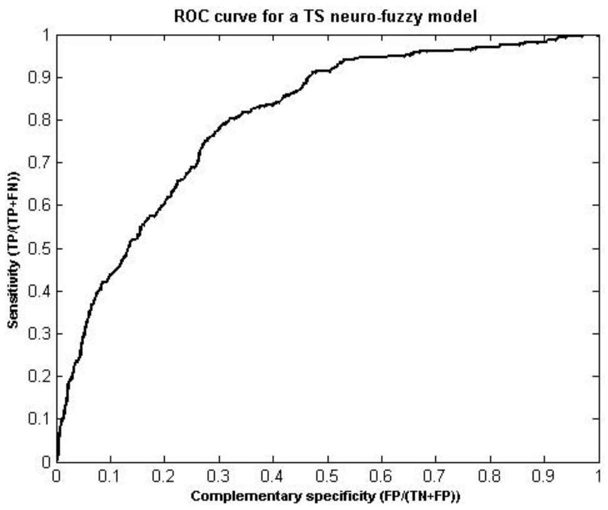


Fig. 1. A ROC curve illustrating the performance of the neuro-fuzzy model

Table 1. Results for the TS neuro-fuzzy model

	Conflict cases	Peace cases
Correctly predicted	315	17967
Incorrectly predicted	77	8378

maximum peace and conflict prediction accuracy. This threshold is found to be 0.488 and the results are expressed as a confusion matrix shown in Table 1.

The results show the TS neuro-fuzzy model predict conflict cases with an accuracy of 80.36% while predicting peace cases with an accuracy of 66.93%. In the work done by Habtemariam *et al* [10], it was found that SVM predicts peace and conflict with 79% and 75%, respectively. From these results it is clear that the TS neuro-fuzzy model predicts conflict with a much higher accuracy but sacrifices its performance on peace outcomes. This biased result will be accepted as the prediction of conflict is considered more important than the prediction of peace.

5 Fuzzy Rule Extraction

The TS neuro-fuzzy model used for forecasting in the previous section can also be used for rule extraction. Two fuzzy rules can be extracted from the model and they are shown below.

1. **If** u_1 is A_{11} **and** u_2 is A_{12} **and** u_3 is A_{13} **and** u_4 is A_{14} **and** u_5 is A_{15} **and** u_6 is A_{16} **and** u_7 is A_{17} **then**
   $y(k) = -1.86 \cdot 10^{-1}u_1 - 1.33 \cdot 10^{-1}u_2 + 0.00 \cdot 10^{0}u_3 - 6.05 \cdot 10^{-1}u_4 - 1.26 \cdot 10^{-1}u_5 - 1.33 \cdot 10^{0}u_6 + 4.71 \cdot 10^{-1}u_7 + 8.95 \cdot 10^{-1}$

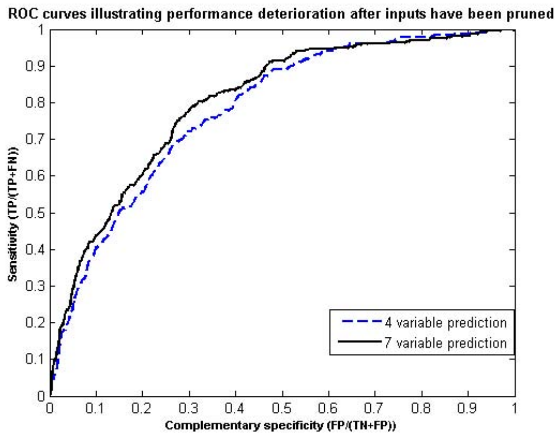


Fig. 2. A ROC illustrates the performance degradation of the neuro-fuzzy model when several inputs are pruned

2. **If** u_1 is A_{21} **and** u_2 is A_{22} **and** u_3 is A_{23} **and** u_4 is A_{24} **and** u_5 is A_{25} **and** u_6 is A_{26} **and** u_7 is A_{27} **then**
   $y(k) = -2.79 \cdot 10^{-1}u_1 + 6.26 \cdot 10^{-2}u_2 + 2.47 \cdot 10^{-1}u_3 - 7.56 \cdot 10^{-1}u_4 - 8.85 \cdot 10^{-1}u_5 - 9.04 \cdot 10^{0}u_6 + 0.00 \cdot 10^{0}u_7 + 3.73 \cdot 10^{-1}$

The symbols from u_1 to u_7 are the input vector which consists of *People oriented Government, Dependancy, Capability, Alliance, Contiguity, Distance* and *Major power*. The rest of the symbols are as previously defined.

It is clear that the rules are quite complex and need to be simplified in order to obtain a didactic interpretation. In fact it is often found that when automated techniques are applied to obtaining fuzzy models, unnecessary complexity is often present [13]. Setnes *et al* [13] present a similarity measure for fuzzy rule based simplification. Two methods of simplifying rules that are proposed are the removal and/or merging of similar fuzzy sets. Removal of a fuzzy set is proposed in the event that it is similar to the universal set i.e. $\mu \approx 1$ and the merging of fuzzy sets is proposed in the event that fuzzy sets from different rules but belonging to the same premise are similar.

In our case the TS fuzzy model contains only two fuzzy rules. The removal of a fuzzy sets similar to the universal set leaves only one remaining fuzzy set. This results in the input being partitioned into only one fuzzy set and therefore introduces difficulty when expressing the premise in linguistic terms. To simplify the fuzzy rules and avoid the redundant fuzzy sets the number of inputs into the TS neuro-fuzzy model have been pruned down to four variables. These variables are *People oriented Government, Dependancy, Alliance* and *Contiguity*. Fig 2 illustrates how the output deteriorates when three of the inputs are pruned. The ROC curve shows that the performance degradation is minimal.

The rules extracted can then be converted so that they are represented in the commonly used linguistic terms. However it is only possible to translate the antecedent of the fuzzy statement into english. The consequent part together with the firing strength of the rule are still expressed mathematically. The translated fuzzy rules with the firing strengths omitted can be written as shown below.

1. **If** Government orienting towards people is low **and** Alliance is strong **and** Contiguity is true **then**
   $y(k) = -3.87 \cdot 10^{-1}u_1 - 9.19 \cdot 10^{-1}u_3 - 7.95 \cdot 10^{-1}u_4 + 3.90 \cdot 10^{-1}$

2. **If** Government orienting towards people is high **and** Alliance is weak **and** Contiguity is false **then**
   $y(k) = -1.25 \cdot 10^{-1}u_1 - 5.62 \cdot 10^{-1}u_3 - 2.35 \cdot 10^{-1}u_4 + 4.23 \cdot 10^{-1}$

To validate the model we can then apply expert knowledge of the problem domain. For instance if the level of *people oriented Government* of two countries is *low*, they have a *weak* alliance and they *share a border* there is a reasonable chance that the countries can find themselves in a conflict situation. If we find values of *Majority rule, Alliance* and *Contiguity* which have a membership value of one, we can then use these as inputs to the model to see if it confirms our existing knowledge. It is found that by using these values and and an arbitrary *Dependency* value the model gives a prediction of 0.6743 which is above the conflict threshold of 0.5360 calculated from the ROC curve. By validating the model with similar statements, we can get a feel for how much confidence we can put in the system. The neuro-fuzzy model therefore offers an improved method of forecasting international conflict as it allows for accurate prediction of dispute outcomes while also catering for the cases where causal interpretations are required.

6 Conclusion and Recommendataion

A background on conflict management has been presented in the form of a literature survey. The methods that have been used in the past to predict interstate disputed have been highlighted. The performance criteria that is expected of a forecasting model has been stated i.e. a satisfactory model must be able to accurately predict dispute outcome and at the same time be able to give a causal explaination of the results. It has been found that previous models have either lacked a sufficient prediction ability or transparency. A Takagi-Sugeno fuzzy model which is trained using concepts from neural network learning is then proposed. It is shown that the TS model is able to predict conflict cases with an accuracy of 80.36%. Further, the TS model is expressed as a set of fuzzy rules which are made readable by expressing them using common linguistic terms. The TS model therefore is able to meet the specified peformance criteria.

However, we recommend the Takagi-Sugeno model for prediction only as it is able to predict the conflict cases accurately. The transparency of the fuzzy model allows for improved validation and therefore increases the chances of user acceptance. It is not advisable for the system to be used to source expert knowledge because of its accuracy. A prediction accuracy of 80.36% is considered good especially on a rare-event prediction problem. However, this means that under certain conditions the fuzzy rules may offer incorrect knowledge. To increse the validity of the model we also recommend that the way some of the interstate variables are expressed be reviewed. For instance, variable such as contiguity have values of either true or false. In the fuzzy domain these are considered exceptions and are commonly termed as fuzzy singleton. Therefore, expressing these fuzzy sets as Gaussian membership functions might introduce flaws into the model.

References

1. N. Beck, G.K., Zeng, L.: Improving quantitative studies of international conflict: A conjecture. American Political Science Review **94**(1) (2000) 21–35
2. Jones, D., Bremer, S., Singer, J.: Militarized interstate disputes, 1816-1992 rationale, coding rules and empirical patterns. Conflict Management and Peace Science **15**(2) (1996) 585–615
3. Gochman, C., Maoz, Z.: Militarized interstate disputes 1816-1976. Conflict resolution **28**(4) (1984) 585–615
4. : Correlates of war. (Internet Listing) URL: `http://www.correlatesofwar.org/`.
5. Lagazio, M., Russett, B.: A Neural Network Analysis of Militarised Disputes, 1885-1992: Temporal Stability and Causal Complexity. In: Temporal Stability and Causal Complexity. University of Michigan Press, New Jersey (2003) 28 –62
6. Schrodt, P.: Patterns, rules and learning: Computational models of international behaviour. (Unpublished manuscript available at) URL: `http://www.ku.edu/`*sim*`keds/papers.dir/Schrodt.PRL.2.0.pdf`.
7. Oneal, J., Russet, B.: Prediction and classification with neural network models. Sociological Methods and Research **4**(3) (1999) 499–524
8. Mackay, D.: A practical bayesian framework for backpropagation networks. Neural Computing **4**(3) (1992) 448–472
9. Marwala, T., Lagazio, M.: Modeling and controlling interstate conflict. In: Proceedings of the IEEE International Joint Conference on Neural Networks, Budapest, Hungary, IEEE (2004) 1233–1238
10. Habtemariam, E.A., Marwala, T.: Artificial intelligence for conflict management. In: Proceedings of the IEEE International Joint Conference on Neural Networks, Montreal, Canada, IEEE (2005) 2583–2588
11. Jang, J., Sun, C., Mizutani, E.: Neuro-fuzzy and soft computing: A computational approach to Learning and Machine Intelligence. First edn. Prentice Hall, New Jersey (1997)
12. Babuska, R., Verbruggen, H.: Neuro-fuzzy methods for nonlinear sytem identification. Annual Reviews in Control **27** (2003) 73–85
13. Sentes, M., ska, R.B., Kaymak, U., van Nauta Lemke, H.: Similarity measures is fuzzy rule based simplification. IEEE Transactions on Systems, Man and Cybernatics-Part B: Cybernatics **28**(3) (1998) 376–386

Hardware Implementation of a Wavelet Neural Network Using FPGAs

Ali Karabıyık and Aydoğan Savran

Ege University, Department of Electrical and Electronics Engineering,
35100, İzmir, Turkey
alikarabiyik@yahoo.com, aydogan.savran@ege.edu.tr

Abstract. In this paper, hardware implementation of a wavelet neural network (WNN) is described. The WNN is developed in MATLAB and implemented on a Field-Programmable Gate Array (FPGA) device. The structure of the WNN is similar to the radial basis function (RBF) network, except that here the radial basis functions are replaced by orthonormal scaling functions. The training of the WNN is simplified due to the orthonormal properties of the scaling functions. The performances of the proposed WNN are tested by applying for the function approximation, system identification and the classification problems. Because of their parallel processing properties, the FPGAs provide good alternative in real-time applications of the WNN. By means of the simple scaling function used in the WNN architecture, it can be favorable to multilayer feedforward neural network and the RBF Networks implemented on the FPGA devices.

1 Introduction

Recently, the WNN's have become a popular tool in networks. Wavelets have been applied successfully to multi scale time-frequency analysis and synthesis in signal processing, function approximation and fault detection [4],[5],[6]. The multi layer perceptron (MLP) with the back-propagation (BP) training algorithm is the mostly used type of neural network in practical application. But this structure is complicated due to multilayer structure and its convergence is too slow. The RBF network has a simpler structure (one hidden layer) than the MLP. The training of the RBF networks can be obtained much easier than the MLP networks by preprocessing the training data, such as clustering, However, due to the basis functions in the RBF, the RBF network representation is not unique and not the most efficient for a given function [3]. A WNN is obtained by using wavelet scaling functions as the basis functions [3]. Unlike the basis functions in RBF [6], the scaling functions have orthonormal properties. Therefore, the wavelet neural networks have improved training performances and convergence rates [7].

In real-time applications, parallel implementation of the neural networks is an important property. The FPGA consists of many logic gates. We can enter a description of our logic designs using a hardware description language (HDL) such as VHDL or VERİLOG. These devices allow to implement neural network designs in parallel architectures in order to use for real-time applications [9-10].

In this paper, an FPGA implementation of a WNN consisting of the haar wavelet function [8] is described. The performance of the WNN is tested by applying for the function approximation, the system identification and classification problems.

The remaining of the paper is organized as follows. In section 2, a brief review of the wavelets is given. The architecture of the wavelet neural network with haar scaling function is described in section 3. In section 4, the steps of the implementation of the wavelet neural network in the FPGA are given. The simulation and the FPGA implementation results are given in section 5. Section 6 concludes the paper.

2 Wavelets

A wavelet is a base (basis) function which has the ability to allow simultaneous time and frequency analysis. Wavelets have adjustable and adaptable parameters so that they are ideal for adaptive systems to give a tool for the analysis of transient, nonstationary, time-varying phenomena [1].

The wavelets are functions whose dilations and translations form a orthonormal basis of $L^2(R)$, the space of all square integrable functions on R. There exists a function Ψ(t) (the "mother wavelet") such that

$$\Psi_{m,n}(t) = 2^{m/2}\Psi(2^m t - n) \tag{1}$$

form an orthonormal basis of $L^2(R)$. Therefore, the wavelet basis induces an orthogonal decomposition of $L^2(R)$.

$$L^2(R) = \bigoplus_m W_m \tag{2}$$

where W_m is a subspace spanned by

$$\left\{ 2^{m/2}\Psi(2^m t - n) \right\}_{n=-\infty}^{n=+\infty} \tag{3}$$

The wavelet Ψ(t) is often generated from a companion Φ(t) known as the scaling function (the "father wavelet") and V_m be the subspace spanned by

$$\Phi_{m,n}(t) = 2^{m/2}\Phi(2^m t - n)_{n=-\infty}^{n=+\infty} \tag{4}$$

The relation between W_m and V_m is

$$\begin{aligned} &W_i \perp V_i \\ &V_{i+1} = V_i \oplus W_i \end{aligned} \tag{5}$$

The dilations and translations of the scaling function induce a multiresolution analysis (MRA) of $L^2(R)$:

$$\ldots\ldots. \subset V_{-1} \subset V_0 \subset V_1 \ldots\ldots\ldots\ldots.. \subset L^2(R) \tag{6}$$

$$\bigcap_m V_m = \{0\}, clos\{\bigcup_m V_m\} = L^2(R) \tag{7}$$

The Haar scaling and wavelet function [1] are shown in Fig.1.

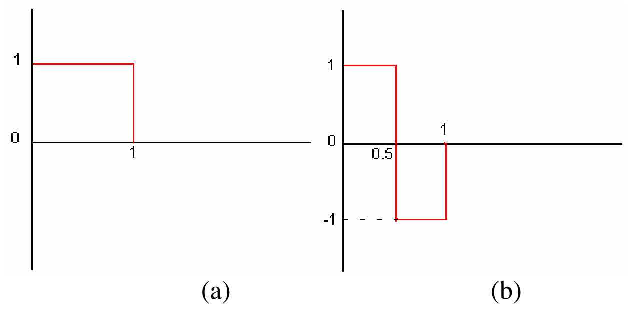


Fig. 1. Haar Wavelet and Scaling Functions; a)Scaling Function: b) Wavelet Function:

$f(x) \in L^2(R)$ can be extract inverse wavelet transform:

$$f(t) = \sum_{m,n} \langle f, \Psi_{m,n} \rangle \Psi_{m,n}(t) \tag{8}$$

and

$$f(t) = \sum_{n} \langle f, \Phi_{m0,n} \rangle \Phi_{m0,n}(t) + \sum_{m \geq m0,n} \langle f, \Psi_{m,n} \rangle \Psi_{m,n}(t) \tag{9}$$

where <.,.> represents the inner product and $m0$ is an arbitrary integer, representing the lowest resolution or scale in the decomposition [1],[4],[6,[8]. In fact that, any $f(x) \in L^2(R)$ can be approximated arbitrarily closely in V_m, for some integer M. That is, for any $\in$ >0, there exists an M sufficiently large such that

$$\left\| f(t) - \sum_{n} \langle f, \Phi_{M,n} \rangle \Phi_{M,n}(t) \right\| \langle \in \tag{10}$$

In the d-dimensional analysis and synthesis, the scaling function is defined by the tensor products of one-dimensional scaling functions [2],[4],[6],[8]. For example, a d-dimensional scaling function can be generated by

$$\Phi_d(t) = \Phi_d(t_1, t_2, ...t_d) = \prod_{j=1}^{d} \Phi(t_j) \tag{11}$$

3 Wavelet Neural Network

In this section, the structure and training procedure of the wavelet neural network is described.

Let $f(t) \in L^2(R)$ be an arbitrary function which has to be approximated. $f(t)$ can be estimated by means of a set of given training data.

$$T_N = \{(t_i, f(t_i)\}_{j=1}^{N} \tag{12}$$

From the wavelet theory explained Section 2, $f(t)$ can be approximated with an arbitrarily accuracy by selecting a sufficiently large M such that

$$f(t) \cong \sum_k \langle f, \Phi_{M,k} \rangle \Phi_{M,k}(t) \tag{13}$$

$$= \sum_k c_k \Phi_{M,k}(t) \tag{14}$$

where

$$\Phi_{M,k}(t) = 2^{M/2} \Phi(2^M t - k) \tag{15}$$

The network structure is shown in Fig. 2. It is a three-layer network similar to the multilayer feedforward neural networks. The input layer is one node with input *x*. The hidden layer contains a countable number of nodes indexed by *k*, and the threshold for the k-th node of the hidden layer is *k*. The weights and nonlinearities of the hidden-layer nodes are identical, i.e., 2^M and $\Phi(x)$, respectively. The weights of output layer are c_k and which has one linear node. The hidden nodes run from –K to K for some positive integer K [3-4-6-8].

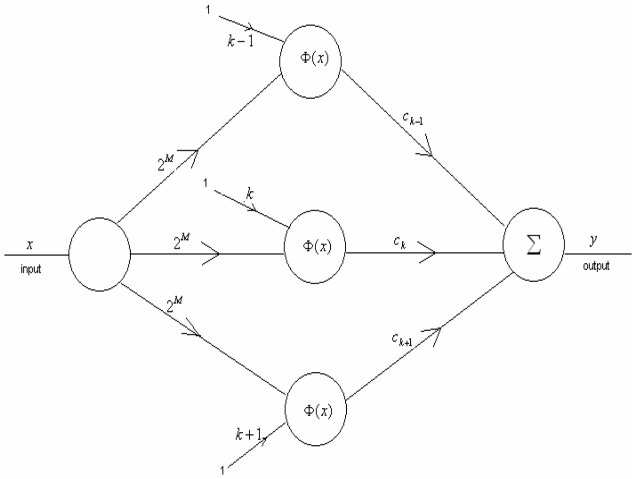


Fig. 2. The Wavelet Neural Network

For a given set of M and K, the WNN described above implements the function $g(t)$

$$g(t) = \sum_{k=-K}^{K} c_k \Phi_{M,k}(t) \tag{16}$$

In this equation, if c_k are properly selected, the $g(t)$ could be used to approximate $f(t)$.

In WNN, the number of hidden nodes is, for a given resolution M, determined by the input signal range which is $2(2^M+p)$ for 1-dimensional case. If we use d-dimensional, it will be $2^d(2^M+p)^d$ [8].

$$p \geq 1 \tag{17}$$

If the input signal range lies in the interval [-1,1], the number of hidden nodes is $2(2^M+1)$ for 1-dimensional case.

With BP training algorithm, when the training data set T_N is avaible, c_k can be found by minimizing the mean square (training) error $e_N(f,g)$,

$$e_N(f,g) = \frac{1}{N}\sum_{i=1}^{N}(f(t_i) - g(t_i))^2 \tag{18}$$

[3] and N is the length of training data. For minimizing the mean square error, c_k can be updated by

$$c_k(t+1) = c_k(t) - \lambda \frac{\partial e_N(f,g)}{\partial c_k} \Big|_{C_k(t)} \tag{19}$$

for some λ, $0\langle\lambda\langle 1$,
where λ is learning rate.

4 Implementation of Wavelet Neural Network in FPGA

The FPGA implementation of the WNN, which is firstly simulated with MATLAB, is performed by using the VHDL (Very High Speed Integrated Circuit Hardware Description Language). The choice of VHDL is justified by its flexibility and the fact that it is a standard language (IEEE 1076). Furthermore, it is a target device independent designing tool, i.e. with one design description many device architectures can be included. After learning process, the WNN is implemented in FPGA devices for parallel processing.

The data types of the WNN's weights, bias coefficients and inputs have to be changed from real type into binary. The latter can be used immediately in the VHDL. To be more specific, it can be easily represented as std_logic_vector type, which is the most suitable data type for digital processing in an FPGA. If the input and coefficients lie in the interval [-5,5], which can be represented with 10 bit binary number.

The structure of WNN in FPGA consists of three units.

1.Input Layer-Coder: The input signal has to be changed from real type into binary. The wide of binary number is generally 10 bit. For this purpose, we described a look-up-table (LUT). If the input signals lie in the interval [-5,5], the resolution of 10 bit binary number is:

$$resolution = \frac{5-(-5)}{2^{10}} = 0.009765 \tag{20}$$

2. Hidden Layer-Neuron Structure: The neuron structure in hidden layer consists of multiplier, accumulator and activation function which are described by VHDL. The input signal is multiplied with input weights by using multiplier. The output of multiplier is summed with bias coefficient to give the net results for activation function. The activation function is described by LUT. The Haar scaling function is used in this work. Every neurons in the hidden layer work parallel at the same clock signal.

3.Output Layer-Decoder: The output of the activation functions are weighted by multiplying with the output weights. In the output layer, the output of each hidden layer neurons are summed by accumulator and then the results are transferred from binary digit to the decimal digit by using LUT in order to show on the displays of the FPGA card.

We enter describe our network using VHDL. Implementation on an FPGA usually requires a number of steps. After using a logic synthesizer program to transform the HDL into netlist, the WNN structure is designed by various logic gates. And then, we use the implementation tools to map the logic gates and interconnections into the FPGA. The logic operations are performed by using configurable logic blocks in the FPGA. Ones the implementation phase is completed, a program extracts the state of the switches in the routing matrices and generates a bitstream where the ones and zeroes correspond to open or closed switches. And finally, the bitstream is downloaded into a physical FPGA chip.

We use the parallel port of host computer for data transfer to the designed WNN in the FPGA. The input of WNN is carried out in the Pentium computer using MATLAB from parallel port of host computer and then the output of WNN can be seen on FPGA displays.

5 Experimental Results

The examples in this section are simulated using MATLAB. Then, the results obtained from the FPGA are compared with the simulation results. The FPGA board is Xilinx Spartan 2E XC2S200E which has 200.000 gates, 50 MHz oscillator, RS-232 serial port, parallel port, seven segment and LED-pushbutton for basic I/O.

Example 1: The first example is the XOR problem. In this problem, the output is one when the inputs are different, otherwise it is zero. The WNN has 2-inputs, 16 neurons in the hidden layer and 1 output. The WNN is scaled with M=0. In 1000 epochs, the sum-squared error between the desired output and the output of WNN is 10^{-8}. After training process, the structure is implied on the FPGA. The parallel port is used to data transfer between the host computer and the FPGA. The output of the system is displayed on the 7-segment displays of FPGA. The sum-squared error through the training process, the desired-the WNN output (blue line) and the WNN output (red line) obtained in the FPGA are shown in Fig. 3.

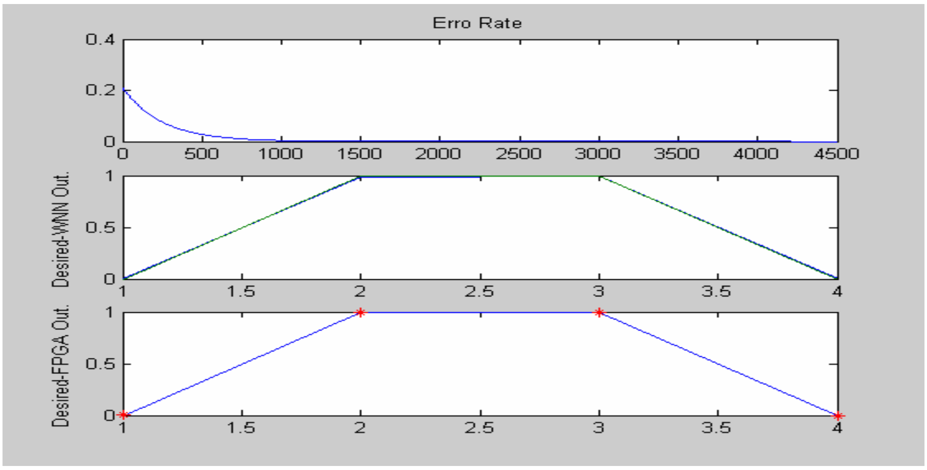


Fig. 3. Simulation -FPGA Results for XOR

Example 2: This example illustrates a classification problem which has two-dimensional input and one-dimensional output. Where the inputs are

$$xd=[-3\ -2\ -2\ 0\ 0\ 0\ 0\ 2\ 2\ 3;\quad 0\ 1\ -1\ 2\ 1\ -1\ -2\ 1\ -1\ 0]; \tag{21}$$

and the output is

$$d=[0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0] \tag{22}$$

The WNN is described by 2-dimensional inputs, 16 neurons in the hidden layer and 1-dimensional output in MATLAB. WNN is scaled with M=0, the sum-squared error is 10^{-8} in 344 epoch and the sum-square error and the desired-WNN output (green) in MATLAB and the FPGA output (red) are given in Fig. 4.

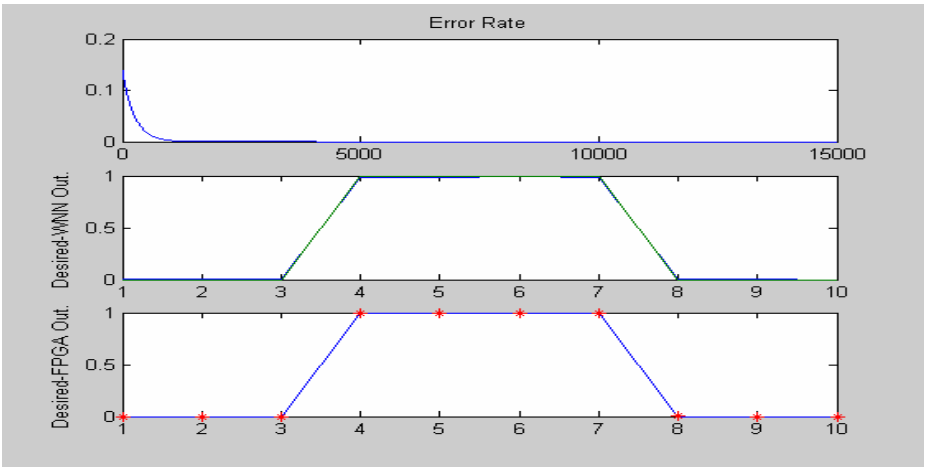


Fig. 4. Simulation-FPGA Results for Classification

Example 3: We consider a function approximation problem which is expressed as [8]

$$f\left[y_p(k)\right] = \frac{y_p(k)}{y_p^2(k)+1} \tag{23}$$

where $y_p(k)$ is the output of system. The system is described by the first order difference equation

$$y_p(k+1) = 0.5y_p(k) + u(k) \tag{24}$$

where $u(k) = 0.2\sin(2\pi\frac{k}{25}) + 0.3\sin(2\pi\frac{k}{10})$. The WNN is described by 1-dimensional input, 36 neurons in the hidden layer and 1-dimensional output in the MATLAB taking M=4. In 300 epochs, the sum-squared error between the desired output and the output of the WNN is 10^{-8}. The sum-square error in learning phase, the WNN output in the FPGA and the MATLAB, are given Fig. 5, respectively.

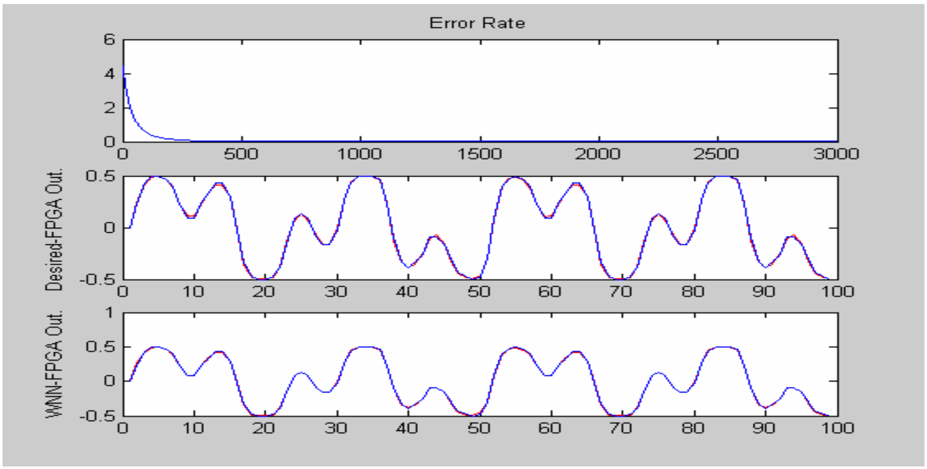


Fig. 5. Results for the function approximation application

Example 4: In this example, the identification of a nonlinear system is considered. It is given as

$$y(t) = 0.5y(t-1) + u(t-1).81 + 0.2y(t-1)^2] + u(t-1)^3 \tag{25}$$

where, *u* and *y* is the input and output variable of the system, respectively. The inputs are generated randomly in the interval [-0.5,0.5]. The WNN has 2 inputs, 100 neurons in the hidden layer and 1 output. The WNN are trained with 2000 data samples and test with 200 data samples. Taking *M*=2, the the sum-squared error between the desired output and the output of the WNN decreased to 2.10^{-4} in the 40 epochs,. The sum-square error , the desired and FPGA outputs, and the WNN outputs for MATLAB simulation and the FPGA implementation are all given in Fig. 6.

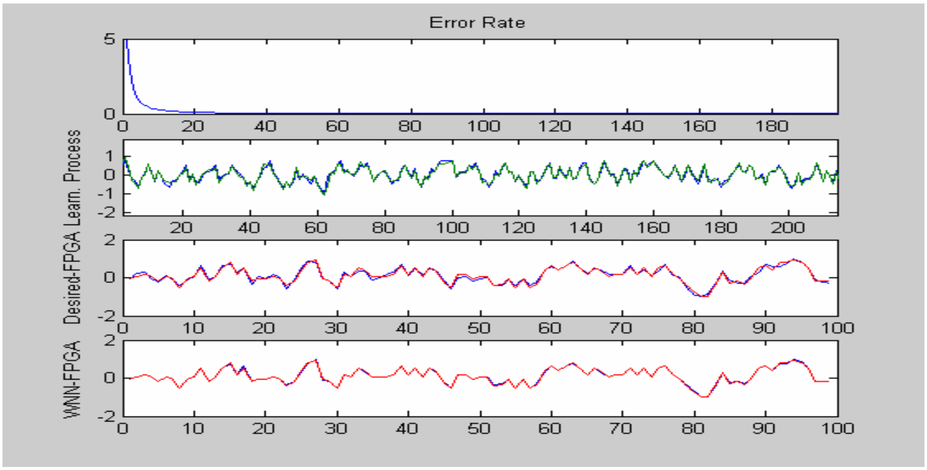


Fig. 6. Results for the system identification application

6 Conclusions

In this paper, a wavelet neural network implemented in an FPGA device has been described. The Haar wavelet function which has a simple structure has been used to simplify the FPGA implementation and form an orthonormal basis.

The good performances for the WNN are obtained by applying for the classification, the function approximation, and the system identification problems.

As a future work, the online training of the WNN in the FPGA will be studied

References

1. Raghuveer M. Rao, Ajit S. Bopardikar, "Wavelet Transform, Introduction to Theory and Applications,".
2. Yuehui Chen, Bo Yang, Jiwen Dong, "Time-series prediction using a local linear wavelet," 2005 Elsevier Science B.V.
3. C. Chen, C. Mo, "A method for intelligent fault diagnosis of rotating machinery", Elsevier Science, Digital Signal Processing 14, pp. 203-217,2004.
4. Dongbing Gu, Huosheng Hu, "Neural predictive control for a car like mobile robot", 2002 Elsevier Science, Robotics and Automation, vol. 39, pp. 73-86,2002.
5. X. Z. Wang, B. H. Chen, S. H. Yang, C. McGreavy,"Application of wavelet and neural networks to diaonostic system development, 2, an integrated framework and its application," 1999 Elsevier Science, Computer and Chemical Engineering, vol. 23, pp. 945-954, 1999.
6. Jun Zhang, Gilbert G. Walter, Yubo Miao, Wan Ngai Wayne Lee, "Wavelet Neural Networks for Function Learning", IEEE Transactions on Signal Processing, vol.43, no. 6, June 1995.

7. C. Yin, W. Guo, T Lin, S. Liu, R. Fu, Z. Pan, L. Wang, “Application of Wavelet Neural Network to the Prediction of Gas Chromatographic Retention Indices of Alkanes”, Journal of the Chinese Chemical Society, 48, pp.739-749, 2001.
8. “Sokho Chang, Seok Won Lee, Boo Hee Nam, “Identification and Control of Nonlinear Systems Using Haar Wavelet Networks”, Transaction on Control, Automation and Systems Engineering, vol.2, no. 3, September, 2000.
9. Serkan Ünsal, “Neural Chip Design using Field Programmable Gate Array”, Ege Uni. Electric&Electronic Engineering Department, Master Thesis, 2003
10. Y. Taright, M. Hubin, “FPGA implementation of a multilayer perceptron neural network using VHDL”, ICSP’98, pp.1311-1314.

Neural Network Implementation in Hardware Using FPGAs

Suhap Sahin, Yasar Becerikli*, and Suleyman Yazici

Department of Computer Eng., Kocaeli University, Izmit ,Turkey
suhapsahin@kou.edu.tr, ybecerikli@kou.edu.tr, syazici@kou.edu.tr

Abstract. The usage of the FPGA (Field Programmable Gate Array) for neural network implementation provides flexibility in programmable systems. For the neural network based instrument prototype in real time application, conventional specific VLSI neural chip design suffers the limitation in time and cost. With low precision artificial neural network design, FPGAs have higher speed and smaller size for real time application than the VLSI design. In addition, artificial neural network based on FPGAs has fairly achieved with classification application. The programmability of reconfigurable FPGAs yields the availability of fast special purpose hardware for wide applications. Its programmability could set the conditions to explore new neural network algorithms and problems of a scale that would not be feasible with conventional processor. The goal of this work is to realize the hardware implementation of neural network using FPGAs. Digital system architecture is presented using Very High Speed Integrated Circuits Hardware Description Language (VHDL) and is implemented in FPGA chip. The design was tested on a FPGA demo board.

1 Introduction

Artificial Neural Networks (ANNs) can solve great variety of problems in areas of pattern recognition, image processing and medical diagnostic. The biologically inspired ANNs are parallel and distributed information processing systems. This system requires the massive parallel computation. Thus, the high speed operation in real time applications can be achieved only if the networks are implemented using parallel hardware architecture [1].

Implementation of ANNs falls into two categories: Software implementation and hardware implementation. ANNs are implemented in software, and are trained and simulated on general-purpose sequential computers for emulating a wide range of neural networks models. Software implementations offer flexibility. However hardware implementations are essential for applicability and for taking the advantage of ANN's inherent parallelism [2]. Specific-purpose fixed hardware implementations (i.e. VLSI) are dedicated to a specific ANN model. VLSI implementations of ANNs provide high speed in real time applications and compactness. However, they lack flexibility for structural modification and are prohibitively costly.

* Corresponding author. ybecerikli@kou.edu.tr, ybecer@ieee.org

We are interested in building a different class of hardware environment, i.e. FPGA-based reconfigurable computing environment for implementing ANNs. FPGA offer speed comparable to dedicated and fixed hardware systems for parallel algorithm acceleration, while as with a software implementation, retaining a high degree of flexibility for device reconfiguration as the application demands [3]. With the introduction of FPGAs, it is feasible to provide custom hardware for application specific computation design. The changes in designs in FPGAs can be accomplished within a few hours, and thus result in significant savings in cost and design cycle. A method of implementing a fully connected feed forward network with Xilinx FPGAs for image processing that the single processing node was partitioned into two XC3090 chips is proposed [4]. A neural associative memories implementation based RAMs and XC3090 FPGAs is reported [2].

This paper explores that how to efficiently use 32 bit floating-point numeric representation in FPGA based ANNs. By making use of the features of SpartanIIE series FPGAs. A VHDL library was designed for using ANN's on FPGAs. The library supports to the IEEE-754 standards for single-precision (32-bit) floating point arithmetic, and it is referred to fp_lib.

2 Artificial Neural Network

The concept of ANNs is emerged from the principles of brain that are adapted to digital computers. The first works of ANNs were the models of neurons in brain using mathematics rule [5]. These works show that each neuron in ANNs take some information as an input from another neuron or from an external input. This information is propagated as an output that are computed as weighted sum of inputs and applied as non-linear function.

Architectural ANNs parameters such as number of inputs per neuron and each neuron’s conductivity change remarkably from application to application. Thus, for special purpose network architectures parameters must be carefully balanced for efficient implementation.

It is apparent that there are three kinds of parallelism to explain within ANNs when carefully exanimate to the data flow and structure of ANNs. The first is spatial parallelism i.e. every neuron in the same layer runs simultaneously. The second is algorithmic parallelism that is related to the formulation of the algorithm itself. In addition, computation on successive layers can be pipelined [3].

3 Field Programmable Gate Arrays and Very-High Hardware Description Language

FPGAs consist of three basic blocks that are configurable logic blocks, in-out blocks and connection blocks. Logic blocks perform logic function. Connection blocks connect logic blocks with in-out blocks. These structures consist of routing channels and programmable switches. Routing process is effectively connection logic blocks exist different distance the others [6].

FPGAs are chosen for implementation ANNs with the following reason:

- They can be applied a wide range of logic gates starting with tens of thousands up to few millions gates.
- They can be reconfigured to change logic function while resident in the system.
- FPGAs have short design cycle that leads to fairly inexpensive logic design.
- FPGAs have parallelism in their nature. Thus, they have parallel computing environment and allows logic cycle design to work parallel.
- They have powerful design, programming and syntheses tools.

The architecture of ANNs must be specified with schematic or algorithmic at first step of FPGAs based system design. When ANNs based FPGAs system design specify the architecture of ANNs from a symbolic level. This level allows us using VHDL which stands for VHSIC (Very High Speed Integrated Circuit) Hardware Programming Language [7]. VHDL allows many levels of abstractions, and permits accurate description of electronic components ranging from simple logic gates to microprocessors. VHDL have tools needed for description and simulation which leads to a lower production cost.

4 Data Representation

There are two problems during the hardware implementation of ANNs. How to balance between the need of reasonable precision (number of bit), that is important for ANN and the cost of more logic area associated with increased precision. How to choose a suitable number format that dynamic range is large enough to guarantee that saturation will not occur for a general-purpose application. So before beginning ANN's based FPGAs system design with VHDL, number format (floating point, fixed point etc.) and precision which used for inputs, weighs and activation function must be considered. This important that precision of the numbers must be as high as possible are used during training phase. Because, precision has a great impact in the learning phase [9]. However low precision is used during the propagation phase [10]. So especially in classification's applications the resulting errors will be small enough to be neglected [10,5,11].

Floating point offers the greatest amount of dynamic range, making it suitable for any application so it would be the ideal number format to use. The objective of this paper is to determinate feasibility of 32 bit floating point arithmetic in FPGAs based ANNs.

5 Application

In this section FPGA based ANN's architecture, works and system results is represented. The application is implementing fully parallel neural network in FPGA. The network is implemented in Xilinx Spartan IIE chip consist of 200000 typical gates, 2352 slices.

5.1 Arithmetic Architecture for ANN's

The first work must be, trained ANN's is mapped on FPGA in application phase. So ANN's architecture was developed using VHDL with 32 bit floating point arithmetic. Because of floating point have greatest amount of dynamic range for any applications. Unfortunately, there is currently no clear support for floating-point arithmetic in VHDL [4, 12]. As a result, a VHDL library was designed for using ANN's on FPGAs. The library supports to the IEEE-754 standards for single-precision (32-bit) floating point arithmetic, and it is referred to fp_lib. The fp_lib has tree separate library, for floating point addition fp_add, floating point subtraction fp_sub and floating point multiplication fp_mul.

The single precision floating point numeric representation supports to IEEE-754 standard shown in Figure 1.

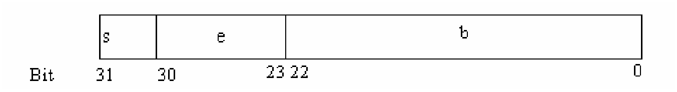


Fig. 1. 32 bit Floating Point Format

The floating point number (n) is computed by:

$$n = -1^{s} 2^{|e-127|} (1.b) \tag{1}$$

In Figure 1, sign field is referred to 's' is bit 31 and is used to specify the sign of the number. Exponent field is referred to 'e' is bits 30 down to 23 are the exponent field. The bias of 127 is used. Because of 8 bit quantity is a signed number representation. To store binary representation (b) of floating point number bits 22 down to 0 are used. The leading one in the mantissa is implicit. So the mantissa is (1.b).

5.2 Network Architecture

By using of the FPGA features hardware implementation of fully parallel ANN's is possible. In the fully parallel ANN's architecture number of multipliers per neuron equals to number of connections to this neuron and number of the full adders equals to number of connections to the previous layer mines one [9]. For example in 2-4-1 network output neuron have 4 multipliers and 3 adders. In this work a VHDL library were designed for floating point addition fp_add and floating point multiplication fp_mul. But most resources of FPGAs are used by multiplication and addition algorithm. So in fully parallel ANN's must be used low number precision (for example 8 bit). With the low number precision fully parallel network is not suitable for any application. With the using fp_lib (32 bit floating point number precision)in ANN's is suitable for any application. But the architecture has one multipliers and one adders per layer and is not full parallel because of area resource of FPGAs.

In this structure there is one multiplier and one adder per layer. The inputs from previous layer enter the layer parallel and multiplier serially with their corresponding weights. The results of multiplication are stored in their neuron area in the addition storage ROM. Multiplied value of per neuron are inputs for adder. The inputs of adder are added serially and each addition are inputs for sigmoid lookup table. The results of look up table are stored for next layer. This ANNs architecture is shown in Figure 2. In this design number of layer and number of neuron are changed easily during the working phase.

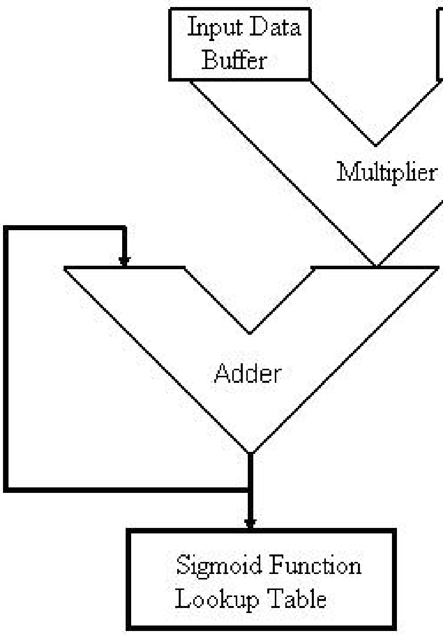


Fig. 2. Block diagram of ANNs

5.3 Modeling Tree Layer (2-3-1) ANNs

ANNs consist of input layer, one hidden layer and output layer as shown in Table 1. Sigmoid function is used as an activation function. Sigmoid function input vector consist of 100 value from -10 to 10 by chosen 0.2 step size. Results of sigmoid function are stored in the ROM

Weights are using this application are shown in Table 1.

Table 1. Weights Used in the Application

w111	w121	w131	w112	w122	w132
0.5	1	0.5	1	0.5	1
w211	w212	w213			
0.5	1	0.5			

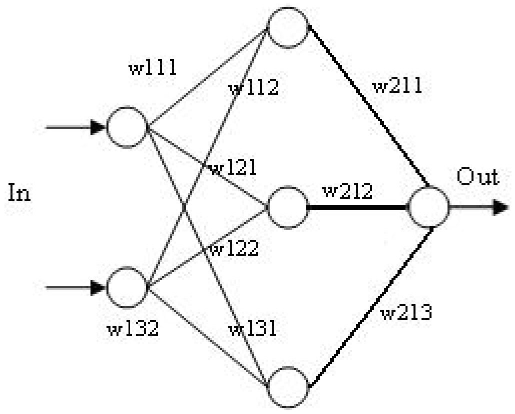


Fig. 3. A three layer MLP

5.4 Tree Layer (2-3-2) ANNs Architecture

In our design the external inputs entered the first layer serially. Input value and tree control signal that are start signal (1 bit), finish signal (1 bit) and neuron count signal (4 bit) must be entered. Before entered input value start signal must be set and after

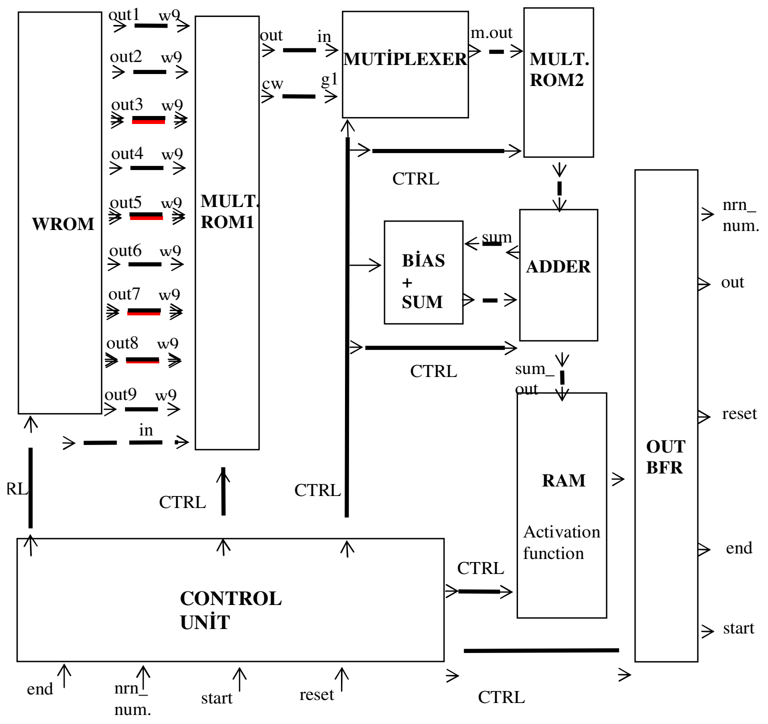


Fig. 4. Block diagram of the reconfigurable ANN

the entered input value finish signal must be set. If finish signal must be set system calculate entered input value number. So, the input number depends on the ones. Weight number is multiplied of input number in the BFR and neuron number.

Then first input, its corresponding and weights are stored in MULT_ROM1 and they are multiplied serially. The results are stored in area of first neuron in MULT_ROM2. The same process is repeat for other inputs. The value and bias in the same neuron area are added serially. The results are input for sigmoid look-up table. Look-up table outputs are stored OUT_BFR. The value that is this layer outputs must be next layer inputs. This working is controlled by control unit. Control unit controls time and makes several signals (for example enable signal) for other unit (see .Fig. 4)

6 Implementation Results

Digilentic demo board is used for implementation. The board has Xlinx Spartan II 2s200epq208-6 and 50 MHz clock. Spartan II chip has 2352 slices and 14 block RAM. VHDL libraries that it is referred to fp_lib were designed for using ANN's on FPGAs. The *fp_lib* has tree separate library are shown in Table 2. The comparison of FPGA based tree layer (2-3-1) ANNs and software based tree layer (2-3-1) ANNs are shown in Table 3 with inputs g1 = 0.5 and g2 = -0.25.

Table 2. Summary of custom arithmetic VHDL libraries

HDL Design	**Description**
fp_lib	IEEE 32-bit single precision floating point library
fp_mul	IEEE 32-bit single precision floating point piplined parallel multiplier
fp_add	IEEE 32-bit single precision floating point piplined parallel adder
Log_rom	IEEE 32-bit single precision floating point Single Port Block Memory

Table 3. Comparison of FPGA base ANNs and software based ANNs

	Software based ANNs	**FPGA based ANNs**	**ERROR**
g1=0.5 g2=-0.25	2.3274321322	2.268109	0.059323

7 Conclusions

In general, it is shown that implementation of neural networks using FPGAs. The resultant neural networks are modular, compact, and efficient and the number of neurons, number of hidden layers and number of inputs are easily changed.

The choice of dynamic range, precision and arithmetic hardware architecture used in neural networks application has a direct impact on the processing density achieved. Using suitable precision arithmetic design, one can be achieve adequately high speed and small size for real time ANNs implementations.

However this study shows that FPGAs are versatile devices for implementing many different applications. The VHDL-FPGA combination is shown to be a very powerful embedded system design tool, with low cost, reliability, and multi-faceted applications. As FPGAs allow the hardware design via configuration software control, the improvement of circuitry design is just a matter of modifying, debugging and downloading the new configuration code in a short time.

References

[1] POLIAC M., ZANETTI J., SALERNO D. 1993. Performance Mesuraments of Seismocardiogram Interpretation Using Neural Networks, Computer in Cardiology, IEEE Computer Society, pp 573-576

[2] RUCKET, U., FUNKE, A. and PINTASKE, C. 1993 Acceleratorboard for Neural Associative Memories, Neurocomputing, Vol.5, No.1, pp 39-49.

[3] ZHU, J., GUNTHER, B.K.,1999. Towards an FPGA Based Reconfigurable Computing Environment for Neural Network Implementations, Proceedings of the Ninth International Conference on Artificial Neural Networks (ICANN'99). In IEE Conference Proceedings 470, pp.661-667, IEE.

[4] COX, C., BLANZ, W. 1992. GABGLION-A Fast Field Programmable Gate Array Implementation of a Connectionist Classifier, IEEE Journal of Solid-satate Circuits, Vol.27, No.3, pp 288-299.

[5] HAYKIN, S. 1999. Neural Networks A Comprehensive Foundation. 2nd edition, Prentice Hall Publishing, New Jersey 07458, USA, Vol.1, pp 6-7

[6] BROWN, S.D., FRANCIS, R.J., VRANESIC Z.G. 1992. Field Programmable Gate Arrays, Kluwer Academics Publishers

[7] ASHEDEN, P., J. September-October 2001, VHDL Standards, IEEE Design & Test of Computers, vol. 18, n. 6, pp. 122-123.

[8] SAVRAN A., ÜNSAL S., "Hardware Implementation of a Feed forward Neural Network Using FPGAs", The third International Conference on Electrical and Electronics Engineering (ELECO 2003), 3-7 December, Bursa, Turkey, 2003.

[9] STEVENSON, M., WEINTER, R. and WIDOW, B. 1990 Sensitivity of Feedforward Neural Networks to Weigh Errors, IEEE Transactions on Neural Networks, Vol.1, No 2, pp71-80

[10] BLAKE, J.J., MAGUIRE, L.P., MCGINNITY, T.M., ROCHE, B., MCDAID, L.J. 1998. The Implementation of Fuzzy Systems, Neural Networks using FPGAs, Information Sciences, Vol. 112, pp. 151-168

[11] KRIPS, M., LAMMERT T., and KUMMERT, A. 2002, FPGA Implementation of a Neural Network for a Real-Time Hand Tracking System, Proceedings of first IEEE Internaional Workshop on Electronic Design, Test and Applications.

[12] YU X., DENI D. 1994. Implementing Neural Networks In FPGAs, The Institution of Electrical Engineers, IEE published, Savoy Place, London WC2R 0BL, UK.

FPGA Discrete Wavelet Transform Encoder/Decoder Implementation

Pedro Henrique Cox[1] and Aparecido Augusto de Carvalho[2]

[1] Fundacão Universidade Federal de Mato Grosso do Sul, Cidade Universitária, 79070-900 Campo Grande MS, Brazil
phcox@del.ufms.br
[2] Universidade do Estado de São Paulo, Faculdade de Engenharia de Ilha Solteira, 15385-000 Ilha Solteira SP, Brazil
aac@dee.feis.unesp.br

Abstract. In a multi-input an multi-output feedforward wavelet neural network, orthogonal wavelet basis functions are used as activate function instead of sigmoid function of feedforward network. This paper adresses the solution on processing biological data such as cardiac beats, audio and ultrasonic range, calculating wavelet coefficients in real time, with processor clock running at frequency of present ASIC's and FPGA. The Paralell Filter Architecture for DWT has been improved, calculating wavelet coefficients in real time with hardware reduced up to 60%. The new architecture, which also processes IDWT, is implemented with the Radix-2 or the Booth-Wallace Constant multipliers. One integrated circuit Encoder/Decoder, ultrasonic range, is presented.

1 Introduction

In artificial neural networks, wavelet series has the ability of functional approximation. The Discrete Wavelet Transform (DWT) is a good tool for time-frequency localization. Combining the localization feature of DWT and the self-modification function of Wavelet Neural Network (WNN), parameters of dyadic wavelet functions are modified through training [5]. The DWT algorithm [1,2,6] provides efficient multi-resolution subband coding representation in the time-scale plane. In each step, the signal is high-pass and low-pass filtered. An algorithm for the calculation of 1-D DWT is proposed [1]. In this algorithm, DWT coefficients in one level are calculated with DWT coefficients of the previous level. The input data sequence l^0 has $N_0 = p2^J$ samples, where p is an integer and J is the number of levels of the transform. Each decomposition level j, $1 \leq j \leq J$, can be seen as the further decomposition of the sequence l^{j-1}, which has N_{j-1} samples, into two subbands l^j and h^j, both with $N_j = N_{j-1} / 2$ samples. Such a decomposition is produced by two convolutions followed by a decimation by two. In equation (1), a_i and c_i denote coefficients on low-pass L_j and high-pass H_j, M tap filters, $l_n^j = 0$ for $n < 0$ and $n \geq J$.

$$h_n^j = \sum_{i=0}^{M-1} c_i \cdot l_{2n-i}^{j-1} \qquad 0 \leq n \leq N_j - 1 \tag{1}$$

$$l_n^j = \sum_{i=0}^{M-1} a_i \cdot l_{2n-i}^{j-1} \qquad 0 \le n \le N_j - 1 \quad . \tag{2}$$

For computing the DWT coefficients of the input discrete-time data, it is assumed that the input data represents DWT coefficients of a higher resolution level. Coefficients of subsequent levels are obtained from equations (1) and (2). Hence, DWT extracts information from the signal at different scales. The first level of wavelet decomposition extracts the high-frequency components of the signal, while the second and all subsequent wavelet decompositions extract, progressively, lower frequency components. A few levels are enough to have a good approximation of the signal with discrete wavelet coefficients. Four level 1-D DWT with low-pass eight order filter wavelet coefficients are presented in equation (3). Numerical equations for high-pass direct and 1-D IDWT inverse filters are obtained from equations (2), (4) and (5).

$$\begin{aligned} l_1(0) &= a_0 l_0(0) + a_1 l_0(-1) + a_2 l_0(-2) + a_3 l_0(-3) + \\ &\quad + a_4 l_0(-4) + a_5 l_0(-5) + a_6 l_0(-6) + a_7 l_0(-7) \end{aligned} \tag{3a}$$

$$\begin{aligned} l_1(2) &= a_0 l_0(2) + a_1 l_0(1) + a_2 l_0(0) + a_3 l_0(-1) + \\ &\quad + a_4 l_0(-2) + a_5 l_0(-3) + a_6 l_0(-4) + a_7 l_0(-5) \end{aligned} \tag{3b}$$

$$\begin{aligned} l_1(4) &= a_0 l_0(4) + a_1 l_0(3) + a_2 l_0(2) + a_3 l_0(1) + \\ &\quad + a_4 l_0(0) + a_5 l_0(-1) + a_6 l_0(-2) + a_7 l_0(-3) \end{aligned} \tag{3c}$$

$$\begin{aligned} l_1(6) &= a_0 l_0(6) + a_1 l_0(5) + a_2 l_0(4) + a_3 l_0(3) + \\ &\quad + a_4 l_0(2) + a_5 l_0(1) + a_6 l_0(0) + a_7 l_0(-1) \end{aligned} \tag{3d}$$

$$\begin{aligned} l_1(8) &= a_0 l_0(8) + a_1 l_0(7) + a_2 l_0(6) + a_3 l_0(5) + \\ &\quad + a_4 l_0(4) + a_5 l_0(3) + a_6 l_0(2) + a_7 l_0(1) \end{aligned} \tag{3e}$$

$$\begin{aligned} l_1(10) &= a_0 l_0(10) + a_1 l_0(9) + a_2 l_0(8) + a_3 l_0(7) + \\ &\quad + a_4 l_0(6) + a_5 l_0(5) + a_6 l_0(4) + a_7 l_0(3) \end{aligned} \tag{3f}$$

$$\begin{aligned} l_1(12) &= a_0 l_0(12) + a_1 l_0(11) + a_2 l_0(10) + a_3 l_0(9) + \\ &\quad + a_4 l_0(8) + a_5 l_0(7) + a_6 l_0(6) + a_7 l_0(5) \end{aligned} \tag{3g}$$

$$\begin{aligned} l_1(14) &= a_0 l_0(14) + a_1 l_0(13) + a_2 l_0(12) + a_3 l_0(11) + \\ &\quad + a_4 l_0(10) + a_5 l_0(9) + a_6 l_0(8) + a_7 l_0(7) \end{aligned} \tag{3h}$$

$$\begin{aligned} l_2(0) &= a_0 l_1(0) + a_1 l_1(-2) + a_2 l_1(-4) + a_3 l_1(-6) + \\ &\quad + a_4 l_1(-8) + a_5 l_1(-10) + a_6 l_1(-12) + a_7 l_1(-14) \end{aligned} \tag{3i}$$

$$\begin{aligned} l_2(4) &= a_0 l_1(4) + a_1 l_1(2) + a_2 l_1(0) + a_3 l_1(-2) + \\ &\quad + a_4 l_1(-4) + a_5 l_1(-6) + a_6 l_1(-8) + a_7 l_1(-10) \end{aligned} \tag{3j}$$

$$\begin{aligned} l_2(8) &= a_0 l_1(8) + a_1 l_1(6) + a_2 l_1(4) + a_3 l_1(2) + \\ &\quad + a_4 l_1(0) + a_5 l_1(-2) + a_6 l_1(-4) + a_7 l_1(-6) \end{aligned} \tag{3k}$$

$$l_2(12) = a_0\, l_1(12) + a_1\, l_1(10) + a_2\, l_1(8) + a_3\, l_1(6) + + a_4\, l_1(4) + a_5\, l_1(2) + a_6\, l_1(0) + a_7\, l_1(-2) \quad (3l)$$

$$l_3(0) = a_0\, l_2(0) + a_1\, l_2(-4) + a_2\, l_2(-8) + a_3\, l_2(-12) + + a_4\, l_2(-16) + a_5\, l_2(-20) + a_6\, l_2(-24) + a_7\, l_2(-28) \quad (3m)$$

$$l_3(8) = a_0\, l_2(8) + a_1\, l_2(4) + a_2\, l_2(0) + a_3\, l_2(-4) + + a_4\, l_2(-8) + a_5\, l_2(-12) + a_6\, l_2(-16) + a_7\, l_2(-20) \quad (3n)$$

$$l_4(0) = a_0\, l_3(0) + a_1\, l_3(-8) + a_2\, l_3(-16) + a_3\, l_3(-24) + + a_4\, l_3(-32) + a_5\, l_3(-40) + a_6\, l_3(-48) + a_7\, l_3(-56) \quad . \quad (3o)$$

To reconstruct the analyzed signal, equations (4) and (5), 1-D DWT coefficients are upsampled and inverse filtered with transfer functions P_j and Q_j, coefficients sets a' and c',

$$p_n^j = \sum_{i=0}^{M/2-1} a'_{2i} \cdot p_{2n-i}^{j+1} + \sum_{i=0}^{M/2-1} c'_{2i} \cdot q_{2n-i}^{j+1} \qquad 1 \le n \le N_j \quad (4)$$

$$p_{n+1}^j = \sum_{i=0}^{M/2-1} a'_{2i+1} \cdot p_{2n-i}^{j+1} + \sum_{i=0}^{M/2-1} c'_{2i+1} \cdot q_{2n-i}^{j+1} \qquad 1 \le n \le N_j \ . \quad (5)$$

This analysis on the signal is done with power of two, or dyadic bands. For computing the DWT coefficients of the input discrete-time data, it is considered that the input data represents DWT coefficients of a higher resolution level. Coefficients of subsequent levels are obtained from equation (1). Hence, DWT extracts information from the signal at different scales. The first level of wavelet decomposition extracts the high-frequency components of the signal, while the second and all subsequent wavelet decompositions extract, progressively, lower frequency components. A few levels are enough to have a good approximation of the signal with discrete wavelet coefficients. It has a very wide array of applications such as biomedicine [3], signal processing, ultrasonic analysis [4], speech compression, numerical analysis, statistics, etc. One original Common Architecture for the DWT and the IDWT is designed. One logical circuit for data compression is presented. This circuit controls compression rates in compression schemes used with the Percent of Root-mean-square Difference (PRD) control index. With one DWT module, one IDWT module and delay memory, one Encoder/Coder is designed. It was simulated with three and four levels to evaluate precision for several lengths data samples and filter coefficients.

To process high frequency signals, combinational multipliers are required. The Booth multiplier implements a paralell serial multiplier with additional control logic. However, it requires less operations. One combinatorial version is designed, the proposed multiplier is faster, needs less area and is simple to implement. With this high efficiency element processor the Encoder/Decoder frequency range is extended to one eighth processor clock frequency. One four-level Asynchronous Folded

Paralell Filter Architecture (AFPFA) with the Radix-2 multiplier and eight filter coefficients is implemented in VHDL.

This work presents an original hardware, 1-D Encoder/Decoder, for VLSI and FPGA integrated circuits. The PRD quality criterion to evaluate precision on DWT and IDWT processing modules is one of the most widely adopted nowadays [13] in data compression algorithms. A wavelet coder/decoder general architecture presented in [12] has a low frequency response band and shows some graphics about results from simulations. Selected wavelet band synthesizers such as [10] employs a Digital Signal Processor board connected to a VXI standard interface to process power line frequency range signals with three levels DWT. The most important achievement in this work is the perfect synthesis architecture for DWT algorithm with any number of levels. Synthesized data is obtained with precision depending on the word length on filter coefficients and input data. This work is the solution to implement data compression algorithms with integrated circuits. Equations for perfect synthesis are implemented in DWT and IDWT algorithms on the Encoder/Decoder. The precision evaluation was made quantizing data and filter coefficients for n bits word processing, $n = 4 + 4i$, $1 \leq i \leq 7$. With synchronous input and constant processing elements, real time analysis and synthesis is assured for signal sampling in mega Hertz range.

2 Asynchronous Folded Paralell Filter Architecture

The Paralell Filter Architecture is optimal with respect to both area and computing time [6]. For each N data samples, N wavelet coefficients are output. It is an architecture that has simple register allocation scheme and two filters, with high processor efficiency. The proposed architecture has only one filter to calculate both low-pass and high-pass wavelet coefficients in each algorithm step. Real time transform is achieved with two clocks. The data sampling clock and the processor clock. The ratio between the two clocks is a real number, the new design (Fig. 1) employs an Asynchronous Control Circuit rather than the classical approach presented in [7]. With ACL, maximum sampling frequency is $f_p/2m$, where f_p is the processor clock frequency and m is the number of processor cycles in each step. Wavelet coefficients are obtained multiplying M samples by M coefficients in a M tap FIR digital filter. For each data sample, two wavelet coefficients are calculated and the result is output to a bidirectional bus with the Recursive Pyramid Algorithm (RPA) [8].

2.1 Timing

When computing DWT, the first octave are scheduled every even data sampling clock cycle 2k. Second octave computations are executed in clock cycles $4k + 1$. Third octave computations are done at $8k + 3$ clock cycles and final results, fourth octave computations, at $16k + 7$ clock cycles. The delay to present first results is the period, in data sampling clock cycles, to fill up CRB_3 for the first time, in addition to the number of periods to compute the next fourth level wavelet coefficient. First results were output with 71 data sampling clock cycles.

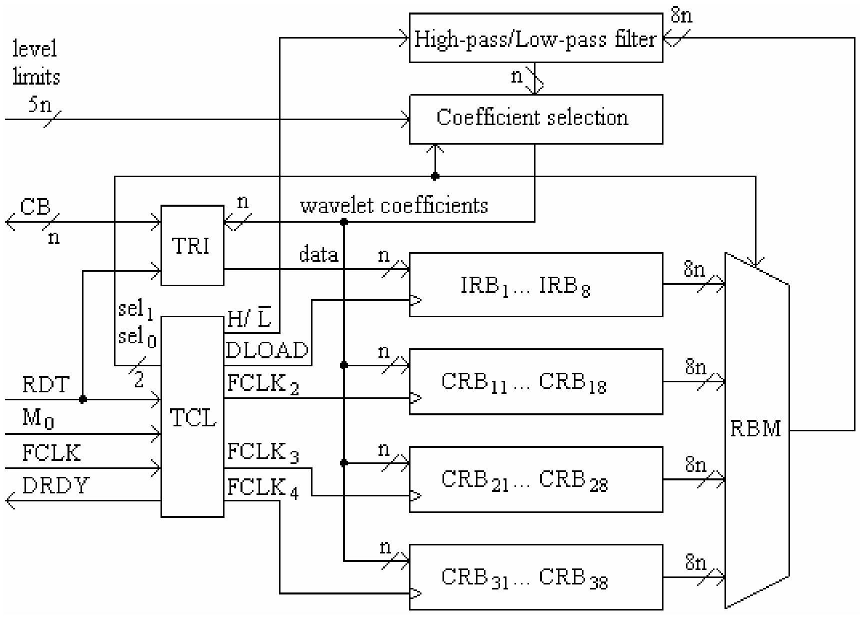


Fig. 1. Asynchronous Folded Paralell Filter Architecture with four levels

3 Common Architecture for Encoding and Decoding

To encode a signal with the Discrete Wavelet Transform consists on calculating two coefficients, outputs of a high-pass and a low-pass Finite Input Filter (FIR). To synthesize a signal consists on inverse filtering two output signals from the low-pass and high-pass FIRs, with two sets of odd and even inverse filter coefficients for each set of input samples. Slight modifications on AFPFA, such as splitting each coefficient register bank in a set of two, and on the synchronization, accessing twice the same set of coefficients to calculate subsequent reconstructed data, are necessary to implement the synthesis module. For inverse operation, the Inverse Recursive Pyramid Algorithm (IRPA) is used. The IRPA is structurally similar to the RPA. In each step, both wavelet coefficient sequences are upsampled inserting zeros. Inverse filtering an add operations are done with two sets of even and odd coefficients. Sets of registers IRB, CRB_1, CRB_2, CRB_3 for DWT are split in two for IDWT and multiplexers B_0, B_1, B_2, and B_3 are inserted (Fig. 2) to form the Common Architecture for DWT and IDWT.

When D=1, the outputs of BL_i are connected to the inputs of BH_i, i = 0, 1, …, 3, and eight data samples or low-pass wavelet coefficients from sets IRB, CRB_1, CRB_2 or CRB_3 in Figure 1 are selected on RBM multiplexer for high-pass or low-pass filtering.

When D=0, four high-pass wavelet coefficients from BH i and four low-pass reconstructed wavelet coefficients from BL_i, i = 0, 1, …, 3 are selected on RBM and multiplied by even and odd coefficient sets for synthesis wavelet coefficients and data. Table 1 presents filters coefficients sets selection with control lines D and F. The

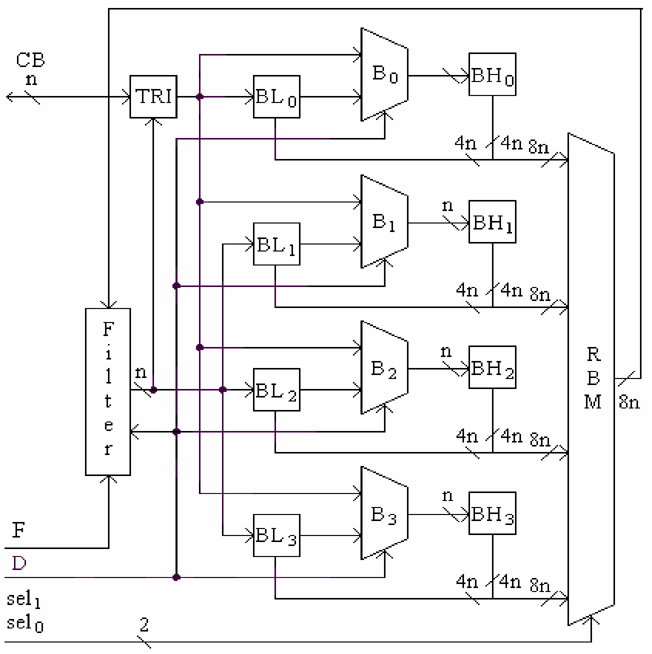


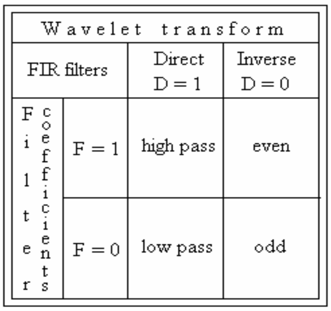
Fig. 2. Common Architecture for DWT and IDWT

control line D defines the architecture mode, Direct (DWT) or Inverse Direct Wavelet Transform (IDWT). The control line F chooses high-pass or low-pass filters in DWT mode, and even or odd inverse filtering coefficient sets in IDWT mode.

4 Experimental Results

FPGA prototyping tools reduces development time to a minimum. Reconfigurable processors are viable platforms for a broad range of specialized applications such as DWT algorithms. Other DWT algorithms have been implemented using CMOS technology [2,6] or DSP-based architecture [10].

Table 1. Filter coefficients sets



Wavelet transform			
FIR filters		Direct D = 1	Inverse D = 0
Filter coefficients	F = 1	high pass	even
	F = 0	low pass	odd

The AFPFA has been implemented in VHDL. Numerical equations define low-pass and high-pass eight order filter operations. The Radix-2 multiplier was implemented first, the 8 tap filter was developped next and one four level DWT algorithm was implemented.

4.1 DWT Algorithm

Control lines sel_1 sel_0 select one of the four register banks for filter input data in multiplexer RBM. Control lines $FCLK_2$, $FCLK_3$ and $FCLK_4$ stores wavelet coefficients at the end of each processor cycle (Fig. 3). External control line RDT access to send data samples to IRB or wavelet coeficients to CB (Fig. 1).

In this paper, we have presented the VHDL implementation of a DWT architecture for real time processing with minimum area. FPGAs like the ACEX EP1K50 have high density and speed to implement complex algorithms directly in hardware. The 8 bits four level VHDL AFPFA requires about 1630 logic cells (56%). The clock has been set to 30 MHz. The implementation defines bit to bit control lines, data buses and high level digital system design. DWT with different analytic wavelets is performed during operation. It was first developped for real-time analysis and compression of biological signals such as ECG. Due to its outstanding performance, AFPFA process audio and ultrasonic signals up to 450 KHz. To extend frequency response to MHz range, the Radix-2 multiplier is replaced by the Booth-Wallace Constant multiplier, one improved version of the Booth-Wallace Tree multiplier.

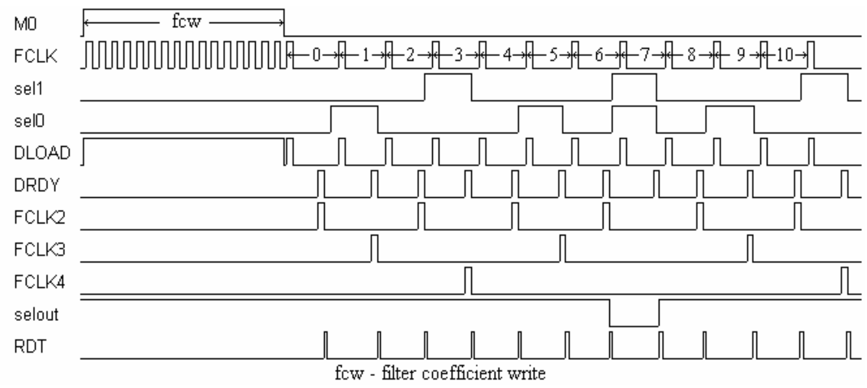


Fig. 3. DWT algorithm timing for filter coefficients write and first states 0 – 10

5 Conclusion

This article presents an original Common Architecture for DWT or IDWT, the Asynchronous Folded Paralell Filter Architecture and an Encoder/Decoder. In this work, the IRPA is implemented including output reconstructed data in the calculations with one processor.

In literature, from 1996 to 2000, we find only a few publications about DWT/IDWT hardware implementations.

The folded common architecture drawn in [11], scheduled with IRPA, requires twice the number of filters and buffers to calculate IDWT than AFPFA, presented in details. The VXI signal analyzer presented in [10] performs only power line analysis, with a DSP. The FPGA coder/decoder in [12] presents some simulations for DWT and IDWT algorithms.

The AFPFA, developped from 2002 to 2003 in São Paulo State University, Department of Electrical Engineering, Ilha Solteira, SP Brazil has a flexible design. The number of levels on DWT/IDWT is changed without affecting algorithm state chart. Only the size and the number of memory buffers is changed, the control logic is the same. The asynchronous feature improves processing speed on biological signals and audio. On an FPGA with Radix-2 processing elements, the analytical wavelet may be software configured. With this processor, the hardware required for a complete Encoder/Decoder is minimized. The implementation with Radix-2 multipliers reduces the hardware up to 60% the hardware on the implementation with Wallace multipliers. Depending on the signal frequency response, the required circuit area is reduced.

An example, for classifying ECG data, only one arithmetic unit implemented with Radix-2 processing elements is required, instead of four when implementing the Encoder/Decoder with the same number of levels with Booth-Wallace Constant processing elements, ultrasonic range. PRD precision for the four level integrated circuit is 0.043 % on 16 bits input data and filter coefficients. Table 2 presents the PRD index on synthesized data for 8, 12, 16, 20, 24, 28 and 32 bits input data and filter coefficients, fixed point processing elements. Two architectures are evaluated, one with three and other with four levels. Precision for DWT module only, fixed point processing elements, is performed synthesizing DWT wavelet coefficients with a floating point ALU in the IDWT module and then calculating the difference between original and reconstructed data. Three and four level IDWT precisions calculated this manner, in a 32 bits microcomputer, are 1.58 10^{-10} % and 2.39 10^{-10} % respectively. Each PRD measured index is the mean value for 20 ECGs.

Table 2. Encoder/Decoder precision evaluation

word length (bits)	Three levels		Four levels	
	DWT	DWT/IDWT	DWT	DWT/IDWT
8	4.59	5.63	4.65	5.95
12	4.24 10^{-1}	5.95 10^{-1}	4.01 10^{-1}	5.40 10^{-1}
16	3.02 10^{-2}	4.33 10^{-2}	2.89 10^{-2}	4.27 10^{-2}
20	1.84 10^{-3}	2.59 10^{-3}	1.69 10^{-3}	2.46 10^{-3}
24	1.13 10^{-4}	1.60 10^{-4}	1.09 10^{-4}	1.51 10^{-4}
28	6.96 10^{-6}	1.05 10^{-5}	6.85 10^{-6}	9.97 10^{-6}
32	4.63 10^{-7}	6.38 10^{-7}	4.48 10^{-7}	6.20 10^{-7}

References

1. Mallat S. G.: A theory for multiresolution signal decomposition: The wavelet representation. IEEE Trans. P. Anal. Machine Intell., vol. 2, pp. 674–693 (1989).
2. Mandal Mrinal K., Panchanathan Sethuraman: VLSI Implementation of Discrete Wavelet Transform. IEEE Trans. on VLSI Syst., vol. 4, n^o 4 (1996).
3. Lu, Zhitao, Kim, Pearlman, William A.: Wavelet Compression of ECG Signals by the Set Partitioning in Hierarchical Trees Algorithm. IEEE Trans. On Biomedical Eng. vol. 47, n^o 7 (2000).
4. Rajoub, Bashar A.: An Efficient Coding Algorithm for the Compression of ECG Signal Using the Wavelet Transform. IEEE Tr. on B. Eng. vol. 49, n^o 4 (2002).
5. Iyengar, S. S.: Foundations of wavelet network and applications. Chapman & Hall/CRC CRC Press LLC (2002).
6. Chakrabarti, Chaitali, Vishwnath, Mohan.: Efficient realizations of the Discrete and Continuous Wavelet Transforms: From Single Chip Implementations to Mappings on SIMD Array Computers. IEEE Trans. on Signal Processing, vol. 43, n^o 3 (1995).
7. Parhi, Keshab K.: Synthesis of Control Circuits in Folded Pipelined DSP Architectures. IEEE Journal of Solid-State Circuits, vol. 27, n^o 1 (1992).
8. Vishwanath, Mohan : The Recursive Pyramid Algorithm for the Discrete Wavelet Transform. IEEE Trans. on Signal Processing, vol. 42, n^o 3. (1994).
9. Huluta, E., Petriu, E. M., Das, S. R., Al-Dhaer, Abdul H.: Discrete Wavelet Transform Architecture Using Fast Processing Elements. IEEE Inst. and Meas. Techn. Conference (2002).
10. Angrisani, L., Daponte, P., D'Apuzzo, M., Pietrosanto, A.: A VXI Signal Analyzer based on the Wavelet Transform. IEEE Inst. and Meas. Techn. Conference, pp. 440-445, may (1997).
11. Vishwanath, M., Owens, R. M.: A Common Architecture For the DWT and IDWT. IEEE, pp. 193-198, (1996).
12. Habib, S. E.-D.: An Efficient FPGA Implementation of a Wavelet Coder/Decoder. The 12th International Conference on Microelectronics, Tehran, october 31 – november 2, (2000).
13. Miaou, Shaou-Gang, Lin, Chih-Lung: A Quality-on-Demand Algorithm for Wavelet-Based Compression of electrocardiogram signals. IEEE Trans. on Biomedical Engineering, vol. 49, n^o 3 (2002).

Randomized Algorithm in Embedded Crypto Module

Jin Keun Hong

Division of Information & Communication, Baekseok University, 115 Anseo-Dong, Cheonan-Si, Chungnam, 330-704, Korea
Jkhong@bu.ac.kr

Abstract. The hardware random number generator is a source of unpredictable, statistically random stream sequences. Critical cryptography applications require the production of an unpredictable and unbiased stream of binary data derived from a fundamental noise mechanism. In this paper, we analyzed hardware random number generator with Gaussian noise using randomized algorithm in respect of security consideration. In this paper, hardware random number system on embedded Linux on chip (LOC) processor, MC68328, is reviewed to reduce the statistical property of the biased bit stream in the output of a random number generator. In experiments of the randomness evaluation for the randomized algorithm, we evaluated the statistical evaluation for 10 test samples, the severe biased and the moderate biased stream. Although the random bit stream has the biased characteristics. But the differential quantities are compensated using the randomized process by chaos function. Therefore in the randomness evaluation of hardware generator, the proposed randomized algorithm is always satisfied the randomness test condition.

1 Introduction

An hardware random number generator uses a non-deterministic source to produce randomness, and more demanding random number applications, such as cryptography, a crypto module engine, and statistical simulation, then benefit from the sequences produced by an random number generator (RNG), a cryptographic system based on a hardware component [1]. As such, a number generator is a source of unpredictable, irreproducible, and statistically random stream sequences, and a popular method for generating random numbers using a natural phenomenon is the electronic amplification and sampling of a thermal or Gaussian noise signal. However, since all electronic systems are influenced by a finite bandwidth, $1/f$ noise, and other non-random influences, perfect randomness cannot be preserved by any practical system. Thus, when generating random numbers using an electronic circuit, a low-power white noise signal is amplified, then sampled at a constant sampling frequency. Yet, when using an RNG with only a hardware component, as required for statistical randomness, it is quite difficult to create an unbiased and stable random bit stream. The studies reported in [3-4] show that the randomness of a random stream can be enhanced when combining a real RNG, linear feedback shift register (LFSR) number generator, and hash function. Hence, in previous studies about RNG schemes in the security area, Fabrizio Cortigiani, et al. (2000) examined a very high speed true random noise generator, S. Rocchi and V. Vignoli (1999) proposed a high speed chaotic

CMOS true random analog/digital white noise generator, Adel et al. (2001) investigated the design and performance analysis of a high speed AWGN communication channel emulator, and a noise-based random bit generator IC for applications in cryptography was considered (Craig S, et al. 1998 [4]).

However, the randomness of such combined methods is still dependent on the security level of the hash function and LFSR number generator. Thus, a previous paper proposed a real RNG that combines an RNG and filtering technique that is not dependent on the security level of the period. In particular, it is important that the RNG hardware offers an output bit stream that is always unbiased. Even though the hardware generating processor generates an output bit stream quickly, if the software randomized algorithm is inefficient, the RNG becomes time consuming, thereby restricting the conditions when an RNG can be applied. Accordingly, this paper proposes an efficient method of randomized filtering for an RNG processor in the embedded crypto module. To consistently guarantee the randomness of the output sequence from an RNG, the origin must be stabilized, regardless of any change of circumstances. Therefore, an RNG is proposed that applies a randomized algorithm that guaranteed the pass probability of random bit stream, plus the computational burden is analyzed when the randomized algorithm is applied. Hereinafter, in chap. 2, the framework of hardware random number generator is introduced. In chap. 3 & 4, experimental results using the system are presented and concluded in respect of security considerations.

2 The Framework of Hardware Random Number Generator

We would like to stress that the class/style files and the template should not be manipulated and that the guidelines regarding font sizes and format should be adhered to. This is to ensure that the end product is as homogeneous as possible. The hardware random number generator includes common components for producing random bitstreams, classified as follows: characteristics of the noise source, amplification of the noise source, and sampling for gathering the comparator output. The embedded Linux on chip system using dragonball CPU, MC68328, which is an integrated controller for handhel products, based on MC68EC000 microprocessor core, is used to generate hardware random bit stream.

The probability density $f(x)$ of the Gaussian noise voltage distribution function is defined by Eq. (1):

$$f(x)=\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{x^2}{2\sigma^2}} \tag{1}$$

Where σ is the root mean square value of Gaussian noise voltage. The noise diode is used the diode with white Gaussian distribution. The power density for noise is constant with frequency from 0.1Hz to 10MHz and the amplitude has a Gaussian distribution. When the frequency range is given, the voltage of noise is decided by a factor of frequency. The crest factor of a waveform is defined as the ratio of the peak to the *rms* value. A crest value of approximately 4 is used for noise. When the output of a fast oscillator is sampled on the rising edge of a slower reference clock, the comparator features differential analog inputs and TTL logic outputs with an active internal pull-up and supplies a fast propagation delay for the sampling circuits. The applied voltage is ±5Vdc and propagation delay is 8nsec. The frequency range can be stably operated up to 100MHz. First, the sampled stream is gathered, then a randomized algorithm is used to enhance the statistical randomness.

3 The Characteristics of the Random Bit Stream

3.1 Random Bit Stream for the Uniformed Distribution

The relationship of characteristics of output random bit stream and the entropy, as followed the Shannon's theory, is defined by Eq.(2).

$$H(x) = -M \sum_{i=1}^{n} p_i \log p_i \tag{2}$$

In this equation, P_i is the probability of state i out of sample n, and M is a constant. The entropy of the output random bit stream of the random number generator is approached at the length of a bit; the probability P_i to generate k bits in the random number generator is 2^{-k}; and the entropy of the random bit stream is equal to k bits and is presented as equal to the possible output bit stream. If k is 4, the output bit stream is as follows:

$$U_o = (0,0,0,0), U_1 = (0,0,0,1),..., U_{14} = (1,1,1,0), U_{15} = (1,1,1,1) \tag{3}$$

The class U_i, is consists of 4 bits block, is the pattern of 16 symbols, if the each pattern give the uniform distribution, then the each pattern is approached the probability of 2^{-4}. In Eq.(4), the duty information (D) during the one cycle is as follows:

$$D = \frac{t}{T} \tag{4}$$

Where t is the number of "1" bit streams, and T is the number of "1" and "0" during one period. In the considerations of the duty cycle, during the one cycle, the class about "0" bit pattern and "1" bit pattern is as follows:

$$\begin{aligned} U_o &= \{(0,0,0,0)\}, U_1 = \{(0,0,0,1),(0,0,1,0),(0,1,0,0),(1,0,0,0)\} \\ U_2 &= \{(0,0,1,1),(0,1,0,1),(0,1,1,0),(1,1,0,0),(1,0,1,0),(1,0,0,1)\}... \end{aligned} \tag{5}$$

During the one cycle, if the duty value will be 0.5, the class will have characteristics of random, and in the respect of mono bit test, the characteristic distribution of the pattern "0" bit and "1" bit is uniformed, and in poker test, the characteristic distribution of the pattern is uniformed. The proposed filter mechanism is driven statistically the uniformed distribution through the reduction of the biased distributed characteristics from the biased random bit stream. In the random number generator, to generate the un-biased output random bit stream, it is needed to sustain the uniformed distribution of random bit stream. In the discrete time series, the output bit stream during the one period is consists of even time series and odd time series, as follows by Eq.(6).

$$V(t) = \sum_n (V_{t_n} + V_{t_{n+1}}) \tag{6}$$

Where n is the even number. The total discrete time series V(t) is summed to the even time series V_{t_n} and the odd time series $V_{t_{n1}}$. The even time series *$V_1(t)$* is as follows:

$$V_1(t) = \sum_n V_{t_n} \tag{7}$$

The odd time series *$V_2(t)$* is as follows:

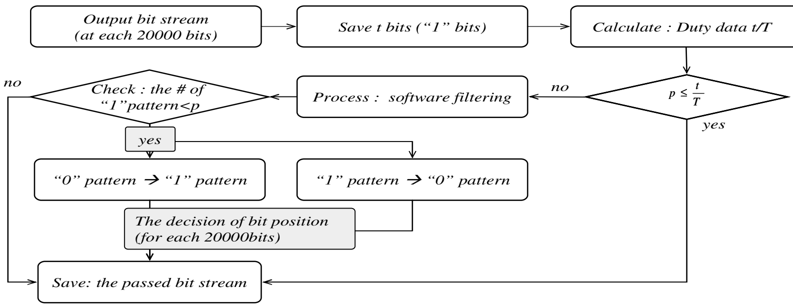
$$V_2(t) = \sum_n V_{t_n+1} \tag{8}$$

Let the density distribution of random bit stream in the odd time series during the one period be 1 and the density distribution of “1” pattern bit stream be 0.6, the bias characteristics of the odd time series has 20%. If the density distribution of random bit stream in the even time series during the one period be 1 and the density distribution of “1” pattern bit stream be 0.6, the bias characteristics of the even time series has 20%. The total bias distribution of random bit stream during one period has 40%. However the duty value of the ideal total time series V(t) is required 0.5 to gather always the un-bias of output bit stream. To sustain statistically the randomness of output bit stream, it is required to alter the bias distribution through the decision of the biased position and the control of the biased quantities. In Eq.(9), to the control of biased quantities, it is proposed to alter the bias distribution for the biased bit stream.

$$V(t) = \sum_n (V_{t_n} + inv|V_{t_{n+1}}|) \tag{9}$$

3.2 Mechanism of Randomized Algorithm

The proposed randomized algorithm is applied to the duty information in Eq. (4). The duty factor is the critical factor for an enhancement of the randomness, and the bit steam during one period is set at 20,000 bits. When the unit of the output bit stream is 20,000 bits, the number of the “1” pattern bits is t bits. If the value of the duty information within one period is included within the significance level (p), the decision will be considered as a state of “pass”.



Output bit stream (at each 20000 bits)
Save t bits (“1” bits)
Calculate : Duty data t/T
$p \le \frac{t}{T}$
no
yes
Process : software filtering
Check : the # of “1”pattern<p
no
yes
“0” pattern → “1” pattern
“1” pattern → “0” pattern
The decision of bit position (for each 20000bits)
Save: the passed bit stream

Fig. 1. Mechanism of randomized Algorithm

If the condition of "pass" is determined, this is added as pass data to the buffer memory. If "fail" is determined through the software filtering process, this is included in the decision process. When the size of the desired bit stream is gathered, the process is then completed. If the value of the duty cycle of the collected output bit stream, P is not satisfied the condition of significance level, then the conversion of the bit pattern is started, as much as the number of bit, which is included in the significance level. If the density distribution of "0" bit pattern is more than the number of the significance level, it is accomplished the conversion process, for example, "0" bit pattern is converted "1" bit pattern, as much as the number of the marginal bit. In the process of the proposed randomized algorithm of Fig.1, it is generated the output bit stream, 20,000bits, to calculate the duty information, the number of "1" bit pattern is saved. If the calculated duty value is included in significance level, it is evaluated about the unbiased bit stream. Then the bit stream is not accomplished the randomized process and defaulted the bit stream. But if it is not included within the significance level, then the randomized process is started about the bit stream. In this case, the range of the defined significance level about "1" bit pattern is decided the value between 9,654 bits and 10,346 bits in the reference of mono bit test, which is defined FIPS 140-1, for the unit of 20,000bits. During one cycle, when the number of "1" bit pattern of random bit stream is not included within the significance level (9,654bits out of 20,000bits), the randomized process is started, as much as the difference bits. Otherwise when the number of the "1" bit pattern is upper than that of the significance level (10,346 bits out of 20,000bits), it is randomized as much as the differential bits from "1" to "0" bit pattern. If the randomized process is accomplished, the algorithm for the decision of location is applied the uniform distribution of the chaos function and the randomized iteration about the same position is processed only by one round. The logistic function, which is the discrete chaos map is applied for the randomized process, is as follows:

$$X_{n+1} = -\alpha X_n (1 - X_n) \tag{10}$$

Where the range of α is $0 \le \alpha \le 4$, and the range of the initial value X_0 is $0 \le X_0 \le 1$. The value of X_{n+1} is derived from the previous state value X_n. Inversely, given X_{n+1}, X_0 has resolved the two values of the solution in an equation of the second degree. The logistic map has the characteristic of irreversibility, and α is the sensitivity parameter that determines the dependence of the next value derived from the initial value. If the value of α is increased, the resulting value varies greatly from the result of the slight variation of the initial value after the recursive calculation. For the condition $\alpha < 1$, when the process of X_n is performed recursively, the value of X_n converges to 0. For the convergence to the direction of the chaos domain, which continues to infinity, the value of α must be $\alpha > 3.56$. However, the distribution of the output bit stream of the chaos function has an independent and uniform distribution, and is an integral number between [0, d-1]. To apply the random number generator, the integral number between [0, 1] is distributed uniformly by the tent transformation function. By the tent function $x' = h(x)$, the non-linear value of the discrete chaos function becomes a linear value.

$$x' = h(x) = \sin^2(\pi x / 2) \tag{11}$$

In tent transformation process, if the initial value is x_0, the transformed value is $x'(0) = h(x_0)$.

$$x(1) = T(x_0),\ x(2) = T^2(x_0),\ldots,\ x(k) = T^k(x_0),\ldots \tag{12}$$

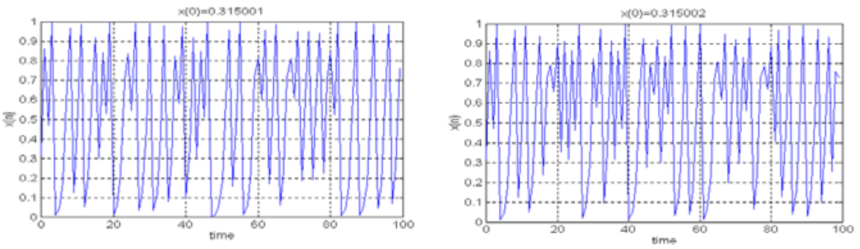
Let y_0 be x'_0, it is driven the equation $y_1 = f(y_0)$, $y_2 = f^2(y_0)$,..., $y_k = f^k(y_0)$. From f function and T function, we can be driven as follow:

$$f^k(h(x)) = h(T^k(x)) \tag{13}$$

Where k is 0, 1, 2, .., . The non-linear value of the x' axis by the tent function is transformed to a linear value of the x axis, which has a uniform distribution. Therefore, the output of the chaos function is uniformly generated by the position of the randomized region during one duty cycle, and the randomized process is performed using the generated position information.

3.3 Experimental Results

For the decision regarding the position information in the randomized process, the chaos function is used through the utilization of a logistic map.



(a) At the initial value=0.315001 (b) At the initial value =0.315002

Fig. 2. X_n according to the logistic initial value of X_0

A logistic map when the initial value of X_0 is 0.315001 and the initial value of another X_0 is 0.315002 after 100 iterations is shown in Fig.2. In the initial state, the logistic map according to the initial value is slightly varied; however, with additional iteration rounds, any deviation is magnified by the factor in the each iteration. Given an initial deviation, it will eventually become as large as the actual signal itself. After a number of iterations, the error will be of the same order of magnitude as the correct values. This section presents the results obtained from the proposed system. First, the effect of the randomness of the output random bit stream was investigated. More quantitative tests for randomness can be found in technical literature [9-10]. To diagnosis the output bit stream of random number generator, it is used the randomness test items such as, mono bit test, poker test, run test of FIPS 140-1. The mono bit test is evaluated for the successive 20,000 bits out of random bit stream. If the tested bit stream is included in significance level [9,654, 10,346], the bit stream is evaluated the non-biased stream. The poker bit test is consists of 500 classes (1 class is 4 bits), and the number of the pattern, which is consists of 4btis, is 16 (0<i<15), From Eq.(14), if the range of result value is between 1.03 and 57.4, its stream is evaluated the non-biased stream.

$$Z = (\frac{16}{5000}) * (\sum C(i)^2) - 5000 \tag{14}$$

Where *C(i)* is the tested bit stream, the run test can detect a monotonic trend in a time series *x(i)*, *i*=1,…, N, by evaluating the number of runs in a second time series derived from *x(i)*. A "run" is defined as a sequence of identical observations that is followed by a different observation. If *x(i)* is a stationary random process, the number of runs is a random variable with mean = N/2+1 and variance =(N(N-2))/(4(N-1)). In experiments concerning the randomness evaluation for the randomized algorithm, a statistical evaluation for 10 test samples consisting of severely biased and moderately biased streams was evaluated. Although the random bit stream has biased characteristics, the differential quantities are offset using the randomized process by the chaos function. Therefore, in the randomness evaluation of a hardware generator, the proposed randomized algorithm always satisfies the randomness test conditions in comparison with the algorithm without software filtering. For the biased bit stream, which is not applied to the software algorithm, the probability of the random bit stream generated successively in a specific pattern is high, and the condition of the significance level for the poker test is not satisfied. In the run test, for the severely biased bit stream, the length of the run is dependent on a specific length, such as 3 or 4, and the tested data from the experiments can be evaluated readily, especially when the length of run is 3. In Fig. 3, the loss quantity of a random bit stream is represented according to the probability of whether a pass will occur. If the randomized algorithm is not applied, the distribution of the loss quantity is suggested in Fig.3. In typical pass/fail boundaries, the measured average of 10 iteration tests based on 7.2x10^8 bits samples per hour is used, depending on whether the sequence passes all the trial tests.

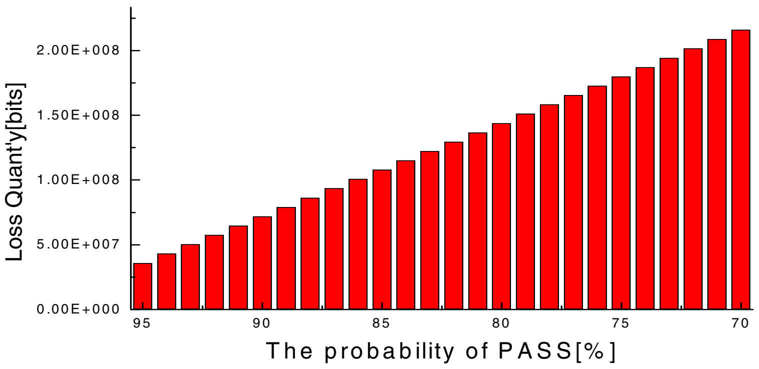


Fig. 3. Loss Quantity according to the probability of pass

The test results were extremely positive; here the proposed system passed all of the trial tests. In Fig.4, the relationship of delay time and probability of pass is presented, for each pass probability on embedded LOC processor. Given in the previous test condition, if the probability of pass is 95%, it is consumed about 200seconds. If the pass probability is reduced, it needs additional consumed time, due to position process by chaos function, to gather the non-biased bit stream from experimental by embedded LOC process.

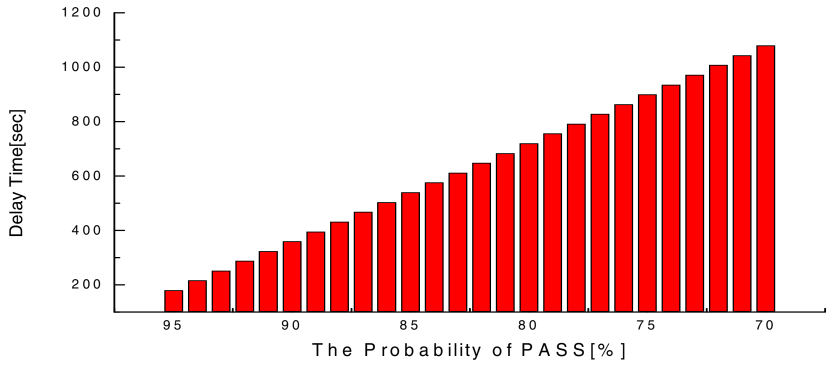


Fig. 4. Delay Time according to the probability of pass

4 Conclusion

The current paper presented and tested a hardware random number generator using the randomized algorithm by chaos function on embedded LOC processor. The proposed method, directly derived from results obtained from the Gaussian noise generator, is based on filtering a noise sequence using a software filter. The hardware random number generator is well suited for applications such as data encryption, circuit testing and measurement, and mathematical simulations. A method of combing real noise technology and software filter mechanism by logistic chaos function is presented, extending this concept to a hardware random generator.

References

1. Alireza h., Ingrid V.: High-Throughput Programmable Crypto-coprocessor, IEEE Computer Society (2004)
2. Jalal A. M, Anand R., Roy C., M. D. M: Cerberus: A Context-Aware Security Scheme for Smart Spaces, proc. IEEE PerCom'03 (2003)
3. Robert Davies: True random number, http://webnz.com/robert/true_ rng.html.
4. C. S. Petrie and J. A. Connelly: A Noise-Based Random Bit Generator IC for Applications in Cryptography, Proc. ISCAS'98, June (1998)
5. M. Delgado-Restituto, F. Medeiro, and A. Rodriguez-Vasquez: Nonlinear switched-current CMOS IC for random signal generation, IEE electronic letters, vol. 29, December (1993)
6. Http://www.io.com/~ritter/RES/NOISE.HTM.

7. Http://www.clark.net/pub/cme/P1363/ranno.html.
8. Boris Ya, Ryabko and Elena Matchikina: Fast and Efficient Construction of an Unbiased Random Sequence, IEEE Trans. on information theory, vol. 46, no. 3, May (2000)
9. FIPS 140-1: Security Requirements for Cryptographic Modules, U.S. Department of Commerce/NIST [National Technical Information Service] Springfield, Virginia (1994)
10. Diehard: http://stat.fsu.edu/~geo/diehard.html Oct. (1998)

Hardware Implementation of an Analog Neural Nonderivative Optimizer

Rodrigo Cardim, Marcelo C.M. Teixeira, Edvaldo Assunção, Nobuo Oki, Aparecido A. de Carvalho, and Márcio R. Covacic

UNESP - São Paulo State University,
Department of Electrical Engineering, Campus of Ilha Solteira,
Avenida Brasil, 56, 15385-000, Ilha Solteira, São Paulo, Brazil
marcelo@dee.feis.unesp.br

Abstract. Analog neural systems that can automatically find the minimum value of the outputs of unknown analog systems, described by convex functions, are studied. When information about derivative or gradient are not used, these systems are called analog nonderivative optimizers. An electronic circuit for the analog neural nonderivative optimizer proposed by Teixeira and Żak, and its simulation with software PSPICE, is presented. With the simulation results and hardware implementation of the system, the validity of the proposed optimizer can be verified. These results are original, from the best of the authors knowledge.

1 Introduction

The most popular optimization algorithms for unconstrained optimization use derivative or gradient information of the objective function. However, in many applied problems the gradient of the objective function may not be accessible or too expensive to evaluate. Thus, there is a need for nonderivative optimization algorithms. Korovin and Utkin [1] proposed an approach to nonderivative optimization using ideas from the theory of variable structure sliding mode systems. They proposed and analyzed in detail an one-dimensional nonderivative optimizer and suggest using the one-dimensional network to solve multi-dimensional optimization problems by varying the direction of search in a multi-dimensional parameter space. However, the implementation issues of how and when to change the search direction were left open. In [2] and [3], an one-dimensional nonderivative optimizer that can be used to solve multi-dimensional unconstrained optimization problems was proposed. The proposed optimizer is robust. It has the property of disturbance rejection in the sense that it will be performing the correct line search in spite of the presence of disturbances with bounded time derivatives. In this paper, an electronic circuit for the analog neural nonderivative optimizer, described in [2] and [3] is proposed.

2 Nonderivative Line Search Network

We now describe our network [2], [3], that performs a line search with no derivative information. One of the main components of our network is a minimum peak detector.

2.1 Minimum Peak Detector

A minimum peak detector can be implemented using standard circuit components like a capacitor, a diode, and an operational amplifier—see, for example, Stout [4, Chapter 8]. It tracks the input signal and holds the minimal value of its output since the last reset. The input signal is continuously compared with the stored minimal value to determine if the stored value should be updated.

In the following, we analyze a class of minimum peak detectors. Let $y = y(t)$ be the input waveform, and $y_d = y_d(t)$ be the output waveform of a minimum peak detector. Then, a mathematical model of a class of minimum peak detectors can be represented as

$$\dot{y}_d = \begin{cases} 0 & \text{if } y_d - y \leq 0 \\ -M & \text{if } y_d - y > 0, \end{cases} \tag{1}$$

where $M > 0$ is a design parameter. A particular realization has the form

$$\dot{y}_d = -\frac{M}{2}\left(\operatorname{sign}(y_d - y) + 1\right).$$

Let $\varepsilon = y_d - y$. We assume that the time derivative of the function $y = y(t)$ is bounded by a known constant $M > 0$, that is,

$$\left|\frac{d}{dt}y(t)\right| < M.$$

We now consider two cases. The first case is when $\varepsilon \leq 0$. In this case $y \geq y_d$, and the network holds the previously detected minimum value of the input signal y. The second case is when $\varepsilon > 0$, that is, when $y < y_d$. We will show that in this case the error $\varepsilon \to 0$. Let

$$v(t) = \frac{1}{2}\varepsilon^2(t) \tag{2}$$

be the measure of the error magnitude. Then, the time derivative of $v = v(t)$ evaluated on the trajectories of (1) is

$$\frac{d}{dt}v(t) = \varepsilon\dot{\varepsilon}(t) = \varepsilon(\dot{y}_d - \dot{y}) \leq \varepsilon(-M + |\dot{y}|) < 0.$$

The speed with which the error goes to zero depends on the magnitude of the design parameter M. Thus, in this case, the output of the network will converge to the input waveform.

2.2 The Proposed Optimizer

Suppose that the function $y = f(x)$ that we wish to minimize has only one minimizer, say x^*, in the interior of the closed interval $[a, b]$. We assume that the slope of f in $[a, b]$ is bounded, that is, there is a finite constant $L > 0$ such that

$$\left|\frac{d}{dx} f(x)\right| \leq L. \tag{3}$$

A block diagram of the proposed optimizer is shown in Figure 1. Note that the optimizer depicted in Figure 1 contains a minimum peak detector as a component.

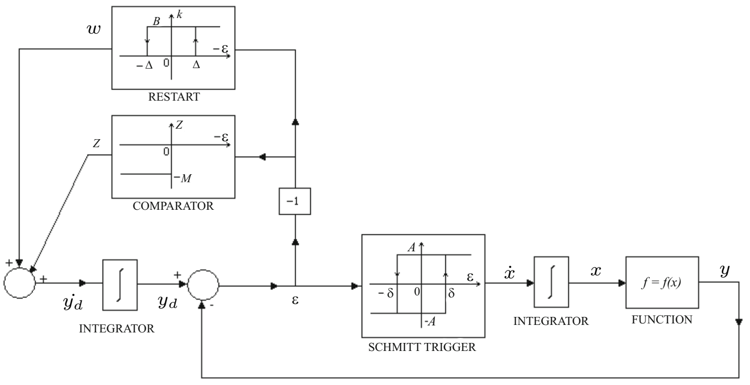


Fig. 1. A block diagram of the analog neural optimizer

The minimum peak detector operates on its input signal y. The output signal of the minimum peak detector is y_d. There are two hysteresis elements in the optimizer, where $\delta < \varDelta$. Both δ and $\varDelta$ are design parameters. Other design parameters are A, B, and M. We now describe the network operation. A typical trajectory of the network is shown in Figure 2. For the sake of argument let the initial condition $x_0 = x(t_0) = x(0)$ be such that

$$\frac{df}{dx}(x(0)) > 0,$$

that is, the initial point x_0 is located on the right side of the minimizer x^*. Next, let $\varepsilon = 0$, $w = 0$, and $u = A$. Then, since $\dot{x} = u$, we have for $t > t_0 = 0$,

$$x(t) = At + x(0).$$

This means that initially we will be moving away from the minimizer denoted as x^*. The output of the minimum peak detector y_d will be constant because $\dot{y}_d = 0$. As a result, the error $\varepsilon = y_d - y$ will become negative and will be decreasing until at some time t_1 it reaches the value $\varepsilon = -\delta$. This will force u to change from A to $-A$ and for $t > t_1$,

$$x(t) = -A(t - t_1) + x(t_1),$$

which means that x will start moving toward x^* and the value of $y = f(x)$ will decrease. Once $\varepsilon = 0$ is reached, the signal out of the minimum peak detector changes its value from $y_d = y(x(0))$ to follow the trajectory $y(x(t))$. The parameter value M should be chosen so that

$$|\dot{y}(t)| = \left| \frac{df(x)}{dx} \dot{x}(t) \right| \le LA < M.$$

The output of the minimum peak detector y_d tracks y until x reaches x^*. Then, since ε is still zero, x will still be moving to the left. But now y_d will be locked at $y(x^*) = y^*$ and the error ε will start decreasing, assuming negative values, till it reaches the value of $\varepsilon = -\Delta$ at some time $t = t_2$. At this instant of time the signal w kicks in, that is, w takes on the value B. This implies that for $\varepsilon \le 0$, we have

$$\dot{\varepsilon} = \dot{y}_d - \dot{y} = B - \dot{y}.$$

For $\varepsilon > 0$, we have $\dot{\varepsilon} = B - M - \dot{y}$. We assume that $B > 2M$. Thus, as long as $w \ne 0$, we have $\dot{\varepsilon} > 0$. The minimum peak detector is reset to a new value.

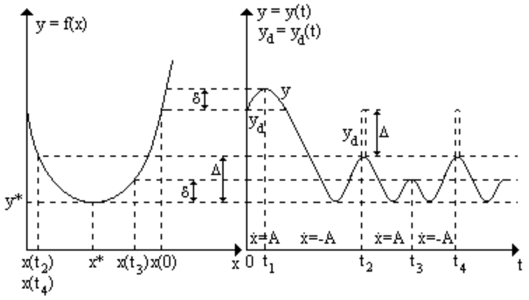


Fig. 2. A typical trajectory of the optimizer

Details of this process are illustrated in Figure 3.

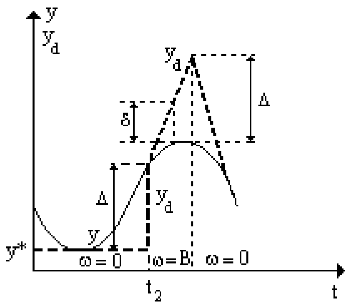


Fig. 3. Resetting the minimum peak detector

Over the time interval when ε is growing from $-\Delta$ to δ, we have $\dot{x} = -A$, and hence y will be increasing. When ε reaches the value of δ, the signal u will take the value of A, that is $\dot{x} = A$, and y will start decreasing, while ε is still increasing. The trajectory x will start moving back toward x^* because now $\dot{x} = A$. When ε achieves the value of Δ the signal w will be set to zero, and ε will start decreasing till it becomes zero. The output of the minimum peak detector, y_d, will track y till x moves past x^*. Then, y_d will take on the value $y(x^*)$ and $z = 0$. The error ε will decrease till it reaches the value of $-\delta$. At the time instance $t = t_3$, the trajectory x will reverse its travel because now $\dot{x} = -A$. The error is negative and it will increase till it reaches the zero value. The trajectory x will still be moving to the left. The error will be taking on negative values till it reaches the value of $-\Delta$ which will activate the signal w at time t_4. We note that this is exactly the same scenario that took place at time t_2. The oscillations of y around the minimum value y^* will now be repeated, with their period being $t_4 - t_2$ and peak-to-peak value approximately equal to Δ. Since Δ is a design parameter, we can choose its value as small as we wish, thereby reducing the amplitude of oscillations of x around the minimizer x^*. In the above analysis, for the sake of argument, we assumed specific initial conditions. However, in the course of this analysis we covered all other possible initial conditions.

The proposed minimizer has the attractive property of disturbance rejection. Indeed, suppose that a disturbance signal d corrupts the output of the integrator as shown in Figure 4 (a). For clarity, we only show a relevant portion of the network. The complete diagram of the network is shown in Figure 1. An equivalent representation of the diagram shown in Figure 4 (a) is given in Figure 4 (b), where we use the well known fact that $d(t) = d(t_0) + \int_{t_0}^{t} \dot{d}(s)ds$. Using this equivalent representation we conclude that as long as $\left|\dot{d}(t)\right| < A$, the sign of the input signal to the integrator is the same as it would be with no disturbance. Therefore, the network will be performing the correct line search in spite of the presence of the disturbance.

According to [5], analog nonlinear dynamical systems with nonlinearities such as hard limiter, hard limiter with hysteresis and comparator, can be considered an Analog Neural Network (ANN). Therefore, the dynamical system in Figure 1 is an ANN.

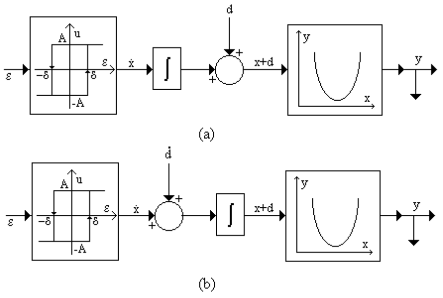


Fig. 4. An illustration of the disturbance rejection property of the proposed network

3 Optimizer Applications

The application of the Korovin and Utkin optimizer in Anti-lock Braking Systems (ABS) was presented in [6] and [7]. The ABS was designed to prevent wheel lock-up. By preventing the wheels from locking, it enables the driver to maintain steering control and to stop in the shortest possible distance. In the system presented in [6] and [7] was necessary to identify the type of the road surface. At this moment, there is not a sensor able to give this information. However the type of road surface can be deduced from the brake pressure, the skid extension and the car decelerating. In [6,7,8,9,10] was showed that the typical adhesion curve of asphalt with ice versus wheel skid extension has only one maximum point. In this way is possible to use the optimizer suggested in [3] to find this point. Other application, presented in [11], was to use the optimizer in power factor correction of electrical systems. In [11] was showed that the apparent power versus reactance curve presents only one minimum point. In this way the optimizer can be used to control de capacitor reactance or the variable inductance in order to adequate the power factor with different loads.

4 Optimizer Implementation

In this work a simulation of the optimizer using the PSPICE software and the electronic breadboard implementation in laboratory, using operational amplifiers, resistors and capacitors was presented. Figure 5 shows the optimizer complete diagram. The circuits blocks are designated as the block diagram illustrated in Figure 1.

In Figure 5, the circuits of the adder and gain blocks were not showed because they can be easily designed using operational amplifiers. The integrators behave as a low-pass filter. The integrator unitary gain frequency (w_{int}) is defined as $w_{int} = \frac{1}{RC}$. In this frequency the gain is 0 dB, but the gain changes for different frequencies. For calibration purpose a multiplier is put in series with each integrator in order to allow the calibration. The integrator can also saturate due the high DC gain. This problem can be solved with a high resistance put in parallel with the capacitor. The resistor makes the integrator DC gain finite, far from the ideal response [12]. Based on [13], the Schmitt trigger, comparator and restart blocks, of the optimizer in Figure 1, can also be easily designed.

4.1 Examples

For circuit simulation and results verification, the system showed in Figure 5 was calibrated with the parameters showed in Table 1 and with the objective function $y = f(x) = x^2$. The simulations illustrated in Figure 6 and 7 show the results obtained with initial condition equal to zero, and Figure 8, with initial condition $y(0) = 11V$ and $y_d(0) = 0V$.

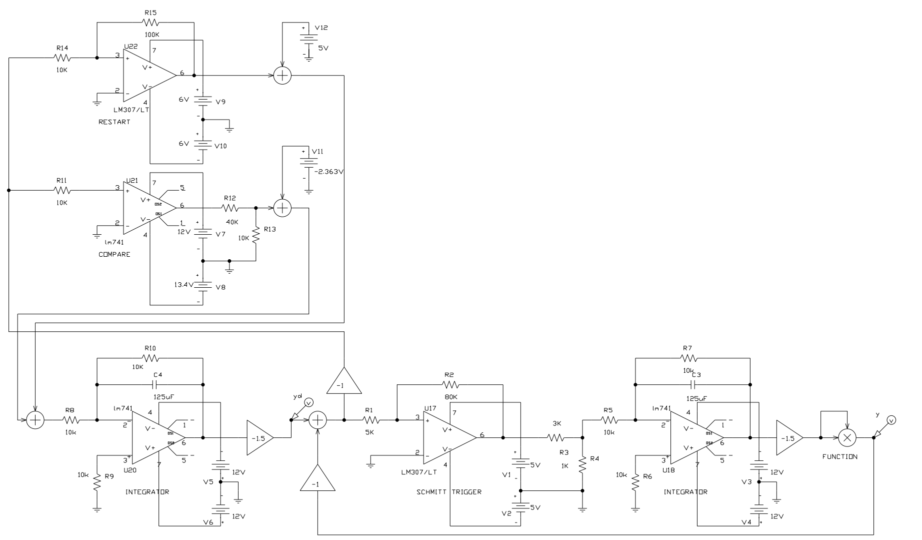


Fig. 5. Circuit of the Analog Nonderivative Optimizer with the software PSPICE

Table 1. Optimizer Parameters for Figure 5

A	1 V
B	11 V
M	5 V
δ	0.25 V
Δ	0.50 V

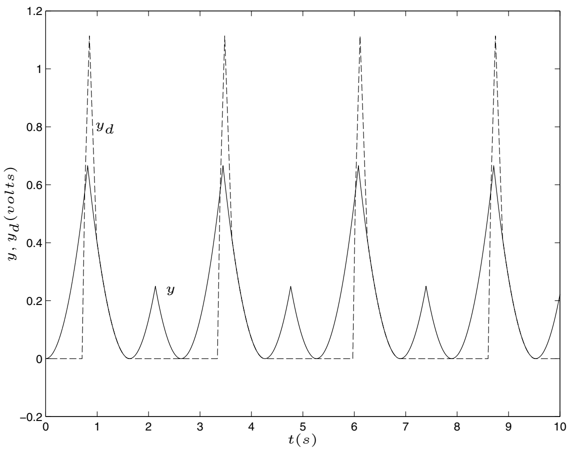


Fig. 6. Simulation of the circuit using the software SIMULINK with initial conditions $y(0) = 0V$ and $y_d(0) = 0V$ (theoretical result)

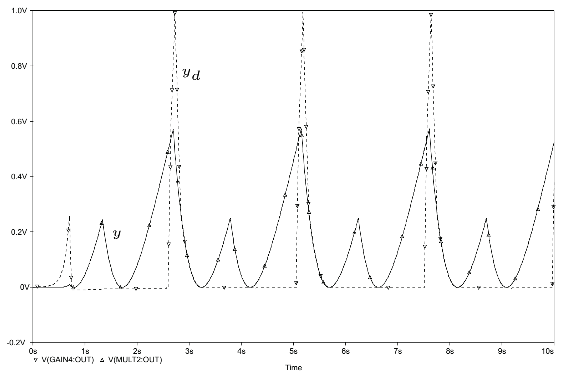


Fig. 7. Simulation of the circuit with the software PSPICE with initial conditions $y(0) = 0V$ and $y_d(0) = 0V$

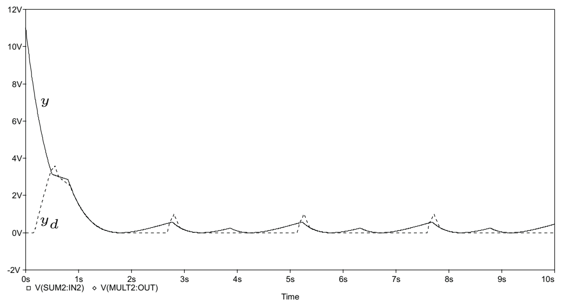


Fig. 8. Simulation of the circuit in the software PSPICE with initial conditions $y(0) = 11V$ and $y_d(0) = 0V$

Figure 9 shows the implementation result obtained in our Control Laboratory at São Paulo State University, in Ilha Solteira - SP, Brazil (Figure 11). The circuit was breadboard as showed in diagram of Figure 5 and the obtained results are satisfactory. From Figure 7 and 9 can be observed that the signal obtained has a low offset. This problem can be solved with integrator calibration in the operation frequency.

Figure 10 shows the result when the minimum point of function is modified. Initially it was considered that the function is $f(x) = x^2$ and in the time $t = 5s$, the function was modified to $f(x) = x^2 + 3$. In both cases, the minimum value of the function was found.

Better optimizer efficiency are expected when the values of δ and Δ are reduced because there is less output variation around the minimum point. However, if this parameters are drastically reduced, more precise electronic components must be used, that will make the circuit more expensive. In this way the values of δ and Δ can be determined to meet the design requirements.

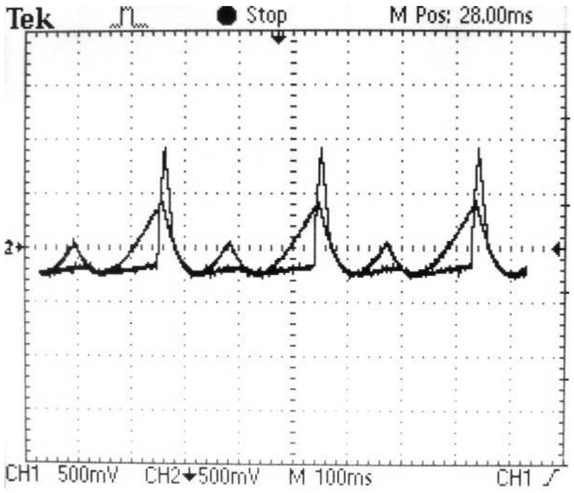


Fig. 9. Circuit waveform obtained in the laboratory

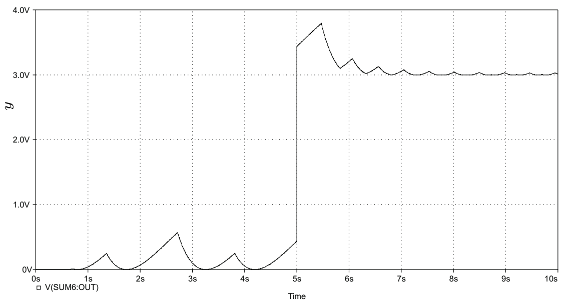


Fig. 10. Circuit Simulation using the software PSPICE with function change

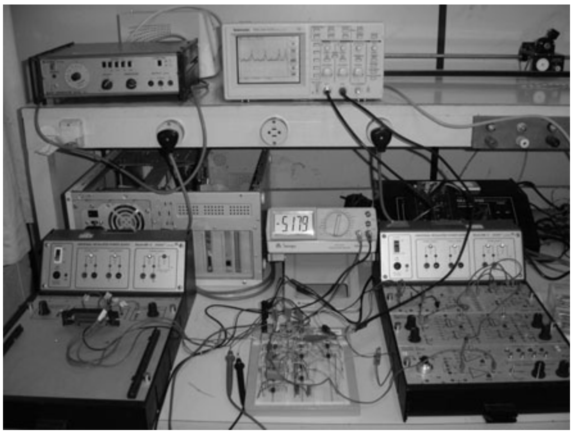


Fig. 11. Picture of the system working at the laboratory

5 Conclusions

We proposed and analyzed the hardware implementation of an analog neural network [2], [3] for solving a class of convex unconstrained programming problems. The optimizer is robust in the sense that it can tolerate disturbance signals with bounded derivatives. The neural nonderivative optimizer simulation in the PSPICE software and the experimental results obtained in the laboratory were very satisfactory. The proposed circuit is simple and uses a small number of electronic components.

Acknowledgments. The authors acknowledge the financial support by FAPESP and CNPq, from Brazil.

References

1. Korovin, S., Utkin, V.I.: Using sliding modes in static optimization and nonlinear programming. Automatica **10** (1974) 525–532
2. Teixeira, M.C.M., Żak, S.H.: Analog nonderivative optimizers. In: American Control Conference - ACC, Albuquerque, New Mexico, USA (1997) 3592–3596
3. Teixeira, M.C.M., Żak, S.H.: Analog neural nonderivative optimizers. IEEE Transactions on Neural Networks **9**(4) (1998) 629–638
4. Stout, D.F.: Handbook of Operational Amplifier Circuit Design. M. Kaufman, McGraw-Hill, New York (1976)
5. Cichocki, A., Unbehauen, R.: Neural Networks for Optimization and Signal Processing. John Wiley & Sons, Chichester, New York, Brisbane, Toronto, Singapore (1993)
6. Will, A.B.: Intelligent Vehicle Steering and Braking Control Systems. PhD thesis, School of Electrical Engineering, Purdue University, West Lafayette, IN (1997)
7. Żak, S.H., Will, A.B.: Sliding mode wheel slip controller for an antilock braking system. International Journal of Vehicle **19** (1998) 523–539
8. Czernichovski, S.: Otimizadores analógicos não derivativos. Msc thesis, UNESP - São Paulo State University, Ilha Solteira - SP, Brazil (2001)
9. Lee, Y., Żak, S.H.: Genetic neural fuzzy control of anti-lock brake systems. In: American Control Conference - ACC, Arlington, Virginia, USA (2001) 671–676
10. Lee, Y., Żak, S.H.: Designing a genetic neural fuzzy antilock-brake-system controller. IEEE Transactions on Evolutionary Computation **6**(2) (2002) 198–211
11. Cardim, R., Teixeira, M.C.M., Assunção, E.: Utilização de um otimizador analógico não-derivativo para a correção do fator de potência. In: II Congresso Temático de Dinâmica e Controle da SBMAC, São José dos Campos - SP, Brazil (2003) 1474–1483
12. Sedra, A.S., Smith, K.C.: Microeletrônica. Makron Books Ltda, Brazil (2000)
13. Franco, S.: Design Operational Amplifiers and Analog Integrated Circuits. McGraw-Hill, New York (1998)

Synchronization Via Multiplex Spike-Trains in Digital Pulse Coupled Networks

Takahiro Kabe, Hiroyuki Torikai, and Toshimichi Saito

EECE Dept, Hosei University, Koganei-shi, Tokyo, 184–8584 Japan
kabe@nonlinear.k.hosei.ac.jp,
{torikai, saito}@k.hosei.ac.jp

Abstract. This paper studies pulse-coupled network of digital spiking neurons and its basic dynamics. The neuron is constructed by coupling two shift registers and has a variety of spike-trains which correspond to digital codes through a inter-spike interval (ISI) modulation. The pulse-coupled network has master-slave configuration. All the spike-trains of neurons in the master side are multiplexed additionally and are transmitted to the slave side via single line. Neurons in the slave side are connected by dynamic winner-take-all function. As parameters are selected suitably, the slave can realize demultiplexing and master-slave synchronization is achieved. VHDL simulation is also discussed for FPGA implementation and this digital network is compared with an analog network.

1 Introduction

This paper studies a digital spiking neuron (DSN), pulse-coupled network of DSNs (PCDSN) and its basic behavior [1]. The DSN has digital state variable for discrete time and can be regarded as a digital version of bifurcating neuron (BN) having rich phenomena [2]-[7]. Roughly speaking the DSN is two coupled shift registers with shift-and-reset switching that corresponds to integrate-and-fire switchings in BNs. Adjusting the coupling configuration, the DSN can exhibit a variety of spike-trains. The spike-trains coexist for initial state and correspond to digital codes through a spike-interval modulation. The PCDSN has master-slave configuration. The master side consists of N pieces of DCNs each of which outputs a spike-train corresponding to a binary code. All the spike-trains are multiplexed additionally and are transmitted to the slave side via single line. The slave side consists of N pieces of DSNs without shift-and-reset switching. The DNSs are connected by dynamic winner-take-all (WTA) function [5] [6]. As parameters are selected suitably the WTA connection can realize demultiplexing of input signal from the master and master-slave synchronization is achieved. Investigation of such synchronization phenomena provides basic information to develop spike-based multiplex communication systems and digital neural networks with efficient signal processing function as suggested in [5] [6]. VHDL simulation is also discussed for FPGA implementation and the PCDSN is compared with an analog pulse-coupled network of BNs (PCBN) having rich synchronization phenomena [6]. Motivations for studying PCDSNs include the following.

(1) It is suitable for reconfigurable hardware implementation using FPGA and so on. The hardware is useful not only for confirmation of rich dynamics but also for practical applications with "siliconization".
(2) It has advantages of both digital and spike-based systems: robustness for analog noise, suitable for computer sibilation, low power consumption, low interception probability, fast transient and so on.
(3) It relates to applications of both digital and spike-based systems: CDMA encoder [8], UWB communication [14], pseudo-random number generator [9], image processing [10]-[12], associative memories [4] [13] and so on. It should be noted that [1] presents PCDNS but does not discuss WTA function and digital vs analog comparison sufficiently.

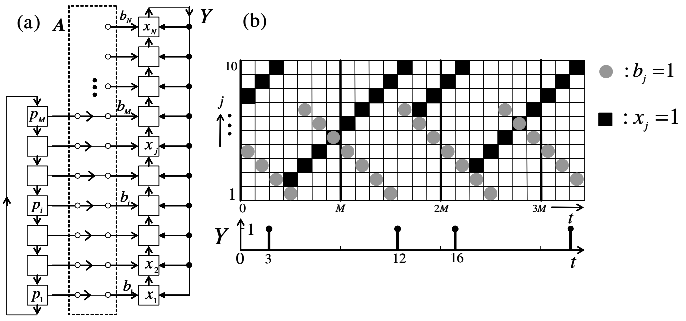


Fig. 1. (a) Digital Spike Neuron (DSN). (b) Basic dynamics for M=7 and N=10.

2 Digital Spiking Neuron

Fig.1(a) shows *digital spiking neuron* (DSN). The DSN has p-cells and x-cells. p-cells have digital state $p_i \in \{0,1\} \equiv \boldsymbol{B}$. $i \in \{0,1,\cdots,M\}$ is an index for p-cells. The p-cells are ring-coupled. Their operation is

$$p_i(t+1) = p_{i(modM)}(t), \quad i = 1,2,\cdots,M. \tag{1}$$

The p-cells oscillate periodically with period M. The state p_i is illustrated by gray circles in Fig.1(b). We introduce a notation for a state vector of the p-cells:

$$(p_1(t),\cdots,p_M(t))^t \equiv \boldsymbol{P}(t) \in \boldsymbol{B}^M. \tag{2}$$

We consider the wirings from p-cells to x-cells. Let the number of x-cells be denoted by N, where $N \geq M$. $j \in \{0,1,\cdots,N\}$ is an index for x-cells. In dotted box of Fig.1(a), the left M terminals accept the state vector $\boldsymbol{P}(t) \in \boldsymbol{B}^M$. The right N terminals output a signal vector $(b_1(t), b_2(t),\cdots,b_N(t))^t \equiv \boldsymbol{b}(t) \in \boldsymbol{B}^N$ which is referred to as a *base signal*. We define $N \times M$ *wiring matrix* $\boldsymbol{A}$. The matrix $\boldsymbol{A}$ has binary element $a_{ji} \in \boldsymbol{B}$. The base signal is given by

$$\boldsymbol{b}(t) = \boldsymbol{A}\boldsymbol{P}(t). \tag{3}$$

In the case of Fig.1(a), the elements of the wiring matrix $\boldsymbol{A}$ are givin by a_{ij}=1 for $i = j$ and a_{ij}=0 for $i \neq j$. Then, $\boldsymbol{b}$(t)=$(p_1(t), \ldots, p_M(t), 0, \ldots, 0)^t$. Each x-cells have digital state $x_j \in \{0, 1\} \equiv \boldsymbol{B}$. The x-cell has three digital inputs $b_j \in \boldsymbol{B}$, $x_N \in \boldsymbol{B}$ and $x_{j-1} \in \boldsymbol{B}$, where $x_0 \equiv 0$. If x_N=0, $x_j(t+1)$=$x_{j-1}(t)$. If $x_N(t)$=1, the $x_j(t+1)$=$x_{j-1}(t) \cup b_j(t)$. We introduce a notation for a state vector of the x-cells:

$$(x_1(t), \cdots, x_N(t))^t \equiv \boldsymbol{x}(t) \in \boldsymbol{B}^N. \tag{4}$$

We define an up-shift operator

$$S_+(\boldsymbol{x}(t)) = (0, x_1, \cdots, x_{N-1})^t. \tag{5}$$

The dynamics of the x-cells is described by

$$\boldsymbol{x}(t+1) = \begin{cases} S_+(\boldsymbol{x}(t)) & \text{for } Y(t) = 0, \\ \boldsymbol{b}(t) \cup S_+(\boldsymbol{x}(t)) & \text{for } Y(t) = 1. \end{cases} \tag{6}$$

Basic dynamics of the x-cells is illustrated by black boxes in Fig.1(b). The state x_j=1 is shifted upward. At $t = 3$, the N-th x-cell has state x_N=1. The state vector $\boldsymbol{x}$ reset to $\boldsymbol{x}(4)$=$\boldsymbol{b}(3)$, where $S_+(\boldsymbol{x}(3)$=$(0, \cdots, 0)^t$. Repeating such *shift-and-reset* dynamics, the x-cells oscillate as shown in Fig.1(b). The state x_N of the N-th x-cell is used as an output Y of the DSN. Then the DSN outputs a discrete-time spike-train as shown in Fig.1(b):

$$Y(t+1) = x_N(t) \in \boldsymbol{B}, \ t = 0, 1, 2, \cdots \tag{7}$$

Note that shift-and-reset dynamics of the DSN can be regarded as digital version of integrate-and-fire dynamics of an analog spiking neuron. The digital state $\boldsymbol{X}$(t) correspond to integration dynamics. The reset of $\boldsymbol{X}(t)$ and generation of a spike $Y(t)$=1 correspond to firing dynamics of the analog spiking neuron. Hence, we refer to the presented circuit in Fig.1(a) as *digital spiking neuron*.

The DSN in Fig.1(a) has 7 co-existing periodic spike-trains and generates one of them depending on the initial state. Fig.2 shows one of such periodic spike-train. Let t_n denote the n-th spike position of $Y(t)$, where $n = 1, 2, 3, \cdots$. In Fig.2, (t_1, t_2, t_3)=(3, 12, 16). Let us define an inter-spike interval (ISI):

$$\Delta_n = t_{n+1} - t_n. \tag{8}$$

In Fig. 2, (Δ_1, Δ_2, Δ_3)=(9, 4, 8). We consider the following ISI modulation.

$$\omega(\Delta_n) = \begin{cases} 0 & \text{for } \Delta_n \leq (M+1)/2, \\ 1 & \text{for } \Delta_n > (M+1)/2. \end{cases} \tag{9}$$

In the case of Fig. 2, the spike-train $Y(t)$ can be encoded by ISI code ($\omega(\Delta_1)$, $\omega(\Delta_2)$, $\omega(\Delta_3)$)=(1,0,1). Moreover, the DSN in Fig.1 can generate periodic spike-trains that correspond to all the 3-bit binary codes except for all 1's. In general, let the number of p-cells is $M = 2^k - 1$ and $N = \frac{3k-1}{2}$. In this case the DSN can generate periodic spike-trains that correspond to all the k-bit binary codes except for all 1's.

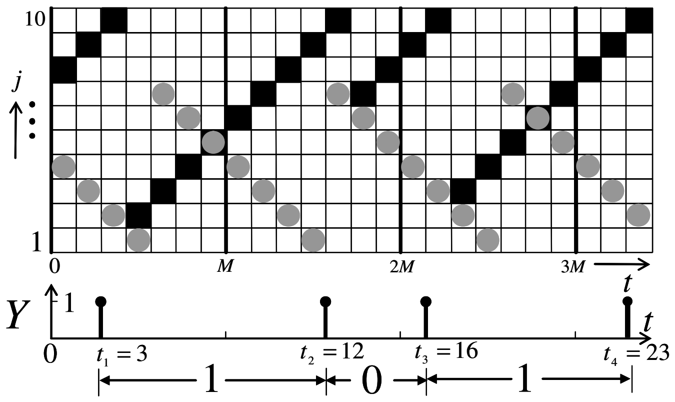


Fig. 2. ISI modulation, spike-train can be encoded by a digital code (1,0,1)

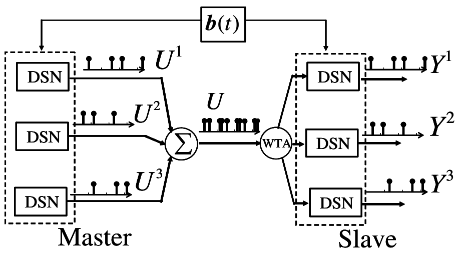


Fig. 3. Pulse-Coupled Network of DSNs(PCDSN)

3 Pulse-Coupled Network of DSNs

3.1 Master Dynamics

We propose a pulse-coupled network of DSNs (PCDSN). Fig.3 shows the PCDSN. The PCDSN has DSNs as masters and slaves. The PCDSN has a common base signal $\boldsymbol{b}(t)$. Let us consider the master in the left side. In Fig.3 the PCDSN has three DSNs (having the same parameters) as masters. Fig.4(a) shows basic dynamics of the master. The n-th master has a digital state vector $(x_1^n(t), \cdots, x_N^n(t))^t \equiv \boldsymbol{x}^n(t) \in \boldsymbol{B}^N$, where n is an index for master. In the case of Fig.3, $n \in \{1, 2, 3\}$. The n-th master outputs a spike train U^n. The spike-trains of the masters are multiplexed: $U(t)=\sum_{l=1}^{l} U^k(t)$. The dynamics of the masters is described by

$$\boldsymbol{x}^n(t+1) = \begin{cases} S_+(\boldsymbol{x}^n(t)) & \text{for } U^n(t) = 0, \\ \boldsymbol{b}(t) \cup S_+(\boldsymbol{x}^n(t)) & \text{for } U^n(t) = 1. \end{cases} \tag{10}$$

Using the ISI modulation in Eq. (9), we can encode each spike-train U^n by 3-bit codes. In Fig.4(b), (1,0,1), (0,1,1) and (1,1,0) are encoded by U^1, U^2 and U^3. So, three digital codes, (1,0,1), (0,1,1) and (1,1,0) are encoded into spike-train U.

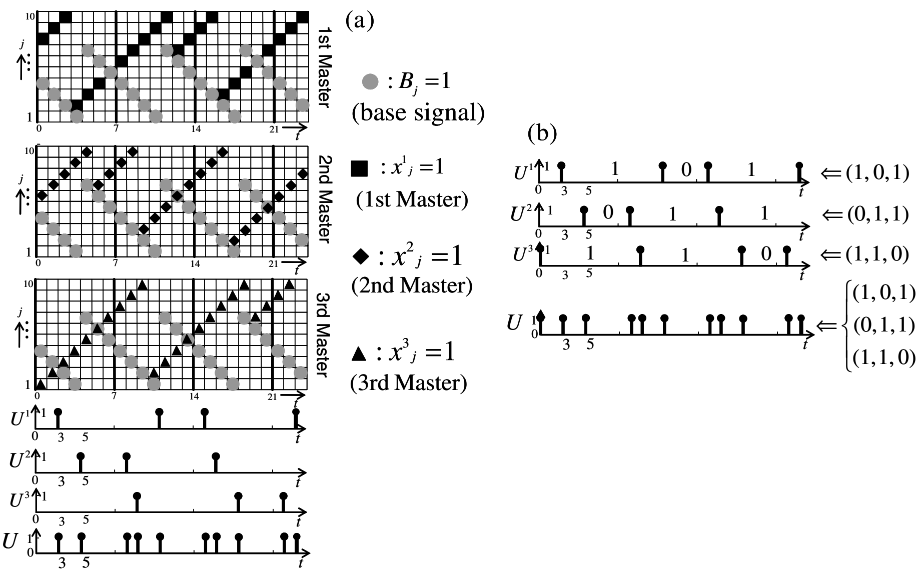

(a)
● : B_j =1 (base signal)
■ : x^1_j =1 (1st Master)
◆ : x^2_j =1 (2nd Master)
▲ : x^3_j =1 (3rd Master)
1st Master
2nd Master
3rd Master
(b)
⇐(1,0,1)
⇐(0,1,1)
⇐(1,1,0)

Fig. 4. Dynamics 0f PCDSN'masters, M=7, N=10

3.2 Slave Dynamics

Let us consider the right side of PCDSN in Fig.3. The PCDSN has three DSNs as slaves. Each m-th slave has a digital state vector $(X^m_1(t), \cdots, X^m_N(t))^t \equiv \boldsymbol{X}^m(t) \in \boldsymbol{B}^N$, where m is index for slave. In the case of Fig.DPCN, $m \in \{1, 2, 3\}$. The slave has input spike train $U(t)$. The m-th master outputs a spike train $Y^m(t)$. Let $h^m(t)$ denote position of j such that $X^m_j(t) = 1$. Then the slaves are governed by the following *winner-take-all* (WTA) based dynamics:

$$\boldsymbol{X}^m(t+1) = \begin{cases} \boldsymbol{b}(t) & \text{for } U(t) = 1 \text{ and} \\ & h^m(t) = \max_z(h^z), \\ S_+(\boldsymbol{X}^m(t)) & \text{otherwise.} \end{cases} \tag{11}$$

We explain slave dynamics based on Fig.5. At $t = 0$, all X^m_j are shifted upward. At $t = 3$, an input spike $U(3) = 1$ arrives and h^1(=8) is highest. In this case, the 1st slave becomes a *winner* and X^1_j is reset to the *base signal* $\boldsymbol{b}$ (gray circle). Then, the 1st slave outputs a spike $Y^1(3) = 1$. The 2nd and 3rd slaves are shifted upward. At $t = 5$, an input spike $U(5) = 1$ arrives and h^2(=8) is highest. In this case, the 2nd slave becomes a *winner* and X^2_j is reset to the $\boldsymbol{b}$. Then, the 2nd slave outputs a spike $Y^2(5) = 1$. The 1st and 3rd slaves are shifted upward.

Repeating such a WTA algorithm, each k-th slave in Fig.5 is synchronized with the k-th master in Fig.4 (i.e., $Y^k(t) = U^k(t)$, k=1, 2, 3). Using the ISI modulation in Eq. (9), the digital codes (0,1,0), (1,0,1), (0,0,1) can be retrieved in the slave side. In general, the 3 masters of the PCDSN in Fig.3 can multipex any combination of different 3 digital codes with 3-bits (except for (1,1,1)).

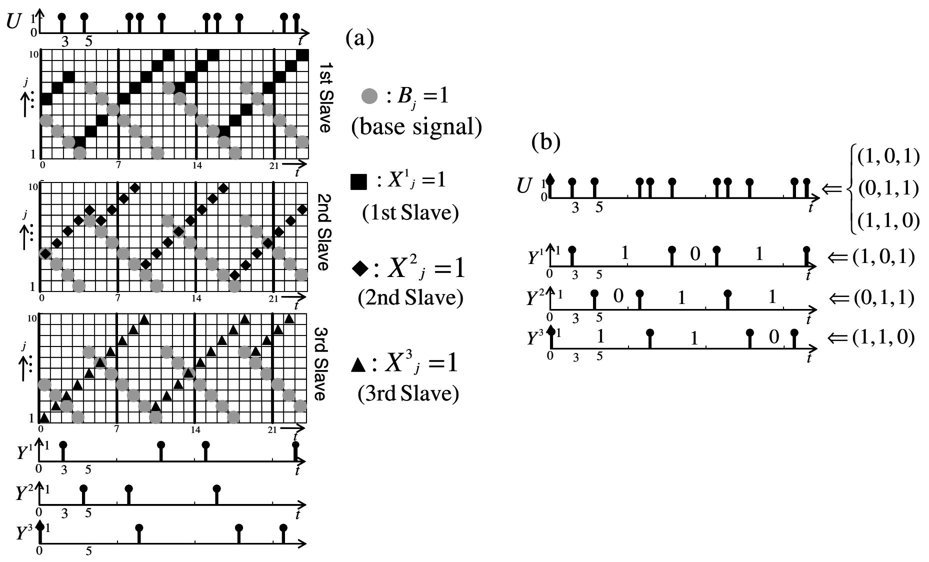

(a)
● : $B_j = 1$ (base signal)
■ : $X^1_j = 1$ (1st Slave)
◆ : $X^2_j = 1$ (2nd Slave)
▲ : $X^3_j = 1$ (3rd Slave)
1st Slave
2nd Slave
3rd Slave
(b)
(1,0,1)
(0,1,1)
(1,1,0)
U
Y^1
Y^2
Y^3

Fig. 5. Dynamics of PCDSN'slaves, M=7, N=10

The 3 slaves can retrieve the 3 digital codes. That is, the PCDSN can realize a spike-based multiplex communication. If the PCDSN has L masters and L slaves ($L \leq M$), the PCDSN can realize a multiplex communication of L digital codes. Also, the bit-length can be generalized: if each master and slave have $M = 2^k - 1$ p-cells and $N = (3M - 1)/2$ x-cells, the bit-length is given by k. In this section we considered the case of $k = 3$.

3.3 Hardware Implementation

We have implemented the DSN in a *field programmable gate array* (FPGA). Fig.6 shows an HDL implementation of the PCDSN corresponding to Fig.4 and Fig.5. We can confirm that k-th slave is synchronized by k-th master.

4 Conclusion

Here let us compare the PCDSN with a pulse-cpuled network (PCN) of analog spiking neuron in [5] as summarized in 1. First, let us compare stability. The analog spiking neuron is originally chaotic. The chaotic neuron can not realize an encoding of digital code into an output spike-train and the PCN can not realize a stable synchronization. Then we had to proposed a stabilization method for the neuron based on modulation of a firing thereshold by a higher frequency periodic signal. On the other hand the DNS has discrete state and time, and then it can realize stable operation against analog perturbations. Second, let us compare implementation. The analog PCN in [5] is implemented by discrete

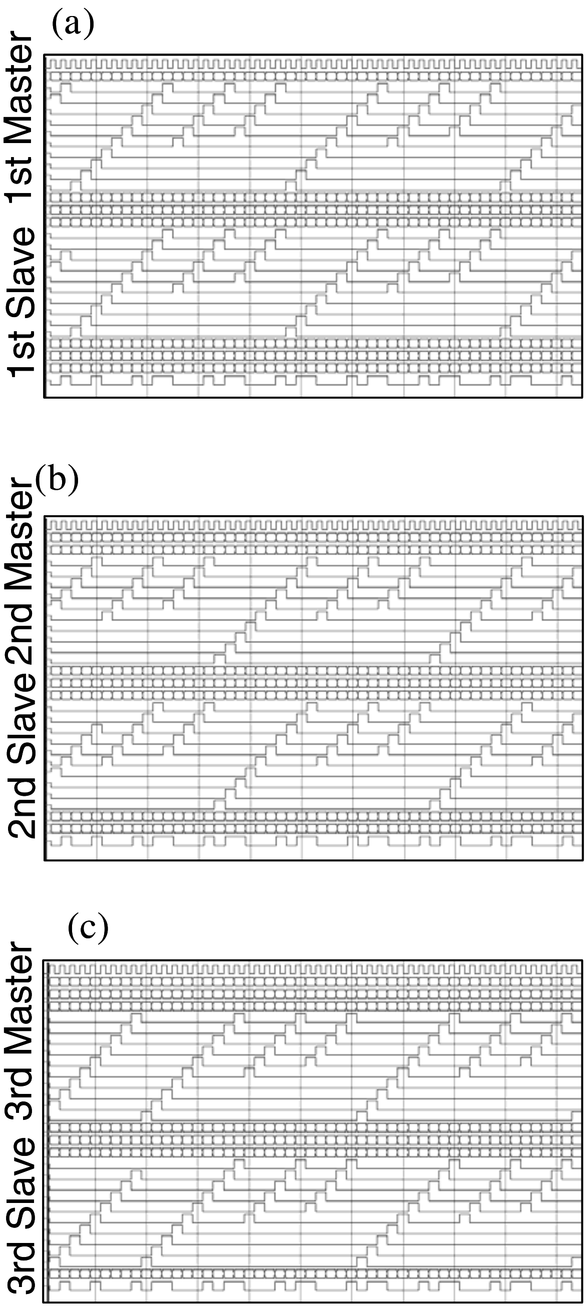


Fig. 6. FPGA implementation (M=7, N=11)

components. In order to realize a desired dynamics, analog parameters must be adjusted carefully because the system operation is very seisitive against the analog parameters. On the other hand the PCDSN can be easily implemented by an FPGA. Also the digital parameters of the PCDSN can be adjusted easily and dynamically. The dynamic parameter adjustment will be a key to realize an on-chip learning of the PCDSN in the future.

Table 1. Analog vs digital

	Analog PCN [IEEE WTA]	**PCDSN**
Master neuron	Originally chotic. Must be stabilized to realize encoding.	Stable operation and encoding against analog perturbations.
Synchronization of PCN	Each master neuron must be stabilized to realize stable synchronization.	Stable synchronization without stabilization.
Implementation	Implementation by discrete components. The analog parameters must be adjusted carefully.	Implementation by an FPGA. Easy and dynamic parameter adjustment is possible.

Future problems include: (a) detailed analysis of the ISI coding of the DSN and synchronization phneomena of the PCDSN; (b) proposing learing algorithm for the PCDSN; (c) FPGA implementation; and (d) comparison with existing Pulse Coupled Neural Networks.

References

1. H. Torikai, H. Hamanaka and T. Saito, Novel digital spiking neuron and its pulse-coupled network: spike position coding and multiplex communication, Proc. IJCNN, pp. 3249 - 3254, 2005.
2. L. Glass and M. C. Mackey, A simple model for phase locking of biological oscillators, J. Math. Biology, 7, pp.339-352, 1979.
3. R. Perez and L .Glass, Bistability, period doubling bifurcations and chaos in a periodically forced oscillator, Phys. Lett. 90A, 9, pp.441-443, 1982.
4. G. Lee and N. H. Farhat, The bifurcating neuron network 1, Neural networks, 14, pp. 115-131, 2001.
5. H. Torikai, T. Saito and W. Schwarz, Synchronization via multiplex pulse-train, IEEE Trans. Circuits Syst. I, 46, 9, pp.1072-1085, 1999.
6. H. Torikai and T. Saito, Synchronization phenomena in pulse-coupled networks driven by spike-train inputs, IEEE Trans. Neural Networks, 15, 2, pp.337-347, 2004.
7. Y. Kon'no, T. Saito and H. Torikai, Rich dynamics of pulse-coupled spiking neurons with a triangular base signal, Neural Networks, 18, pp. 523-531, 2005.

8. S. C. Kim and B. G. Lee, A theory on sequence spaces and shift register generators, IEEE Trans. Comm., 44, 5, pp. 609-618, 1996.
9. S. Guan and S. Zhang, An evolutionary approach to the design of controllable cellular automata structure for random number generation, IEEE Trans. Evolutionary Computation, 7, 1, pp. 23-26, 2003.
10. J. J. Hopfield and A. V. M. Herz, Rapid local synchronization of action potentials: Toward computation with coupled integrate-and-fire neurons, Proc. Natl. Acad. Sci., 92, 15, pp. 6655-6662, 1995.
11. S. R. Campbell, D. Wang and C. Jayaprakash, Synchrony and desynchrony in integrate-and-fire oscillators, Neural Comput., 11, pp. 1595-1619, 1999.
12. H. Nakano and T. Saito, Grouping synchronization in a pulse-coupled network of chaotic spiking oscillators, IEEE Trans. Neural Networks, 15, 5, pp. 1018-1026, 2004
13. E. M. Izhikevich, Weakly pulse-coupled oscillators, FM interactions, synchronization, and oscillatory associative memory, IEEE Trans. Neural Networks, 10, 3, pp. 508-526, 1999
14. G. M. Maggio, N. Rulkov and L. Reggiani, Pseudo-chaotic time hopping for UWB impulse radio, IEEE Trans. Circuits Syst. I, 48, 12, pp. 1424-1435, 2001.
15. H. Torikai, H. Hamanaka and T. Saito, Reconfigurable Digital Spiking Neuron and its Pulse-Coupled Network: Basic Characteristics and Potential Applications IEEE Trans. Circuits Syst. II, 2006.

A Bit-Stream Pulse-Based Digital Neuron Model for Neural Networks

César Torres-Huitzil

Computer Science Department, INAOE
Apdo. Postal 51 & 216, Puebla, Mexico
ctorres@inaoep.mx

Abstract. An area-efficient pulse mode hardware neuron model with sigmoid-like activation function for artificial neural networks implementations is presented. The neuron activation function is based on an enhanced version of the voting circuit previously reported in the literature. The proposed model employs pulse stream computations and statistical saturation to deal with the nonlinearities inherent to neural computations. This approach provides an embedded hardware implementation feasibility favoring silicon area efficiency rather than speed. Implementation results on Field Programmable Gate Array (FPGA) technology shows the proposed neuron model requires fewer hardware resources than previous implementations and it is especially attractive for neurons with wide receptive fields in large neural networks. Experimental results are presented to highlight the improvements of the proposed model.

1 Introduction

Although technological improvements and the inherent computational capabilities of neural networks have increased their utilization in a wide range of applications, their full utilization in practical systems is still limited [1]. In spite of most applications of neural networks are implemented using general purpose processors, currently, more research is done for developing Very Large Scale Integration (VLSI) implementations due to the potential for real-time performance and embedded processing. Hardware complexity issues of neural networks such as area greedy-operators and the high interconnectivity requirements related to their connection-centric structure, present difficulties to be implemented efficiently into hardware. The need to deal with strong implementation and application constraints is a major research domain to find ways to map neural connectivity and nonlinear functionality onto hardware [2]. The hardware plausibility is related to a very fine grain parallelism that fits parallel hardware devices, as well as to the emergence of very large digital reconfigurable devices, such as FPGAs, that become able to handle both adaptability and massive parallelism of neural networks on a single chip [3]. While these are benefits for digital implementations, the large amount of logic required is still a drawback to overcome, Motivated by the potential gains in the design cycle, the availability of high density reconfigurable devices, and the development of high level design tools, in this paper, the FPGA implementation of area saving pulse-based neural networks is addressed.

Recently, an increasing number of arithmetical and architectural choices are being proposed to alleviate the hardware implementation problems of neural networks searching for the potential massive parallelism driven by low area solutions [2]. An effective approach for the neural networks implementation is the use of pulse stream based computations [4]. Particularly, stochastic pulse modulation [5], where signal strengths are represented by probabilities and coded as random pulse streams, has gained popularity since neural computations are performed with simple computational elements, i.e. logic gates, and the ease to handle signals under noise and power constraints [6][7]. However, the flexibility and accuracy to deal with nonlinearities in the activation function under this approach are limited. Recent improvements have lead to the development of new kind of hardware-friendly neuron models with enhanced activation function. Above all, the use of a voting circuit instead of an OR gate has been beneficial to improve the characteristics of the activation function at the cost of increasing the circuit complexity [8].

In this paper, a pulse-coded approach for an area-efficient neuron hardware model is presented which can benefit from technology improvements to allow the implementation of a large number of neurons on a single chip. The neuron model is an adapted and enhanced version of the proposed by Hikawa and the recent architecture presented in [9]. An optimization for the voting circuit to reduce the silicon area is presented through the use of bit-oriented computations. The neuron activation function is analyzed through simulations and experiments and the enhanced neuron model is implemented on an FPGA device. The rest of the paper is organized as follows. Section 2 presents a brief description of the voting circuits reported previously in the literature. Section 3 provides the details and the proposed enhancements to the voting circuit in the neuron hardware model. Results on the VHDL modeling and the FPGA implementation of a single neuron is presented in section 4. A comparison and a discussion of the obtained results are presented in section 5. Finally conclusions and further work are presented.

2 Pulse Neuron Models Based on the Voting Circuit

In the original neuron architecture presented in [8], stochastic streams are used to represent the weights and the neuron inputs. Digital Frequency Synthesizers (DFSs) are used to perform multiplications and model the synapse functionality. A voting circuit is employed for modeling the sigmoid-like activation function through statistical saturation. A block diagram of the voting circuit is shown in figure 1.

In general terms, on each clock cycle, if the excitatory inputs, N_E, exceed the inhibitory ones, N_I, then an output pulse, synchronized with the clock signal, is generated. The circuit uses statistical saturation to implement the activation function. The S-stage register is used to reach saturation faster since its effect is equivalent to have more pulses simultaneously, Thus, changing S, the steepness of the activation function can be changed. In spite of the advantages of the voting circuit to reproduce sigmoid-like activation function with reduced hardware resources, it still uses area-consuming arithmetic modules such as adders and comparators. For a full description of the neuron hardware model see [8].

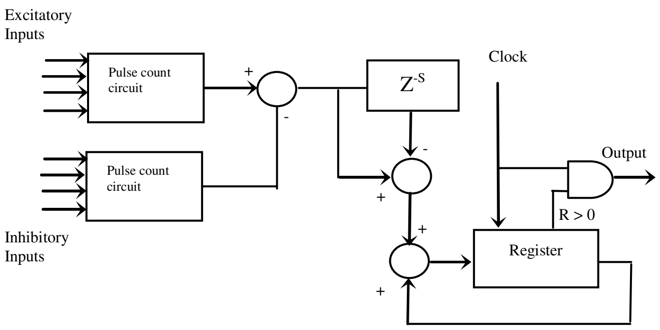


Fig. 1. Block diagram of the pulse mode voting circuit with adjustable activation function adapted from [8]

A recent work proposes a new voting circuit having as main goal to reduce further the silicon area [9]. For this purpose, a bit manipulation and time-multiplexing scheme was chosen as opposed to integer operations in [8] to generate a pulse whenever the excitatory inputs exceed the inhibitory ones.

A block diagram of the bit oriented voting circuit is shown in figure 2. It is based on a shift register and a couple of multiplexers to scan the pulse inputs on each clock cycle. The register is built with N_E+N_I+1 cells. At the initial state the excitatory cells are reset to zero and the inhibitory one set to one. On each clock cycle data is moved downwards if an inhibitory input is present and upwards if an excitatory pulse is present. If excitatory and inhibitory pulses are both present the shift operation is disabled. At the end of the input scanning, a pulse output is generated if the inhibitory exceeds the excitatory inputs.

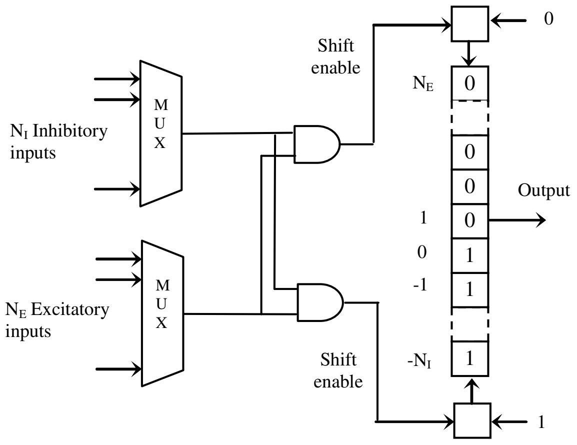


Fig. 2. Voting circuit oriented to bit-stream computation adapted from [9]

The bit-oriented approach reduces the hardware resource utilization of the original voting circuit but clearly increases the latency of the neuron hardware model because of the sequential input scanning, thus an area-performance tradeoff exist. In spite of the improvements under this approach, from an area efficiency point of view, a disadvantage of this voting circuit is that the number of cells in the shift register increases directly with the number of inputs to the voting circuit, which is a drawback in terms of area efficiency if full connected and high density neural networks are being utilized for solving a given application. In the following section an improved version of the voting circuit is implemented though a rather simple modification but with good results.

3 Enhanced Voting Circuit

In this section a description of the proposed enhanced voting circuit for the pulse-based hardware neuron model is presented. As mentioned in the previous section the latency of the voting circuit with bit-oriented computations is increased by several clock cycles per output evaluation equal to the number of excitatory or inhibitory inputs. According to results reported in [9], the bit-oriented voting architecture shows promising figures when area is privileged with respect to performance. However, an improvement can be done to the proposed circuit that reduces further the number of registers significantly. The enhanced voting circuit proposed in this work is shown in the block diagram of figure 3.

The voting circuit is essentially composed of two multiplexers for scanning the inhibitory and excitatory inputs with a common control word and a binary counter. The enhanced voting circuit mainly changes the shift register by an up/down counter that counts down if an excitatory input is present at a given time and counts up if an inhibitory input is present. If both excitatory and inhibitory inputs are present at the same time, the counter is disabled and its internal state is not modified, as well as if not pulses are present in the inputs. Taking the internal state of the counter as a 2-complement number, at the end of the input scan, the most significant bit (MSB) determines if a pulse is generated or not. If the MSB is one then the number of excitatory inputs is greater than the inhibitory inputs.

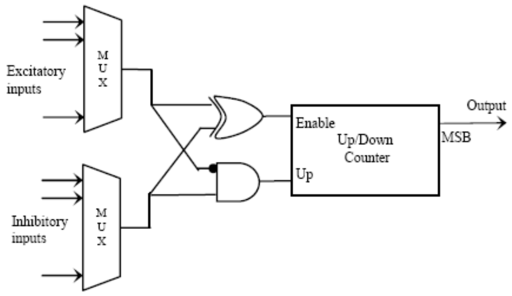


Fig. 3. Block diagram of the enhanced voting circuit

The three-state finite state machine, illustrated in figure 4, shows, conceptually, the counter functionality according to excitatory and inhibitory inputs.

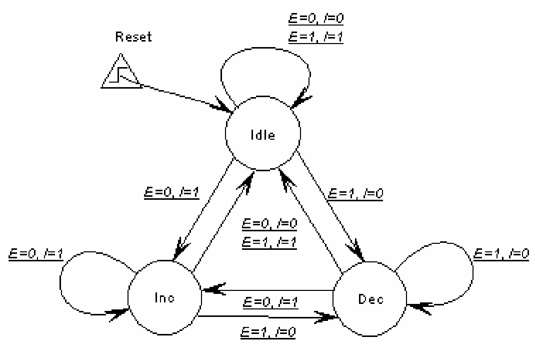


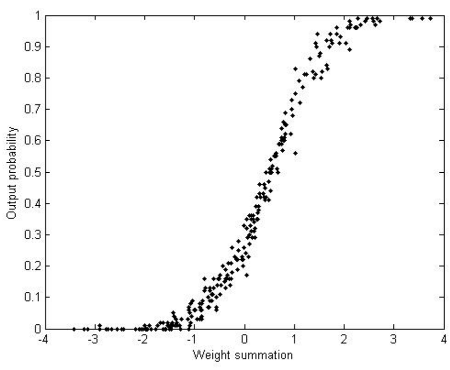
Fig. 4. Finite state machine showing the counter behavior in the voting circuit. *E* and *I* stand for excitatory and inhibitory inputs, respectively.

The voting circuit optimization can be seen as a binary coding of the neuron internal state in contrast with the one-hot coding in the architecture proposed in [9]. With this rather simple modification, the enhanced voting circuit reduces the number of registers. The number of registers used in this approach is in the order of base-two logarithm of the number of inputs. This optimization is especially important when the number of inputs to the neuron is large and for high dense and full connected neural networks. As in the case of the [9] the latency of the circuit is increased by some clock cycles per output evaluation equal to the number of excitatory or inhibitory inputs. However, the hardware resource utilization is reduced and the advantages of the flexibility of the activation function characteristic are preserved. In the following section, the VHDL modeling and FPGA implementation results are presented.

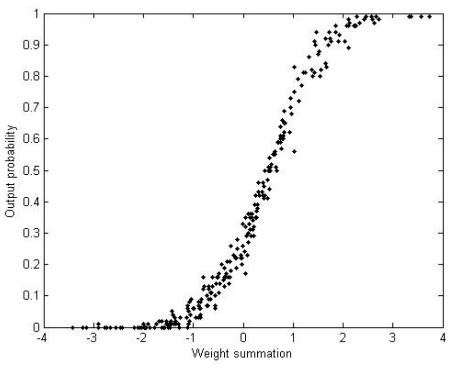
4 Implementation Results of a Single Neuron

The proposed enhanced voting circuit and the new hardware neuron model have been modeled, simulated and validated in VHDL Hardware Description Language. A full parameterizable neuron model was developed in order to carry out several experiments for different number of inhibitory and excitatory inputs to evaluate the hardware requirements for FPGA implementation.

The neuron model was simulated and validated following a similar procedure exposed in [8][9]. A neuron with five inputs was defined and the inputs were forced to be constant streams with probabilities equal to one. The weights were changed randomly between +1 y -1 using a 9-bit 2-complement fixed point representation, stored in a file and exported as a test vector from MATLAB to the digital logic simulator. In the experiments, the synapses were implemented with DFSs, for details of this module see [8]. The neuron was simulated in Modelsim and the activation function values were extracted with MATLAB to plot the shaped activation function. As expected, the resultant values closely resemble the logistic sigmoid activation function. The results of the activation function produced by the neuron model used in [9] and the proposed in this work are shown in figure 5(a) and 5(b), respectively.



(a)



(b)

Fig. 5. (a) Sigmoid-like activation function obtained with the shift-register based voting circuit and (b) activation function obtained with the up/down counter based voting circuit

To obtain the activation function plot in this experiment, the neuron output pulses were analyzed in a time window of 256 operation cycles and then normalized using the maximum number of pulses. Each neuron operation cycle requires 5 main clock cycles to scan all the inputs and produce a single output result. Thus, the processing speed for this class of hardware implementation is not so fast due to the increased clock cycles required for the input scanning and pulse accumulating operations in a bit-stream pulse-based approach.

According to experiments and results shown in figure 5, the activation function produced for both variants of the voting circuit are identical. The variance in the neuron output is much smaller than a pure stochastic implementation where the voting circuit is changed by a simple OR logic gate. Similar results in changing the steepness of the activation function through the reset of the voting circuit are obtained as those reported in [9]

For hardware resource utilization and comparison purposes, a VHDL five-input neuron model was synthesized targeted to an FPGA Virtex device with Xilinx ISE 7.1i tools. The FPGA synthesis results for the voting circuit used in the 5-input neuron model are summarized in table 1.

Table 1. Virtex II synthesis results for the proposed voting circuit and the corresponding reported in [9]

	Proposed design	Reference design
LUTs	14	17
Flip-Flops	4	11
Slices	8	9
IO's	11	

For comparison purposes the results obtained in [9] for the voting circuit, reference design, are also shown in table 1. As seen in table 1, a hardware resource reduction is obtained mainly in the number of registers used in the voting circuit without increasing the complexity of the associated hardware for the internal control logic and the interconnection logic with other neurons in a given network. The reported results do not include the hardware resources for the synapses hardware modules since the architectural optimization was focused on the voting circuit.

In order to provide an estimation, in different implementation scenarios, of the hardware resource utilization of the complete hardware neuron model proposed in this work, the FPGA synthesis results of the neuron model for different number of inputs, 5, 8 and 16 inputs, are summarized in table 2.

Table 2. Virtex II synthesis results for the complete neuron model for different number of inputs. The neuron synapses are modeled by Digital Frequency Synthesizers (DFSs) as proposed in [8].

	5 inputs	8 inputs	16 inputs
LUTs	75	109	207
Flip-Flops	59	92	181
Slices	46	71	136

5 Discussion

As shown in the previous section the hardware resource utilization is lower in the proposed enhanced voting circuit than previous implementations. Particularly, the number of registers is significantly reduced. As exposed in [9], in the presence of N_E excitatory inputs and N_I inhibitory ones, the number of registers in the voting circuit, N_R, is given by equation 1:

$$N_R = N_E + N_I + 1, \tag{1}$$

In the enhanced voting circuit proposed in this work, the number of registers is proportional to the base-two logarithm of the maximum number of inputs. Formally, the number of used registers is given by equation 2:

$$N_R = \lceil \log_2(\max(N_E, N_I)) \rceil + 1, \tag{2}$$

where [°] denotes the closest integer greater than the number in the argument.

The comparison of equations1 and 2 shows the achievable hardware reduction for different input number when the enhanced voting circuit is used. According to the results in table 2, and targeting the neuron model to a XC2V2000-5 Virtex-II device, it would be possible to map over 150 8-input neurons with an estimated clock frequency of 150 MHz, a neuron density and performance good enough for neural embedded processing such as the implementation of neurocontrollers for mobile robots. Most of the hardware resource utilization is related to the DFS type synapse multipliers. For the current implementation, 9-bit wordlength for weights, the hardware resources required for each synapse are 11 slice flip-flops, and 12 4-input LUTs for a total occupation of 11 slices of the FPGA device. For hardware reduction it would be possible to use stochastic synapse multipliers, AND gates for instance, at the cost of a greater variance in the neuron output due to the inherent variance in estimating the value of a stochastic signal through simple logic gates.

As stated in [10], due to the wide variety of network architectures and hardware implementations reported in the literature, it is difficult to establish faithful figures of merit to highlight the hardware capabilities of a given approach since aspects such as the implementation technology and accuracy are diverse for different implementation solutions. A common hardware performance metric is the connections-per-second (CPS), which is defined as the rate multiplication and accumulate operations are performed during the recall processing in the neural network. For the proposed implementation approach, the CPS for a specified neural network with N_N neurons and N inputs per neuron, usually the same number of excitatory and inhibitory inputs is used, is given by equation 3.

$$CPS = \frac{F \cdot N_N}{N_W} \tag{3}$$

where, F is the maximum clock frequency achievable by the targeted hardware technology, and N_w is the number of operation cycles in the time window used to evaluate the neuron output.

Note that in equation 3, the number of inputs per neuron, N, does not appear as a significant factor in the performance metric. This is an effect of the neuron input scanning process that limits the achievable speed by a factor equal to N, the increased number clock of cycles required to accomplish a computation, but with a significant hardware utilization reduction due to the bit-oriented processing.

According to the reported results, it is possible to place around 8-input 150 neurons into and XC2V2000-5 Virtex-II device. This means that a network with 1200 synapses could be placed into a single chip. On the other hand, 256 operation cycles (256×8 clock cycles) are required for an update of the neuron. With a maximum clock frequency of 150 MHz, the overall time for an update to occur is about 13.6 microseconds. The number of updates per second is around 75000. For a network with 1200 synapses, a maximum speed of 87 millions of CPS can be achieved according to equation 3.

Although the accuracy and number of neurons which can fit onto a single chip is similar to the results presented in [10], the processing speed of the proposed approach is significantly less, one magnitude order, due to the underlying bit-oriented processing. However, as mentioned previously, a detailed and careful analysis is required for a more fair comparison. The reported timing is sufficient for good performance in embedded applications. However, much work stills remains to be done in order to show the usefulness of this approach in specific applications where such as hardware plausibility and possibilities could be fully exploited such as in the case of neurcontrollers for mobile robots where small and medium size neural networks are required.

According to the presented results and literature revision, it is possible to remark the following key aspects of pulse-mode hardware implementation of neural networks.

1. The neural network hardware complexity, area-greedy operators and the highly dense interconnection scheme, is reduced by the nature of pulse coding and bit stream computations.
2. The use of a voting circuit instead of simple logic gates, as usual in stochastic computations, is beneficial to improve the accuracy and flexibility of the activation function.
3. The variance of the pulse-based neuron output depends on factors such as, nature of the voting circuit, the number of inputs to the neuron and the accumulation time for neuron operation.
4. An area-performance tradeoff can be tailored for a given application by controlling neuron parameters such as the precision of the synapses and the accumulation time required to evaluate the neuron output.
5. The spatial efficiency can be improved at the cost of temporal efficiency. A rough estimation indicates that ten times integration can be achieved and the processing speed is reduced by a similar factor when compared to conventional digital implementations.
6. Synaptic plasticity or neural network training is feasible to be implemented on-chip since relatively simple logic circuits are used to produce learning.

6 Conclusions

In this paper an improved architecture for the voting circuit for a stochastic pulse coded neuron with a sigmoid-like activation function has been presented. Even with the rather simple optimization, the area saving for hardware implementation is considerable especially for full connected neural networks whit large receptive field neurons, i.e., neurons with a large number of inputs.

As the architecture is functionally equivalent to the one proposed in [9], it posses the same latency drawback that tradeoffs performance. The presented results show promising alternative implementation options for different application constraints where area efficiency is a key aspect. The hardware neuron models can be used to build high density networks large enough, over 10^3 neurons in most up to date reconfigurable devices, to solve complex problems on a single chip favoring an embeddable neural processing. Most work is required in order to show the usefulness of the proposed approach in applications where the neuron density and performance could be exploited. For this purpose, an integrated design framework is being developed to allow the automatic synthesis of neural networks from high level specifications where a hardware/software codesign is beneficial for the hardware implementation of neural networks. Another key aspect to be addressed is the on-chip learning capabilities for embedded real-world applications and for different hardware implementation options of neuron models.

References

[1] B. Colwell, “Machine Intelligence Meets Neuroscience,” IEEE Computer, Vol. 38, No. 1, pp. 12-15, January 2005.

[2] Leonardo Maria Reyneri, “Implementation Issues of Neuro-Fuzzy Hardware: Toward HW/SW Codesign,” IEEE Transactions on Neural Networks, Vol. 14, No. 1, pp. 176-194, January 2003.

[3] S. W. Olridge and S. J. E. Wilton, “A novel FPGA Architecture supporting Wide, Shallow Memories,” IEEE Transactions on Very Large Scale Integration Systems, Vol. 13, No. 6, pp. 758-762, June 2005.

[4] Leonardo Maria Reyneri, “A Performance Analysis of Pulse Stream and Fuzzy Computing Systems,” IEEE Transactions on Circuits and Systems, Vol. 52, pp. 642-660, October 1995.

[5] Bradley D. Brown, Howard C. Card, “Stochastic Neural Computations I: Computational Elements,” IEEE Transaction on Computers, Vol. 50, No. 9, pp. 891-905, September 2001.

[6] David Braendler, Tim Hendtlas, and Peter O’Donoghue, “Deterministic Bit-stream Digital Neurons,” IEEE Transactions on Neural Networks, Vol. 13, No. 6, pp. 1514-1525, November 2002.

[7] Shigeo Sato, Ken Nemoto, Shunsake Akimoto, Mitsunaga Kinjo, and Koji Nakajima, “Implementation of a New Neurochip Using Stochastic Logic,” IEEE Transactions on Neural Networks, Vol. 14, No. 5, pp. 1122-1127, September 2003,

[8] H. Hikawa, “A New Digital Pulse-Mode Neuron with Adjustable Activation Function,” IEEE Transactions on Neural Networks, Vol. 14, No. 1, pp. 236-242, January 2003..

[9] Matteo Martincingh and Antonio Abramo, “A New Architecture for Digital Stochastic Pulse-Mode Neurons Based on the Voting Circuit,” IEEE Transactions on Neural Networks, Vol. 16, No. 6, pp. 1685-1693, November 2005.

[10] David Braendler, Tim Hendtlass, and Peter O’Donoghue, “Deterministic Bit-Stream Digital Neurons,” IEEE Transaction on Neural Networks, Vol. 13, No. 6, November 2002. pp. 1514-1525.

From Hopfield Nets to Pulsed Neural Networks

Ana M.G. Guerreiro[1] and Carlos A. Paz de Araujo[2]

[1] Rio Grande do Norte Federal University, Natal RN, Brazil
[2] University of Colorado, Colorado Springs CO, USA

Abstract. Considering the first two generations of Artificial Neural Networks, Hopfield model is the only active system. Studying this type of network, a relation between this artificial neural network and the third generation, characterized by spiking neurons, was noticed. This paper presents the relationship between the Hopfield Neural Networks and the Pulsed Neural Networks. This relation is shown by the integration of the Hopfield neuron model, ending in an integrate-and-fire model, with the appropriate choice of the input kernels.

1 Introduction

Mathematical models for integrate-and-fire neurons can be traced back to 1907. There exists a number of variations to this model, which are described and compared in a recent survey [1]. These mathematical models for spiking neurons do not provide a complete description of the biological neuron. The spiking models [2], however, are of great importance because they represent more closely the manner in which signals are transmitted in biological neural networks where there are often voltage (action potential) spikes rather than analog signals. We need to recognize, however, that there is no single mathematical model that can capture all aspects of neuronal signaling.

The Pulsed Neural Networks models uses a spiking neuron as an computational unit. Biological neural systems use the timing of a single action potential to encode information [3], [4]. This model proposes a better approximation to the biological neural system than the first and second generations.

The first and second generations are based on threshold units and non-linear activation functions. These two generations seem to be passive systems, and thus unable to process continuous time signals, or even discrete time with varying attributes. Only the Hopfield model is an active system. A link between the Hopfield Model and the pulsed neurons is found, when an integration of a Hopfield model was done and it becomes the basis of the integrate-and-fire model.

The paper is organized as follow: section 2 shows a review of the well known Hopfield model, followed in section 3 by a review of the integrate-and-fire model in continuous and discrete time. Section 4 shows the integration of the Hopfield model and the relation with the integrate-and-fire model. Finally in section 5, conclusions are giving.

2 Hopfield Model

The architecture of recurrent (cyclic) neural networks is, in general, a cyclic graph, and the computations of these devices may not terminate in a bounded number of steps. Special attention has been paid to Hopfield networks.

The Hopfield network is a dynamic system which can be used as an associative memory with no hidden neurons. An associative memory is a very important resource for intelligent behavior.

The Hopfield model of a neuron is shown on figure 1. By applying Kirchoff's current law to the input node of the nonlinearity of Figure 1, we get:

$$\mathrm{C}_j \frac{dv_j(t)}{dt} + \frac{v_j(t)}{\mathrm{R}_j} = \sum_{i=1}^{N} \mathrm{w}_{ji}\mathrm{x}_i(t) + \mathrm{I}_j \tag{1}$$

The capacitive term on the left side adds dynamics to the neuron model. The output can be described as:

$$\mathrm{y}_j(t) = \varphi(v_j(t)) \tag{2}$$

The activation function relating the output is the logistic function:

$$\varphi(v_j) = \frac{1}{1 + \exp(-v_j)} \tag{3}$$

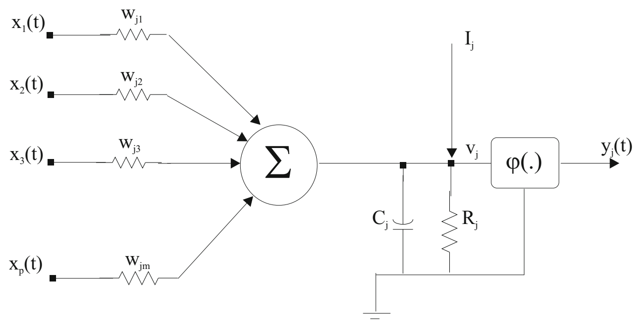


Fig. 1. The Hopfield model

The dynamics of the network is constrained by a Liapunov, or energy function, E, which is a bounded function defined on the state space decreasing along any productive computation. It follows from the existence of such a function that the network state converges towards some stable state corresponding to a local minimum of E. The energy equation can be given as:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ji}x_i y_j + \sum_{j=1}^{N} \frac{1}{R_j}\int_0^{x_j} \varphi_j^{-1}(x)dx \tag{4}$$

In the case of discrete time the energy function is reduced to the first term and the second term becomes negligibly small as shown in [5]. This approach is very similar to the spin-glass model in solid state physics. It is also similar to a one-dimensional Ising model. An integration of the Hopfield model leads to the integrate-and-fire model as shown in section 4.

3 Pulsed Neural Networks

The spiking neuron or pulsed neuron model has the characteristic of generating a spike, or a pulse, when a certain threshold is reached. The threshold is a determined value that biologically depends on the physiological characteristics of the neurons. The synapses are modelled as leaky integrators representing the behavior of the signals when they pass through neuron interconnections. Based on this concept the integrate-and-fire model was developed. There are continuous and discrete time versions of this model. Now we are going to describe the continuous time type 2.

A neuron i is said to fire, if the state variable, $mp_i(t)$, reaches a threshold ν. The moment of firing time is defined as $t_i^{(f)}$. After the neuron fires, there is a period called the refractory period in which the neuron cannot fire.

The mathematical description of the membrane potential can be summarized as:

$$mp_i(t) = \sum_{t_i^{(f)} \in F_i} \eta_i(t - t_i^{(f)}) + \sum_{t_j^{(f)} \in \Gamma_j} \sum_{t_j^{(f)} \in F_j} \omega_{ij} \varepsilon_{ij}(t - t_j^{(f)}) \tag{5}$$

where $mp_i(t)$ is the state variable of neuron i, $t_i^{(f)}$, $t_j^{(f)}$ are the firing times of neuron i and neuron j, respectively, F_i is the set of all firing times of the neuron i, Γ_i is the set of the neuron j presynaptic of i, $\eta_i(.)$ is the refractioness kernel, ω_{ij} is the adaptable weights, and $\varepsilon_{ij}(.)$ is the kernel to model the postsynaptic of neuron i.

The refractory kernel can be modelled as:

$$\eta_i(t - t_i^{(f)}) = -\eta_o \exp\left(\frac{-(t - t_i^{(f)})}{\tau_i}\right) h(t - t_i^{(f)}) \tag{6}$$

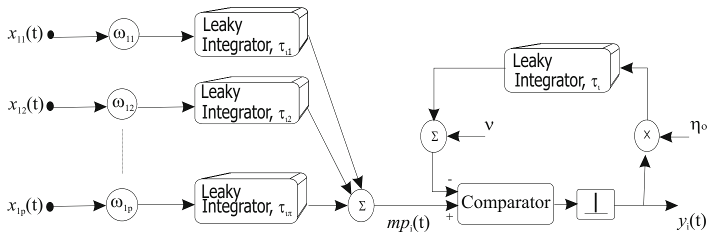
where $h(.)$ is the step function, τ_i is the decay time constant of the neuron i, and η_o is the amplitude of the refractioness kernel.

The presynaptic impulses is the pulse response of an integrator that can be described as:

$$\varepsilon_{ij}(t - t_j^f) = \exp\left(-\frac{(t - t_j^{(f)})}{\tau_{ij}}\right) h(t - t_j^{(f)}) \tag{7}$$

where τ_{ij} is the decay time constant related to the synaptic ij.

For a better understanding of how the integrate-and-fire neuron operates, a Simulink model in MATLAB was built. The neuron model has two presynaptic inputs ($j = 2$). The inputs are scaled by the weights, ω_1 and ω_2 and pass through



$x_{11}(t)$
$x_{12}(t)$
$x_{1p}(t)$
ω_{11}
ω_{12}
ω_{1p}
Leaky Integrator, τ_{i1}
Leaky Integrator, τ_{i2}
Leaky Integrator, $\tau_{i\pi}$
Leaky Integrator, τ_i
Σ
ν
η_o
X
Comparator
$mp_i(t)$
$y_i(t)$

Fig. 2. The continuous integrate-and-fire neuron model

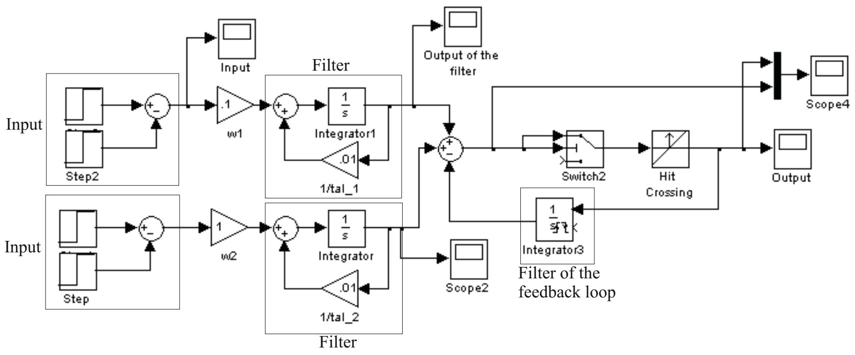


Fig. 3. The simulink model of the continuous integrate-and-fire neuron model

the integrators. A third integrator is responsible for the refractory period. All the signals are summed, resulting in the membrane potential or soma. The membrane potential is composed of *switch* and *hit crossing* modules, which simulate the charging and threshold of the soma. The firing of the axon is shown on the scope of the block diagram on Figure 3.

Two simulations were done. The first simulation uses a threshold value of 0.5 volts which is set in the switch and hit crossing blocks of figure 3. The neuron emits a spike when the membrane potential reaches the threshold. The threshold value is increased by 0.5 volts in the second simulation. Figure 4 (A) shows the membrane potential curve (dotted line) as soon as the the threshold (black line) is reached, a spike is generated. Figure 4 (B) shows the same simulation but with the threshold value set at 1.0 volt. The spike of the second simulation occurs later than the spike in the first one, because with the same conditions we only increased the threshold value by taking more time for the soma value to reach the threshold.

Because, Boolean value spikes are more analogous to the biological neuron (for coding) it is sometimes more suitable to use the discrete time version of the integrate-and-fire model. The continuous time model has a leaky integrator, which accumulates incoming spikes from the preceding neurons and decreases their effect depending on the time arrival of the spikes. An integrator circuit can be designed with a resistor and a capacitor. The RC circuit component values

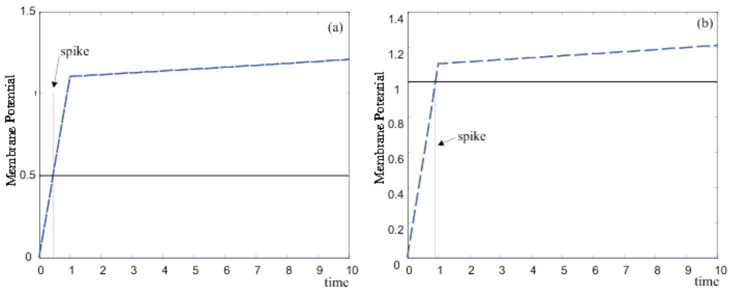


Fig. 4. (A) Threshold of 0.5 volts, and 1.0 volts

are responsible for the decay constant of the circuit. The impulse response of the RC circuit can be described as:

$$h(t) = \frac{1}{RC} \exp\left(\frac{-t}{RC}\right) u(t) \tag{8}$$

Considering $t = nT$:

$$h(nT) = \frac{1}{\tau} \exp\left(\frac{-nT}{\tau}\right) \tag{9}$$

where τ = RC. Taking the Z-transform of the (9), we obtain:

$$\mathbf{Z}\ (h(nT)) = \sum_{n=0}^{\infty} \left(\frac{1}{\tau} \exp\left(\frac{-nT}{\tau}\right)\right) z^{-n} = \frac{1}{1 - \exp\left(\frac{-T}{\tau}\right) z^{-1}} (10) \tag{10}$$

The membrane potential can now be described using discrete kernels as:

$$\begin{aligned} mp_i(n) &= -\{\eta_o y(n) + \eta_o \exp\left(\frac{-T}{\tau_i}\right) u_{fi}(n-1) + \nu\} \\ &\quad + \sum_{j \in \Gamma_i} (\omega_{ij} x_{ij}(n) + \exp\left(\frac{-T}{\tau_{ij}}\right) u_{fij}(n-1)) \end{aligned} \tag{11}$$

where T is the time slice representing the duration of the action potential, u_{fi} and u_{fij} are the infinite impulse response filter output of the feedback loop and the input dendrite tree, respectively.

4 Integrating the Hopfield Neuron Model

Equation (1) is rewritten as

$$C_j \frac{dv_j(t)}{dt} = -\frac{v_j(t)}{R_j} + \sum_{i \in \Gamma_j} c_{ij} x_i(t) + I_j \tag{12}$$

The neuron j will receive input from many different neurons $i \in \Gamma_j$. The current $I_j(t)$ can be seen as the outgoing current pulse of negligible width:

$$I_j(t) = -C_j(\nu - v_r) \sum_{t_j^{(f)}} \delta(t - t_j^{(f)}) \tag{13}$$

where $\delta(.)$ is the Dirac δ-$function$, v_r is the initial condition, and ν is the threshold.

If we don't insert a train of spikes and consider the initial value equal to zero than (13) reduces to the same threshold function considered in the other artificial neural networks.

The input, $x_i(t)$, can be considered as a Dirac δ-function or, more generally, as a pulse of finite width with a time window $\alpha(t - t_i^{(f)})$ for $t > t_i^{(f)}$.

$$\alpha(k) = \frac{1}{\tau_k} e(-\frac{k}{\tau_k}) u(k) \tag{14}$$

where τ_k is a time constant, u(k) is the Heaviside step function and $k = t - t_j^{(f)}$.

Considering the input as a Dirac δ-function. We can rewrite (12) as

$$\frac{dv_j(t)}{dt} + \frac{v_j(t)}{\tau_m} = \frac{R_j}{\tau_m} [\sum_{i \in \Gamma_i} c_{ji} \sum_{t_i^{(f)} \in F_i} \delta(t - t_i^{(f)}) - C_j(\nu - v_r) \sum_{t_j^{(f)} \in F_j} \delta(t - t_j^{(f)})] \tag{15}$$

where τ_m is a time constant proportional to $R_j C_j$.

The integrating factor method is used to solve (15). The integrating factor is described as

$$I_{factor} = e^{\int_0^\infty \frac{1}{\tau_m} dt} = e^{\frac{1}{\tau_m} t} \tag{16}$$

Equation (15) then becomes

$$v_j(t) e^{\frac{1}{\tau_m} t} = \int_0^\infty \frac{R_j}{\tau_m} [\sum_{i \in \Gamma_i} c_{ji} \sum_{t_i^{(f)} \in F_i} \delta(t - t_i^{(f)}) - C_j(\nu - v_r) \sum_{t_j^{(f)} \in F_j} \delta(t - t_j^{(f)})] e^{\frac{1}{\tau_m} t} dt \tag{17}$$

Solving the second part of the right side of (17) gives:

$$-\frac{R_j}{\tau_m} \int_0^\infty C_j(\nu - v_r) \sum_{t_j^{(f)} \in F_j} \delta(t - t_j^{(f)}) e^{\frac{1}{\tau_m} t} dt = -(\nu - v_r) \sum_{t_j^{(f)} \in F_j} \int_0^\infty \delta(t - t_j^{(f)}) e^{\frac{1}{\tau_m} t} dt \tag{18}$$

where $\tau_m = R_j C_j$.

Applying the properties of the Dirac δ-function (19) in (20):

$$\int_0^\infty x(t) \delta(t - t_0) = x(t_0),\ t_0 \ > \ 0 \tag{19}$$

Solving the first part of the right side of the (17) we get

$$\frac{R_j}{\tau_m}\int_0^\infty \sum_{i\in\Gamma_i} c_{ji} \sum_{t_i^{(f)}\in F_i} \delta(t-t_i^{(f)})e^{\frac{1}{\tau_m}t}dt = \sum_{i\in\Gamma_i} w_{ji} \sum_{t_i^{(f)}\in F_i} e^{\frac{1}{\tau_m}t_i^{(f)}} u(t-t_j^{(f)}) \quad (20)$$

where $w_{ji} = \frac{R_j c_{ji}}{\tau_m}$

Applying the results of (18) and (20) in (17) gives

$$v_j(t)e^{\frac{1}{\tau_m}t} = -(\nu - v_r) \sum_{t_j^{(f)}\in F_j} e^{\frac{1}{\tau_m}t_j^{(f)}} u(t-t_j^{(f)}) + \sum_{i\in\Gamma_i} w_{ji} \sum_{t_i^{(f)}\in F_i} e^{\frac{1}{\tau_m}t_i^{(f)}} u(t-t_i^{(f)}) \quad (21)$$

Simplification of (21):

$$v_j(t)e^{\frac{1}{\tau_m}t} = -(\nu - v_r) \sum_{t_j^{(f)}\in F_j} e^{\frac{1}{\tau_m}t_j^{(f)}} u(t-t_j^{(f)}) + \sum_{i\in\Gamma_i} w_{ji} \sum_{t_i^{(f)}\in F_i} e^{\frac{1}{\tau_m}t_i^{(f)}} u(t-t_i^{(f)}) \quad (22)$$

$$v_j(t) = -(\nu - v_r) \sum_{t_j^{(f)}\in F_j} e^{-\frac{1}{\tau_m}(t-t_j^{(f)})} u(t-t_j^{(f)}) + \sum_{i\in\Gamma_i} w_{ji} \sum_{t_i^{(f)}\in F_i} e^{-\frac{1}{\tau_m}(t-t_i^{(f)})} u(t-t_i^{(f)}) \quad (23)$$

Now, the input will be consider to be a pulse of finite width and with a time window $\alpha(t-t_i^{(f)})$ for $t > t_i^{(f)}$, (17) can be written as

$$v_j(t)e^{\frac{1}{\tau_m}t} = \int_0^\infty \frac{R_j}{\tau_m}[\sum_{i\in\Gamma_i} c_{ji} \sum_{t_i^{(f)}\in F_i} \alpha(t-t_i^{(f)}) - C_j(\nu - v_r) \sum_{t_j^{(f)}\in F_j} \delta(t-t_j^{(f)})]e^{\frac{1}{\tau_m}t}dt \quad (24)$$

The second part of the equation is the same as before, so the first part is solved as

$$\frac{R_j}{\tau_m}\int_0^\infty \sum_{i\in\Gamma_i} c_{ji} \sum_{t_i^{(f)}\in F_i} \alpha(t-t_i^{(f)})e^{\frac{1}{\tau_m}t}dt \quad (25)$$

If we consider $k = t - t_i^{(f)}$ in (14) and apply to (25):

$$\frac{R_j}{\tau_m}\sum_{i\in\Gamma_i} c_{ji} \int_0^\infty \frac{1}{\tau_k} e^{-\frac{k-k'}{\tau_k}} e^{\frac{k'}{\tau_k}u(k-k')}dk \quad (26)$$

$$\sum_{i\in\Gamma_i} \frac{R_j c_{ji}}{\tau_m} \frac{1}{1-\frac{\tau_k}{\tau_m}} e^{-\frac{k}{\tau_m}} - e^{-\frac{k}{\tau_k}} u(k) \quad (27)$$

$$\sum_{i\in\Gamma_i} w_{ji} \frac{1}{1-\frac{\tau_k}{\tau_m}} e^{-\frac{k}{\tau_m}} - e^{-\frac{k}{\tau_k}} u(k) \quad (28)$$

where $w_{ji} = \frac{R_j c_{ji}}{\tau_m}$.

The integrated model can be summarized as

$$v_j(t) = \sum_{t_j^{(f)} \in F_j} \eta(t - t_j^{(f)}) + \sum_{i \in \Gamma_i} w_{ji} \sum_{t_i^{(f)} \in F_i} \varepsilon(t - t_i^{(f)}) \qquad (29)$$

$$\eta(k) = -(\nu - v_r)e^{-\frac{1}{\tau_m}k}u(k) \qquad (30)$$

$$\varepsilon(k) = e^{-\frac{1}{\tau_m}k}u(k) \quad if \quad x_j(k) = \delta(k) \qquad (31)$$

$$\varepsilon(k) = \frac{1}{1 - \frac{\tau_k}{\tau_m}}[e^{-\frac{k}{\tau_m}} - e^{-\frac{k}{\tau_k}}]u(k) \quad if \quad x_j(k) = \alpha(k) \qquad (32)$$

The integrated model of Hopfield networks is going to be used in the pulsed neural networks, with the appropriate choice of kernels. It formally shows that the Hopfield model is related to an integrate-and-fire network with distinct ε and η. Experiments of associative memory were done using Hopfield Nets and the Pulsed Neural Networks for pattern recognition of alpha numeric incomplete and corrupted data. Both systems had a similar performance as shown in [6].

Depending on the input, if it is considered to be just a impulse function or a pulse with a finite width, the kernels for the refractory kernel and for the pre and pos-synaptic will be different following the equations, (30), (31) and (32).

5 Conclusions

This paper shows the relationship between the Hopfield Neural Networks and the Pulsed Neural Networks. The description of both models were presented here as a tutorial. A simulink model of the integrate-and-fire more is presented and simulated varying the threshold values. Later the integration of the Hopfield model is described step by step, considering two types of inputs, as a Dirac δ-function and a finite width pulse. The resulting kernels were showed to be the kernels for the integrate-and-fire model.

References

1. W. Gerstner.: Time structure of the activity in neural networks models. Phys. Riview, vol.51, 1995, pp. 738-748.
2. Usher M., Schuster H. G., Niebur E.: Dynamics of populations of integrate and fire neurons, Neural Computation, vol.5, 1993, pp. 570-586.
3. F. Rieke, D. Warland, R. de Ruyter van Steveninck and W. Bialek.: Spikes: Exploring the Neural Coding, MIT Press, Cambridge, 1996.
4. Wolfgang Maass and Christopher M. Bishop.: Pulsed Neural Networks, MIT Press, England, 2001.
5. Haykin, S., Neural Network: A comprehensive foundation, Prentice Hall, New Jersey, 1999.
6. Guerreiro, A. M. G., A New Paradigm for Neurocomputation Using the Boolean Complete Neuron, Ph.D Thesis, University of Colorado, May 2004.

A Digital Hardware Architecture of Self-Organizing Relationship (SOR) Network

Hakaru Tamukoh, Keiichi Horio, and Takeshi Yamakawa

Graduate School of Life Science and Systems Engineering,
Kyushu Institute of Technology,
2-4 Hibikino, Wakamatsu-ku, Kitakyushu-shi 808-0196, Japan
tamukoh@brain.kyutech.ac.jp

Abstract. This paper describes a new algorithm of self-organizing relationship (SOR) network for an efficient digital hardware implementation and also presents its digital hardware architecture. In SOR network, the weighted average of fuzzy inference takes heavy calculation cost. To cope with this problem, we propose a fast calculation algorithm for the weighted average using only active units. We also propose a new generating technique of membership function by representing its width on power-of-two, which suits well with the digital hardware bit-shift process. The proposed algorithm is implemented on FPGA with massively parallel architecture. The experimental result shows that the proposed SOR network architecture has a good approximation ability of nonlinear functions.

1 Introduction

Self-organizing relationship (SOR) network is a powerful tool which can extract a desired I/O relationship of a target system in its learning mode and generate an output in its execution mode [1]. In the learning mode, with the applied learning data along with its evaluation values, fuzzy if-then rules are obtained which are eventually used in the intended systems. As for in the execution mode, fuzzy rules obtained during the learning mode is used to generate outputs corresponding to the inputs. SOR network is currently being utilized in wide range of technical applications [2]-[5]. SOR network, however, requires high computing specs according to its scale which led to the development of an effective custom hardware.

The learning mode of SOR network is similar to that of self-organizing map (SOM) [6][7], and we have developed a digital hardware of the learning mode of SOM [8]. In this paper, we describe a digital hardware design of execution mode of SOR network.

The process in the execution mode of SOR network is almost same to a fuzzy inference. At first similarity between input and each reference vector is calculated, then the output is obtained by a weighted average. The calculations of the similarity and the output are considered as a calculations of membership value and defuzzification, respectively. In the calculation of the output, summation of

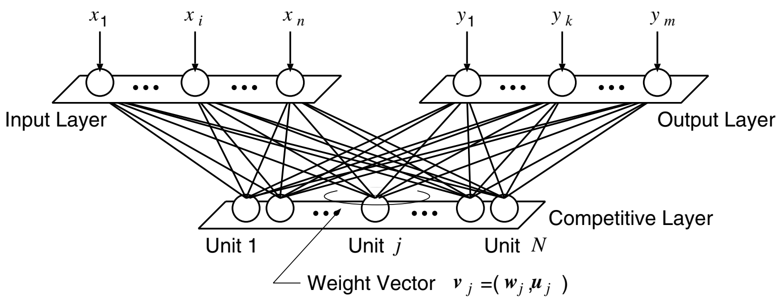


Fig. 1. Structure of the SOR network

membership values for all rules is needed. It takes a long computational time, because it can not be realized in parallel architecture. To reduce the computational time, only rules which have large influence on the calculation of the output is used. To realize it, rough comparison winner-take-all(WTA) algorithm [8] is employed.

Furthermore, we propose a new membership function generation technique which suits well with the digital hardware implementation. In conventional fuzzy inference, membership functions are defined for each input variable and the membership value for each rule is obtained by producing the membership values for all input variables [9]-[11]. On the other hand, in the proposed method, the membership value for each rule is calculated based on the distance between the input and the center of membership function. The shape of membership function is represented as power-of-two which make its digital hardware implementation very easy and effective.

2 SOR Network

Fig.1 shows the structure of SOR network which is almost same to that of SOM. In SOM, each competing unit has only one reference vector, on the other hand, the unit has two reference vectors $\boldsymbol{w}_i$ and $\boldsymbol{u}_i$ in SOR network. We define the reference vector of the i-th unit $\boldsymbol{v}_i = [\boldsymbol{w}_i, \boldsymbol{u}_i]$. In learning of SOR network, I/O vector pairs $\boldsymbol{I} = [\boldsymbol{x}, \boldsymbol{y}]$ and their evaluations E of the target system are used as the learning vectors. Although the learning procedure of SOR network is almost same to that of the ordinary SOM, the method of updating the reference vector is different. The reference vectors are updated in accordance with evaluation of the learning vector by:

$$\boldsymbol{v}_i^{\mathrm{new}} = \boldsymbol{v}_i^{\mathrm{old}} + \begin{cases} \alpha(t)E(\boldsymbol{I} - \boldsymbol{v}_i^{\mathrm{old}}) & E \geq 0 \\ \beta(t)E\exp(- \parallel \boldsymbol{I} - \boldsymbol{v}_i^{\mathrm{old}} \parallel)\mathrm{sgn}(\boldsymbol{I} - \boldsymbol{v}_i^{\mathrm{old}}) & E < 0, \end{cases} \tag{1}$$

where, $\boldsymbol{v}_i^{\mathrm{new}}$ and $\boldsymbol{v}_i^{\mathrm{old}}$ are reference vectors after and before update, respectively. $\alpha(t)$ and $\beta(t)$ are learning coefficients for attractive and repulsive learning, respectively. As the result of the learning, the desired I/O relationship of the target system is obtained. The desired I/O relationship is represented by reference vectors as set of if-then rules as follows.

$$Rule\ i:\ If\ \boldsymbol{x}\ is\ \boldsymbol{w}_i,\ then\ \boldsymbol{y}\ is\ \boldsymbol{u}_i \tag{2}$$

After the learning, the desired output can be generated when the input vector is applied to the network. This process is called execution mode. First the similarity z_i between input vector $\boldsymbol{x}$ and reference vector $\boldsymbol{w}_i$ is calculated by:

$$z_i = \exp(-\frac{d_i}{\gamma}) \tag{3}$$

where, d_i is the distance between $\boldsymbol{x}$ and $\boldsymbol{w}_i$, and γ is the parameter. Then the output vector is calculated by:

$$y_k = \frac{\sum_{i=1}^{N} z_i u_{ki}}{\sum_{i=1}^{N} z_i} \tag{4}$$

The output vector $\boldsymbol{y}$ is expected to be a desired output for the applied input.

The process of the execution mode is similar to fuzzy inference. The calculation of the similarity represented in Eq.(3) corresponds to calculation of membership value for rule i in fuzzy inference. And calculation of the output in Eq.(4) corresponds to defuzzification employing center-of-gravity method.

3 A Bit-Shifting-Based Fuzzy Inference for SOR Network

We have developed SOM hardware, and the learning architecture of SOR network is almost same to that of SOM's. Thus, we focus on the hardware design of the execution mode of SOR network in this article. In the execution mode of SOR network, calculation of the similarities between inputs and reference vectors can be implemented in parallel architecture. However, the calculation of the output by weighted averaging can not be implemented in parallel architecture, and calculation time increases as the number of units increase. Furthermore, it is difficult to realize the Eq.(3) with digital hardware, because the very large storage is needed to memorize the shape of the membership functions.

In this paper, we propose a new calculation method of a membership value which is suitable for implementation in digital hardware. In this method, the reference vector $\boldsymbol{w}_i$ is considered as the center of membership function. The similarity shown in Eq.(3), that is the membership value for the rule i, is redefined by:

$$z_i = \begin{cases} 2^{-s_i}, if\ flag_i = 1 \\ 0, \quad otherwise \end{cases} \tag{5}$$

where,

$$s_i = \frac{d_i}{2^{r-\log_2 a}} \tag{6}$$

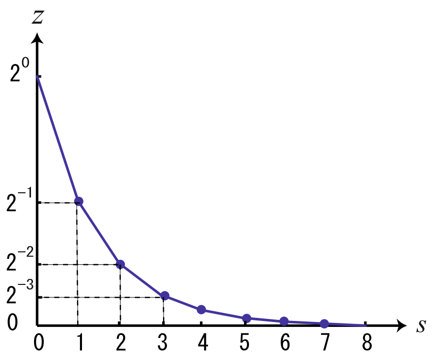


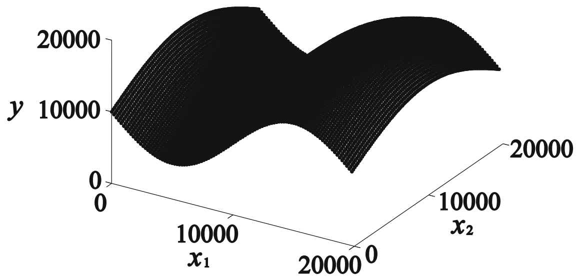
Fig. 2. Shape of the membership function used in this study. This membership function represents $r = 3$ with $a = 8$ bit accuracy.

Table 1. Simulation results

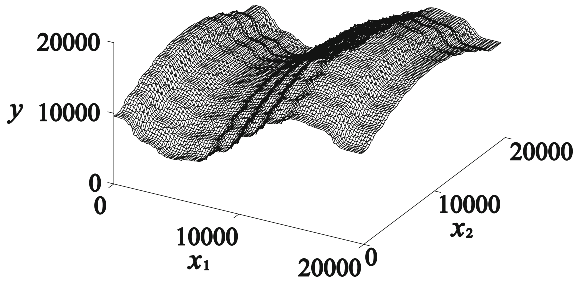
	RMSE	calculation cost
original SOR network	476.98	$N = 100$
proposed SOR network	474.98	$C = 7.31$ (average)

where, d_i is the distance between $\boldsymbol{x}$ and $\boldsymbol{w}_i$, and a is the calculation accuracy. Parameter r is introduced to represent the width of membership functions. The shape of a membership function is shown in Fig.2. By assigning various values to the parameter r, a various shape of membership functions can be easily produced. r is increased when a wide membership function is needed and decreased for a narrow one. This membership value z_i is suitable for hardware implementation, because it is realized by only the bit shifters. $flag_i$ in Eq.(5) means whether the distance d_i is smaller than the threshold or not. In other words, the unit i with $flag_i = 1$ has large membership value with a large influence on the calculation of the output. We call it, "active unit". In our previous work, the winner-take-all (WTA) circuit which finds the minimum distance was proposed [8]. In this circuit, comparison accuracy of the distance can be changed. In case of high comparison accuracy, only one weight vector which has minimum distance to the input vector is selected. On the contrary, if the accuracy is low (i.e. rough comparison), plural number of candidates for the winner will be selected. In this study, we propose an active unit selector employing the rough comparison WTA. The rough comparison WTA can select active units which have small distance its weight vector $\boldsymbol{w_i}$ to the input $\boldsymbol{x}$. As the result, the $flag_i$ of the active unit is set to '1'.

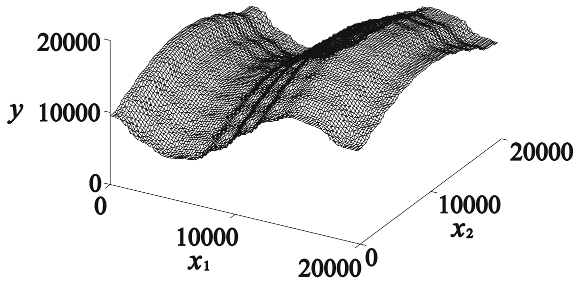
The above calculation can be processed in parallel architecture, however, in calculation of defuzzification represented by Eq.(4), serial architecture have to be included, because the summation of N elements is needed. However, in usual cases the number of units which have significant influence on the calculation is not that large. Therefore, we propose a new fast calculation method for



(a)



(b)



(c)

Fig. 3. Results of mappings generated by SOR network. (a) The ideal mappings. (b) Mappings resulting from the conventional algorithm. (c) Mappings resulting from the proposed algorithm. Proposed algorithm generates mappings very similar to that of the conventional algorithm.

defuzzification which employs only the active units. Through the propose method, Eq.(4) can be redefined as follow,

$$y_k = \frac{\sum_{i \in P} z_i u_{ki}}{\sum_{i \in P} z_i}, \tag{7}$$

P is the set of flag = '1' units. C is defined as the number of units which has flag = '1' and expected to be $C \ll N$. Therefore, Eq.(7) would lead to a huge reduction of clock cycles comparing to Eq.(4). Furthermore, the multiplication in Eq.(7) is realized by only bit shifter, because z_i is represented as factorial of 2.

4 Simulation Results

To evaluate the performance of the proposed fuzzy inference for execution mode of SOR network, approximation of the nonlinear I/O relationship of 2-input and 1-output shown in Eq.(8) is done.

$$y = (0.5\sin(\pi x_1) + 0.5\cos(0.5\pi x_2)) \times 10^4 + 10^4 \tag{8}$$

Fig.3 (a) shows the desired I/O relationship. In the simulation, the number of units in the competitive layer is 10×10. To evaluate only the execution mode of SOR network, the weight vectors are arranged in the desired position. In the original execution mode, the parameter γ in Eq.(3) was 500. In the proposed execution mode, the parameter r and a in Eq.(6) are 12 and 16, respectively. As testing input vectors, 100×100 vectors obtained in the input space are used. Fig.3 (b) and (c) shows I/O relationship generated by the original and the proposed execution modes, respectively. Simulation results are summarized in Table 1. The root-mean-square-errors of the original and the proposed methods are 476.98 and 474.98. In the proposed method, the average of the number of active units which are selected to generate the output is 7.31. This means that 93 % of reduction on calculation cost for Eq.(4) is realized by using Eq.(7).

5 Hardware Architecture

In this section, we explain about the proposed hardware architecture of SOR network. The main idea of the hardware design is based on a massively parallel system, where each processing element corresponds to one competing unit in SOR network. In this paper, we have divided the hardware structure for SOR network into two modules, a global circuit and a local circuits that represent the whole SOR network structure and competing units, respectively.

5.1 Global Circuit

Fig.4 shows the global circuit which represents the whole SOR network structure. The global circuit consists a global controller, a certain number of local circuits, an active unit selector and a defuzzification module. In the proposed hardware, all local circuits run in parallel.

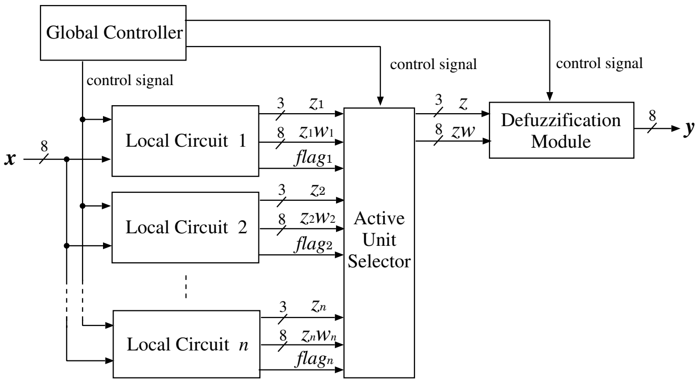


Fig. 4. Block diagram of the global circuit

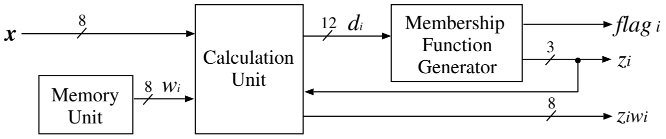


Fig. 5. Block diagram of the local circuit

5.2 Local Circuit

Fig.5 shows the architecture of a local circuit. The local circuit consists a calculation unit, a memory unit and a membership function generator. The calculation unit calculates Manhattan distance, and the memory unit stores weight vector.

Fig.6 shows the architecture of the membership function generator. The membership function generator calculates the membership value corresponding to input vector by using Eq.(5) and Eq.(6). In this study, rough comparison WTA in the membership function generator is realized in a 9-input NOR gate. When all high 9 bits of variable s_i are assigned to ‘0’, the weight vector w_i is selected as a winner candidate. Thus, w_i can be determined whether it is an active unit or not by inputting s_i into the 9-input NOR gate. The membership function generator is realized by bit shifters, 9-input NOR gate and RAM for parameter r, thus easing it to be realized with digital hardware.

5.3 Active Unit Selector

The active unit selector serially selects local circuit i which holds $flag_i = 1$ and then outputs z_i and $z_i w_i$ of local circuit i to the defuzzification module on every clock cycle. When the number of active units is C, then C clock cycles are required for transfering z_i and $z_i w_i$ of all active units.

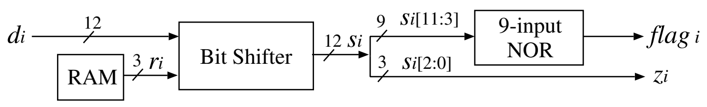


Fig. 6. The membership function generator for the proposed fuzzy inference

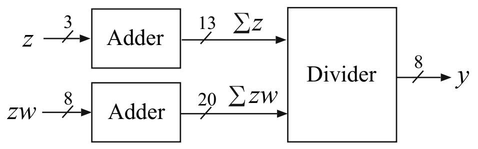


Fig. 7. The block diagram of the defuzzification module

5.4 Defuzzification Module

Fig.7 shows the architecture of the defuzzification module which consists two adders and a divider. Each adder receives z and zw from active unit selector, and then calculates the sum. After the summation of z and zw from all active units, y is calculated by the divider.

5.5 Performance

In the proposed architecture, the execution mode of SOR network is divided into three steps. The three steps are distance calculation, weighted summation and division, and clock cycles required for each steps are C_{dist}, C_{wsum} and C_{div}, respectively. The number of clock cycles C_{exe} that is necessary to generate an output element from an input vector is given by:

$$C_{exe} = C_{dist} + C_{wsum} + C_{div}. \tag{9}$$

In the case of the simulation, the number of clock cycles is 10.31 (average) per an input vector. In the simulation described in section 4, the proposed hardware took 2.062 ms when executed on 50 MHz. When the same simulation executed on a PC (CPU: Intel Xeon 2.8GHz, Memory: 2Gbytes, MMX without extended instructions), it resulted in an average time of 172 ms. Thus proving that the proposed hardware architecture can be prospected to be 80 times faster.

To evaluate the proposed hardware performance of the architecture on state-of-the-art technology, we designed the SOR network hardware by using the Xilinx xc2v6000-6 FPGA which can hold 76,000 Logic Cells equivalent circuit. The designed SOR Network hardware which consists of 25 local circuits described in VHDL is synthesized using Leonardo Spectrum FPGA compiler and Xilinx tools for place and route. In each local circuit, the weight vector with up to 16 dimensions and 8 bits accuracy is available. The circuit size and speed of the designed hardware are estimated by Xilinx tools. Estimation results are summarized in Table 2. The designed hardware can run in 50 MHz clock.

Table 2. Circuit size and speed

Device Utilization for 2V6000ff1517			
Resource	Used	Avail	Utilization
Function Generators	4662	67584	6.90 %
CLB Slices	2331	33792	6.90 %
Dffs or Latches	941	70696	1.33 %
Clock Frequency			50.1 MHz

6 Conclusion

In this paper, we proposed a new fuzzy inference for the execution mode of SOR network and its digital hardware architecture. In the proposed fuzzy inference, a bit-shifting-based membership function generator and a fast defuzzification method employing only active units were proposed. The proposed membership function generator can be realized without ROM for memorizing the shape of membership functions. The proposed defuzzification method reduces 93% clock cycles for the execution mode of SOR network. Simulation result shows that the proposed SOR network is similar to the original SOR network on approximation ability of nonlinear functions. The proposed SOR network was implemented on FPGA. The performance of the proposed hardware architecture was superior to CPU implementation of SOR network.

We estimate the proposed hardware which consists 25 local circuits. As a result, the hardware use about 7 % of 76,000 Logic Cells FPGA with 50 MHz speed. The largest FPGA on the current state is the Xilinx xc5vlx330 which equips 330,000 Logic Cells. In the real world application, large network is required (e.g. the trailer-truck back-up control system uses 625 competitive units in [2]). The proposed hardware is based on the massively parallel architecture. Thus, it can extend easily to large network by adding local circuits up to the maximum size of FPGA.

Acknowledgments

This work was supported by a Center of Excellence Program granted in 2003 to Department of Brain Science and Engineering, Kyushu Institute of Technology by Japan Ministry of Education, Culture, Sports, Science and Technology.

References

1. T. Yamakawa and K. Horio, "Self-organizing relationship (SOR) network," *IEICE Trans. on Fundamentals* vol. E82-A, no. 8, pp.1674-1678, 1999.
2. T. Koga, K. Horio and T. Yamakawa, "The Self-Organizing Relationship (SOR) Network Employing Fuzzy Inference Based Heuristic Evaluation," *Neural Networks*, in press.

3. T. Koga, K. Horio and T. Yamakawa, “Applications of brain-inspired SOR network to controller design and knowledge acquisition,” *Proc. of 5th Workshop on Self-Organizing Maps (WSOM’05)*, pp.687-694, 2005.
4. K. Horio, T. Haraguchi and T. Yamakawa, “An Intuitive Contrast Enhancement of an Image Data Employing the Self-Organizing Relationship (SOR),” *Proc. of Int. Joint Conf. on Neural Networks (IJCNN’99)*, pp.10-16, 1999.
5. K. Horio and T. Yamakawa, “Adaptive Self-Organizing Relationship Network and Its Application to Adaptive Control,” *Proc. of the 6th Int. Conf. on SoftComputing and Information/Intelligent Systems*, pp.299-304, 2000.
6. T. Kohonen, “Self-Organized Formation of Topologically Correct Feature Maps,” *Biological Cybernetics*, vol.43, pp.59-69, 1982.
7. T. Kohonen, *Self-Organizing Maps 3rd ed.*, Berlin: Springer, 2001.
8. H. Tamukoh, K. Horio and T. Yamakawa, “Fast learning algorithms for self-organizing map employing rough comparison WTA and its digital hardware implementation,” *IEICE Trans. on Electronics* vol. E87-C, no. 11, pp.1787-1794, Nov., 2004.
9. P. Y. Chen, ”VLSI implementation for fuzzy membership-function generator,” *IEICE Trans. on Inf* & *Syst.* vol. E86-D, no. 6, pp.1122-1125, June, 2003.
10. J. M. Jou, P. Y. Chen and S. F. Yang, ”An adaptive fuzzy logic controller: its VLSI architecture and applications,” *IEEE Trans. on Very Large Scale Integr. (VLSI)* vol. 8, no. 1, pp.52-60, Jan., 2000.
11. A. Gabrielli and E. Gandolfi, ”A fast digital fuzzy processor,” *IEEE Micro* vol. 19, no. 1, pp.68-79, Jan., 1999.

Towards Hardware Acceleration of Neuroevolution for Multimedia Processing Applications on Mobile Devices

Daniel Larkin*, Andrew Kinane, and Noel O'Connor

Centre for Digital Video Processing, Dublin City University, Dublin 9, Ireland
{larkind, kinanea, oconnorn}@eeng.dcu.ie

Abstract. This paper addresses the problem of accelerating large artificial neural networks (ANN), whose topology and weights can evolve via the use of a genetic algorithm. The proposed digital hardware architecture is capable of processing any evolved network topology, whilst at the same time providing a good trade off between throughput, area and power consumption. The latter is vital for a longer battery life on mobile devices. The architecture uses multiple parallel arithmetic units in each processing element (PE). Memory partitioning and data caching are used to minimise the effects of PE pipeline stalling. A first order minimax polynomial approximation scheme, tuned via a genetic algorithm, is used for the activation function generator. Efficient arithmetic circuitry, which leverages modified Booth recoding, column compressors and carry save adders, is adopted throughout the design.

1 Introduction

Artificial neural networks (ANN) have found widespread deployment in a broad spectrum of classification, perception, association and control applications [1]. However, finding an appropriate network topology and an optimal set of weights represents a difficult multidimensional optimisation problem. Ideally, the topology should be as small as possible, but large enough to accurately fit the training data. Failure to find a suitable configuration will cause poor generalisation ability with unseen data and/or excessive execution time. One possible solution to this issue is to use a genetic algorithm to evolve an optimum topology and/or weights. This approach is also sometimes known as Evolutionary Artificial Neural Networks (EANN) or Neuroevolution (NE) [2][3]. As well as reducing the requirement for trial and error design exploration for the ANN, the approach is more robust at avoiding local minima and has the scope for finding a minimal topology [2]. A minimal topology is hugely beneficial since fewer neurons and synaptic connections lead to reduced computation, which in turn means higher throughput and lower power consumption.

* The support of the Informatics Commercialisation intiative of Enterprise Ireland is gratefully acknowledged.

However, even with minimal topologies, when processing high dimensional datasets, for example multimedia data, the NE process may not meet system requirements (throughput and/or power consumption). In the case of multimedia data, poor performance is likely to be a consequence of the unavoidable combination of a requirement for extremely high throughput for real time processing and large complex ANN topologies caused by a high number of inputs. The associated computational complexity is highly undesirable from a real time operation and low power consumption perspective. Clearly, this poses considerable problems when incorporating NE on constrained computing platforms (e.g. on mobile devices for gaming or multimedia processing applications) which suffer from limitations such as low computational power, low memory capacity, short battery life and strict miniaturisation requirements.

One possible solution to NE complexity issues, is to offload the computational burden from the host processor to a dedicated hardware accelerator. Although a general consensus emerging in recent times is that the viability of dedicated hardware ANN processors are questionable [4]. However, due to the aforementioned throughput and power consumption issues, ANN hardware acceleration still provides an attractive and viable solution particularly in the context of constrained computing platforms. This has motivated us to design an efficient and flexible hardware ANN accelerator, which is suitable for NE tasks. We have chosen to investigate the widely used open source Neuro Evolving Augmented Topologies (NEAT) software library [3]. Our profiling has revealed the computational burden of the ANN evaluation is suitable for hardware off load, whilst the genetic algorithm routines, which use moderate computational resources can reside in software. This scalable co-design methodology combines the reconfigurability of software with the speed of hardware [5]. This also facilitates application re-usability, where the core could be re-deployed for a number of applications. It should be noted that hardware acceleration will not be applied to the genetic algorithm itself.

The rest of this paper is organised as follows: Section 2 details related prior research in the area. Section 3 discusses NE hardware architecture design challenges and choices. Section 4 outlines the hardware implementation of the proposed architecture. Section 5 details hardware synthesis results and power consumption estimates. Future work is outlined in Section 6, whilst Section 7 draws conclusions about the work presented.

2 Related Research

There are many approaches to NE, differing principally in the choice of genome encoding scheme and the operators chosen for mutation and crossover. NEAT is an example of a direct encoded node-based NE approach. Each genome consists of a number of link and neuron genes. Fig. 1 shows an example genome and the principal constituent elements of each gene type. NEAT uses a genome historical marking scheme (shown as the *innovation number* in Fig. 1), which avoids many of the problems associated with other NE approaches. This scheme allows

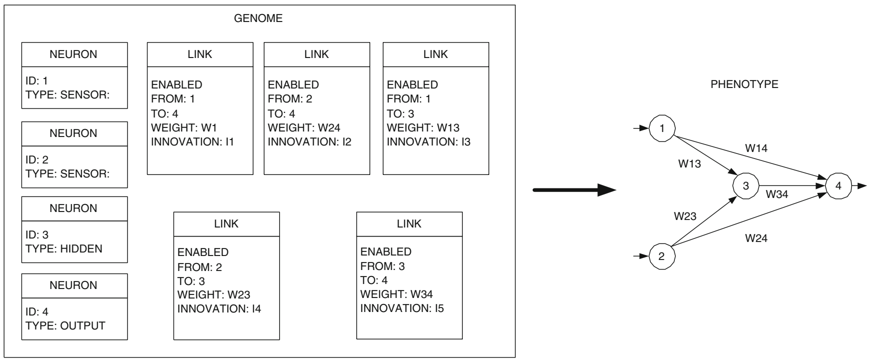


Fig. 1. Example of an NEAT genome and resultant phenotype

meaningful topology crossover to occur, and furthermore, it avoids any computational overhead of topology analysis when producing the valid offspring. Taking inspiration from biological evolution, NEAT introduces the concept of *complexification*, whereby processing starts with a minimal topology with no hidden nodes and each input is connected to an output node. Each successive generation systemically elaborates the complexity by adding neurons and/or synaptic connections. This is attractive in the context of our interest in mobile devices since minimal topologies are favoured, thus reducing computational complexity for a given problem. During complexification, topology innovation occurs. However, adding structure will typically reduce the fitness function, since smaller structures optimise faster. Consequently, there is little opportunity for the new topology to survive [3]. Again taking inspiration from biological evolution, where an innovation needs time to reach its potential, NEAT uses a process of *specification.* This allows genomes to compete only against their own species and gives the new offspring a better chance of survival. The combination of these features, allows NEAT to outperform other NE approaches [3], and for this reason, we have chosen it for further investigation.

2.1 ANN Hardware Implementations

There has been considerable research in analog and digital hardware ANN implementations. Analog implementations typically have the benefit of high speed operation and smaller area compared to a digital implementation. However an analog design has a number of drawbacks including susceptibility to component process variation, electrical noise and environmental conditions, along with resolution issues for the weights and activation function. A digital hardware implementation, on the other hand, does not suffer from these drawbacks, and furthermore allows ease of design and computational accuracy. For these reasons we have adopted a digital design approach.

A digital implementation typically stores the network topology and/or synaptic weights in memory. These values are then later retrieved and processed in discrete chunks by the parallel processing elements (PE). This is advantageous because any network size and potentially any topology can be handled. A PE usually consists of multiply accumulate circuitry [6] and sometimes depending on the configuration, an activation function generator. The number of processing elements implemented is a design space trade off between area, power and performance. This design space extends from a single PE to a PE for each neuron or even a PE for each synaptic calculation. A discussion of the PE implementation challenges is given in Section 4.

One of the principal challenges for ANN acceleration using time shared PE's is how to get the ANN data from slow, bandwidth limited memories to the high speed PEs in a fast, bandwidth efficient manner. Systolic array architectures have been the pervasive choice for bridging this gap [6]. It is well established that systolic arrays offer many benefits for ANN with regard to using memory bandwidth effectively by maximising data reuse, in addition to permitting highly regular PE control logic. However, as will be demonstrated in Section 3, topologies with sparse synaptic connections considerably reduce the efficiency of systolic array approaches, in addition to increasing memory requirements. Furthermore, as sparse topologies result in fewer synaptic calculations, it could be exploited to reduce power consumption. Despite this, the literature contains very few hardware implementations, which attempt to exploit topology sparseness. Rather, it focuses on dense connectivity neural algorithms.

Our work provides a hardware ANN accelerator extension for NEAT, using an architecture which is suitable for any evolved topology, as well as having the ability to exploit sparse connectivity. The intended target of this work is as an accelerator for mobile devices, consequently the focus is on power efficiency, rather than ultra high performance acceleration.

3 Proposed Hardware Architecture

Unlike a conventional ANN, an evolved network can have any topology, potentially with a mixture of forward synaptic connections, recurrent synaptic connections and looped current synaptic connections. Furthermore, owing to the complexification process, NEAT will naturally favour sparse topologies. These factors have important consequences for the hardware architecture. The efficiency of systolic array architectures is dramatically reduced when operating on sparse neural topologies. This is because the data flow through the systolic array is frequently interrupted and thus the throughput benefit of multiple PEs is not being achieved. To overcome this, the PE can be either disabled for that synaptic calculation or have a weight of zero. However both solutions are undesirable. Disabling the PE leads to additional control logic for each individual PE. Whilst storing weights with a zero value leads to increased memory sizes, which further exacerbates power consumption issues, particularly as memory power consumption disproportionately increases with size. The systolic array efficiency is further reduced due to the presence of recurrent synaptic connections. This causes unpredictable feedback synaptic connections from other neurons, causing further data flow interruptions. However more importantly, as systolic array architectures favour layered ANN, where the inputs to each PE layer are well defined, it is not a trivial matter to dynamically reconfigure the PE inputs for the evaluation of alternative topologies, which contain recurrent links. This situation would be necessary for an application where NEAT is evolving in real-time, such as artificial intelligence for computer gaming [7]. Whilst it is possible to modify the genetic algorithm so that only forward synaptic connections are added, this compromises the quality of the evolved ANN solution.

These factors have motivated us to explore alternative architectures rather than a traditional systolic array approach. Examining the NEAT genome data structures, it is clear that the essential information in the link and neuron genes, can be mapped easily to hardware memories. Essentially, we propose "parsing" through the *LINK* memory to retrieve the relevant weights and using the "*LINK*$\rightarrow$*FROM*" field as the index to retrieve the output from the appropriate neuron in the *NEURON* memory. Once the values are retrieved the multiply accumulate operation is performed. This operation is repeated for all synapses associated with that neuron, before the activation function is calculated. The process then repeats for all neurons.

To increase throughput, two PEs operating in parallel are used, as can be seen from the proposed architecture in Fig. 2. The PE datapath is 64 bits and each PE has access to the memory (via a memory controller), thus two PEs were chosen so as to give a good trade off between throughput and bus addressing complexity. Each PE is equipped with a local "*LINK*" SRAM. This is loaded with the incoming synaptic connections for that neuron. To reduce stalling, the synaptic connections for neuron $N + 1$ are prefetched, whilst in parallel the PE processes neuron N. The PE datapath is 64 bits wide because 4×16 bit entries from the local *LINK* SRAM are retrieved in a burst. Parallel processing is possible since the "*LINK*$\rightarrow$*WEIGHT*" values are processed sequentially and they do not have interdependencies. The decision to use 4 parallel arithmetic units per PE was chosen as a trade off between throughput, bus width size and to minimise the complexity of the hierarchical memory system.

Unless a fully connected topology is being evaluated, the 4 "*NEURON*$\rightarrow$*ID*" addresses decoded from "*LINK*$\rightarrow$*FROM*" will not necessarily be contiguous. Therefore if a single "*NEURON*" SRAM is used, only one "*NEURON*" address could be processed per clock cycle. However to maximise the performance of the multiple arithmetic units in the PE datapath, valid data needs to be available on each clock cycle. To overcome this issue, we propose partitioning the neuron memory into 8 smaller SRAMs, as can be seen in Fig. 2. To further increase the probability that all data units can be retrieved within one clock cycle, we propose using a data cache to maximise data reuse. The cache provides backup should two or more data requests occur for the same "*NEURON*" SRAM bank. This could occur depending on the connectivity of the topology, in particular if the sparse synapses were modulo 8 apart.

When the memory control logic receives a request for 8 new values from the 2 PEs, the cache is firstly examined to see if it can fulfil these requests. Should

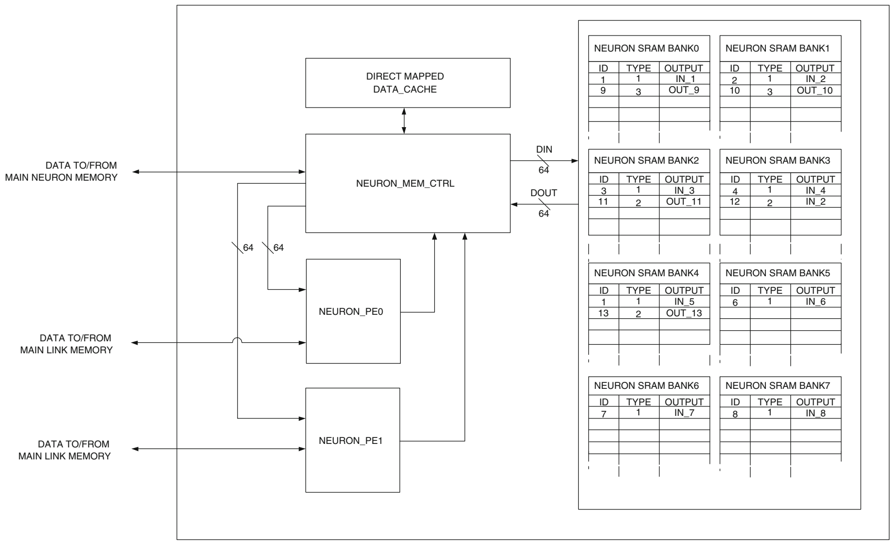


Fig. 2. Simplified block diagram of the proposed NE hardware accelerator datapath

cache misses occur, the "*NEURON→ID*" is decoded to indicate the relevant SRAM bank. If 2 or more requests attempt to access the same SRAM bank, the data must be retrieved over multiple clock cycles, otherwise all requests can be handled in one clock cycle. In the worst case scenario, when none of the data is present in the cache and all requested data is located in the same "*NEURON*" SRAM bank, one multiply accumulate operation occurs per clock cycle. On the other hand, if all the requested data is either in the cache or different Neuron SRAM banks, 8 multiply accumulates occur per clock cycle. For a fully connected feed forward ANN, the performance of this proposed system will be similar if not identical to a systolic array with the same number of arithmetic units, and we believe, the system will also statistically outperform a systolic array architecture when processing a mixed forward and recurrent sparse connection topology. Performing this comparison is targeted as future work.

4 Hardware Implementation

A fundamental digital implementation design decision is what format to use for the data representation. There are at least three popular approaches for digital ANN – stochastic, fixed point and floating point. Stochastic digital ANN implementations encode the value of a signal using the probability of a given bit in a stochastic pulse stream being a logic 1 [8]. This has the benefit that many common arithmetic operations require very simple logic [9]. Whilst beneficial from an area perspective, there are issues concerning representation variance. Furthermore, due to the increased latency from the inherently serial operation, a higher clock frequency is required to match the throughput from a more parallel fixed

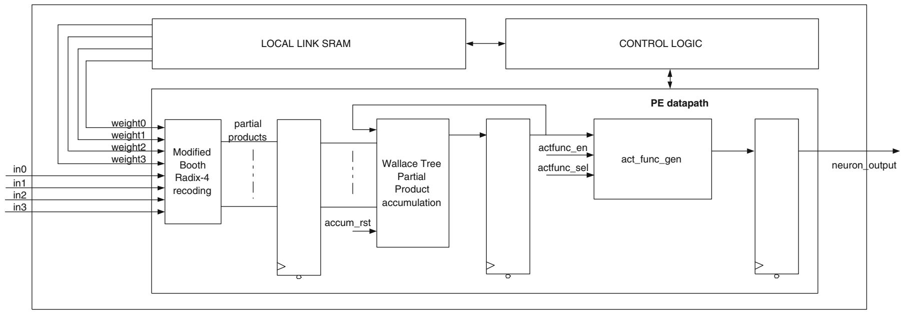


Fig. 3. Simplified block diagram of the Neuron PE

point implementation. The substantially increased clock frequency is of considerable concern in the clock tree network from a power consumption perspective, particularly in deep sub micron technologies. A floating point implementation on the other hand offers a wide dynamic range suitable for the typical distribution of ANN weight values, however it has a considerable area overhead for arithmetic operators. Consequently we have chosen a fixed point representation, as we believe it offers a reasonable trade off between area, power and performance.

The width of the datapath in a fixed point implementation is a vital design decision. To minimise area and power consumption, the minimum number of bits should be chosen, which will result in an acceptable error. Reduced fixed-point precision hardware ANN issues were explored in [10]. It was found that 10 precision bits were sufficient for multi-layer perceptrons trained via back propagation [10]. Using fewer precision bits than this will effect the convergence speed, and in some cases may completely prevent convergence. Related to this, input/output standardising (sometimes called scaling) is advised in most cases. Standardising the inputs will result in a smaller integer range. This leads to fewer bits switching and consequently power savings. For these reasons, we propose using 16 bits in a 6.10 fixed point representation (6 integers bits and 10 fractional bits) throughout the design, with the input data standardised to a range of -1 to 1. The 6 integer bits allows net accumulation values to grow to levels, which maximally exploit the resolution achievable from the proposed activation function generator. The remainder of this section will discuss the architecture implementation issues and decisions.

4.1 PE Implementation

The function of the PE is to generate the neuron weighted sum of inputs, this clearly requires multiply accumulate circuitry. We have also chosen to add the activation function generator to each PE (see Fig. 3). An alternative approach is to time share a single activation function generator between multiple PEs. This approach is typically adopted when using a systolic array architecture due to the highly regular processing. For our proposed architecture, control logic would need to be designed to ensure that only a single PE has control of the activation function generator at any clock cycle. However as we are currently using only two PEs and that the area overhead for the activation generator is not considerable ($< 2,000$ gates), we chose to integrate the activation function logic into the PE.

In a system where the ANN weights are static, canonic signed digit representation and multiplier-less distributed arithmetic architectures can be employed in the generation of the weighted sum. However, static weights are clearly not suitable for NE. Fortunately, owing to the prevalence of the sum of products operation in signal processing algorithms and matrix arithmetic, there has been considerable research on efficient hardware implementations. Consequently, we have chosen a fused multiply add approach as can be seen in Fig. 3 [11]. The number of partial products has been halved by using modified Booth radix 4 coding and the accumulation step is merged with the addition of the partial products using a Wallace tree [11]. Furthermore, the generation of the two's complement for the modified Booth algorithm uses a simple inversion for the one's complement and delays adding the additional one, until the Wallace tree stage, thereby reducing the critical path. The final sum and carry are then added using an efficient carry propagate adder.

Each PE has ≈ 3KB of local SRAM to store "*LINK→FROM/WEIGHT*" data, providing enough storage for the details of over 1000 synaptic connections. Obviously the amount of memory can be adjusted based upon the timing constraints of the main memory and connectivity characteristics of a particular application. The PE control logic, which governs access to the local SRAM and the control signals for the PE datapath is outlined in Algorithm 1.

Algorithm 1. Neuron PE datapath control flow

```
1 MEM_SETUP setup;
    Load PE SRAM with "LINK→FROM/WEIGHT" data for first "LINK→TO" neuron;
    Regular processing starts once loaded;
    In parallel, prefetch "LINK→FROM/WEIGHT" data for the next neuron;
2 LINK_DECODE Stage;
    PE requests 4 "LINK→FROM/WEIGHT" entries from local SRAM;
3 INPUT_FETCH Stage;
    Using the 4 retrieved "FROM" addresses, the PE requests these values from the
    "NEURON" SRAM memory banks;
    The "WEIGHT" values are set up on the inputs to the partial product generation logic;
4 PP_GEN Stage;
    With the input values returned from the "NEURON" SRAM banks, the partial product
    generation logic is enabled;
5 ACCUM Stage;
    Partial products and the previous accumulation are added in the Wallace Tree compressor;
6 ACT_FN Stage;
    If necessary the activation function generator is enabled;
7 WRITE_BACK Stage;
    If the activation function is enabled, output is written to memory;
```

We have recently proposed an efficient hardware architecture for an activation function generator [12], which improves the approximation error over prior research. Our approach uses a minimax polynomial spline approximation, with a genetic algorithm (GA) leveraged to find the optimum location of the approximating polynomials. The GA typically improved the approximating error by

30% to 60% relative to an even distribution of the approximating polynomials. Using a spline-based approach has the benefit that multiple activation functions can be accommodated by merely changing the coefficients of the approximating polynomial. This is beneficial from an evolutionary perspective and uses minimal extra hardware to support the additional functions.

4.2 Cache Design

To minimise the area and control logic overhead a direct mapped cache implementation has been chosen [13]. The selection of the cache size represents an important design trade off, a larger cache will have fewer cache misses, but will have a larger area. In our initial design investigation we use 2 parallel PEs each with 4 arithmetic units, as a result we believe a cache size of 64 blocks will be appropriate. However, further optimisations should be possible with the cache size by tailoring it to the statistics of the ANN topology for a particular application, e.g. video analysis. Each block consists of 2 elements, the neuron address and the neuron data. A benefit of using a direct mapped cache is that logic for a cache block replacement strategy is not required. When a value from the activation function generator is written back to SRAM, a stall signal is issued and if necessary the cache is updated. This avoids any potential data mismatches and the need for additional storage to hold a "block dirty" value in the cache [13].

5 Results

Prior to hardware implementation, we carried out profiling on two classical ANN/GA problems, the XOR problem and the double pole balancing problem, using the NEAT software library. Despite the fact that both evolved topologies resulted in a small number of neurons, the evaluation of the ANN was the clear computational hot spot. Additionally, it is fair to assume that as the number of neurons in the genome increases (for example in multimedia tasks), this computational hot spot will only worsen.

The hardware design was captured in Verilog HDL and synthesised using Synopsys Design Compiler using a 90nm TSMC ASIC library. Power consumption was estimated from a gate level netlist with a 1.2 volt voltage source. A summary of the preliminary synthesis and power results can be seen in table 1. It should be noted that the area figures do not take into account the *NEURON* and *LINK* memory storage elements. The design was synthesised using a 200MHz and a 300MHz clock frequency, these are typical speeds of mobile microprocessors. As would be expected, the higher clock frequency design has a marginally larger area. Power estimates were generated using the data derived from the "Winning" topology for the XOR and double pole balancing tests generated in software. Owing to the fact that the topologies were small, little discernible difference was noted in the average power between the two data sets. We believe these power consumption figures are appropriate for deployment on a mobile device. Under optimum conditions, our proposed hardware calculates 8 multiply accumulate operations per clock cycle and requires 1 clock cycle for the calculation of the activation function. This compares favourably with modern mobile processors, which typically achieve a sustained 1 multiply accumulate operation per clock cycle. Comparison of results with other ANN implementations is difficult, as no other approach from the outset attempts to accelerate the ANN evaluation within neuroevolution and exploit the associated sparse topologies. Normalisation of results will be necessary to give a fair comparison and this is targeted as a future work item.

Table 1. Preliminary synthesis results

	Frequency [MHz]	Area [Gates]	Average Power [mW]
Proposed architecture	200	52,186	42.27
Proposed architecture	300	55,726	69.55

6 Future Work

The current bottleneck in the design is retrieving the topology from memory for the multiple PEs. Alternative micro-architectures are currently being investigated, which could give throughput benefits and improve the scalability of the architecture. The cache module warrants further investigation, in particular the effects of different caches sizes, architectures and different input datasets from NEAT. Before integration with the NEAT software library begins, a comparative power consumption study between fixed point, reduced word length floating point and stochastic implementations is planned. Benchmarking will be necessary to compare the performance (throughput and power consumption) of the dedicated hardware core to a software implementation running a general multimedia processing task such as face detection in video sequences.

7 Conclusions

The computational complexity associated with Neuroevolution for multimedia applications on mobile devices is highly undesirable from a throughput and power consumption perspective. This paper has proposed a viable hardware architecture for accelerating the neural network genome calculation. The architecture is flexible enough to process any ANN topology, whilst still providing a good trade off between area, power and throughput.

References

[1] Widrow, B., Rumelhart, D.E., Lehr, M.A.: Neural Networks: Applications in Industry, Business and Science. Communications of the ACM **37** (1994) 93–105

[2] Fogel, D.B., Fogel, L.J., Porto, V.W.: Evolving neural networks. Biol. Cybern. **63** (1990) 487–493

[3] Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evol. Comput. **10** (2002) 99–127

[4] Omondi, A.R.: Neurocomputers: A Dead End? International Journal of Neural Systems **10** (2000) 475–481

[5] Reyneri, L.: Implementation issues of neuro-fuzzy hardware: going toward hw/sw codesign. IEEE Transactions on Neural Networks **14** (2003) 176–194

[6] Kung, S., Hwang, J.: A Unified Systolic Architecture for Artifical Neural Networks. Journal of Parallel and Distributed Computing **6** (1989) 358–387

[7] Stanley, K., Bryant, B., Miikkulainen, R.: Real-time neuroevolution in the NERO video game. IEEE Transactions on Evolutionary Computation **9** (2005) 653–668

[8] Gaines, B.: Stochastic Computing Systems, Advances in Information Systems Science. Plenum Press New York (1969)

[9] Brown, B.D., Card, H.C.: Stochastic Neural Computation I: Computational Elements. IEEE Transactions on Neural Networks **50** (2001) 891–905

[10] Holt, J., Hwang, J.: Finite Precision Error Analysis of Neural Network Hardware Implementations. IEEE Transactions on Computers **42** (1993) 280 – 291

[11] Koren, I.: Computer Arithmetic Algorithms. 2nd edn. A K Peters Ltd (2001)

[12] Larkin, D., Kinane, A., Muresan, V., O'Connor, N.: An Efficient Hardware Architecture for a Neural Network Activation Function Generator. In: International Symposium on Neural Networks, Chengdu, China (2006)

[13] Hennessy, J.L., Patterson, D.A.: Computer Architecture: A Quantitative Approach. 3rd edn. Morgan Kaufmann Publishers (2003)

Neurocomputing for Minimizing Energy Consumption of Real-Time Operating System in the System-on-a-Chip

Bing Guo[1,*], Dianhui Wang[2], Yan Shen[3], and Zhishu Li[1]

[1] School of Computer Science & Engineering, SiChuan University, ChengDu 610065, China
guobing@sohu.com
[2] Department of Computer Science & Engineering, La Trobe University, Melbourne, VIC 3086, Australia
csdhwang@ieee.org
[3] School of Mechatronics Engineering, University of Electronic Science & Technology of China, ChengDu 610054, China
shenyan02@163.com

Abstract. The RTOS (Real-Time Operating System) is a critical component in the SoC (System-on-a-Chip), which consumes the dominant part of total system energy. A RTOS system-level power optimization approach based on hardware-software partitioning (RTOS-Power partitioning) can significantly minimize the energy consumption of a SoC. This paper presents a new model for RTOS-Power partitioning, which helps in understanding the essence of the RTOS-Power partitioning techniques. A discrete Hopfield neural network approach for implementing the RTOS-Power partitioning is proposed, where a novel energy function, operating equation and coefficients of the neural network are redefined. Simulations are carried out with comparison to other optimization techniques. Experimental results demonstrate that the proposed method can achieve higher energy savings up to 60% at relatively low costs.

Keywords: Hopfield neural network, Power optimization, RTOS, Hardware-software partitioning, SoC.

1 Introduction

Along with the development of global energy crisis, power issues in the vast number of embedded systems have been increasingly concerned. As a new type of embedded systems, a SoC (System-on-a-Chip) almost implements the functionality of an overall computer system in a single IC (Integrated Chip). In general, embedded software in the SoC contains RTOS (Real-Time Operating System) and user applications. The RTOS in the SoC is shortly called SoC-RTOS. Embedded software execution drives the circuit activities of the underlying hardware, including MPU, memory and I/O peripherals, while the manner in which software uses hardware can have a substantial impact on the power dissipation of a SoC [1, 2]. The SoC-RTOS forms a critical component of embedded software, and provides benefits ranging from hardware abstraction and resource management to user applications. The SoC-RTOS not only

[*] Corresponding author.

occupies a significant portion of machine cycles but also can consume the domain part of total system energy. Therefore, it is becoming crucial to optimize the energy consumption of the SoC-RTOS [3, 4].

Hardware-software partitioning is a well-established design methodology with the goal to increase the performance of a system at some costs. Gupta and De Micheli developed an iterative improvement algorithm to partition real-time embedded systems between a co-processor and a general-purpose processor [5]. Eles et al. proposed a simulated annealing and taboo search hardware-software partitioning algorithm [6]. Saha et al. applied a genetic algorithm for hardware-software partitioning [7]. Filho et al. designed a Petri Nets based approach for hardware-software partitioning of embedded systems [8]. Xiong et al. suggested a dynamic combination of genetic algorithm and ant algorithm for hardware-software partitioning of embedded systems [9]. Arató et al. discussed an algorithm based on integer linear programming to solve the partitioning problem optimally even for quite big systems [10]. Stitt et al. considered a solution for dynamic hardware-software partitioning [11]. In [12], we presented a hardware-software partitioning approach of the SoC-RTOS based on Hopfield neural networks, which optimize the running time of the SoC-RTOS under the constraint of hardware area, remarkably improve the performance of the multi-task SoC-RTOS. However, most of these solutions' objective is to meet performance constraints while keeping the system cost (e.g. total chip area) as low as possible. None of them provide power-aware optimization and estimation strategies.

In [13], Henkel suggested a comprehensive system-level power optimization approach for core-based embedded systems that deploys hardware-software partitioning based on fine-grained (instruction/operation-level) power estimation analysis. Contrast with some local power optimization techniques as described in [1] and [2], this system-level power optimization approach based on hardware-software partitioning can yield high energy savings between 35% and 94% compared to the initial design. Still, there is no guarantee that a hardware-software partition actually meets imposed power constraints because accurate estimations are very difficult at the instruction or operation-level. Meanwhile, it is hard to assure that the partitioning solution of an embedded system solved by a simple list schedule method is optimal or near optimal.

However, *hardware-software power-aware automated partitioning of the SoC-RTOS* (RTOS-Power partitioning) is significant to the SoC design, *i.e.* determining which components of the SoC-RTOS should be realized in the hardware and which ones should be in the software, and minimizing energy consumption for the same function but also perform it more quickly. Due to the characteristics of the SoC-RTOS, the RTOS-Power partitioning is quite different from the partitioning of embedded systems and SoCs. Usual hardware-software partitioning methods are inadequate for such RTOS-Power partitioning tasks in many aspects. The composition of hardware and software elements in the SoC-RTOS creates some new problems, such as power modeling and estimation of the SoC-RTOS, refining power constraints and multi-object conditions, designing an appropriate power optimization algorithm, evaluating the partitioning results, and system architecture issues. In this paper, we focus on the optimization algorithm development and design of RTOS-Power partitioning [1].

2 Description of the RTOS-Power Partitioning Problem

The main objective of the RTOS-Power partitioning is to optimally allocate the functional behavior of the RTOS to the hardware-software system of the SoC so that the system energy consumption is minimized. The functional behavior of the SoC-RTOS can be modeled by a task graph, in which each task has it own energy attribute, and task-based energy modeling and estimation strategies are adopted. For software, a task is a set of coarse-grained operations with definite interface, which can be an algorithm procedure, an object or a component; for hardware, a task is a specific IP (Intellectual Property) module with clear functions, interface and constraints [11, 12].

To formulate our problem, the following notations are used in this paper:

G : A *directed acyclic graph* (DAG) and also refers to the task graph of a SoC-RTOS, $G = (V, E)$

V : The task node set that has to be partitioned, $V = \{v_1, v_2, \ldots, v_n\}$

E : The directed edge set that represents the control or data dependency and communication relationship between two nodes, $E = \left\{ e_{ij} \middle| v_i, v_j \in V, i \neq j \right\}$

N : Total number of task nodes belonging to G , $N = |V|$

M : Total number of directed edges belonging to G , $M = |E|$

P : A power-aware hardware-software partitioning of G

V_H : The subset of nodes partitioned into the hardware, $V_H \subseteq V$

V_S : The subset of nodes partitioned into the software, $V_S \subseteq V$

$s(v_i)$ (or s_i) : The software costs of v_i

$h(v_i)$ (or h_i): The hardware costs of v_i

$c(v_i, v_j)$ (or c_{ij}): The communication costs between v_i and v_j if they are in different contexts (hardware or software) whereas the communication costs between the nodes in the same context are neglected

c_i : The sum of c_{ji} , $c_i = \sum_{j=1, j\neq i}^{N} c_{ji}$

H_P : The hardware costs of P , $H_P = \sum_{v_i \in V_H} h_i$

S_P : The software costs of P , $S_P = \sum_{v_i \in V_S} s_i$

C_P : The communication costs of P , $C_P = \sum_{v_i \in V_S, v_j \in V_H \text{ or } v_i \in V_H, v_j \in V_S} c_{ij}$

$g_p(V_H, V_S)$: The total costs of P

$f_P(V_H, V_S)$: The total energy consumption of P

Definition (k-way partitioning). Given $G = (V, E)$, it is called k-way partitioning if there exists a cluster set $P = \{p_1, p_2, \ldots, p_k\}$ such that $\bigcup_{i=1}^{k} p_i = V$ and $p_i \cap p_j = \phi, i \neq j$.

When $k = 2$, P is called bi-partitioning, which means that only one software context (e.g. one general-purpose processor) and one hardware context (e.g. one ASIC or FPGA) are considered in the target system; when $k > 2$, P is called multi-way partitioning, which means that multiple software contexts and multiple hardware contexts are considered in the target system. According to the architecture of the target system, the RTOS-Power partitioning can be categorized into bi-partitioning and multi-way partitioning. Bi-partitioning is the foundation of the multi-way partitioning, and is widely applied in domain applications. In this paper, the partitioning only refers to the bi-partitioning without any additional declaration.

Problem (RTOS-Power partitioning). Given $P = (V_H, V_S)$, $V_H \cup V_S = V$ and $V_H \cap V_S = \phi$, the RTOS-Power partitioning is formulated as the following constrained optimization problem:

$$\min f_P(V_H, V_S), \tag{1}$$

$$s.t.\ C_{\min} \leq g_p(V_H, V_S) = H_P + S_P + C_P \leq C_{\max}, \tag{2}$$

$$v_i \in V, e_{ij} \in E, 1 \leq i, j \leq n, i \neq j,$$

where $C_{\min} > 0$ and $C_{\max} > 0$ are the two given cost values of the SoC-RTOS.

3 A Novel Discrete Hopfield Neural Network Approach

The *discrete Hopfield neural network approaches* (DHNNA) have been successfully applied to signal and image processing, pattern recognition and optimization. In this paper, we employ this type of neural network to solve the RTOS-Power partitioning optimization problem.

3.1 Neuron Expression

A neural network with N neurons is used to give a response for each of the N nodes in the graph G. The *i*-th neuron belongs to the subset with node *i* and has an input U_i and output V_i. The activation functions of the neuron are given by:

$$V_i = f\left(\sum_{j=1, j\neq i}^{N} W_{ji} V_j - \theta_i \right) = f(U_i) = \begin{cases} 0, & if\ U_i > 0 \\ 1, & if\ U_i \leq 0 \end{cases}, \tag{3}$$

where $W_{ji}V_j = h_j(1-V_j)+s_jV_j+c_{ji}((1-V_j)+V_j)$, $\theta_i = \alpha_i(1-V_j)+\beta_iV_j+\varphi_{ij}((1-V_j)+V_j)$, the weights W_{ji} is a sum of the costs of v_j (i.e., s_j or h_j) and the communication costs c_{ji}, θ_i is the threshold value of *i*-th neuron (α_i, β_i and φ_{ij} are explained in subsection 3.2). Meanwhile, the neuron output $V_i = 0$ indicates $v_i \in V_H$ and $V_i = 1$ indicates $v_i \in V_S$.

To avoid the local optimum caused by initial conditions, the neuron input value should be restricted within a certain range. The upper limit $U_{\max}$ and the lower limit $U_{\min}$ of the neuron input are set as follows, where the average value U_{avg} of c_{ij} is calculated from all connected task nodes [14]:

$$U_{avg} = \frac{2}{N}C_P\,, \quad U_{\min} = -\frac{U_{avg}}{2}\,, \quad U_{\max} = \frac{U_{avg}}{2}\,, \tag{4}$$

$$U_i = \begin{cases} U_{\min}\,, & if\ U_i < U_{\min} \\ U_{\max}\,, & if\ U_i > U_{\max} \end{cases}. \tag{5}$$

3.2 Energy Function

In response to the constraint and objective condition of the RTOS-Power partitioning, an energy function consisting of the following two terms is defined by:

$$E = \frac{A}{2}E_1 + \frac{B}{2}E_2\,, \tag{6}$$

$$E_1 = \sum_{i=1}^{N}\sigma_i^2\left(\sum_{j=1}^{N}\left(h_jV_i(1-V_j)+s_j(1-V_i)V_j+c_{ij}(V_i(1-V_j)+(1-V_i)V_j)\right)-C_{\min}\right), \tag{7}$$

$$\begin{aligned} E_2 &= f_p(V_H, V_S) \\ &= \sum_{i=1}^{N}\left(\sum_{j=1, j\neq i}^{N}\left(\alpha_iV_i(1-V_j)+\beta_i(1-V_i)V_j+\varphi_{ij}(V_i(1-V_j)+(1-V_i)V_j)\right)\right), \end{aligned} \tag{8}$$

where A and B are two positive coefficients which are specified in Subsection 3.4 below, α_i is the energy consumption of one task realized in the hardware and has the different values for different tasks, β_i is the energy consumption of one task realized in the software and has the different values for different tasks, φ_{ij} is the energy consumption of one corresponding communication and has the different values for different communications [12, 13, 15].

The function $\sigma_i(x)$ used in Eq. (7) is given by:

$$\sigma_i(x)=\begin{cases} x+C_{\min}-(h_i+s_i+c_i), & if\ x<-C_{\min}+(h_i+s_i+c_i), \\ 0, & if\ 0\le x\le C_{\max}-C_{\min}, \\ x-C_{\max}+(h_i+s_i+c_i), & if\ x>C_{\max}-(h_i+s_i+c_i). \end{cases} \tag{9}$$

The E_1 represents one term of the energy function, which is associated to the constraint $g_P(V_H,V_S)=\sum_{i=1}^{N}\left(\sum_{j=1,j\neq i}^{N}\left(h_jV_i(1-V_j)+s_j(1-V_i)V_j+c_{ij}\left(V_i(1-V_j)+(1-V_i)V_j\right)\right)\right)$, while E_2 is associated to the objective function $f_P(V_H,V_S)$, which indicates the energy consumption of the SoC-RTOS [4, 12].

3.3 Operating Equation

The operating equation for the *i*-th neuron is governed by:

$$\begin{aligned} \frac{dU_i}{dt}&=-\frac{\partial E}{\partial V_i} \\ &=-A\sigma_i\left(\sum_{j=1}^{N}\left(h_jV_i(1-V_j)+s_j(1-V_i)V_j+c_{ij}(V_i+V_j-2V_iV_j)\right)-C_{\min}\right)\times \\ &\left(\sum_{j=1}^{N}\left(h_j(1-V_j)-s_jV_j+c_{ij}(1-2V_j)\right)\right)-\frac{B}{2}\sum_{j=1,j\neq i}^{N}\left((\alpha_i+\varphi_{ij})-(\alpha_i+\beta_i+2\varphi_{ij})V_j\right). \end{aligned} \tag{10}$$

In order to avoid the local optimum and obtain a high quality solution within a limited computation time, a noise term given by Eq. (11) is added to the operating Eq. (10); that is,

$$D=\eta(1-2V_j)=\begin{cases} +\eta, & if\ \ V_j=0 \\ -\eta, & if\ \ V_j=1 \end{cases}. \tag{11}$$

If the noise term D is kept adding in the updating rule, the state changes excessively and even a local optimum solution may not be reachable. Hence, the term D will be discarded in the operating equation as $[t/T_0]\ge\lambda$, where $[\bullet]$ is a round-off operator, i.e., it gives an integer most close to the entity, $\lambda=T_0-(t\times T_0)/T_{\max}-1$, T_0 is a positive coefficient and $T_{\max}$ is a maximal step of iterations.

3.4 Setting of Coefficients for the Operating Equation

The coefficient A depends on the average value $\overline{\omega}$ of the task node costs; that is,

$$\overline{\omega} = \frac{H_P + S_P}{N}. \tag{12}$$

The coefficient B depends on the U_{avg}. In this study, we set $A\overline{\omega} = KBU_{avg}$, where K is a regularizing constant. In our simulations, we take $K = 1/3$, $B = 1$, $T_0 = 20$, $T_{\max} = 300$ and $\eta = U_{avg}/(3 + 10 \times \rho)$. Here, ρ is the edge generation ratio of the graph G ($0 < \rho \leq 1$), which is expressed by $M/(N(N-1)/2)$ [14].

The values of α_i, β_i and φ_{ij} can be collected by a low-level energy simulator *EMSIM*, which simulates a SoC platform featuring the Intel StrongARM processor and RTOS μC/OS. Additionally, the execution time of each task in different contexts and conditions can be collected by this simulator [4].

Note that $C_{\min} = \sum_{i=1}^{N} s_i$ is a minimal cost if the SoC-RTOS is totally realized by the software, and $C_{\max} = \sum_{i=1}^{N} h_i$ is a maximal cost if the SoC-RTOS is totally realized by the hardware. To achieve a more practical and favorable solution, we amend the maximal cost to be $C_{\max} = \frac{1}{2}\sum_{i=1}^{N} h_i$ in this study [15, 16].

4 Experimental Results

To verify the feasibility and effectiveness of the proposed optimization techniques in this paper, we employed the similar simulation methods used in [4], [9] and [12]. Also, a comparative study was carried out with the *Genetic Algorithm* (GA) and *Ant Algorithm* (AA), where parameters are set with the same in [9].

4.1 Simulation Conditions

To date, no standard benchmark for this topic is available. The methods commonly adopted in the literature are to generate the random DAG and to assign some attributes to the nodes and edges.

In this simulation, we constrain the settings as follows:

(1) Use the GVF (Graph Visualization Framework) software package to generate 5 groups of random DAG as our task graphs. The number of task nodes (N) in each group is 50, 200, 500, 1000 and 1500, respectively. Each group has 20 sample graphs, in which each graph has the different edge generation ratios

(ρ). The average value of 20 samples in each group is taken as the final energy consumption result of this group.

(2) The costs of task nodes and communication costs of edges, each task node is related with two functions for one is a hardware function and another is a software function, while each edge is associated with one function. The output of function is taken as the cost of task node and edge. The appropriate function for each task node and edge are chosen from the MediaBench benchmark program package [9].

(3) As the initial partitioning, $N/2$ task nodes are assigned to each subset.

(4) Target system architecture contained one Intel StrongARM core and one Xilinx Spartan-3 S1000 FPGA model.

(5) The simulation environment used the Intel Celeron 2.6GHz processor, 512MB SDRAM, Linux 9.0 operating system and KDevelop 3.2 IDE.

4.2 Simulation Results and Analysis

Table 1 shows the experimental results of energy saving percentage with the purely software realized SoC-RTOS and $f_P(V_H, V_S)$ (unit: mJ) produced by the DHNNA, GA and AA on the different node number. Fig. 1 shows the relationship between $f_P(V_H, V_S)$ and count of tasks nodes in these three algorithms.

Table 1. Results of $f_P(V_H, V_S)$ and energy saving percentage produced by the DHNNA, GA and AA

Total DAG Nodes	DHNNA		GA		AA	
	$f_P(V_H,V_S)$	Sav (%)	$f_P(V_H,V_S)$	Sav (%)	$f_P(V_H,V_S)$	Sav (%)
50	221.73	-42.41	281.29	-26.94	234.26	-39.22
200	538.86	-48.87	686.21	-34.89	564.55	-46.45
500	985.04	-51.24	1276.87	-36.79	1033.67	-48.84
1000	4308.65	-57.80	6236.32	-38.92	4879.74	-52.21
1500	6412.37	-60.08	9605.82	-40.20	7216.95	-55.07

The aim of this experiment is to optimize the energy consumption of the SoC-RTOS under the constraint of total chip area. The additional hardware overhead of FPGA occupied by the SoC-RTOS is less than 30k cells. It is observed that the energy consumption from the DHNNA is significantly less than that obtained by GA, and slightly less than that obtained by the AA. In fact, combined with our research results in [12], since some parts of the SoC-RTOS are realized in the hardware, they not only consume less energy but also perform more quickly. This is of paramount importance: we achieved high energy savings but not at the cost of performance. Meanwhile, the DHNNA can be also applied to the hardware-software partitioning of embedded systems and SoCs, while taking into account other constraints and optimization performances, such as hard real-time and multi-processor.

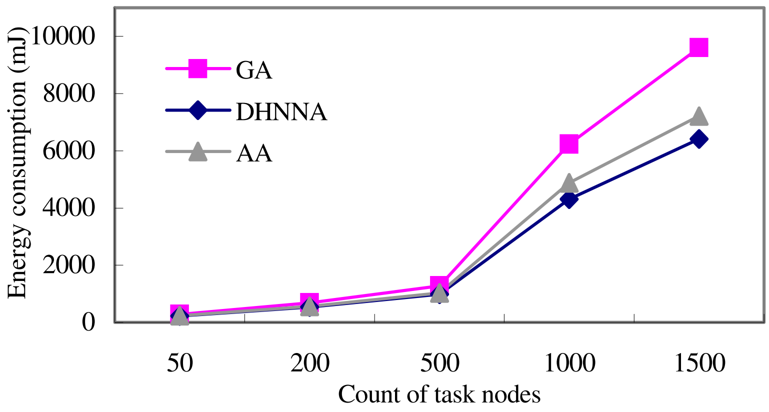


Fig. 1. Energy consumption/Nodes curve

Along with an increase of the node number, the value of $f_P(V_H, V_S)$ is increasing continuously. This is a main disadvantage of the DHNNA. Practically it is expected to give an exact and more stable solution for the $f_P(V_H, V_S)$. The one reason resulting in such a case may be of the initial condition and network parameters. Another reason may be of that the power modeling and estimation of the SoC-RTOS is not exact enough. Therefore, the energy macromodeling of the SoC-RTOS need to be further investigated, which is a more exact power modeling method based on service routine level, and can effectively improve the RTOS-Power partitioning results.

5 Conclusions

A discrete Hopfield neural network approach for solving a problem of RTOS-Power partitioning is proposed in this paper. According to the characteristics of the SoC-RTOS partitioning, a new energy function for a Hopfield neural network is defined, and some practical considerations on the state updating rule are given. Simulation results demonstrate that our method outperforms other power optimization approaches. Our technique can achieve tremendous energy savings of up to 60% at relatively small costs. Further work will concentrate on the RTOS-Power partitioning based on energy macromodeling of the SoC-RTOS with comparative studies.

References

1. Jerraya, A.A., Yoo, S., Verest, D., When, N.(eds): Embedded Software for SoC. Kluwer Academic Publishers, Netherlands , ISBN 1-4020-7528-6 (2003) 3-11
2. Dick, R.P.: Multi-objective Synthesis of Low-Power Real-time Distributed Embedded Systems. Ph.D Dissertation, the Department of Electrical Engineering, Princeton University (2002) 11-21
3. Li, T., John, L.K.: Run-time modeling and estimation of operating system energy consumption. In proceeding of SIGMETRICS'03, San Diego (2003) 160-171

4. Tan, T.K., Raghunathan, A., Jha, N.K.: Energy Macromodeling of Embedded Operating Systems. ACM Transactions on Embedded Computing Systems (TECS), 4(1) (2005) 231-254
5. Gupta, R.K., De Micheli, G.: Hardware-Software Co-synthesis for Digital Systems. IEEE Design and Test of Computers, 10(3) (1993) 29-41
6. Eles, P., Peng, Z., Kuchcinski, K., Doboli, A.: System Level Hardware/Software Partitioning Based on Simulated Annealing And Tabu Search. Design Automation for Embedded Systems, 2(1) (1997) 5-32
7. Saha, D., Mitra, R.S., Basu, A.: Hardware/Software Partitioning Using Genetic Algorithm. Proc. of Int. Conf. on VLSI Design, Hyderabad (1998) 155-159
8. Filho, F.C., Maciel, P., Barros, E.: A Petri Nets Based Approach for Hardware/Software Partitioning. Integrated Circuits and system design, 8(6) (2001) 72-77
9. Xiong, Z.H., Li, S.K., Chen, J.H.: Hardware/Software Partitioning Based on Dynamic Combination of Genetic Algorithm and Ant Algorithm. Journal of software, 16(4) (2005) 503-512
10. Arató, P., Juhász, S., Mann, Z.Á., Papp, D.: Hardware-Software Partitioning in Embedded System Design. Proceedings of the IEEE International Symposium on Intelligent Signal Processing, Budapest (2003) 63-69
11. Stitt, G., Lysecky, R., Vahid, F.: Dynamic Hardware/Software Partitioning: A First Approach. Design Automation Conference (DAC), Anaheim (2003) 74-81
12. Guo, B., Wang, D.H., Shen,Y., Li, Z.S., Liu, Z.: Hardware-software Partitioning of Real-time Operating Systems Using Hopfield Neural Networks. NeuroComputing (2006) (In press)
13. Henkel, J.: A low power hardware/software partitioning approach for core-based embedded systems. Design Automation Conference (DAC) , New Orleans (2002) 122-127
14. Tamaki, Y., Funabiki, N., Nishikawa, S.: A Binary Neural Network Algorithm for the Graph Partitioning Problem. Electronics and Communications in Japan, Part 3, Vol. 82, No. 12 (1999) 34-42
15. Pereira, C., Raghunathan, V., Gupta, S., Gupta, R., Srivastava, M.: A software architecture power aware real time operating systems. CECS Technical Report 02-02, University of California, Irvine (2002)
16. He, Z.T., Mok, A., Peng, C.: Timed RTOS modeling for embedded system design. Proceedings of the IEEE Real-time and Embedded Technology and Applications Symposium (RTAS'05) , San Francisco (2005) 54-63

A Novel Multiplier for Achieving the Programmability of Cellular Neural Network

Peng Wang, Xun Zhang, and Dongming Jin

Institute of Microelectronics, Tsinghua University, Beijing
100084, P.R. China
{wangpeng00, zhangxun97}@mails.tsinghua.edu.cn,
jdmime@tsinghua.edu.cn

Abstract. A novel CMOS four-quadrant analog-digital multiplier for implementing a programmable Cellular Neural Network (CNN) is presented. The circuit, which can be fabricated in a standard CMOS process, performs the four-quadrant weighting of interconnect signals. Using this multiplier a programmable CNN neuron can be implemented with little expense. Both simulation and test results are given for the circuit fabricated in a standard, mixed signal, 0.18μm, CMOS process. According to this design, one input is analog voltage and the other input is digital signal. The linearity deviation is less than 1% in the dynamic range (1.0V,2.2V) centered on V_{ref}=1.6V. The power supply voltage is 3.3V.

1 Instruction

Cellular Neural Network[1] consists of arrays of elementary processing units (cell), each one connected only to a set of adjacent cells (neighbors), and exhibits potential application in various processing tasks[2]-[4], such as pattern recognition, motion detection, etc. The local connectivity property makes CNN's routing easy, allowing increased cell density per silicon area and making these computation paradigms very suitable for VLSI implementation.

Up to now, several proposals of practical CNN implementations have been covered in different papers[5]-[7]. These papers mostly focus on VLSI implementation and use analog techniques. In this paper, a novel multiplier is presented to implement CNN neuron cell, where one input variable is of the form of analog voltage signal, the other is weight in the form of digital signal, and the output is a current representing the product. The precision of digital weight is 4-bits and can be extended. The design presented here is in simple structure and can be implemented in standard CMOS process.

2 Cellular Neural Network

The operation of the conventional CNN's basic cell is defined by the following equation:

$$C_x \frac{dv_{xij}(t)}{dt} = -\frac{1}{R_x} v_{xij}(t) + \sum_{C(k,l)\in N_r(i,j)} A(i,j;k,l) v_{ykl}(t) + \sum_{C(k,l)\in N_r(i,j)} B(i,j;k,l) v_{ukl} + I \qquad (1)$$

Where 1≤*i*,*k*≤M, 1≤*j*,*l*≤N, C_x, R_x and *I* are the integrating capacitor, the lossy element and the offset current, respectively. Cells which is adjacent to the neuron *C*(*i,j*) constitute the neighborhood $N_r(i,j)$. *A*(*i,j;k,l*) is the element (*k,l*) of the feedback template *A* associated with the state of cell C(*i,j*), which represents the impact of the output of the neighboring neuron *C*(*k,l*) on the *C*(*i,j*). *B*(*i,j;k,l*) is the element (*k,l*) of the control template *B* associated with the input of the cell *C*(*i,j*) , which represents the control of the input of the neighboring neuron *C*(*k,l*) on the *C*(*i,j*). v_{xkl}, v_{ykl} and v_{ukl} are the state, output and input of the *C*(*k,l*), respectively. Fig. 1 shows the simplified block diagram of a neuron cell, in which the *B* template is omitted. The output and the state of the cell are related by a non-linear equation, typically given by

$$v_{yij}(t) = \frac{1}{2}\left(\left|v_{xij}(t)+1\right| - \left|v_{xij}(t)-1\right|\right) \tag{2}$$

These neuron cells can be directly connected in parallel using a simple structure with each cell forming one node of the network.

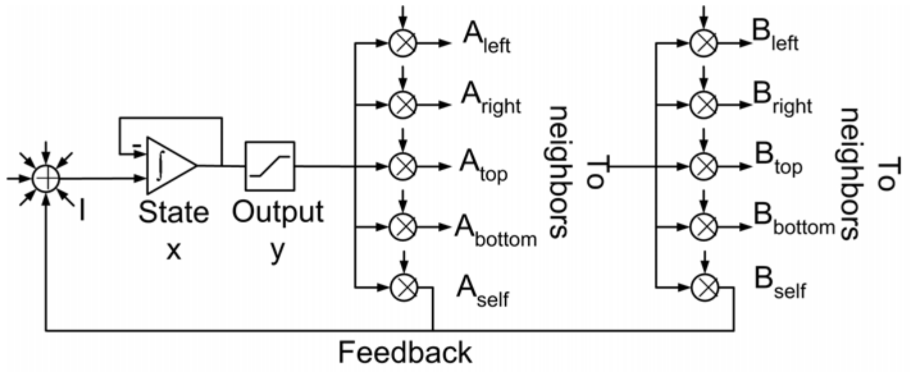


Fig. 1. Block diagram for a cell

The core of neuron cell is multiplier. Various applications of CNN require suitable implementation of multiplier. For example, implementation of CNN with fixed cloning template must be different to that one with programmable cloning template. The multiplier should perform a multiplication between a variable signal (the input or the output of the cell) and a fixed value (the weight of the cloning template), which value should be modified in accordance with different algorithms for different applications.

Most of these programmable implementations and designs refer to one common analog multiplier used in neuron cell. It is Gilbert multiplier, which consists of three differential pairs operating in saturation region. There are some drawbacks to above Gilbert multiplier. The dynamic range of Gilbert multiplier is limited due to more transistors in one shunt. Additionally, it can only perform multiplication of two analog voltage signals. Thus, the incoming cloning template should be in the form of analog signal, so it needs to design additional circuit block in order to communicate with external digital device.

3 Design of the Multiplier

Fig. 2 shows the schematic of the present multiplier. The basic idea behind this circuit is very simple in concept: that is, the input voltage variable is converted to a differential current output by a transconductance differential amplifier (OTA in the following), then the current is modified proportionally by a set of current mirrors in accordance with the product of an input signal and a dynamically-refreshed digital weight. The basic building block of multiplier is a transconductance differential amplifier (M_{B1}~M_{B6}, M_{01}~M_{08}), a middle stage (M_{09}~M_{18}) and a set of current mirrors (MD_{01}~MD_{24}).

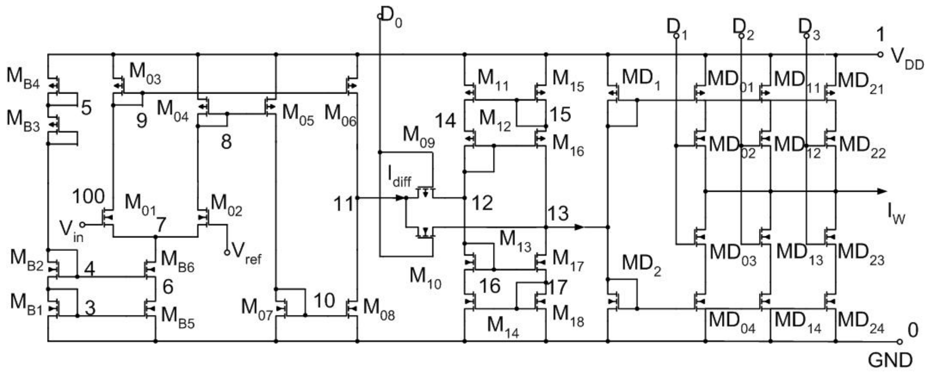


Fig. 2. Schematic circuit diagram of the designed CMOS multiplier

Transistors M_{B1}~M_{B6} constitute the bias circuit and supply a steady tail current to the OTA. The bias current is 10μA. M_{01}~M_{08} constitute the OTA, wherein M_{01} and M_{02} are operated in linear region, the difference of output current I_{01} and I_{02} is given by

$$\begin{aligned} I_{diff} &= I_{01} - I_{02} \\ &= \frac{\beta_{01}}{2}\left[2(V_{in} - V_7 - V_{TN})V_{DS01} - V_{DS01}{}^2\right] - \frac{\beta_{02}}{2}\left[2(V_{ref} - V_7 - V_{TN})V_{DS02} - V_{DS02}{}^2\right] \quad (3) \\ &= \beta(V_{in} - V_{ref})V_{DS} \end{aligned}$$

where $\beta = \mu_0 C_{ox} W/L$, V_{ref} is bias voltage, and V_{TN} is the threshold voltage of the n-type MOSFET.

The equation (3) gives the relation of a two-quadrant multiplier where V_{in}-V_{ref} may have both positive and negative values and V_{DS} is positive. In order to be compatible with the digital weight signal, the bias voltage V_{ref} connected to the gate of M_{02} is set to 1.6V.

The middle stage (M_{09}~M_{18}) is controlled by the MSB bit of the digital weight signal, and can provide a countercurrent of the output of the OTA. In order to keep linearity, the maximum of the current is limited to10μA. For the current mirror we have used improved Wilson Current mirror to reduce mismatch.

The programmable weighting circuit (MD_{01}~MD_{24}) shown at the right of the schematic is a binary-weighted current source array with the capability of four-quadrant multiplication. The magnitude of multiplication is done by n-1 LSB bits, the polarity of the output I_W is controlled by swapping the differential current through the MSB bit as described above. In this design, n=4 and the sizes of transistors for weights are chosen such that $|I_W| \leqslant 3.5|I_{diff}|$ in a step of $0.5I_{diff}$. So the ratios of the current mirrors are 2:1, 1:1, and 1:2, respectively. Because the current mirrors are used many times, it is important to match them as closely as possible through a careful layout design. Then, the output of the multiplier is shown to

$$I_w=D_3\times 2I_{diff}+D_2\times I_{diff}+ D_1\times 1/2I_{diff} \tag{4}$$

Because during the operation of the CNN the template A and B keep steady, the proposed multiplier is very applicable to implement feedback and control operator. It is apparent that the LSB bits can be easily extended and the ratios of current mirrors can also be changed according to the practical application.

4 Simulation and Test Results

The multiplier chip was fabricated using SMIC 0.18μm Mixed Signal CMOS process. Fig.3 depicts the layout of the multiplier.

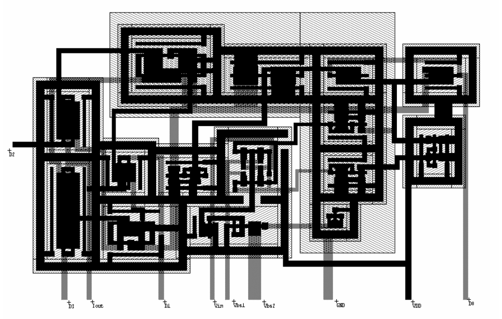
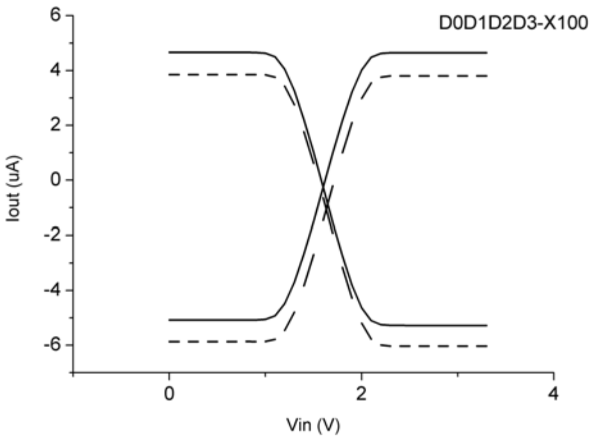


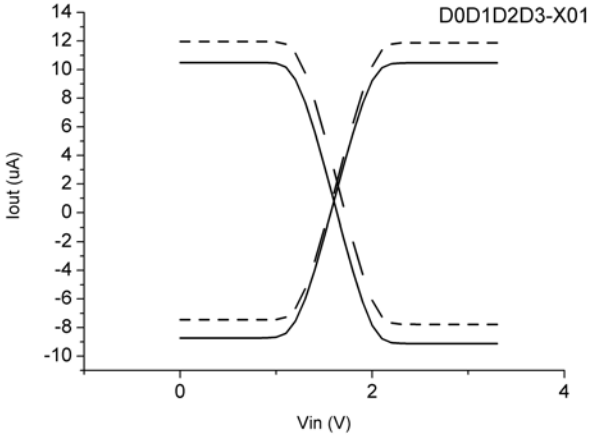
Fig. 3. Layout of the multiplier

The power supply voltage is 3.3V, and the reference voltage is 1.6V. Both the resulting SPICE simulations and test results of the four-quadrant multiplier are shown in Fig. 4. Moreover, Fig.4(a)-(e) illustrate the plots obtained when apply "X100", "X010", "X001", "X110"and"X111" to digital signal lines D0, D1, D2, and D3, respectively. The solid lines illustrate the simulation result, and the dot lines illustrate the test result.

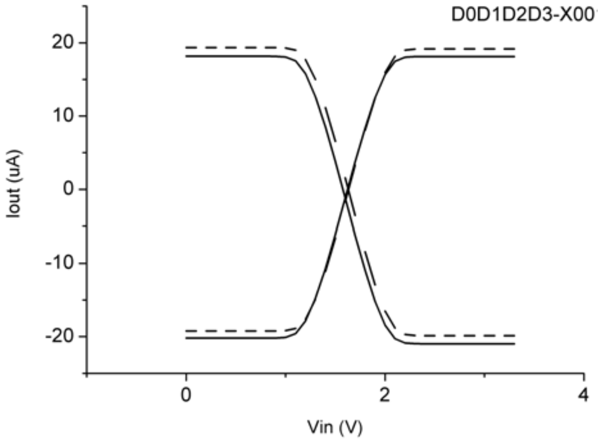
Results indicate that the linearity deviation less than 1% can be achieved within the dynamic range (1V~2.2V), the zero drift is little, and the precision increases as the digital factor increases.



(a)

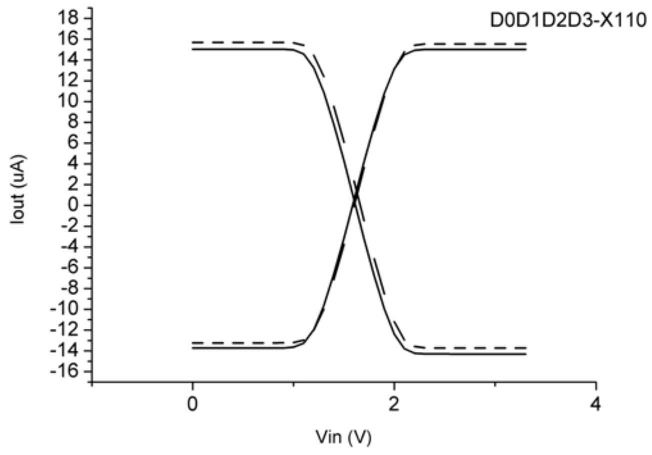


(b)

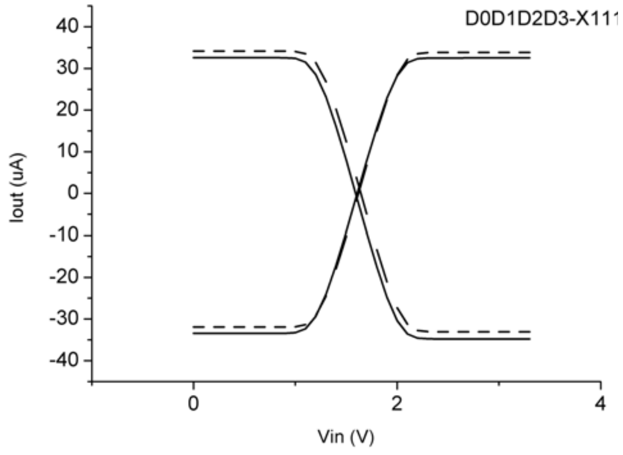


(c)

Fig. 4. Simulation and test result



(d)



(e)

Fig. 4. (*continued*)

5 Conclusion

A novel multiplier based on analog design techniques using for implementing programmable CNN has been proposed. The input and output of the neuron cell are in the form of analog voltage signal and the fixed template A and B templates are in the form of digital signal. The simulation results have been given and indicate the multiplier has the characteristics of simple structure, large dynamic range and high precision. The proposed multiplier can also be used in other applications like Kohonen and Hamming artificial neural architectures.

References

1. Chua, L.O., Yang, L.: Cellular Neural Networks: Theory. Circuits and Systems, IEEE Transactions on, Vol. 35 Issue: 10, P. 1257-1272 (1988)
2. Harrer, H., Venetianer, P.L., Nossek, J.A., Roska, T., Chua, L.O.: Some Examples of Preprocessing Analog Images with Discrete-Time Cellular Neural Networks. Cellular Neural Networks and their Applications, 1994. CNNA-94, Proceedings of the Third IEEE International Workshop on, P. 201-206 (1994)
3. Tanaka, M., Tanji, Y., Onishi, M., Nakaguchi, T.: Lossless Image Compression and Reconstruction by Cellular Neural Networks. Cellular Neural Networks and Their Applications, 2000. (CNNA 2000). Proceedings of the 2000 6th IEEE International Workshop on, P. 57-62 (2000)
4. Khryasshyov, V.V., Sautov, E.Yu., Sokolenko, E.A.: Cellular Neural Network in Image Filtration Tasks. Circuits and Systems for Communications, 2002. Proceedings. ICCSC '02. 1st IEEE International Conference on , P. 267-270 (2002)
5. Cardarilli, G.C., Lojacono, R., Salerno, M., Sargeni, F.: VLSI Implementation of a Cellular Neural Network with Programmable Control Operator. Circuits and Systems, 1993, Proceedings of the 36th Midwest Symposium on, vol.2, P. 1089-1092 (1993)
6. Espejo, S., Carmona, R., Dominguez-Castro, R., Rodriguez-Vazquez, A.: A CNN Universal Chip in CMOS Technology. International Journal of Circuit Theory and Applications, vol. 24, P. 93-109 (1996)
7. Sindhwani, M., Srikanthan, T., Asari, K.V.: VLSI Efficient Discrete-Time Cellular Neural Network Processor. Circuits, Devices and Systems, IEE Proceedings, Vol. 149 Issue. 3 , P. 167-171 (2002)

Neural Network-Based Scalable Fast Intra Prediction Algorithm in H.264 Encoder

Jung-Hee Suk, Jin-Seon Youn, and Jun Rim Choi

School of Electrical Engineering and Computer Science,
Kyungbook National University, Daegu, Korea
{sukjunghee, gbrave, jrchoi}@ee.knu.ac.kr

Abstract. In this paper, we propose a neural network-based scalable fast intra prediction algorithm in H.264 in order to reduce redundant calculation time by selecting the best mode of 4×4 and 16×16 intra prediction. In this reason, it is possible to encode compulsively by 4×4 intra prediction mode for current MB(macro block)'s best prediction mode without redundant mode decision calculation in accordance with neural network's output resulted from co-relation of adjacent encoded four left, up-left, up and up-right blocks. If there is any one of MBs encoded by 16×16 intra prediction among four MBs adjacent to current MB, the probability of re-prediction into 16×16 intra prediction will become high. We can apply neural networks in order to decide whether to force into 4×4 intra prediction mode or not. We can also control both the bit rates and calculation time by modulating refresh factors and weights of neural network's output depend on error back-propagation, which is called refreshing. In case of encoding several video sequences by the proposed algorithm, the total encoding time of 30 input I frames are reduced by 20% ~ 65% depending upon the test vector compared with JM 8.4 by using neural networks and by modulating scalable refreshing factor. On the other hand, total encoding bits are increased by 0.8% ~ 2.0% at the cost of reduced SNR of 0.01 dB.

1 Introduction

In intra mode a prediction block P is formed based on previously encoded and reconstructed blocks and is subtracted from the current block prior to encoding. For the luma samples, P is formed for each 4×4 block or for a 16×16 MB (macro block). In H.264 reference software (JM 8.4), the encoder typically selects the prediction mode for each block that minimizes the difference between a prediction block P and the block to be encoded after the total 144 mode operations for nine prediction modes of each sixteen 4×4 luma block and for four prediction modes of each 16×16 macro block. For many video sequences encoded in H.264, most of MBs are encoded by 4×4 intra prediction mode instead of 16×16 intra prediction. There are a total of nine optional prediction modes for each 4×4 luma block, four modes for a 16×16 luma block and four modes for chroma components. The encoder typically selects the prediction mode for each block that minimizes the difference between P and the block to be encoded.

1.1 4×4 Luma Prediction Mode

The samples above and to the left (labeled A-M in Fig. 1) have previously been encoded and reconstructed and are therefore available in the encoder and decoder to form a prediction reference. In Fig. 1, the samples a, b, c, … , p of the prediction block P are calculated based on the samples A–M as follows. Mode 2 (DC prediction) is modified depending on which samples A–M have previously been coded; each of the other modes may only be used if all of the required prediction samples are available. If samples E, F, G, H have not yet been decoded, the value of sample D is copied to these positions and they are marked as 'available' [2].

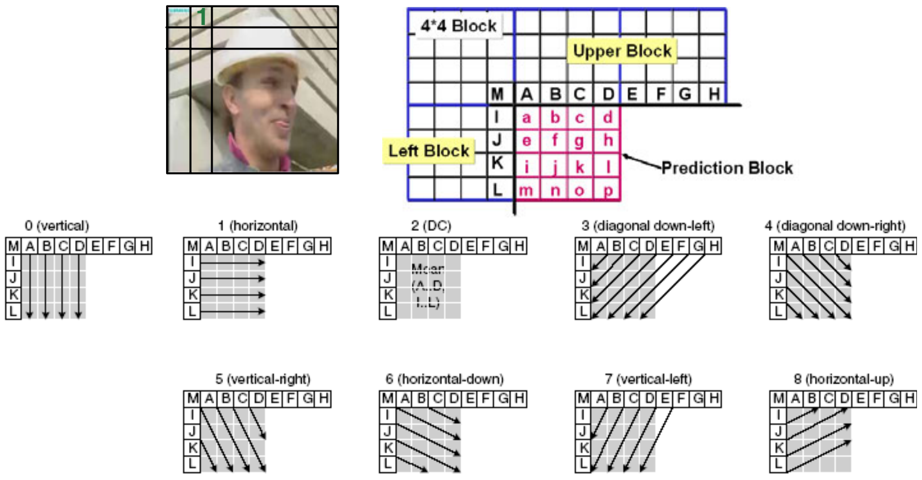


Fig. 1. Labelling of prediction samples (4×4) and nine 4×4 luma prediction modes

1.2 16×16 Luma Prediction Mode

As an alternative to the 4×4 luma modes described in the previous section, the entire 16×16 luma component of a macroblock may be predicted in one operation. Four modes are available, shown in Fig. 2.

1.3 8×8 Chroma Prediction Mode

Each 8×8 chroma component of an intra coded a macroblock is predicted from previously encoded chroma samples above and/or to the left and both chroma components always use the same prediction mode. The four prediction modes are very similar to the 16×16 luma prediction modes described in above section and illustrated in Fig. 2, except that the numbering of the modes is different. The modes are DC (mode 0), horizontal (mode 1), vertical (mode 2) and plane (mode 3).

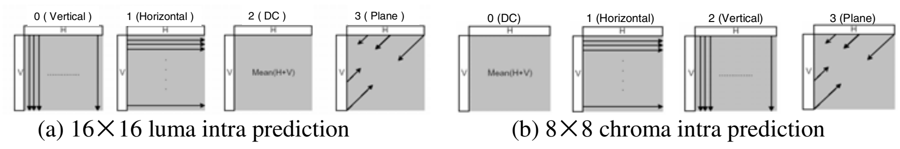


Fig. 2. 16╳16 luma prediction modes and 8╳8 chroma prediction modes

2 Intra Prediction Algorithm of Original JM 8.4

In JM 8.4 reference software, 4×4 intra prediction and 16×16 intra prediction are used together such as Fig. 3. In R-D optimization mode, 16×16 intra prediction mode decision is calculated before 4×4 intra prediction. Intra prediction modes having the minimum cost (SAE) among 4×4 intra prediction and 16×16 intra prediction are selected. If R-D optimization mode is not selected, 4×4 intra prediction mode decision is calculated before 16×16 intra prediction [1]. If one of the intra modes is selected, another computational cost of mode prediction such as DCT, quantization, inverse DCT and inverse quantization is meaningless. Because most of MBs are encoded in 4×4 intra prediction mode, the cost of 16×16 mode decision is dummy calculation. Therefore, we can reduce the dummy calculation by using the characteristics that 4×4 intra prediction mode is mostly selected in H.264 encoder and another MBs to be encoded in 16×16 intra mode mostly appear at adjacent blocks previously encoded in 16×16 intra mode.

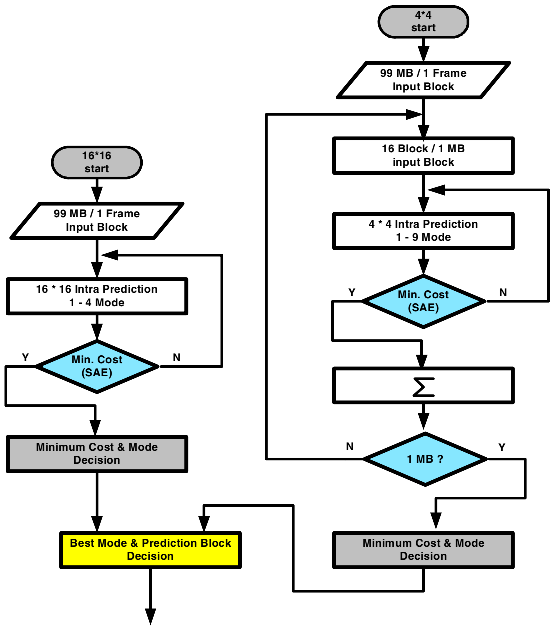
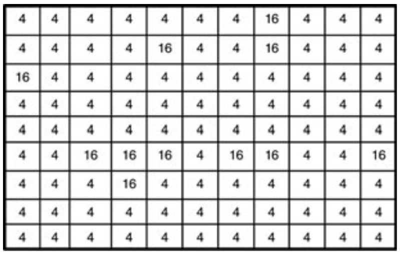

4*4 start
99 MB / 1 Frame Input Block
16 Block / 1 MB input Block
4 * 4 Intra Prediction 1 - 9 Mode
Min. Cost (SAE)
Y
N
Σ
1 MB ?
N
Y
Minimum Cost & Mode Decision
16*16 start
99 MB / 1 Frame Input Block
16 * 16 Intra Prediction 1 - 4 Mode
Min. Cost (SAE)
Y
N
Minimum Cost & Mode Decision
Best Mode & Prediction Block Decision

Fig. 3. Luma intra prediction algorithm of original H.264 reference software (JM 8.4)

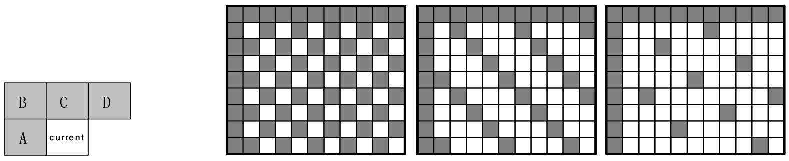
3 Proposed Scalable Fast Intra Prediction Algorithm

The main concept of the proposed algorithm is that JM 8.4 reference software has many dummy calculations so that we can reduce redundant calculations based upon the three characteristics. The first characteristic is 4×4 intra prediction mode is mostly selected for current MB's prediction mode in H.264 encoder. The second characteristic is the ratio of MBs between 4×4 intra mode and 16×16 intra mode is almost 10:1. The third characteristic is the 16×16 intra predicted MBs are adjacent to each other such as Fig. 4. If there is any one of MBs previously encoded in 16×16 intra prediction mode among four MBs adjacent to current MB such as Fig. 4 and Fig. 5(a), the probability of re-prediction into 16×16 intra prediction for current MB will become high. Fig. 5(a) shows that if one of the MB (A, B, C, D)'s prediction mode is 16×16 intra prediction mode, the current MB's mode is easy to be 16×16 intra prediction mode. In this case we should apply original intra prediction method of JM 8.4 using both 4×4 and 16×16 intra prediction. On the other hand, if there is none of MBs previously encoded in 16×16 intra prediction mode among four MBs adjacent to the current MB, we can force the intra prediction mode of the current MB into 4×4 intra mode, which will save the encoding time whereas the total encoded bits will increase only a little.



4	4	4	4	4	4	4	16	4	4	4
4	4	4	4	16	4	4	16	4	4	4
16	4	4	4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4	4	4	4
4	4	16	16	16	4	16	16	4	4	16
4	4	4	16	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4	4	4	4

Fig. 4. Example pattern of luma intra prediction mode encoded by JM 8.4



(a) MBs adjacent to current MB (b) Refresh factor = 2 Refresh factor = 4 Refresh factor = 8

Fig. 5. MBs adjacent to current MB and refresh pattern (gray color) in which current MB is encoded by original JM 8.4's prediction method using both 4×4 and 16×16 intra prediction

First of all, boundary MBs in one frame are the most important MB to predict the next MB's intra mode, because this is the staring point of the intra prediction algorithm in H.264. The errors of intra prediction at boundary MB of a frame can cause continuous intra prediction errors throughout the whole one frame, so these MBs are encoded by the original method using both 4×4 intra prediction and 16×16 intra prediction. The more predicting with proposed correlation on the current block and adjacent blocks, the more prediction errors will be occurred. In order to protect this propagation error we should apply another method that current MB is forcibly encoded by the original intra prediction method although there is none of MBs previously encoded in 16×16 intra prediction mode among four MBs adjacent to the current MB every interval such as in Fig. 5(b). This method is called refreshing and Fig. 5(b) shows several refresh patterns. A refreshing factor means an interval of a block encoded by the original JM 8.4's full cost prediction. The refresh factor is simply obtained by modulo (%) operation with img→current_mb_nr (MB's number) in JM 8.4. The larger the refresh factor, the more encoding times will be reduced, however total encoded bits increase only a little. This result is shown in section 4.

We can use these characteristics to design neural networks for fast mode decision and apply neural networks in order to decide whether the current MB should be encoded in only 4×4 intra mode or in original JM 8.4's method using both 4×4 and 16×16 intra prediction. Neural networks control how strongly current MB should be refreshed and test how much prediction errors were produced to modulate neural network operations. An early use of a recurrent network can be found in the work of Anderson et al. [3], [4]. They used a fully connected neural network called brain state in a box (BSB) to model psychological effects observed in probability learning. In this network each unit, which has no self-connection, is fully connected to every other units in the networks [5].

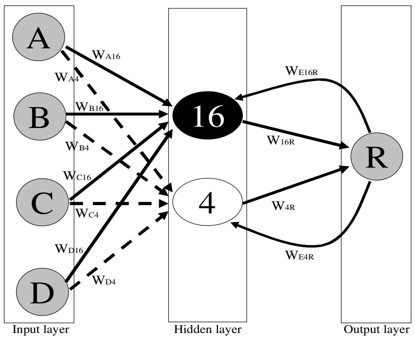


Fig. 6. Proposed neural networks to optimize intra prediction mode decision

In Fig. 6, A, B, C, D are the input units that have '-1' or '1' value and this four nodes are appointed in Fig. 5(a). If A is encoded by 16×16 intra prediction mode, A will be set to '1' or if A is encoded by 4×4 intra prediction mode, A will be reset to '-1'. In case of each B, C and D, the condition applies. If there is any one of an MB encoded by 16×16 intra prediction mode among four MBs, the probability of re-prediction into 16×16 intra prediction for the current MB will become high. We can apply neural networks in order to decide whether to force into 4×4 intra prediction mode or to use original JM 8.4's method. Net_{16} and Net_4 is hidden layer's node controlling the strength of prediction and correcting error propagation. Equation (1) and (2) shows the operation of the proposed neural network. Each weight factor is variable in proportion to the probability of re-prediction into 16×16 intra prediction. The weights of the prior left and up block are the largest because of the probability of re-prediction into 16×16 intra prediction.

$$\begin{aligned} F(\text{total net}) &= W_{16R}\times Net_{16} + W_{4R}\times Net_{4} + W_{E16R} + W_{E4R} \\ &= W_{A16}\times A + W_{B16}\times B + W_{C16}\times C + W_{D16}\times D + W_{A4}\times A + W_{B4}\times B \\ &\quad + W_{C4}\times C + W_{D4}\times D + W_{E16R} + W_{E4R} \end{aligned} \tag{1}$$

$W_{A16}= W_{C16}=1, W_{B16}= W_{D16}=0.5, W_{A4}= W_{C4}=0.5, W_{B4}= W_{D4}=0.25, W_{16R}= W_{4R}=1$

If an error is occurred, $W_{E16R} = -1,\ W_{E4R} = -0.5$, else $W_{E16R} = 1,\ W_{E4R} = 0.5$

$$\text{If } F(\text{total net}) > -4 \text{ output } R=1 \text{ else output } R=0 \tag{2}$$

In equation (2) the output R has ‘1’ or ‘0’ value. When R is ‘1’, it means that the probability of re-prediction into 16× 16 intra prediction for current MB becomes high, we should calculate exact mode decision using both 4× 4 and 16× 16 intra prediction even if it takes more operation time and cost. When R is ‘0’, it means that the probability of 4× 4 intra prediction for current MB becomes high, we only force into 4× 4 intra prediction mode for the current MB’s best prediction mode without 16× 16 intra prediction and without the cost of comparison between 4× 4 intra prediction and 16× 16 intra prediction. Thus we can reduce redundant calculation time to select the best mode. When output R is ‘1’, if the prediction result is 4× 4 intra prediction, an error is occurred, so we use these errors for the back-propagation factor. We can optimize accuracy of neural networks with this factor such as Table 1. Table 1 is a LUT that describes the mechanism of proposed neural networks according to input unit pattern, its output and back-propagation adapted by network’s error. We can notice that the more errors (output R=1 but real prediction result is 4× 4 intra mode), the more back-propagations (output R=0, gray color) will be occurred, which means that the number of MB forcibly encoded in 4× 4 intra mode becomes large.

Table 1. Look-up table that describes the mechanism of proposed neural networks

Input Units				For 1st Input	For 2nd Input		For 3rd Input			
A	B	C	D	Output R	No error at 1st	Error at 1st	No error at both 1st & 2nd	No Error at 1st & Error at 2nd	No Error at 2nd & Error at 1st	Error at both 1st & 2nd
					Output R		Output R			
1	1	1	1	1	1	1	1	1	1	1
1	1	1	-1	1	1	1	1	1	1	1
1	1	-1	1	1	1	1	1	1	1	1
1	1	-1	-1	1	1	1	1	1	1	1
1	-1	1	1	1	1	1	1	1	1	1
1	-1	1	-1	1	1	1	1	1	1	1
1	-1	-1	1	1	1	1	1	1	1	1
1	-1	-1	-1	1	1	1	1	1	1	0
-1	1	1	1	1	1	1	1	1	1	1
-1	1	1	-1	1	1	1	1	1	1	1
-1	1	-1	1	1	1	1	1	1	1	0
-1	1	-1	-1	1	1	0	1	0	0	0
-1	-1	1	1	1	1	1	1	1	1	1
-1	-1	1	-1	1	1	1	1	1	1	0
-1	-1	-1	1	1	1	0	1	0	0	0
-1	-1	-1	-1	0	0	0	0	0	0	0

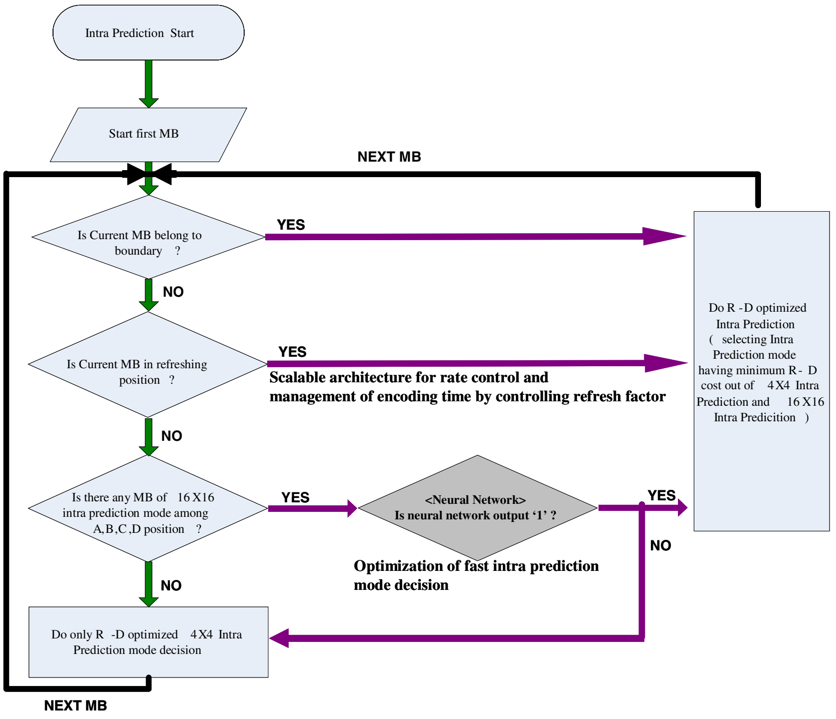
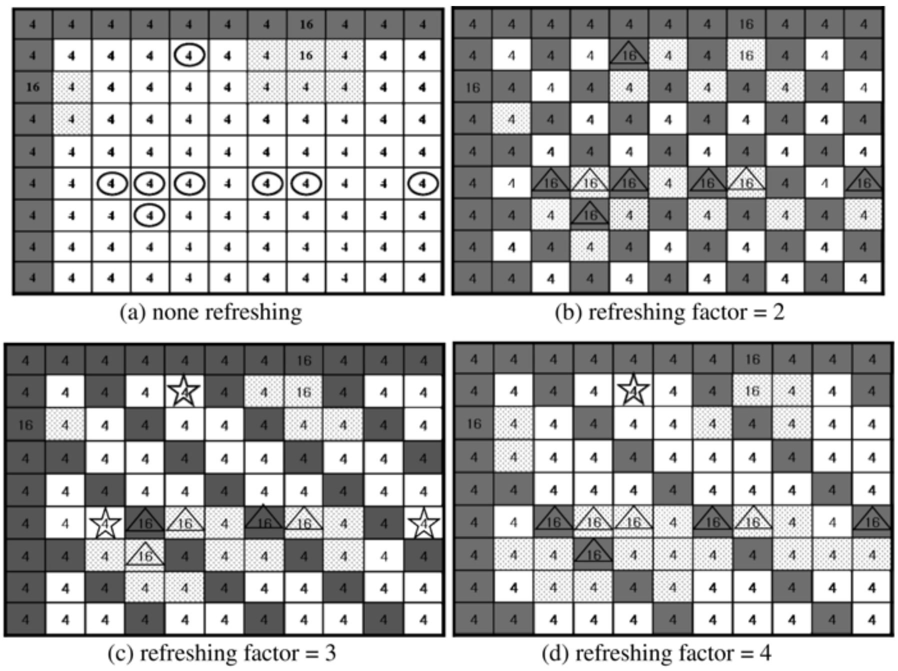

Intra Prediction Start
Start first MB
NEXT MB
Is Current MB belong to boundary ?
YES
NO
Is Current MB in refreshing position ?
YES
Scalable architecture for rate control and management of encoding time by controlling refresh factor
NO
Is there any MB of 16X16 intra prediction mode among A,B,C,D position ?
YES
<Neural Network> Is neural network output '1' ?
YES
NO
Optimization of fast intra prediction mode decision
NO
Do only R -D optimized 4X4 Intra Prediction mode decision
NEXT MB
Do R -D optimized Intra Prediction (selecting Intra Prediction mode having minimum R- D cost out of 4X4 Intra Prediction and 16X16 Intra Predicition)

Fig. 7. Proposed Scalable fast intra prediction algorithm using neural network

The proposed scalable intra prediction algorithm using neural network is described in Fig. 7. If the current MB belongs to boundary, this MB is encoded by R-D optimized intra prediction using both 4×4 and 16×16 intra prediction in JM 8.4. Then, if current MB is in the refreshing position, this MB is encoded as above method. In this refreshing process we can control bit rate and encoding time by controlling refresh factor. This result is shown in section 4. Then, if there is any one of MBs encoded by 16×16 intra prediction mode among A, B, C, D position in Fig. 5(a). This current MB should be tested what is the most probable intra prediction mode by neural network. When neural network output R is '1', it means that the probability of re-prediction into 16×16 intra prediction for current MB becomes high that we should calculate the exact mode decision using 4×4 and 16×16 intra. When neural network output R is '0', it means that probability of 4×4 intra prediction for current MB becomes high, we only force into 4×4 intra prediction mode for the current MB's best prediction mode without 16×16 intra prediction and comparison between 4×4 and 16×16 intra prediction. Fig. 8 shows the mechanism of refreshing method and neural network in order to select best intra mode. Fig. 8(a) has prediction error marked with 8 circles because it does not have a refreshing position. But Fig. 8(b) has no prediction error because it has some refreshing positions and it uses neural network algorithm to optimize the mode decision. Fig. 8(c) and 8(d) show the comparison of refreshing factor 3 and 4. In this case, we will notice that the simple decrease of a refreshing is not good to correct prediction.


(a) none refreshing
(b) refreshing factor = 2
(c) refreshing factor = 3
(d) refreshing factor = 4

■ : MB which is at boundary and refreshing position
□ : MB which is just forced into 4×4 intra prediction without 16×16 intra prediction
▨ : MB which is adjacent to 16×16 intra MB and has '1' value of neural network output R
○ : Prediction error compared with Fig. 4, ☆: Uncorrected Prediction error
△ : Corrected MB compared with Fig. 8(a) by proposed algorithm

Fig. 8. Mechanism of refreshing method and neural networks in order to select best intra mode

4 Experiment and Verification

Experiment is based on H.264's reference encoder/decoder software JM 8.4. We experiment 30 I pictures for each test video sequence. Quantization parameter of 28 rate-distortion optimization is applied for each test video sequence to satisfy H.264 baseline profile. We tested trevor, foreman, salesman and suzie pictures having QCIF (176×144) size with color format. Table 2 shows the experimental result for each color video sequence. Input picture's frame rate is 30Hz and entropy coding method is CAVLC. Each source is tested into 8 cases and the performance parameter is total encoding time, total encoded bit and SNR ratio of Y, U, V components. In the first case (JM 8.4, 4×4 and 16×16 intra prediction), it is tested by original JM 8.4 encoder without any changes. Its total encoded bit is smallest, but its total encoding time is the largest among 8 cases. The second case is tested by only 4×4 intra prediction. The third case is tested by only 16×16 intra prediction. The total encoding time of 16×16 intra prediction is the smallest but incrementation of the total encoded bit is the largest in this case.

Table 2. Performance of proposed algorithm

Tested Video (color, cif, 30frames@1Hz)	Performance		Original (JM 8.4)			Proposed				
			4×4 & 16×16	only 4×4	only 16×16	Refresh Factor				
						2	4	8	16	none
Trevor	Total Encoding Time (sec)		20.630	18.68	5.48	15.500	12.945	10.563	9.353	8.501
	Total Encoded Bit		842088	860190	1026000	848456	852576	856840	858024	858704
	SNR (dB)	Y	37.31	37.32	37.26	37.31	37.31	37.31	37.32	37.31
		U	39.81	39.81	39.76	39.83	39.82	39.79	39.79	39.79
		V	39.34	39.35	39.31	39.34	39.31	39.33	39.33	39.31
Foreman	Total Encoding Time		20.24	18.94	6.12	14.327	10.954	9.110	8.436	7.456
	Total Encoded Bit		789936	801256	101242	791864	796304	796512	796752	797776
	SNR	Y	36.77	36.78	36.65	36.77	36.77	36.77	36.77	36.77
		U	39.50	39.51	39.46	39.48	39.49	39.49	39.49	39.48
		V	41.46	41.47	41.22	41.44	41.41	41.38	41.39	41.41
Salesman	Total Encoding Time		21.537	19.53	7.24	15.254	11.486	9.406	8.435	7.767
	Total Encoded Bit		861080	961240	110542	865368	868408	870136	870816	871168
	SNR	Y	36.34	36.34	36.23	36.34	36.34	36.34	36.34	36.34
		U	39.76	39.76	39.69	39.77	39.76	39.80	39.77	39.76
		V	40.39	40.39	40.36	40.39	40.38	40.37	40.35	40.36
Suzie	Total Encoding Time		17.734	15.652	4.38	14.412	11.735	11.173	9.791	8.983
	Total Encoded Bit		512808	602142	724012	515824	519528	520728	521352	521712
	SNR	Y	37.80	37.81	37.62	37.81	37.80	37.80	37.80	37.80
		U	43.46	43.47	43.24	43.45	43.45	43.42	43.40	43.36
		V	43.24	43.24	42.98	43.20	43.16	43.15	43.16	43.14

Table 3. Performance improvement of proposed algorithm

Tested Video	Refresh Factor	Total Decreased Encoding Time (%)	Total Increased Encoded Bit (%)	SNR(dB)		
				Y	U	V
Trevor	2	24.87	0.76	0	0.05	0
	4	37.25	1.25	0	0.03	-0.08
	8	48.80	1.75	0	-0.05	-0.03
	16	54.66	1.89	0.03	-0.05	-0.03
	none	58.79	1.97	0	-0.05	-0.08
Foreman	2	29.21	0.24	0	-0.06	-0.05
	4	45.88	0.08	0	-0.03	0.01
	8	54.99	0.83	0	-0.03	-0.2
	16	58.32	0.86	0	-0.03	-0.17
	none	63.16	0.99	0	-0.05	-0.12
Salesman	2	29.17	0.498	0	0.025	0
	4	46.67	0.85	0	0	-0.03
	8	56.33	1.05	0	0.1	-0.05
	16	60.83	1.13	0	0.03	-0.1
	none	63.94	1.17	0	0	-0.07
Suzie	2	18.73	0.588	0.03	-0.02	-0.1
	4	33.83	1.31	0	-0.02	-0.19
	8	37.00	1.54	0	-0.01	-0.2
	16	44.79	1.67	0	-0.14	-0.19
	none	49.35	1.74	0	-0.23	-0.23

Last 5 cases are tested by the proposed method and each case is different for the refresh factor to control the bit rate and the total encoding time. Refresh factor 2 has the smallest refreshing interval for delicate intra prediction to reduce the total encoded bits, but its encoding time is not better than for the larger refresh factor. The latest case (none refresh factor) dose not use refreshing method. Table 3 shows that the total encoding time is reduced by 63% than the original JM 8.4 method. Nevertheless, its total encoding bit increments is only 0.99 % than JM 8.4. By proposed algorithm, total average encoding time of 30 frames are reduced by 20% ~ 65% compared with the H.264 reference software (JM 8.4) by using neural networks and modulating scalable refreshing factor. On the other hand total encoding bits are increased by 0.8% ~ 2% at the cost of reduced SNR of 0.01 dB.

5 Conclusion

In this paper, we discussed a scalable intra prediction algorithm in H.264 by using neural networks to reduce redundant calculation time by selecting the best mode of intra prediction. Our proposed method reduce the total encoding time of I frame by 20% ~ 65% compared with H.264 reference software (JM 8.4) and it is possible to control bit rate and reduce encoding time by modulation the of refresh factor. Whereas total encoded bits are only increased by 0.8% ~ 2% compared with H.264 reference software by using neural networks and by modulating scalable refreshing factor. As a result we expect to contribute to fast real time implementation of H.264 encoder.

Acknowledgements. This work is supported by ITRC and IT-SoC.

References

1. P. Topiwala, G Sullivan, A. Joch and F. Kossentini, "Overview and Performance Evaluation of the Draft ITU-T H.26L Video Coding Standard." Proc. SPIE, Appl. Dig. Im. Proc, Aug. 2001.
2. Iain E.G. Richardson "H.264 and MPEG-4 VIDEO COMPRESSION, Video Coding for Next-generation Multimedia", WILEY, 2003.
3. A. Anderson et al., “Distinctive features, categorical perception, and probability learning: Some applications of a neural model,” Psych. Rev. vol. 84, pp. 413–451, 1977.
4. J. F. Kolen, “Exploring the computational capabilities of recurrent neural networks,” Ph.D. dissertation, Ohio State Univ., Columbus, 1994.
5. Seong-Whan Lee and Hee-Heon Song, “A New Recurrent Neural-Network Architecture for Visual Pattern Recognition”, IEEE TRANSACTIONS ON NEURAL NETWORKS, VOL. 8, NO. 2, MARCH 1997.

Author Index