

Derong Liu Shumin Fei
Zengguang Hou Huaguang Zhang
Changyin Sun (Eds.)

LNCS 4493

Advances in Neural Networks – ISNN 2007

4th International Symposium on Neural Networks, ISNN 2007
Nanjing, China, June 2007
Proceedings, Part III

3
Part III

 Springer

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

University of Dortmund, Germany

Madhu Sudan

Massachusetts Institute of Technology, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Moshe Y. Vardi

Rice University, Houston, TX, USA

Gerhard Weikum

Max-Planck Institute of Computer Science, Saarbruecken, Germany

Derong Liu Shumin Fei
Zengguang Hou Huaguang Zhang
Changyin Sun (Eds.)

Advances in Neural Networks – ISNN 2007

4th International Symposium
on Neural Networks, ISNN 2007
Nanjing, China, June 3-7, 2007
Proceedings, Part III

Volume Editors

Derong Liu

University of Illinois at Chicago, IL 60607-7053, USA

E-mail: dliu@ece.uic.edu

Shumin Fei

Southeast University, School of Automation, Nanjing 210096, China

E-mail: smfei@seu.edu.cn

Zengguang Hou

The Chinese Academy of Sciences, Institute of Automation, Beijing 100080, China

E-mail: zengguang.hou@ia.ac.cn

Huaguang Zhang

Northeastern University, Shenyang 110004, China

E-mail: zhanghuaguang@ise.neu.edu.cn

Changyin Sun

Hohai University, School of Electrical Engineering, Nanjing 210098, China

E-mail: cysun@hhu.edu.cn

Library of Congress Control Number: 2007926816

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

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

ISSN 0302-9743

ISBN-10 3-540-72394-3 Springer Berlin Heidelberg New York

ISBN-13 978-3-540-72394-3 Springer Berlin Heidelberg New York

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

springer.com

© Springer-Verlag Berlin Heidelberg 2007

Printed in Germany

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

Printed on acid-free paper SPIN: 12061044 06/3180 5 4 3 2 1 0

Preface

ISNN 2007 – the Fourth International Symposium on Neural Networks—was held in Nanjing, China, as a sequel of ISNN 2004/ISNN 2005/ISNN 2006. ISNN has now become a well-established conference series on neural networks in the region and around the world, with growing popularity and increasing quality. Nanjing is an old capital of China, a modern metropolis with a 2470-year history and rich cultural heritage. All participants of ISNN 2007 had a technically rewarding experience as well as memorable experiences in this great city.

A neural network is an information processing structure inspired by biological nervous systems, such as the brain. It consists of a large number of highly interconnected processing elements, called neurons. It has the capability of learning from example. The field of neural networks has evolved rapidly in recent years. It has become a fusion of a number of research areas in engineering, computer science, mathematics, artificial intelligence, operations research, systems theory, biology, and neuroscience. Neural networks have been widely applied for control, optimization, pattern recognition, image processing, signal processing, etc.

ISNN 2007 aimed to provide a high-level international forum for scientists, engineers, and educators to present the state of the art of neural network research and applications in diverse fields. The symposium featured plenary lectures given by worldwide renowned scholars, regular sessions with broad coverage, and some special sessions focusing on popular topics.

The symposium received a total of 1975 submissions from 55 countries and regions across all six continents. The symposium proceedings consists of 454 papers among which 262 were accepted as long papers and 192 were accepted as short papers. We would like to express our sincere gratitude to all reviewers of ISNN 2007 for the time and effort they generously gave to the symposium. We are very grateful to the National Natural Science Foundation of China, K. C. Wong Education Foundation of Hong Kong, the Southeast University of China, the Chinese University of Hong Kong, and the University of Illinois at Chicago for their financial support. We would also like to thank the publisher, Springer, for cooperation in publishing the proceedings in the prestigious series of

Derong Liu
Shumin Fei
Zeng-Guang Hou
Huaguang Zhang
Changyin Sun

ISNN 2007 Organization

General Chair

Derong Liu, University of Illinois at Chicago, USA, and Yanshan University, China

General Co-chair

Marios M. Polycarpou, University of Cyprus

Organization Chair

Shumin Fei, Southeast University, China

Advisory Committee Chairs

Shun-Ichi Amari, RIKEN Brain Science Institute, Japan

Chunbo Feng, Southeast University, China

Zhenya He, Southeast University, China

Advisory Committee Members

Hojjat Adeli, Ohio State University, USA

Moonis Ali, Texas State University-San Marcos, USA

Zheng Bao, Xidian University, China

Tamer Basar, University of Illinois at Urbana-Champaign, USA

Tianyou Chai, Northeastern University, China

Guoliang Chen, University of Science and Technology of China, China

Ruwei Dai, Chinese Academy of Sciences, China

Dominique M. Durand, Case Western Reserve University, USA

Russ Eberhart, Indiana University Purdue University Indianapolis, USA

David Fogel, Natural Selection, Inc., USA

Walter J. Freeman, University of California-Berkeley, USA

Toshio Fukuda, Nagoya University, Japan

Kunihiko Fukushima, Kansai University, Japan

Tom Heskes, University of Nijmegen, The Netherlands

Okyay Kaynak, Bogazici University, Turkey

Frank L. Lewis, University of Texas at Arlington, USA

Deyi Li, National Natural Science Foundation of China, China

Yanda Li, Tsinghua University, China

Ruqian Lu, Chinese Academy of Sciences, China

John MacIntyre, University of Sunderland, UK
Robert J. Marks II, Baylor University, USA
Anthony N. Michel, University of Notre Dame, USA
Evangelia Micheli-Tzanakou, Rutgers University, USA
Erkki Oja, Helsinki University of Technology, Finland
Nikhil R. Pal, Indian Statistical Institute, India
Vincenzo Piuri, University of Milan, Italy
Jennie Si, Arizona State University, USA
Youxian Sun, Zhejiang University, China
Yuan Yan Tang, Hong Kong Baptist University, China
Tzyh Jong Tarn, Washington University, USA
Fei-Yue Wang, Chinese Academy of Sciences, China
Lipo Wang, Nanyang Technological University, Singapore
Shoujue Wang, Chinese Academy of Sciences
Paul J. Werbos, National Science Foundation, USA
Bernie Widrow, Stanford University, USA
Gregory A. Worrell, Mayo Clinic, USA
Hongxin Wu, Chinese Academy of Space Technology, China
Youlun Xiong, Huazhong University of Science and Technology, China
Lei Xu, Chinese University of Hong Kong, China
Shuzi Yang, Huazhong University of Science and Technology, China
Xin Yao, University of Birmingham, UK
Bo Zhang, Tsinghua University, China
Siyong Zhang, Qingdao University, China
Nanning Zheng, Xi'an Jiaotong University, China
Jacek M. Zurada, University of Louisville, USA

Steering Committee Chair

Jun Wang, Chinese University of Hong Kong, China

Steering Committee Co-chair

Zongben Xu, Xi'an Jiaotong University, China

Steering Committee Members

Tianping Chen, Fudan University, China
Andrzej Cichocki, Brain Science Institute, Japan
Wlodzislaw Duch, Nicolaus Copernicus University, Poland
Chengan Guo, Dalian University of Technology, China
Anthony Kuh, University of Hawaii, USA
Xiaofeng Liao, Chongqing University, China
Xiaoxin Liao, Huazhong University of Science and Technology, China
Bao-Liang Lu, Shanghai Jiaotong University, China

Chenghong Wang, National Natural Science Foundation of China, China
Leszek Rutkowski, Technical University of Czestochowa, Poland
Zengqi Sun, Tsinghua University, China
Donald C. Wunsch II, University of Missouri-Rolla, USA
Gary G. Yen, Oklahoma State University, Stillwater, USA
Zhang Yi, University of Electronic Science and Technology, China
Hujun Yin, University of Manchester, UK
Liming Zhang, Fudan University, China
Chunguang Zhou, Jilin University, China

Program Chairs

Zeng-Guang Hou, Chinese Academy of Sciences, China
Huaguang Zhang, Northeastern University, China

Special Sessions Chairs

Lei Guo, Beihang University, China
Wen Yu, CINVESTAV-IPN, Mexico

Finance Chair

Xinping Guan, Yanshan University, China

Publicity Chair

Changyin Sun, Hohai University, China

Publicity Co-chairs

Zongli Lin, University of Virginia, USA
Weixing Zheng, University of Western Sydney, Australia

Publications Chair

Jinde Cao, Southeast University, China

Registration Chairs

Hua Liang, Hohai University, China
Bhaskhar DasGupta, University of Illinois at Chicago, USA

Local Arrangements Chairs

Enrong Wang, Nanjing Normal University, China
Shengyuan Xu, Nanjing University of Science and Technology, China
Junyong Zhai, Southeast University, China

Electronic Review Chair

Xiaofeng Liao, Chongqing University, China

Symposium Secretariats

Ting Huang, University of Illinois at Chicago, USA
Jinya Song, Hohai University, China

ISSN 2007 International Program Committee

Shigeo Abe, Kobe University, Japan
Ajith Abraham, Chung Ang University, Korea
Khurshid Ahmad, University of Surrey, UK
Angelo Alessandri, University of Genoa, Italy
Sabri Arik, Istanbul University, Turkey
K. Vijayan Asari, Old Dominion University, USA
Amit Bhaya, Federal University of Rio de Janeiro, Brazil
Abdesselam Bouzerdoum, University of Wollongong, Australia
Martin Brown, University of Manchester, UK
Ivo Bukovsky, Czech Technical University, Czech Republic
Jinde Cao, Southeast University, China
Matthew Casey, Surrey University, UK
Luonan Chen, Osaka-Sandai University, Japan
Songcan Chen, Nanjing University of Aeronautics and Astronautics, China
Xiao-Hu Chen, Nanjing Institute of Technology, China
Xinkai Chen, Shibaura Institute of Technology, Japan
Yuehui Chen, Jinan University, Shandong, China
Xiaochun Cheng, University of Reading, UK
Zheru Chi, Hong Kong Polytechnic University, China
Sungzoon Cho, Seoul National University, Korea
Seungjin Choi, Pohang University of Science and Technology, Korea
Tommy W. S. Chow, City University of Hong Kong, China
Emilio Corchado, University of Burgos, Spain
Jose Alfredo F. Costa, Federal University, UFRN, Brazil
Mingcong Deng, Okayama University, Japan
Shuxue Ding, University of Aizu, Japan
Meng Joo Er, Nanyang Technological University, Singapore
Deniz Erdogmus, Oregon Health & Science University, USA

Gary Feng, City University of Hong Kong, China
Jian Feng, Northeastern University, China
Mauro Forti, University of Siena, Italy
Wai Keung Fung, University of Manitoba, Canada
Marcus Gallagher, University of Queensland, Australia
John Qiang Gan, University of Essex, UK
Xiqi Gao, Southeast University, China
Chengan Guo, Dalian University of Technology, China
Dalei Guo, Chinese Academy of Sciences, China
Ping Guo, Beijing Normal University, China
Madan M. Gupta, University of Saskatchewan, Canada
Min Han, Dalian University of Technology, China
Haibo He, Stevens Institute of Technology, USA
Daniel Ho, City University of Hong Kong, China
Dewen Hu, National University of Defense Technology, China
Jinglu Hu, Waseda University, Japan
Sanqing Hu, Mayo Clinic, Rochester, Minnesota, USA
Xuelei Hu, Nanjing University of Science and Technology, China
Guang-Bin Huang, Nanyang Technological University, Singapore
Tingwen Huang, Texas A&M University at Qatar
Giacomo Indiveri, ETH Zurich, Switzerland
Malik Magdon Ismail, Rensselaer Polytechnic Institute, USA
Danchi Jiang, University of Tasmania, Australia
Joarder Kamruzzaman, Monash University, Australia
Samuel Kaski, Helsinki University of Technology, Finland
Hon Keung Kwan, University of Windsor, Canada
James Kwok, Hong Kong University of Science and Technology, China
James Lam, University of Hong Kong, China
Kang Li, Queen's University, UK
Xiaoli Li, University of Birmingham, UK
Yangmin Li, University of Macau, China
Yongwei Li, Hebei University of Science and Technology, China
Yuanqing Li, Institute of Infocomm Research, Singapore
Hualou Liang, University of Texas at Houston, USA
Jinling Liang, Southeast University, China
Yanchun Liang, Jilin University, China
Lizhi Liao, Hong Kong Baptist University, China
Guoping Liu, University of Glamorgan, UK
Ju Liu, Shandong University, China
Meiqin Liu, Zhejiang University, China
Xiangjie Liu, North China Electric Power University, China
Yutian Liu, Shandong University, China
Hongtao Lu, Shanghai Jiaotong University, China
Jinhu Lu, Chinese Academy of Sciences and Princeton University, USA
Wenlian Lu, Max Planck Institute for Mathematics in Sciences, Germany

Shuxian Lun, Bohai University, China
Fa-Long Luo, Anyka, Inc., USA
Jinwen Ma, Peking University, China
Xiangping Meng, Changchun Institute of Technology, China
Kevin L. Moore, Colorado School of Mines, USA
Ikuko Nishikawa, Ritsumeikan University, Japan
Stanislaw Osowski, Warsaw University of Technology, Poland
Seiichi Ozawa, Kobe University, Japan
Hector D. Patino, Universidad Nacional de San Juan, Argentina
Yi Shen, Huazhong University of Science and Technology, China
Daming Shi, Nanyang Technological University, Singapore
Yang Shi, University of Saskatchewan, Canada
Michael Small, Hong Kong Polytechnic University
Ashu MG Solo, Maverick Technologies America Inc., USA
Stefano Squartini, Universita Politecnica delle Marche, Italy
Ponnuthurai Nagaratnam Suganthan, Nanyang Technological University,
Singapore
Fuchun Sun, Tsinghua University, China
Johan A. K. Suykens, Katholieke Universiteit Leuven, Belgium
Norikazu Takahashi, Kyushu University, Japan
Ying Tan, Peking University, China
Yonghong Tan, Guilin University of Electronic Technology, China
Peter Tino, Birmingham University, UK
Christos Tjortjis, University of Manchester, UK
Antonios Tsovdos, Cranfield University, UK
Marc van Hulle, Katholieke Universiteit Leuven, Belgium
Dan Ventura, Brigham Young University, USA
Michel Verleysen, Universite Catholique de Louvain, Belgium
Bing Wang, University of Hull, UK
Dan Wang, Dalian Maritime University, China
Pei-Fang Wang, SPAWAR Systems Center-San Diego, USA
Zhiliang Wang, Northeastern University, China
Si Wu, University of Sussex, UK
Wei Wu, Dalian University of Technology, China
Shunren Xia, Zhejiang University, China
Yousheng Xia, University of Waterloo, Canada
Cheng Xiang, National University of Singapore, Singapore
Daoyi Xu, Sichuan University, China
Xiaosong Yang, Huazhong University of Science and Technology, China
Yingjie Yang, De Montfort University, UK
Zi-Jiang Yang, Kyushu University, Japan
Mao Ye, University of Electronic Science and Technology of China, China
Jianqiang Yi, Chinese Academy of Sciences, China
Dingli Yu, Liverpool John Moores University, UK
Zhigang Zeng, Wuhan University of Technology, China

Guisheng Zhai, Osaka Prefecture University, Japan
Jie Zhang, University of Newcastle, UK
Liming Zhang, Fudan University, China
Liqing Zhang, Shanghai Jiaotong University, China
Nian Zhang, South Dakota School of Mines & Technology, USA
Qingfu Zhang, University of Essex, UK
Yanqing Zhang, Georgia State University, USA
Yifeng Zhang, Hefei Institute of Electrical Engineering, China
Yong Zhang, Jinan University, China
Dongbin Zhao, Chinese Academy of Sciences, China
Hongyong Zhao, Nanjiang University of Aeronautics and Astronautics, China
Haibin Zhu, Nipissing University, Canada

Table of Contents – Part III

Feedforward Neural Networks

Using Three Layer Neural Network to Compute Multi-valued Functions	1
A Novel Global Hybrid Algorithm for Feedforward Neural Networks	9
Study on Relationship Between NIHSS and TCM-SSASD Based on the BP Neural Network Multiple Models Method	17
Application of Back-Propagation Neural Network to Power Transformer Insulation Diagnosis	26
Momentum BP Neural Networks in Structural Damage Detection Based on Static Displacements and Natural Frequencies.....	35
Deformation Measurement of the Large Flexible Surface by Improved RBFNN Algorithm and BPNN Algorithm	41
Evaluation of the Growth of Real Estate Financial System Based on BP Neural Network	49
GA-Based Neural Network to Identification of Nonlinear Structural Systems	57
Approximation Capability Analysis of Parallel Process Neural Network with Application to Aircraft Engine Health Condition Monitoring	66
Neural Network-Based Position Sensorless Control for Transverse Flux Linear SRM	73
Tourism Room Occupancy Rate Prediction Based on Neural Network...	80

Recurrent Neural Networks

Recurrent Neural Networks on Duty of Anomaly Detection in Databases	85
Solving Variational Inequality Problems with Linear Constraints Based on a Novel Recurrent Neural Network	95
Equalization of 8PSK Signals with a Recurrent Neural Network	105
Short-Term Load Forecasting Using BiLinear Recurrent Neural Network	111
Convergence of Gradient Descent Algorithm for a Recurrent Neuron	117
Periodicity of Recurrent Neural Networks with Reaction-Diffusion and Dirichlet Boundary Conditions	123
New Critical Analysis on Global Convergence of Recurrent Neural Networks with Projection Mappings	131
A Study on Digital Media Security by Hopfield Neural Network	140
Two Theorems on the Robust Designs for Pattern Matching CNNs	147
Iterative Learning Control Analysis Based on Hopfield Neural Networks	157
Stochastic Stabilization of Delayed Neural Networks	164

Neural Networks for Optimization

An Evolutionary Multi-objective Neural Network Optimizer with Bias-Based Pruning Heuristic	174
A Hybrid of Particle Swarm Optimization and Hopfield Networks for Bipartite Subgraph Problems	184

Convergence of a Recurrent Neural Network for Nonconvex Optimization Based on an Augmented Lagrangian Function	194
Multi-objective Topology Optimization of Structures Using NN-OC Algorithms	204
A Hybrid Particle Swarm Algorithm for the Structure and Parameters Optimization of Feed-Forward Neural Network	213
Designing Neural Networks Using PSO-Based Memetic Algorithm	219
Simultaneous Optimization of ANFIS-Based Fuzzy Model Driven to Data Granulation and Parallel Genetic Algorithms	225
A Neural Network Based Optimization Method for a Kind of QPPs and Application	231
Multilayer Perceptron Networks Training Using Particle Swarm Optimization with Minimum Velocity Constraints	237
Characterization and Optimization of the Contact Formation for High-Performance Silicon Solar Cells	246
Optimizing Process Parameters for Ceramic Tile Manufacturing Using an Evolutionary Approach	252
Application of RBF Neural Network to Simplify the Potential Based Optimization	261
Satisficing Approximation Response Model Based on Neural Network in Multidisciplinary Collaborative Optimization	267
A New BP Network Based on Improved PSO Algorithm and Its Application on Fault Diagnosis of Gas Turbine	277

The Evaluation of BP-ISP Strategy Alignment Degree with PSO-Based ANN	284
Improved Results on Solving Quadratic Programming Problems with Delayed Neural Network	292
A Modified Hopfield Network for Nonlinear Programming Problem Solving	302
A Novel Artificial Neural Network Based on Hybrid PSO-BP Algorithm in the Application of Adaptive PMD Compensation System	311
Stabilizing Lagrange-Type Nonlinear Programming Neural Networks ...	320
 Support Vector Machines	
Support Vector Machines and Genetic Algorithms for Soft-Sensing Modeling	330
Incremental Nonlinear Proximal Support Vector Machine	336
Machinery Fault Diagnosis Using Least Squares Support Vector Machine	342
Support Vector Machine with Composite Kernels for Time Series Prediction	350
Neuromorphic Quantum-Based Adaptive Support Vector Regression for Tuning BWGC/NGARCH Forecast Model	357
Modulation Classification of Analog and Digital Signals Using Neural Network and Support Vector Machine	368
A Facial Expression Recognition Approach Based on Novel Support Vector Machine Tree	374

Universal Steganalysis Using Multiwavelet Higher-Order Statistics and Support Vector Machines	382
Tree-Structured Support Vector Machines for Multi-class Classification	392
Identification of MIMO Hammerstein Models Using Support Vector Machine	399
Extraction of Filled-In Items from Chinese Bank Check Using Support Vector Machines	407
Online Least Squares Support Vector Machines Based on Wavelet and Its Applications	416
Ultrasound Estimation of Fetal Weight with Fuzzy Support Vector Regression	426
Hybrid Support Vector Machine and General Model Approach for Audio Classification	434
An Improved SVM Based on 1-Norm for Selection of Personal Credit Scoring Index System	441
Combining Weighted SVMs and Spectrum-Based NN for Multi-classification	448
Rotation Invariant Texture Classification Algorithm Based on DT-CWT and SVM	454
A Two-Pass Classification Method Based on Hyper-Ellipsoid Neural Networks and SVM's with Applications to Face Recognition	461
SVM Based Adaptive Inverse Controller for Excitation Control	469
An Adaptive Internal Model Control Based on LS-SVM	479

Regularization Paths for ν -SVM and ν -SVR	486
A Fast and Accurate Progressive Algorithm for Training Transductive SVMs	497
Fast Support Vector Data Description Using K-Means Clustering	506
A Kernel-Based Two-Stage One-Class Support Vector Machines Algorithm.....	515
A Confident Majority Voting Strategy for Parallel and Modular Support Vector Machines	525
Soft-Sensor Method Based on Least Square Support Vector Machines Within Bayesian Evidence Framework	535
Simulation of Time Series Prediction Based on Smooth Support Vector Regression	545
 Fault Diagnosis/Detection	
Extension Neural Network Based on Immune Algorithm for Fault Diagnosis	553
A New Fault Detection and Diagnosis Method for Oil Pipeline Based on Rough Set and Neural Network	561
Probabilistic Neural Network Based Method for Fault Diagnosis of Analog Circuits	570
Gear Fault Diagnosis by Using Wavelet Neural Networks	580
An Adaptive Threshold Neural-Network Scheme for Rotorcraft UAV Sensor Failure Diagnosis	589

KPCA Plus FDA for Fault Detection	597
Distribution System Fault Diagnosis Based on Improved Rough Sets with Uncertainty	607
A Design Method of Associative Memory Model with Expecting Fault-Tolerant Field	616
Classification and Diagnosis of Mechanical Faults Using the RBF Network Based on the Local Bispectra	626
Temperature-Variation Fault Diagnosis of the High-Voltage Electric Equipment Based on the BP Neural Network	633
Diagnosis of Turbine Valves in the Kori Nuclear Power Plant Using Fuzzy Logic and Neural Networks	641
Communications and Signal Processing	
Stochastic Cellular Neural Network for CDMA Multiuser Detection	651
A New Approach of Blind Channel Identification in Frequency Domain	657
Soft Decision-Directed Square Contour Algorithm for Blind Equalization	663
Call Admission Control Using Neural Network in Wireless Multimedia Networks	672
A Neural Network Solution on Data Least Square Algorithm and Its Application for Channel Equalization	678
Multiple Receive-Antennas Aided and RBF-Based Multiuser Detection for STBC Systems	686

Location Management Cost Estimation for PCS Using Neural Network	695
Neural-Based Separating Method for Nonlinear Mixtures	705
Adaptive Natural Gradient Algorithm for Blind Convolutive Source Separation	715
Blind Source Separation in Post-nonlinear Mixtures Using Natural Gradient Descent and Particle Swarm Optimization Algorithm.....	721
Echo State Networks for Real-Time Audio Applications	731
Blind Separation of Positive Signals by Using Genetic Algorithm	741
A Speech Enhancement Method in Subband	751
Sequential Blind Signal Extraction with the Linear Predictor	758
Fast Code Detection Using High Speed Time Delay Neural Networks ...	764
Multiscale Estimation to the Parameter of Multidimension Time Series	774
Gaussianization Based Approach for Post-Nonlinear Underdetermined BSS with Delays	783
Regularized Alternating Least Squares Algorithms for Non-negative Matrix/Tensor Factorization	793
Underdetermined Blind Source Separation Using SVM.....	803
Predicting Time Series Using Incremental Langrangian Support Vector Regression	812

Image/Video Processing

A New Approach for Image Restoration Based on CNN Processor	821
Using Alpha-Beta Associative Memories to Learn and Recall RGB Images	828
A Method for Enlargement of Digital Images Based on Neural Network	834
Neural Network Based Correction Scheme for Image Interpolation	840
Edge Enhancement Post-processing Using Hopfield Neural Net	846
Image Quality Assessment Based on Wavelet Coefficients Using Neural Network	853
Neural Network Approach for Designing One- and Two-Dimensional Quasi-Equiripple FIR Digital Filters	860
Texture Image Segmentation Based on Improved Wavelet Neural Network	869
Local Spatial Properties Based Image Interpolation Using Neural Network	877
Image Processing Applications with a PCNN	884
Image Segmentation Based on Cluster Ensemble	894
Decision-Based Hybrid Image Watermarking in Wavelet Domain Using HVS and Neural Networks	904
MR Image Registration Based on Pulse-Coupled Neural Networks	914

Image Magnification Based on the Properties of Human Visual Processing 923

Incorporating Image Quality in Multimodal Biometric Verification 933

Efficient Motion Estimation Scheme for H.264 Based on BP Neural Network 943

Neural Network Based Visual Tracking with Multi-cue Adaptive Fusion 950

One-Class SVM Based Segmentation for SAR Image 959

A New Segmentation Method Based on SVM for HIFU Image-Guided System 967

Applications of Neural Networks

Greenhouse Air Temperature and Humidity Prediction Based on Improved BP Neural Network and Genetic Algorithm 973

A Modified RBF Neural Network and Its Application in Radar 981

Driving Load Forecasting Using Cascade Neural Networks 988

Minimal Resource Allocation on CAN Bus Using Radial Basis Function Networks 998

Neural Network Based High Accuracy Frequency Harmonic Analysis in Power System 1006

Hardware/Software Partitioning of Core-Based Systems Using Pulse Coupled Neural Networks 1015

Technical and Fundamental Analysis for the Forecast of Financial Scrip Quotation: An Approach Employing Artificial Neural Networks and Wavelet Transform	1024
Multi-view Moving Human Detection and Correspondence Based on Object Occupancy Random Field	1033
Determination of Storage Time of Rice Seed Using ANN Based on NIRS	1043
Selected Problems of Knowledge Discovery Using Artificial Neural Networks	1049
An Intelligent Differential Evolution Algorithm for Designing Trading-Ratio System of Water Market	1058
Neural Network-Based H_∞ Filtering for Nonlinear Jump Systems	1067
A Novel Off-Line Signature Verification Based on Adaptive Multi-resolution Wavelet Zero-Crossing and One-Class-One-Network....	1077
Combining News and Technical Indicators in Daily Stock Price Trends Prediction	1087
Project-Based Artificial Neural Networks Development Software and Applications.....	1097
Effects of Salinity on Measurement of Water Volume Fraction and Correction Method Based on RBF Neural Networks	1107
Implementation of Brillouin-Active Fiber for Low Threshold Optical Logic and Memory Based Neural Networks in Smart Structures	1114

Hydro Plant Dispatch Using Artificial Neural Network and Genetic Algorithm..... 1120

Application of Wavelet Packet Neural Network on Relay Protection Testing of Power System 1130

Robust Stabilization of Uncertain Nonlinear Differential-Algebraic Subsystem Using ANN with Application to Power Systems 1138

A Novel Residual Capacity Estimation Method Based on Extension Neural Network for Lead-Acid Batteries 1145

A Genetic Algorithm-Based Artificial Neural Network Approach for Parameter Selection in the Production of Tailor-Welded Blanks 1155

Development of Artificial Neural Networks-Based In-Process Flash Monitoring (ANN-IPFM) System in Injection Molding..... 1165

Reduce Feature Based NN for Transient Stability Analysis of Large-Scale Power Systems 1176

Semi-active Control of Eccentric Structures Based on Neural Networks 1182

Development of the Multi-target Tracking Scheme Using Particle Filter..... 1192

Author Index..... 1203

Using Three Layer Neural Network to Compute Multi-valued Functions

Nan Jiang¹, Yixian Yang², Xiaomin Ma³, and Zhaozhi Zhang⁴

¹ College of Computer Science, Beijing University of Technology, Beijing 100022, China
jiangnan@bjut.edu.cn

² Information Security Center, Beijing University of Posts and Telecommunications,
Beijing 100876, China

³ Engineering and Physics Department, Oral Roberts University, Tulsa, OK 74171, USA

⁴ Institute of Systems Science, Academia Sinica, Beijing 100080, China
zhzhza@iss.ac.cn

Abstract. This paper concerns how to compute multi-valued functions using three-layer feedforward neural networks with one hidden layer. Firstly, we define strongly and weakly symmetric functions. Then we give a network to compute a specific strongly symmetric function. The number of the hidden neurons is given and the weights are 1 or -1. Algorithm 1 modifies the weights to real numbers to compute arbitrary strongly symmetric functions. Theorem 3 extends the results to compute any multi-valued functions. Finally, we compare the complexity of our network with that of binary one. Our network needs fewer neurons.

1 Introduction

Multi-valued multi-threshold neural networks can be used for solving many kinds of problems, such as multi-valued logic, machine learning, data-mining, pattern recognition and *etc* [1-2].

Diep [3] gives the number of functions that can be calculated by a multi-threshold neuron and gives a lower bound on the number of weights required to implement a universal network. Žunić [4] derives an upper bound of the number of linear multi-valued threshold functions defined on some subsets of $\{0,1\}^n$. Ojha [5] enumerates the number of equivalence classes of linear threshold functions from a geometric lattice perspective in weight space. Ngom [6] points out that one strip (points located between two parallel hyperplanes) corresponding to one hidden neuron and constructs two neural networks based on these hidden neurons to compute a given but arbitrary multi-valued function. Young [7] deals with classification task for real-valued input. Anthony [1] presents two consistent hypothesis finders to learn multi-threshold functions. Obradović and Parberry [8, 9] view multi-threshold networks as analog networks of limited precision. In [9], the authors give two learning algorithms: one is for realizing any weakly symmetric q -valued functions and the other is for realizing arbitrary q -valued functions. But they use seven layer and five layer neural networks respectively. In [8], the authors use a simple three layer network with $q-1$ hidden

neurons to realize n -variable q -valued functions. But they need to increase the inputs to $n+q-1$ variables.

Our method is different from the above algorithms. We extend the technique of Siu [10] (Chapter 3) from binary valued to multi-valued cases.

2 Preliminaries

$R(q)=\{0,1,\dots,q-1\}$ is a residue class ring. In this paper, we assume $q \geq 3$ and use q to replace 0. A q -valued function f can be denoted by $R^n(q) \rightarrow R(q)$ or $\{1,2,\dots,q\}^n \rightarrow \{1,2,\dots,q\}$.

Assume that $X=(x_1,\dots,x_n)^T$ is the input vector of a neuron, where $x_i \in \{q,1,\dots,q-1\}$.

$W=(w_1,\dots,w_n)^T$ is the weight vector, where $w_i \in R$ (R is real field). Denote $W^T X = \sum_{i=1}^n w_i x_i$.

A q -valued (or $(q-1)$ -threshold) neuron Y is defined as $Y(t_1, t_2, \dots, t_{q-1})$

$$= \begin{cases} q & W^T X \in (-\infty, t_1) \\ 1 & W^T X \in [t_1, t_2) \\ \dots & \dots \\ q-1 & W^T X \in [t_{q-1}, +\infty) \end{cases}, \text{ where } t_1 < t_2 < \dots < t_{q-1} \text{ are thresholds.}$$

In order to give the method for computing arbitrary q -valued functions step by step, we give the concepts of weakly symmetric function and strongly symmetric function.

Definition 1 (Weakly Symmetric Function). A q -valued function $f: \{1,2,\dots,q\}^n \rightarrow \{1,2,\dots,q\}$ is said to be weakly symmetric if $f(x_1,\dots,x_n) = f(x_{(1)},\dots,x_{(n)})$ for any permutation $(x_{(1)},\dots,x_{(n)})$ of (x_1,\dots,x_n) .

Definition 2 (Strongly Symmetric Function). A q -valued function $f: \{1,2,\dots,q\}^n \rightarrow \{1,2,\dots,q\}$ is said to be strongly symmetric if f depends only on the sum of its input values $\sum_{i=1}^n x_i$.

3 Computing Arbitrary Multi-valued Functions

3.1 Neurons

Assume that $f(x_1,\dots,x_n)$ is an n -variable strongly symmetric function. Because $x_i \in \{q,1,\dots,q-1\}$, we have $n \leq \sum_{i=1}^n x_i \leq qn$. Divide the interval $[n, qn] = [n, qn+1)$ into $(q+d-3)s$

subintervals $[k_1^{(0)}, k_1^{(1)}), [k_1^{(1)}, k_1^{(2)}), \dots, [k_1^{(q+d-4)}, k_1^{(q+d-3)}), \dots, [k_s^{(0)}, k_s^{(1)}), [k_s^{(1)}, k_s^{(2)}), \dots, [k_s^{(q+d-4)}, k_s^{(q+d-3)}),$ where d is an integer and we will give its value in Theorem 1, $k_j^{(0)}, k_j^{(1)}, \dots, k_j^{(q+d-3)}$ also are integers, $k_j^{(i)} = k_j^{(i-1)} + 2$, $j=1,2,\dots,s$, $i=1,2,\dots,q+d-3$. Without loss of generality, we assume $k_s^{(q+d-3)} = qn+1$ (i.e. $2(q+d-3)(q-1)n+1$ and $s = ((q-1)n+1)/2(q+d-3)$) to avoid the use of cumbersome notations. $\sum_{i=1}^n x_i$ must belong to

one of the subintervals.

In order to compute f , we consider a three layer feedforward neural network. In the second (or hidden) layer, we construct $2ds$ ($q-1$)-threshold neurons $Y_{1j}, Y_{2j}, \dots, Y_{dj}, Y_{(d+1)j}, \dots, Y_{(2d-1)j}, Y_{(2d)j}, j=1, 2, \dots, s$. Their outputs are denoted by $Y_{1j}(k_j^{(0)}, k_j^{(1)}, \dots, k_j^{(q-2)}), Y_{2j}(k_j^{(1)}, k_j^{(2)}, \dots, k_j^{(q-1)}), \dots, Y_{dj}(k_j^{(d-1)}, k_j^{(d)}, \dots, k_j^{(d+q-3)}), Y_{(d+1)j}(-k_j^{(d+q-3)}, \dots, -k_j^{(d)}, -k_j^{(d-1)}), \dots, Y_{(2d-1)j}(-k_j^{(q-1)}, \dots, -k_j^{(2)}, -k_j^{(1)}), Y_{(2d)j}(-k_j^{(q-2)}, \dots, -k_j^{(1)}, -k_j^{(0)}), j=1, 2, \dots, s$, respectively, where we use $k_j^{(0)}, k_j^{(1)}, \dots, k_j^{(d+q-3)}, -k_j^{(d+q-3)}, \dots, -k_j^{(1)}, -k_j^{(0)}$ as thresholds. The outputs of neurons in the hidden layer are listed below.

$$\left\{ \begin{array}{l} Y_{ij}(k_j^{(i-1)}, k_j^{(i)}, \dots, k_j^{(i+q-3)}) = \begin{cases} q, & \sum_{i=1}^n x_i < k_j^{(i-1)} \\ 1, & k_j^{(i-1)} \leq \sum_{i=1}^n x_i < k_j^{(i)}, \quad i=1, 2, \dots, d \\ \dots \\ q-1, & \sum_{i=1}^n x_i \geq k_j^{(i+q-3)} \end{cases} \\ Y_{ij}(-k_j^{(-i+2d+q-2)}, \dots, -k_j^{(-i+2d)}) = \begin{cases} q, & -\sum_{i=1}^n x_i < -k_j^{(-i+2d+q-2)} \\ \dots \\ q-2, & -k_j^{(-i+2d+4)} \leq -\sum_{i=1}^n x_i < -k_j^{(-i+2d)}, \quad i=d+1, \dots, 2d \\ q-1, & -\sum_{i=1}^n x_i \geq -k_j^{(-i+2d)} \end{cases} \end{array} \right. \quad (1)$$

In the following, we omit the thresholds in the neurons $Y_{ij}, i=1, 2, \dots, 2d, j=1, 2, \dots, s$, e.g. $Y_{2j}(k_j^{(1)}, k_j^{(2)}, \dots, k_j^{(q-1)})$ is denoted by $Y_{2j}()$.

3.2 Computing a Specific Strongly Symmetric Function

We assume that the neurons Y_{1j}, \dots, Y_{dj} have weights 1 and $Y_{(d+1)j}, \dots, Y_{(2d)j}$ have weights -1. The third (or output) layer only has one neuron z which is also a q -valued neuron. Its inputs are the $2ds$ outputs of the neurons Y_{ij} . Because all weights of the neuron z are assumed to equal 1, so we need to compute $\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}()$ for various cases using the Eq. (1). For the notation of simplicity, we denote $\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}()$ when $\sum_{i=1}^n x_i = k_j^{(i)} (k_j^{(i)}+1)$ by $\sum_{(i)} (\sum_{(i)+1})$.

Case 1: there exists a j such that $\sum_{i=1}^n x_i = k_j^{(0)}$, then

$$\begin{array}{llll} Y_{1j}()=1 & Y_{1l} = \begin{cases} q-1 & l < j \\ q & l > j \end{cases} & \dots & Y_{dj}()=q & Y_{dl} = \begin{cases} q-1 & l < j \\ q & l > j \end{cases} \\ Y_{(d+1)j}=q-1 & Y_{(d+1)l} = \begin{cases} q & l < j \\ q-1 & l > j \end{cases} & \dots & Y_{(2d)j}=q-1 & Y_{(2d)l} = \begin{cases} q & l < j \\ q-1 & l > j \end{cases} \end{array}$$

So, $\sum_{(0)} = 2dq - d + 1 + (2dq - d)(s-1)$.

Case 2: there exists a j such that $\sum_{i=1}^n x_i = k_j^{(0)} + 1$, then

$$\begin{aligned}
 Y_{1j}()=1 & \quad Y_{1l} = \begin{cases} q-1 & l < j \\ q & l > j \end{cases} \quad \dots \quad Y_{dj}()=q & \quad Y_{dl} = \begin{cases} q-1 & l < j \\ q & l > j \end{cases} \\
 Y_{(d+1)j}=q-1 & \quad Y_{(d+1)l} = \begin{cases} q & l < j \\ q-1 & l > j \end{cases} \quad \dots \quad Y_{(2d)j}=q-2 & \quad Y_{(2d)l} = \begin{cases} q & l < j \\ q-1 & l > j \end{cases}
 \end{aligned}$$

So, $\sum_{(0)+1}=2dq-q-d+(2dq-d)(s-1)$.

Computing the other cases similarly, we can get Eq. (2).

$$\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}() = \begin{cases} \left. \begin{aligned} & 2dq - (i+1)q - d + (i+1) + (2dq-d)(s-1), \quad \sum_{i=1}^n x_i = k_j^{(i)} \\ & 2dq - (i+1)q - d + (2dq-d)(s-1), \quad \sum_{i=1}^n x_i = k_j^{(i)} + 1 \end{aligned} \right\} \text{if } 0 \leq i \leq d-2 \\ \left. \begin{aligned} & dq + (2dq-d)(s-1), \quad \sum_{i=1}^n x_i = k_j^{(i)} \\ & dq - d + (2dq-d)(s-1), \quad \sum_{i=1}^n x_i = k_j^{(i)} + 1 \end{aligned} \right\} \text{if } d-1 \leq i \leq q-3 \\ \left. \begin{aligned} & dq + (i-q+2)(q-1) + (2dq-d)(s-1), \quad \sum_{i=1}^n x_i = k_j^{(i)} \\ & dq + (i-q+3)q - d + (2dq-d)(s-1), \quad \sum_{i=1}^n x_i = k_j^{(i)} + 1 \end{aligned} \right\} \text{if } q-2 \leq i \leq d+q-4 \end{cases} \quad (2)$$

In order to realize q -valued functions, $\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}()$ must have at least q different values. From Eq. (2) we see that this is related to d . Theorem 1 gives the value of d .

Theorem 1. In order to make $\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}()$ have at least q different values, $d=q-1$ is necessary and sufficient.

Proof. From Eq. (2), we know that $\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}()$ always includes $(2dq-d)(s-1)$. For a residue class ring, this has no affection to the number of values. So in this proof, we omit them.

1. Sufficiency: $d=q-1 \Rightarrow \sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}()$ have at least q different values.

(1) When $0 \leq i \leq d-2=q-3$, $\sum_{(i)}=2dq-(i+1)q-d+(i+1)=1-q+i+1=i+2$. Since $2 \leq i+2 \leq q-1$, $\sum_{(i)}$ can take the value $2, 3, \dots, q-1$.

(2) When $0 \leq i \leq d-2=q-3$, $\sum_{(i)+1}=2dq-(i+1)q-d=1-q=1$.

(3) When $q-2 \leq i \leq d+q-4=2q-5$, $\sum_{(i)}=dq+(i-q+2)(q-1)=q-i-2$. Since $3-q \leq q-i-2 \leq q$, $\sum_{(i)}$ can take the value $3, 4, \dots, q$.

(4) When $q-2 \leq i \leq d+q-4=2q-5$, $\sum_{(i)+1}=dq+(i-q+3)q-d=1-q=1$.

From (1)-(4), we can see that $\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}()$ can take the value $1, 2, \dots, q$.

2. Necessity: $d \leq q-2 \Rightarrow \sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}()$ can not take q different values.

- (1) When $0 \leq i \leq d-2$, $\sum_{(i)} = 2dq - (i+1)q - d + (i+1) = i+1-d$. Because $1-d \leq i+1-d \leq q-1$ and $d \leq q-2$, we can get that $\sum_{(i)} \geq 3$.
- (2) When $0 \leq i \leq d-2$, $\sum_{(i+1)} = 2dq - (i+1)q - d = q-d \geq 2$.
- (3) When $d-1 \leq i \leq q-3$, $\sum_{(i)} = dq = q$.
- (4) When $d-1 \leq i \leq q-3$, $\sum_{(i+1)} = dq - d = q-d \geq 2$.
- (5) When $q-2 \leq i \leq d+q-4$, $\sum_{(i)} = dq + (i-q+2)(q-1) = q-i-2$. Because $2-d \leq q-i-2 \leq q$ and $d \leq q-2$, we can get that $\sum_{(i)} \geq 4$.
- (6) When $q-2 \leq i \leq d+q-4$, $\sum_{(i+1)} = dq + (i-q+3)q - d = q-d \geq 2$.

From (1)-(6), we can see that $\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}(\cdot)$ can not take the value 1. \square

Remark 1. According to Theorem 1, the input of neuron z is $(\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}(\cdot) \bmod q)$ instead of $\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}(\cdot)$ directly. Otherwise, if the input of z is $\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}(\cdot)$, $d=q-1$ will only be the sufficient condition. That is to say, $d \leq q-1$.

The inputs and weights of z have been determined. Now we give the $q-1$ thresholds of z . From the sufficiency proof of Theorem 1, we can easily get $\sum_{(0)+1} = 1$, $\sum_{(i)} = i+2$, $0 \leq i \leq q-3$. So they can be taken as the $q-1$ thresholds of z . Then, we obtain a three layer feedforward neural network which realizes a specific strongly symmetric function:

$$f(x_1, \dots, x_n) = z(\sum_{(0)+1}, \sum_{(0)}, \sum_{(1)}, \dots, \sum_{(q-3)}) = \begin{cases} q \\ 1 \\ 2 \\ \vdots \\ q-1 \end{cases} \text{ for } \sum_{i=1}^n x_i = \begin{cases} k_j^{(q-2)} \\ k_j^{(0)} + 1, \dots, k_j^{(2q-5)} + 1 \\ k_j^{(0)} \\ \vdots \\ k_j^{(q-3)}, k_j^{(q-1)} \end{cases} \quad j=1, 2, \dots, s.$$

Since $s = ((q-1)n+1)/2(q+d-3)$, there are $2ds = \frac{qn}{2} + \frac{q+n-1}{2(q-2)}$ neurons in the hidden layer for $q \geq 3$. This is the complexity of the neural network.

3.3 Computing Arbitrary Strongly Symmetric Functions

If we allow Y_{ij} have integer and rational weights, we can prove the following theorem.

Theorem 2. Let $f(x_1, \dots, x_n)$ be a q -valued strongly symmetric function. Then f can be computed using a three layer feedforward neural network with $2ds = \frac{qn}{2} + \frac{q+n-1}{2(q-2)}(q-1)$ threshold neurons in the hidden layer which have integer and rational weights chosen from the set $S = \{\pm 1, \dots, \pm q, \pm 1/2, \dots, \pm q/2, \pm 1/3, \dots, \pm q/3, \dots, \pm 1/q, \dots, \pm (q-1)/q\}$.

Proof. Since f only depends on the sum of its inputs $\sum_{i=1}^n x_i$, then we assume that

$f(x_1, \dots, x_n) = a_m \in \{1, 2, \dots, q\}$, where $m = \sum_{i=1}^n x_i \in [n, qn]$. For proving the theorem, we only

need to prove that we can choose a set of integer and rational weights from the given set S for the neurons Y_{ij} $i=1, 2, \dots, 2d$, $j=1, \dots, s$ such that the constructed network

outputs a_m . For every m , we choose fixed inputs x_1, \dots, x_n such that $\sum_{i=1}^n x_i = m$, without loss of generality, let $(x_1, \dots, x_n) = (1, 1, \dots, 1), (2, 1, \dots, 1), (2, 2, \dots, 1), \dots, (q, q, \dots, q)$. Then we choose weights $w_1^{(ij)}, \dots, w_n^{(ij)}$ for Y_{ij} , where, $w_l^{(ij)} \in S, l=1, 2, \dots, n, i=1, 2, \dots, d$. The weights of the neurons $Y_{(d+1)j}, \dots, Y_{2dj}$ are set to be $-w_1^{(dj)}, \dots, -w_n^{(dj)}, \dots, -w_1^{(1j)}, \dots, -w_n^{(1j)}$, respectively. We require that the output of the neurons satisfying $\sum_{j=1}^s \sum_{i=1}^{2d} Y_{ij}(\cdot) = (2dq - d)(s-1) + a_m$ for all m ($n \leq m \leq qn$). This can be realized by using the following algorithm.

Algorithm 1. Initially, set the weights equal 1 for neurons Y_{1j}, \dots, Y_{dj} and -1 for neurons $Y_{(d+1)j}, \dots, Y_{2dj}, j=1, \dots, s$, and set $t=1$. Set the remnant set $m = \{n, n+1, \dots, qn\}$.

Step 1. Compute $b_m = z(\sum_{(0)+1}, \sum_{(0)}, \dots, \sum_{(q-3)})$, for the remnant set of m .

Step 2. Compare a_m and b_m , delete those m such that $a_m = b_m$.

Step 3. If the remnant set of m is not void, find $m_t =$ the smallest m in the remnant set. Otherwise, stop.

Step 4. If $m_t \in [k_j^{(0)}, k_j^{(2q-4)})$, then set the weights $w_i^{(1j)} = \dots = w_i^{(dj)}, i=1, \dots, n$. such that

$$\sum_{i=1}^n w_i^{(1j)} x_i = \begin{cases} k_j^{(q-2)} \\ k_j^{(0)} + 1 \\ k_j^{(0)} \\ \vdots \\ k_j^{(q-3)} \end{cases} \quad \text{if } a_{m_t} = \begin{cases} q \\ 1 \\ 2 \\ \vdots \\ q-1 \end{cases}$$

where $\sum_{i=1}^n x_i = m_t$. Set $w_i^{((d+1)j)} = \dots = w_i^{(2dj)} = -w_i^{(1j)}, i=1, \dots, n$. Set $t=t+1$, then go to step 1.

Evidently, execute the algorithm one cycle, the remnant set reduces at least one m . Hence the algorithm must stop after executing at most $(q-1)n+1$ cycles. Thus the function f is computed. \square

Corollary 1. If $f(x_1, \dots, x_n)$ is a q -valued strongly symmetric function, then f can be computed using a three layer feedforward neural network with fixed architecture.

There are $T, \frac{qn}{2} + \frac{q+n-1}{2(q-2)} \leq T \leq 2(q-1)((q-1)n+1)$, $(q-1)$ -threshold neurons in the hidden

layer. More precisely, $T = \sum_{j=1}^s t_j$, there are $t_j, 1 \leq t_j \leq 2q$, same neurons $Y_{ij}, i=1, 2, \dots, 2d$. The weights of the neurons Y_{ij} are chosen from the set S .

Proof. It is easy to see that the proof of corollary 1 is a variant of theorem 2. In fact, if the algorithm is executed $t_j, t_j > 1$, cycles for $m_t \in [k_j^{(0)}, k_j^{(2q-4)})$, then we need t_j same neurons $Y_{ij}, i=1, 2, \dots, 2d$ in the hidden layer and choose different weights for them as step 4 of the algorithm. In this case, the weights of the output neuron z must be adapted. For each $m \in [k_j^{(0)}, k_j^{(2q-4)})$, only $2d$ weights of the neuron z equal 1 corresponding to its inputs $\sum_{i=1}^{2d} Y_{ij}(\cdot)$ such that the output of the neuron z equals a_m , the other weights of z equal 0. \square

3.4 Computing Arbitrary q -Valued Functions

As a consequence of Theorem 2, we have the following theorem.

Theorem 3. Any q -valued function $f(x_1, \dots, x_n)$ can be computed by a three layer neural network with $\frac{q-1}{2(q-2)}q^n$ $(q-1)$ -threshold neurons in the hidden layer.

Proof. Since the sum $\sum_{j=1}^n q^{j-1}x_j$ is distinct integers for different inputs $(x_1, \dots, x_n) \in \{1, 2, \dots, q\}^n$. Thus any q -valued function can be regarded as a function of weighted sum of its inputs or a strongly symmetric function of $(q^n-1)/(q-1)$ variables. The result follows from theorem 2. \square

4 Binary Neuron vs. Multi-valued Neuron

Siu [10] has proved that any m -variable Boolean function can be computed by a binary neural network with $\lceil (2^m - 1)/2 \rceil$ neurons in the hidden layer. If we use binary neural networks to compute any n -variable q -valued function $f(x_1, \dots, x_n)$, we must combine at least $q-1$ such networks. Each time, one value is separated from the remnant values. Furthermore, variables (x_1, \dots, x_n) must be altered to Boolean variables (y_1, \dots, y_m) , i.e. a n -variable q -valued function is represented to a set of $(q-1)$ m -variable binary functions. Since, $m = n \log_2 q$, a n -variable q -valued function can be computed by a three layer binary neural networks with $\lceil (2^m - 1)/2 \rceil = \lceil (q^n - 1)/2 \rceil$ hidden neurons. So the total number of hidden neurons is $(q-1)\lceil (q^n - 1)/2 \rceil$.

Theorem 3 asserts that if we use three layer network with $(q-1)$ -threshold neurons, the total number of hidden neurons is $\frac{q-1}{2(q-2)}q^n$.

$\frac{q-1}{2(q-2)}q^n < (q-1)\lceil \frac{q^n-1}{2} \rceil$, this indicates that the complexity of neural networks with $(q-1)$ -threshold neurons is lower than that of binary neural networks.

5 Conclusion

This paper proposes a method to compute arbitrary q -valued functions using three layer feedforward neural networks with one hidden layer. The number of hidden neurons required is given. Estimating the number of hidden neurons is a fundamental problem concerning the complexity of neural networks. Through the constructive proving process, we give the upper bound.

Acknowledgment. This work is supported by Excellent Person Development Foundation of Beijing under Grant No. 20061D0501500191.

References

1. Anthony, M.: Learning Multivalued Multithreshold functions. CDMA Research Report No. LSE-CDMA-2003-03, London School of Economics (2003)
2. Wang, S.: Multi-valued Neuron and Multi-thresholded Neuron: Their Combinations and Applications. Acta Electronica Sinica (1996) 1-6

3. Diep, T.A.: Capacity of Multilevel Threshold Devices. *IEEE Transactions on Information Theory* (1998) 241-255
4. Žunić, J.: On Encoding and Enumerating Threshold Functions. *IEEE Transactions on Neural Networks* (2004) 261-267
5. Ojha, P.C.: Enumeration of Linear Threshold Functions from the Lattice of Hyperplane Intersections. *IEEE Transactions on Neural Networks* (2000) 839-850
6. Ngom, A., Stojmenović, I., Milutinović, V.: STRIP --- a Strip-based Neural-Network Growth Algorithm for Learning Multiple-valued Functions. *IEEE Transactions on Neural Networks* (2001) 212-227
7. Young, S., Downs, T.: CARVE --- a Constructive Algorithm for Real-valued Examples. *IEEE Transactions on Neural Networks* (1998) 1180-1190
8. Obradović, Z., Parberry, I.: Learning with Discrete Multi-valued Neurons. *Journal of Computer and System Sciences* (1994) 375-390
9. Obradović, Z., Parberry, I.: Computing with Discrete Multi-valued Neurons. *Journal of Computer and System Sciences* (1992) 471-492
10. Siu, K.Y., Roychowdhury, V., Kailath T.: *Discrete Neural Computation: a Theoretical Foundation*. Englewood Cliffs, NJ: Prentice-Hall (1995)

A Novel Global Hybrid Algorithm for Feedforward Neural Networks*

Hongru Li, Hailong Li, and Yina Du

Key Laboratory of Integrated Automation of Process Industry (Northeastern University), Ministry of Education, Shenyang, 110004, China
lihongru@ise.neu.edu.cn

Abstract. A novel global optimization hybrid algorithm was presented for training neural networks in this paper. During the course of neural networks training, when the weights are being adjusted with Quasi-Newton(QN) method, the error function may be stuck in a local minimum. In order to solve this problem, a original Filled-Function was created and proved. It was combined with QN method to become a global optimization hybrid algorithm. When the net is trained with our new hybrid algorithm, if error function was tripped in a local minimal point, the new hybrid algorithm was able to help networks out of the local minimal point. After that, the weights could being adjusted until the global minimal point for weights vector was found. One illustrative example is used to demonstrate the effectiveness of the presented scheme.

1 Introduction

Artificial neural networks have been developing in many years. Because of their excellent capability of self-learning and self-adapting, neural networks are very important in many fields [1,2,3,4]. In the numerous models of neural networks, the multilayer Feedforward neural networks(FFN) are widely used. The FFN model has powerful ability in approximation for any continuous functions, therefore, this ability provides basic theory for its modelling and controlling in nonlinear system [2]. And the Back-Propagation algorithm and Quasi-Newton(QN) method etc, which are based on gradient descent algorithm, make the feed-forward neural networks more easily to adjust the weights vector. However, the gradient descent algorithm has the natural drawback: those training algorithms would often stick the error function in a local minimal point and stop searching the optimal weights for goal error function.

For the weakness of those FFN algorithm, the hybrid algorithms with randomness global optimization algorithms ,such as genetic algorithms, simulated annealing algorithm, randomness searching, were presented [5,6,7]. These hybrid algorithms really brought some exciting results. Nevertheless, for the large-scale

* This work is supported by national natural science foundation of P.R. China Grant #60674063, by national postdoctoral science foundation of P.R. China Grant #2005037755, by natural science foundation of Liaoning Province Grant #20062024.

continuous system optimization problem, those algorithms were too slow. In that theory, their random property would make sure that they will find the final solution in the enough time. But in fact, those algorithms would waste too much time in practice projects [1]. In sum, there are still some shortcomings in those randomness hybrid algorithms.

In contrast to randomness global optimization algorithms, deterministic global optimization algorithms were also developing quickly in recent years, for example, the Filled-Function, the Tunnelling method, the Downstairs method etc. [8,9,10]. Deterministic global optimization algorithms have the faster searching speed and now have been an important branch in optimization algorithm field. Among those deterministic algorithms, the Filled-Function method has a better performance. But in the practicable program, those Filled-Function models with two or more parameters and exponential parameters are difficult to be adjusted and often overflow.

To avoid the drawbacks of existing Filled-Function models, a single-parameter Filled-Function model was presented and combined with QN method to optimize the weights and bias to minimize the error function. A new hybrid training algorithm is constructed for FFN. We also compare this new hybrid algorithm with QN method and discuss the merits and flaws.

2 Filled-Function Optimization Algorithm

In this section, the model that we considered is:

$$\min f(X) \tag{1}$$

Where $X \in \Omega \subset \mathbb{R}^n$, $f(X)$ is a multi-extremum function, $f(X)$ has the global minimal point in Ω , and assume that $f(X)$ has second derivative.

Definition 1. ... $X_1, X_2 \in \mathbb{R}^n, X_1 \neq X_2$. $f(\lambda) = f(X_1 + \lambda(X_2 - X_1))$...

Definition 2. ... B_1 ... X_1 ,
) $X_0 \in B_1, X_0 \neq X_1, f(X_0) > f(X_1)$, ...
 $X_0 \notin B_1$, ... X_1

Definition 3. ... X_2 ... X_1 , ...
 $f(X_2) > f(X_1)$, ... B_2 ... B_1

This algorithm’s main idea is that: Assume that X_1 is a local minimal point in $f(X)$. A Filled-Function $P(X)$ could be constructed and make the point X_1 to be the global maximal point of $P(X)$. Furthermore, for the points on the $P(X)$, there is not any stable point in the basins which are higher than the

basin of X_1 on the $f(X)$, however, there is a stable point \bar{X} in the basins which are lower than basin of X_1 . And the point \bar{X} is the minimal point on the line through the X_1 and \bar{X} . After that, the point \bar{X} is viewed as the initial point to optimize the $f(X)$. Then we could find the point X_2 that meets the condition $f(X_2) < f(X_1)$ and replaces X_1 . We could repeat such a process to find the global minimal point X^* . Figure 1 gives us an example to show the process that Filled-Function help the two-dimension curvilinear function $f(X)$ to get out of the local minimal point. To avoid the flaws in existing models, a new single-parameter model was brought into FNN. This Filled-Function model could be implemented and adjusted easily.

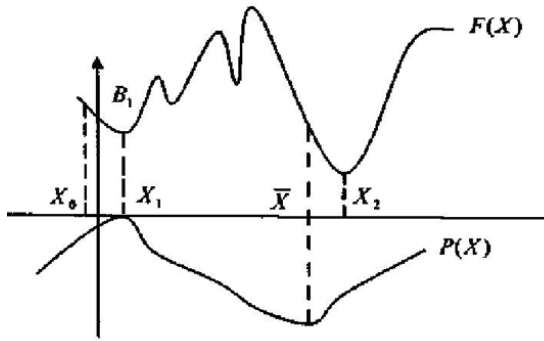


Fig. 1. The filled-function of the two-dimension curvilinear function F

When we got the minimal point X_1 on $f(X)$, a new Filled-Function model on the point X_1 could be constructed as follows:

$$H(X, A) = \frac{1}{1 + \sqrt{F(X) - F(X_1)}} - A \ln(1 + \|X - X_1\|^2) \quad (A > 0) \quad (2)$$

Before put this Filled-Function model into an algorithm, we must prove that this one could satisfy the basic quality of the Filled-Function.

Proposition 1. X_1 is the maximal point on the $H(X, A)$ if $X = X_1$.

When there exists $X = X_1, H(X, A) = 1$ and $f(X) > f(X_1)$, if $A > 0$, then $X \in B_1$ satisfies:

$$\frac{1}{1 + \sqrt{f(X) - f(X_1)}} - A \ln(1 + \|X - X_1\|^2) < 1 = H(X_1) \quad (3)$$

so X_1 is the maximal point on the $H(X, A)$.

Proposition 2. $f(X) > f(X_1)$, $d \dots$

$$\begin{cases} d^T \nabla f(X) \geq 0 \\ d^T (X - X_1) > 0 \end{cases} \quad (4)$$

$$\begin{cases} d^T \nabla f(X) > 0 \\ d^T (X - X_1) \geq 0 \end{cases} \quad (5)$$

Let $H(X, A) = \frac{\nabla F}{2(1 + \sqrt{f(X) - f(X_1)})^2 \sqrt{f(X) - f(X_1)}} - A \left(\frac{2(X - X_1)}{1 + \|X - X_1\|^2} \right)$

The gradient of the Filled-Function $H(X, A)$ is as follows

$$\nabla H(X, A) = - \frac{\nabla F}{2(1 + \sqrt{f(X) - f(X_1)})^2 \sqrt{f(X) - f(X_1)}} - A \left(\frac{2(X - X_1)}{1 + \|X - X_1\|^2} \right) \quad (6)$$

$$d^T \nabla H(X, A) = - \frac{d^T \nabla f}{2(1 + \sqrt{f(X) - f(X_1)})^2 \sqrt{f(X) - f(X_1)}} - 2A \left(\frac{d^T (X - X_1)}{1 + \|X - X_1\|^2} \right) \quad (7)$$

From the assumption on direction d , we could get:

$$d^T \nabla H(X, A) < 0 \quad (8)$$

Therefore $H(X, A)$ will descend along the direction d , for any positive integer A .

Proposition 3. If $f(X) > f(X_1)$, $d^T \nabla f(X) < 0$ and $d^T (X - X_1) > 0$

$$A > - \frac{d^T \nabla f}{4(1 + \sqrt{f(X) - f(X_1)})^2 \sqrt{f(X) - f(X_1)}} \frac{1 + \|X - X_1\|^2}{d^T (X - X_1)} = A^* \quad (9)$$

Let $H(X, A) = \frac{\nabla F}{2(1 + \sqrt{f(X) - f(X_1)})^2 \sqrt{f(X) - f(X_1)}} - A \left(\frac{2(X - X_1)}{1 + \|X - X_1\|^2} \right)$

Because of (6)

$$d^T \nabla H(X, A) = - \frac{d^T \nabla f}{2(1 + \sqrt{f(X) - f(X_1)})^2 \sqrt{f(X) - f(X_1)}} - 2A \left(\frac{d^T (X - X_1)}{1 + \|X - X_1\|^2} \right) \quad (10)$$

If $A > A^*$ then $d^T \nabla H(X, A) < 0$ So $H(X, A)$ would descend along the direction d .

Proposition 4. If $f(X) > f(X_1)$, $d^T \nabla f(X) < 0$ and $d^T (X - X_1) > 0$, $A < A^*$, then $H(X, A) > 0$

Because of (6), if there exists $A < A^*$, then $d^T \nabla H(X, A) > 0$ Therefore, $H(X, A)$ would rise along the direction d .

3 The New Hybrid Training Algorithm

We could train weights and bias with the help of the Filled-Function. Actually, we utilize the Filled-Function to make the error function get rid of the local minimal point. For the nets with n training sample data and q output neurons, the error function and sum error function of the k th sample are defined as follows:

$$E(k, w) = \frac{1}{2} \sum_{i=1}^q (y_i(k) - y_i(k, W))^2 \quad (11)$$

$$E(w) = \frac{1}{n} \sum_{k=1}^n E(k, w) \quad (12)$$

Where $y_i(k)$ is the output of the networks and $y_i(k, W)$ is the expected output, and W is the weight matrix. The bias could be viewed as weights as the input is -1 . So the error function could be transformed to be the function only with argument W . Training the net is a procedure that finds proper weights matrix to minimize the $E(W)$.

The training algorithm with single parameter Filled-Function is as follows:

- Step 1.** Choose a random initial point W_0 in the reign of $[0, 1]$ to minimize the $E(W)$, and set the goal precision $h = 0.001$. Utilize the QN method to find a minimal point W_1^* .
- Step 2.** If the $EW_1^* < h$, then W_1^* is the optimal weights, end the algorithm, or turn to Step 3.
- Step 3.** Construct the Filled-Function as (2) on the point W_1^* , and choose an initial point W_1 next to W_1^* , e.g. $W_1 = W_1^* + \delta e_1$ ($\delta > 0$). e_1 is the first unit vector, and $A > 0$.
- Step 4.** Search from the point W_1 to find the minimal point W_k^* of $H(W, A)$ with QN method.
- Step 5.** Make $W = W_k^*$, if the any one of criterion below is satisfied:
(1) $H(W_k^*, A) \geq H(W_{k-1}^*, A)$, **(2)** $H(W_k^*, A) \geq 0$, **(3)** $d_{k-1}^T \nabla H(W_k^*, A) \geq 0$,
(4) $(W_k^* - W_1^*)^T \nabla H(W_k^*, A) \geq 0$,
(5) $\|\nabla H(W_k^*, A)\| \leq \varepsilon$ (ε is the positive integer).
- Step 6.** Optimize the $E(W)$ from the new minimal point W , then we could get the another minimal point W_2^* .
- Step 7.** If $E(W_2^*) \leq E(W_1^*)$, then let $W_1^* \leftarrow W_2^*$, and turn to Step 2.
- Step 8.** If $E(W_2^*) > E(W_1^*)$, then magnify A , e.g. $A = 2A$, and turn to Step 2.

For the problem of adjusting the important parameter A : In Filled-Function, A is a very important parameter. From the Proposition 2, when $E(W)$ rises, $H(W, A)$ would descend as long as A is a positive integer. From the Proposition 3, when $A > A^*$ and $E(W)$ descend, $H(W, A)$ could still descend along the direction d . From the Proposition 4, in the basin lower than B_1 , when $E(W)$ descends, if $A < A^*$, then $H(W, A)$ would rise along direction d . It indicates that only if the parameter A is in a limited range, $H(W, A)$ would be an available

Filled-Function. This range limited by A^* could be calculated by definite equation. And A^* is a value varying with weights matrix W . In the neighborhood of local minimal point, $E(W)$ is prone to $E(W^*)$, therefore, A^* is prone to infinity. So at the beginning of the algorithm, A is set to be a large positive integer, e.g. $A = 100$. Although there is no such theory to follow in the math field, the experiment could give us a general adjusting trend. The adjusting trend is that with the change of the matrix W , A^* could be altered to a larger integer. In the algorithm imitation, when A is set firstly, if $W_2^* \leq W_1^*$, then the A is a proper parameter. If $W_2^* > W_1^*$, A needs to be changed, such as $A \leftarrow 2A$.

4 Simulation Results

Process Modelling

To test the validity of the new algorithm for real problem, we made a simulation about a project's process modelling. The object is fuel furnace in a certain steel company. Three hundred input-output data were taken from the project. The goal of modelling is construct the nonlinear dynamic mapping relationship between input and output. The simulation used QN method and new hybrid algorithm to training the net, respectively.

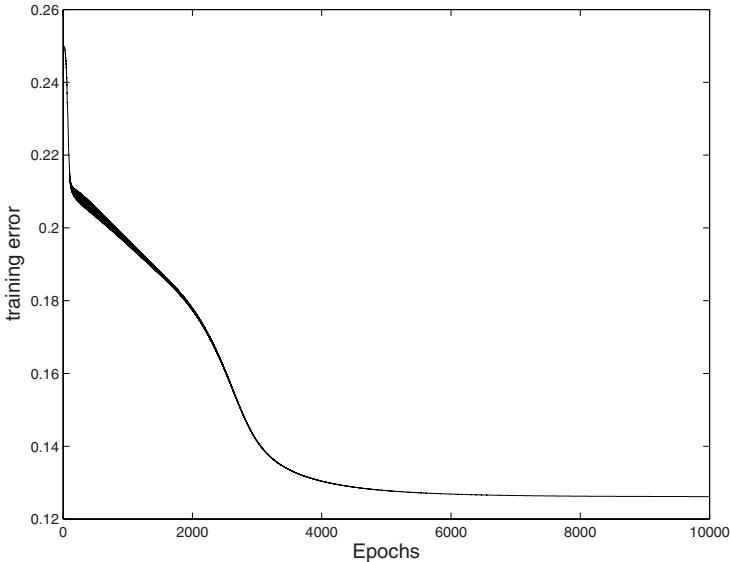


Fig. 2. Figure on training with the QN method

The training results are shown as Figure 2 and Figure 3. In the Figure 2, the error function training with the QN method is tripped in the local minimum and error stays at 0.1295 after 6369 epoch. In the Figure 3, however, the hybrid

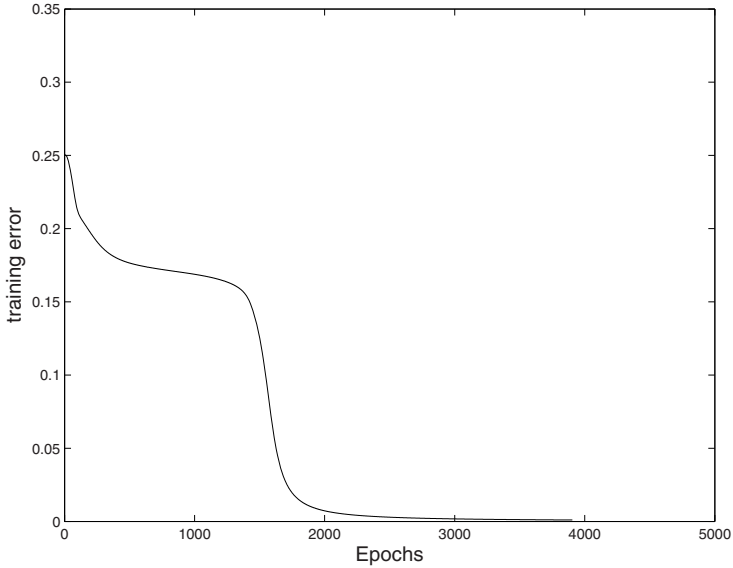


Fig. 3. Figure on training with hybrid algorithm

algorithm could make the error reach to the desired precision at 9.8672×10^{-4} after 3954 epoch. Simulation results show that hybrid algorithm is more available than QN method to search the global minimum.

5 Conclusions

The global optimization algorithm is researched in this paper. Firstly, a new Filled-Function model was constructed and proved. Furthermore, this new model was combined with QN method to be a hybrid neural network training algorithm successfully. Later, the new training algorithm was proved available with the algorithm simulation. Therefore, the new hybrid algorithm could solve the local minimal point problem of QN method effectively. In addition, there are still some problems left. For the problem of adjusting the important parameter A in a more convincing method, the writers are doing some more researches now.

References

1. Li, H., Wan, B.: A New Global Optimization Algorithm for Training Feedforward Neural Networks and Its Application. *System Engineering Theory and Practice*. **8** (2003) 42–47
2. Gao, M., Liu, X., Li, S.: A New Global Optimization BP Neural Networks. *Jouenal of Binzhou Teachers College*. **20** (2004) 37–41
3. Jiang, H.: Study on Class of Filled-Functions for Global Optimization. *Journal of Harbin University of Commerce(Natural Sciences Edition)*. **21** (2005) 230–232

4. Horinik K., Stinchcombe M., White H.: Multilayer Feed Forward Networks Are Universal Approximators. *Neural Networks*. **2** (1989) 359–366
5. Andersen R. S., Bloomfield P.: Properties of the Random Search in Global Optimization. *Journal of Optimization Theory and Applications*. **16** (1975) 383–398
6. David J. J., Frenzel J. F.: Training Product Unit Neural Networks with Genetic Algorithms. *IEEE Expert*. **8** (1993) 26–33
7. Kirkpatrick S., Gelatt C. D., Vecchi M. P.: Optimization by Simulated Annealing. *Science*. **220** (1983) 671–680
8. Zheng, Y.: A Class of Filled-Functions for Global Optimization. *Mathematic and Physical Journal*. **14** (1994) 184–189
9. Levy A. V.: The Tunneling Method Applied to Global optimization presented at the SIAM Conference on Numerical Optimization(Boulder Colorado U.S.A Fune). **4** (1984) 12–14
10. He Y.: A Downstairs Method for Finding a Global Minimizer of a Function of Several variables. *Journal on Numerical Methods and Computer Applications*. **10** (1989) 56–63
11. Shubert. B.O.: A Sequential Method Seeking the Global Maximum of a Function. *SIAM J.Numer. Anal.*. **9** (1972) 379–388
12. Kong,M., Zhuang,J.: A Modified Filled Function Method for Finding a Global Minimizer of a Nonsmooth Function of Several Variables. *Numerical Mathematics A Journal of Chinese Universities*. **2** (1996) 165–174
13. Ge, R.: A Filled Function Method for Finding a Global Minimizer of a Function of Several Variables. *The Dundee Biennial Conference on Numerical Analysis*. (1983) 56–63
14. Zhang L.,Li D., Tian W.: A New Filled Function Method for Global Optimization. *Journal of Global Optimization*. **19** (2004) 37–43

Study on Relationship Between NIHSS and TCM-SSASD Based on the BP Neural Network Multiple Models Method

Zhang Yanxin¹, Xu Junfeng², Gao Ying², and Hou Zhongsheng¹

¹ Institute of Automatic control, School of Electronics and Information Engineering, Beijing Jiaotong University, Beijing, 100044, P.R. China

² Dongzhimen hospital, Haiyuncang St., Dongcheng District, Beijing 100700, China
zyxhyq@yahoo.com.cn
jolia_tara@yahoo.com.cn
hzhsh@telecom.njtu.edu.cn

Abstract. In this paper, the complex nonlinear relationship between NIHSS and TCM-SSASD is analyzed. Much method based on the BP neural networks is induced to approximate this complex nonlinear relationship. At the same time, two schemes of multiple models are proposed to improve the approximation performance of the BP neural network. Through the comparison among these schemes, it is shown that there is exact complex nonlinear relationship between the two diagnosis sheets. The works in the paper would be guidance for the diagnosis of the Apoplexy Syndromes and assistant for the further study of the relation between Western medicine and Traditional Chinese Medicine.

1 Introduction

National Institutes of Health Stroke Scale (abbreviation: NIHSS), which is called Western medicine sheet, is the acknowledged international criterion for the apoplexy syndromes. TCM Standardized Sheet of Apoplexy Syndromes Diagnosis (abbreviation: TCM-SSASD), which is called Chinese medicine sheet, is the standardized half-quantitative Sheet, which is summarized by many Chinese medicine experts based on their clinic experience of many years [1-3]. This sheet divide the apoplexy syndromes into six basic Syndrome, which include wind syndrome, fire syndrome, phlegm syndrome, static blood syndrome, qi vacuity syndrome, preponderance of yang due to deficiency of yin syndrome, and each Syndrome is measured by ascertained value. It is aimed to assistant for the future clinic diagnosis and to build a data flat for discovering the differentiation thought of TCM by the data-mining method [4]. For a long term, the Western medicine and TCM are unified together for the diagnosis and treatment of the apoplexy syndromes. However, there is no scientific criterion guidance for the head layer design. Are there any coherence, difference and complementarities between the single Western medicine and the single TCM? Can the erudite and intensive TCM compete with the advanced western medicine? How to unify the western medicine and TCM to realize the complement of

their advantage? These problems become the focus topic of all Chinese medicine researchers. With the development of the modern methodology and the numerical TCM, the technology of data storage and data mining become impossible. Therefore, the exploring for the relationship between NIHSS and TCM-SSASD is not only important in the theory researches but also have a wise application foreground.

BP (Back Propagation) neural network is a classic multi-level forward neural network. It is welcome by many researchers in the intelligent control field for its simple principle and distinguished approximation function to the nonlinear system. In the concerned problem of TCM, BP neural network can be applied to syndromes diagnosis [5-6], Chinese medicine finger-print chromatogram[7], detection of non contact life parameter signals[8], The analysis and predict for the Chinese medicine properties[9], the rule of prescription compatibility[10], and etc. However, there is no report about the data mining between NIHSS and TCM-SSASD.

In this paper, BP intelligent algorithm is used to approximate the nonlinear relationship between NIHSS and TCM-SSASD based on mass clinic data and information. Through standardization, combination and optimizing, the apoplexy syndromes can be standardization. It will improve the diagnose accuracy of the clinic syndrome, and then perfect the clinic curative effect evaluation system of TCM.

The concept of multiple models comes from the industry process control. In different manufacture environment, while the structure and parameters of the model varying, the processing control model can be regarded as multiple models [11]. In this paper, this concept is induced to approximate the TCM information by multiple nonlinear models and is expected to achieve a better result.

The structure of the paper is as follows: firstly, the background and problem will be introduced. Secondly, NIHSS and TCM-SSASD will be standardized and combined. Based on it, three kinds of BP neural networks will be built to approximate the relationship of them. The algorithms and improving schemes will be presented in the third sections, including the design of four BP algorithms and two multiple models schemes. In the fourth section, a simulation result will be given out based on clinic data. At last, the merit and defect are summarized by comparing all kinds of algorithms in the conclusion section.

2 Design of Neural Network

2.1 Clustering and Standardize of the NIHSS and TCM-SSASD

The data information comes from Dongzhimen hospital, Xiyuan hospital, Chaoyang hospital and Guanganmen hospital. It includes 422 apoplectics form November, 2003 to March, 2006. According to [4], the data is cleaned up and standardized, and a new table is come out. It includes a western medicine value varying form 0 to 35, and six basic TCM Syndromes, which include wind Syndrome, fire Syndrome, phlegm Syndrome, static blood Syndrome, qi vacuity Syndrome, preponderance of yang due to deficiency of yin syndrome, which varying form 0 to 30 (static blood Syndrome, especially, 0-35). The table is as follows:

Table 1. The value of NIHSS and TCM for apoplectics

ID	Western	TCM value					
	medicine Value	Wind	Fire	Phlegm	Static blood	Qi vacuity	Preponderance of yang due to deficiency of yin
1	10	5	13	20	0	4	2
2	5	12	4	13	9	0	1
...
421	2	0	7	0	1	0	14
422	12	12	4	8	8	0	0

2.2 Building of Neural Network

2.2.1 Basic Model

Based on the above table, many kinds of methods are trial and error in search of the relationship between NIHSS and TCM-SSASD. It is shown that there is no a distinct linear relationship between them [1]. Therefore, we suppose that there is a complex nonlinear relationship between them. Thus, we build a neural network to approximate the nonlinear relationship between the six syndromes and western medicine, whose input of the network is the value of the TCM six basic syndromes, and the output is the value of the NIHSS. It is as follows:

$$y = f(x_1, x_2, x_3, x_4, x_5, x_6), \quad (1)$$

where $x_1, x_2, x_3, x_4, x_5, x_6$ are the independent variable, also the six inputs of the neural network, which present the six Syndromes in TCM, including wind Syndrome, fire Syndrome, phlegm Syndrome, static blood Syndrome, qi vacuity Syndrome, preponderance of yang due to deficiency of yin syndrome. y is the output of the neural network, which is the value of the NIHSS. Since there is a hidden level in the forward network, it can be applied widely to classification and diagnose in the medicine field. At the same time, for the property of approximation to an arbitrarily nonlinear function, the neural network can be used to build a model for the nonlinear system.

At first, a 3 layers BP neural network is built, which include a input layer with 6 neurons, a hidden layer with 8-60 neurons and a output layer with 1 neurons. It is as follows:

In the training process, 277 apoplectics are chosen as sample. The input and output in the sample are standardized, which means the value of the TCM six syndromes are divided by 30 (static blood Syndrome divided by 35, especailly) and the value of NIHSS is also divided by 30. Through many experiments, it is found that the

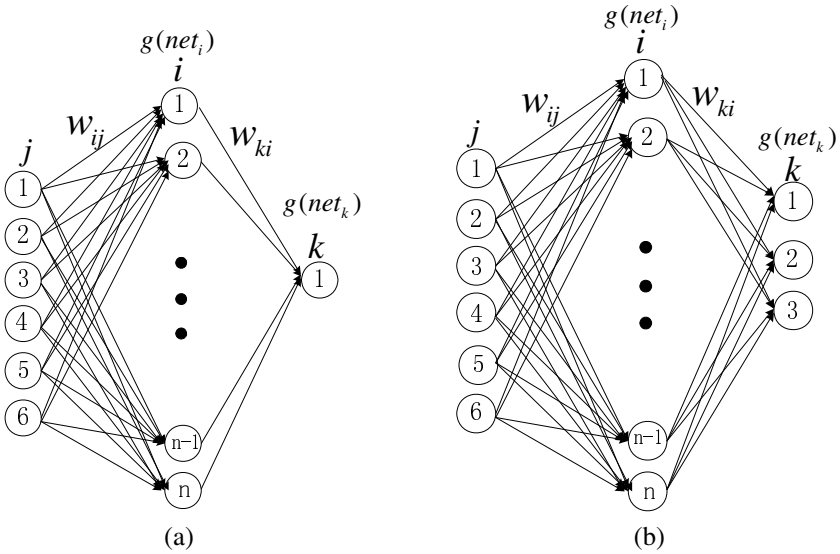


Fig. 1. Three levels BP neural network with single output (a) and multiple outputs (b)

convergence speed is most quickly and the performance is most perfectly while the number of the hidden layer is 10. Here the activation function is adopted as sigmoid function and hyperbolic tangent function.

2.2.2 The Improvement of the Network Model

Model 1: MIMO neural network with a hidden layer. While mass data is input into the neural network, there is much noises and disturbance, which results in the errors between the actual value and the network output. If we needn't the exact information about the value of the NIHSS, and only want to get a distinguishing among the states of an illness, such as light, moderate and severe, the structure of the neural network is simpler, and the error rate can be reduced accordingly. Here we transformed the value of NIHSS into three states: light, moderate and severe as three neurons. If the state of an illness is light, then the output of the network is $(1, 0, 0)$; if the state of an illness is moderate, then the output of the network is $(0, 1, 0)$; if the state of an illness is severe, then the output of the network is $(0, 0, 1)$. The input is not changed, but y becomes a vector in (1). The number of the hidden layer is 10. The structure of this neural network is shown in Fig 1 (b).

Model 2: A neural network with four layers. In generally, if the neurons in the hidden layer are sufficient enough, then the network can approximate an arbitrary nonlinear function. However, considering the complex nonlinear data come from the TCM clinic practice, we choose 4 layers neural network aiming to get a better convergence

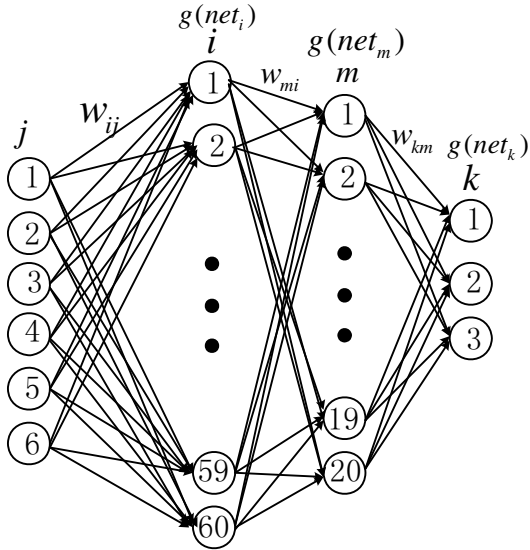


Fig. 2. Four layers BP neural network with multiple outputs

speed and robustness performance. There are two layers in the hidden layer, the first layer has 60 neurons and the second one has 20 neurons. It is shown as fig. 2.

3 Choosing of Algorithms

The forward BP multiple layers neural network has excellent generalization ability and interpolation function. Based on the general BP algorithms, we can choose different algorithms to satisfy different demand, such as convergence speed, memory capability, convergence precision, and so on. Here we choose four algorithms, including GD algorithm (gd), quasi-Newton BP algorithm (bfg), resilient BP algorithm (rp) and Levenberg-Marquardt BP algorithm (lm). The comparison of these algorithms in the TCM diagnose is in the next section.

The simulation indicates that if a fixed number sample arbitrarily in the database is chosen the training sample, different neural network would be built every time. And the accuracy rate of the diagnose ranges from 50% to 78%. The reason lies in two case: 1) The data comes from the clinic open information collection system. There is no strict limitation for the enrolled apoplectics. Since the curing time, age, sex, medical history, diagnose of the apoplectics are different, the values of NIHSS and TCM-SSASD are different too. Such as the value of wind Syndrome, phlegm Syndrome, static blood Syndrome for the ischemic apoplexy within the first three days are always high, but after two weeks, the value of static blood Syndrome and qi vacuity Syndrome for the sufferers rises to a high level. This means that different curing time and the basic of the treatment may result the instability of the model of the neural networks and great errors.

Therefore, if we mix all the different information together arbitrarily, it may destroy the structure of the neural networks in the weight modification process and the useful information may be losing. The weight will be averaged. To avoid the above phenomenon, two schemes are adopted:

Scheme 1: The average estimation by multiple models. Arbitrarily extract a fixed number of samples for n times. Then n neural networks are built.

Definition:

$$\Psi = \{f_i(x); i = 1, 2, \dots, n\}, \quad (2)$$

which presents a model set with the $f_i(x)$ as element. $f_i(x)$ can represent a controlled subject in the industry process control, which can be approximated by the neural networks as:

$$f_i(x) = f(x) + e_i(x), i = 1, 2, \dots, n, \quad (3)$$

where $f_i(x)$ is the i th estimation by neural networks, which will approximate the actual TCM diagnose nonlinear system. $e_i(x)$ is the error of the estimation, which is supposed as i.i.d. random variables with zero expectation $i = 1, 2, \dots, k$.

Choose the average value of the n models f_s as the final estimation of the multiple models:

$$f_s = \frac{f_1(x) + f_2(x) + \dots + f_n(x)}{n} = f(x) + \frac{e_1(x) + e_2(x) + \dots + e_n(x)}{n}, \quad (4)$$

where the error drops to $\frac{1}{n}$ of the original one, which improves the performance of the estimation. The principle figure is as Fig. 3.

Scheme 2: Feature multiple models estimation. From above, it is shown that all apoplextics can be divided into different groups with different feature. Such as: from the diagnose point of view, there are ischemic apoplexy suffers and cerebral hemorrhage suffers; for the curing time, there are suffers cured within 0-1 day, 2-3 days, 6-8 days and 12-16 days; from the sex point of view, there is female and male. These data with distinct features cover all the field of the data sets. Therefore, we can build a independent feature model for every kinds of suffers groups with different features.

Firstly, divide the data set X into k subsets Ω_i with individuation feature, i.e.,

$$X = \{x_i : x_i \in \Omega_i, i = 1, \dots, k\}, \quad (5)$$

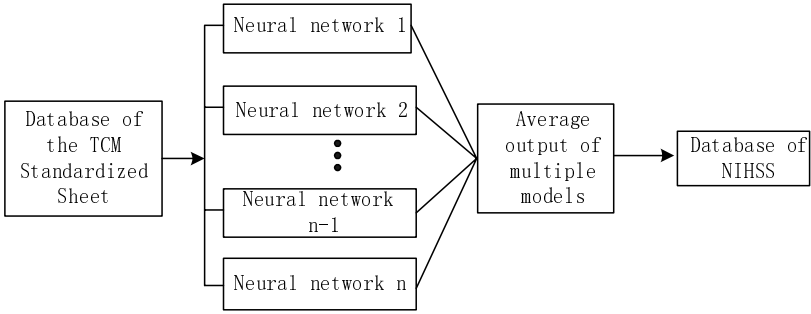


Fig. 3. Scheme of average estimation by multiple models

where X is the overall data set, x_i is the element of the i th feature subset Ω_i . For every subset Ω_i , k models named $f_i(x_i)$ are built. In the diagnose process, if the suffer $x \in \Omega_j$, then the j th neural network model $f_j(x_i)$ is adopted to diagnose. The principle figure is as Fig. 4.

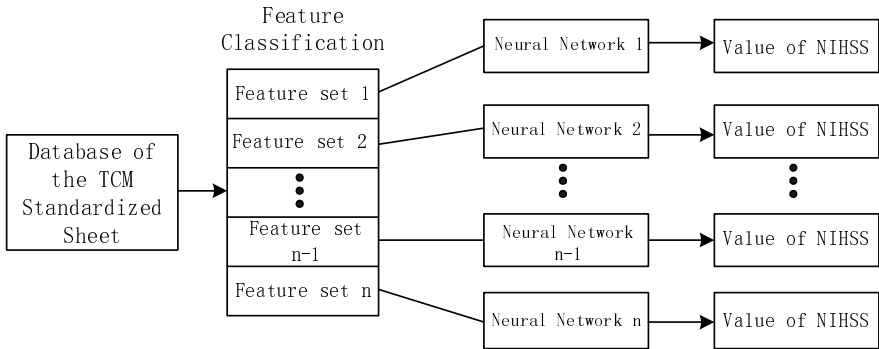


Fig. 4. Feature multiple models scheme

Remark: In the building process of $f_i(x_i)$, scheme 1 can be adopted, which would average the multiple building models and reduce the estimation errors.

4 Simulation

In the BP neural network, we adopt the six Syndrome value in the TCM-SSASD as the input and the value in NIHSS or the corresponding 3 states of illness as the output. There are one or two hidden layers in the network and 6-60 neurons within it. 277 suffers are chosen as the training samples and 145 ones as checking samples. Four algorithms are adopted, including gd, bfg, rp and lm to approximate the nonlinear relationship of the NIHSS and TCM-SSASD. For the multiple models scheme 1, one

thousand models are used to build an average one. The result can be seen in the following table 2. It is shown that one hidden layer with 10 neurons is sufficient enough to approximate the nonlinear relationship. The LM algorithms bring forth a better performance and quickly convergence speed.

Secondly, based on the idea of multiple models scheme 2, we divide all of the apoplectics into 4 parts. The first part is cured within 0-1 day, and the second one within 2-3 days, the third one within 6-8 days, and the fourth one within 12-16 days. Thus 4 neural networks are built, and every one is build by the scheme 1. The result is given in the table 2. It is shown that feature multiple models can reflect the local feature information of the samples and the diagnose identification rate is higher than the general one.

Table 2. Identification rate of the neural networks based on NIHSS and TCM-SSASD

	Training samples / Checking samples	Identification of the general neural network	Identification of the average multiple models	Identification of the feature multiple models
BP(gd)	277 / 145	66.67%	67.67%	70%
BP(bfg)	277 / 145	65%	66.5%	68%
BP(rp)	277 / 145	66.67%	68.1%	68.4%
BP(lm)	277 / 145	66.67%	67.2%	69%

5 Conclusion

In this paper the relationship between NIHSS and TCM-SSASD is studied based on the data mining method. The forward BP neural network is adopted to approximate the complex nonlinear relationship with four algorithms. The definition of the multiple models in the industry control process is induced and two multiple models scheme are designed to improve the approximation performance, including average multiple models and feature multiple models. The simulation illustrates the validation of the design scheme. It is also shown that there is much coexistence information in the Western medicine and the TCM.

In the feature models building, we only classify the data by the curing time. Other features, such as sex, age, syndrome, may be used to classify the apoplectics in the future work.

References

1. Wang, Z.H.: The Study on the Classification Standards of TCM Apoplexy Syndromes Based on the Clustering Analysis of TCM Standardized Sheet of Apoplexy Syndromes Diagnosis. Beijing Jiaotong University, The Institute of Advanced control system, Research report ACSL_06001. www.acsl.bjtu.edu.cn
2. The Scientific Research Group of Apoplexy Syndrome and Clinic Diagnose: The Clinic Study on the Criterion for Diagnose and Differentiation of Syndromes of Stroke. Journal of Beijing University of TCM 6 (1994) 41-43

3. Hu, J.L., Li, J.S., Yu, X.Q.: The Background and Status in Quo of Criterion for the Diagnose of TCM Syndromes. *Journal of Henan University of Chinese Medicine* **20** (11) (2005) 77-79
4. Hou, F.G., Zhao, G., He, X.M.: The Summarize for the Methodology in the Criterion for Diagnose of Syndromes Quantification. *Shanxi Chinese Medicine* **26** (5) (2005) 473-475
5. Li, J.S., Hu, J.L., Yu, X.Q., Wang, M.H., Wang, Y.Y.: The Study on the Syndromes Diagnose by Radial Basic Function Neural Network Based on the Clustering Data Mining. *Chinese Journal of Basic Medicine in Traditional Chinese Medicine* **11** (9) (2005) 685-687
6. Wang, X.W., Zhai, H.B., Wang, J.: A Quantification Diagnose Method for Chinese Medicine Based on the Data Mining. *Journal of Beijing University of TCM* **28** (1) (2005) 4-7
7. Liu, B.: The Application of Neural Network in the Field of Chinese Medicine. *The Journal of Chinese medicine* **5** (2003) <http://www.100md.com/html/Dir/2003/05/12/9132.htm>
8. Yang, D., Wang, J.Q., Jing, X.J., Mi, A.S., Lu, G.H.: Detection of Noncontact Llife Parameter Signals by BP Neural Network. *Journal of the Fourth Military Medical University* **25** (14) (2004) 1249-1251
9. Liu, H.M.: Forecasting the Extraction of Coumarins in *Angelica dahurica* by Supercritical CO₂ with BP Neural Network. *Shizhen Chinese medicine* **4** (2) (2006) 176-178
10. Wei, X.H., Wu, J.J., Liang, W.Q.: Application of an Artificial Neural Network in the Design of Sustained-Release Dosage Forms. *ACTA PHARMACEUTICA SINICA* **36** (9) (2001) 690-694
11. Narendra, K.S., Balakrishnan, J., Ciliz, M.K.: Adaptation and Learning Using Multiple Models Switching and Tuning. *IEEE Control System Magazine* **15** (3) (1995) 37-51

Application of Back-Propagation Neural Network to Power Transformer Insulation Diagnosis

Po-Hung Chen¹ and Hung-Cheng Chen²

¹ St. John's University, Department of Electrical Engineering,
Taipei, Taiwan, 25135, R.O.C.
phchen@mail.sju.edu.tw

² National Chin-Yi University of Technology,
Department of Electrical Engineering, Taichung, Taiwan, 411, R.O.C.
hcchen@chinyi.ncit.edu.tw

Abstract. This paper presents a novel approach based on the back-propagation neural network (BPNN) for the insulation diagnosis of power transformers. Four epoxy-resin power transformers with typical insulation defects are purposely made by a manufacturer. These transformers are used as the experimental models of partial discharge (PD) examination. Then, a precious PD detector is used to measure the 3-D (ϕ -Q-N) PD signals of these four experimental models in a shielded laboratory. This work has established a database containing 160 sets of 3-D PD patterns. The database is used as the training data to train a BPNN. The training-accomplished neural network can be a good diagnosis system for the practical insulation diagnosis of epoxy-resin power transformers. The proposed BPNN approach is successfully applied to practical power transformers field experiments. Experimental results indicate the attractive properties of the BPNN approach, namely, a high recognition rate and good noise elimination ability.

1 Introduction

More than half of the breakdown accidents of power apparatuses are caused by insulation deterioration. The reliability of a power apparatus is affected significantly by the presence of insulation defects. Partial discharge (PD) pattern recognition has been regarded as an important diagnosis method to prevent power apparatuses from malfunction of insulation defect. However, the measurement of PD needs heavy power equipment and precious instruments. The Heavy Power Lab in St. John's University installed a set of precious instrument (Hipotronics DDX-7000 Digital Discharge Detector). Therefore, the school has the capability of doing insulation diagnosis related researches.

Power transformer is one of the most important apparatuses in a power delivery system. Breakdown of a power transformer can cause an interruption in electricity supply and result in a loss of considerable profits [1]. Therefore, detecting insulation defects in a power transformer as early as possible is of priority concern to a power transformer user. PD phenomenon usually originates from insulation defects and is an important symptom to detect incipient failures in power transformers. Classification of different types of PD

patterns is of importance for the diagnosis of the quality of power transformers. PD behavior can be represented in various ways. Because of the randomization of PD activity, one of the most popular representations is the 3-D (φ -Q-N) distribution [2], [3], i.e., the PD pattern is described using a pulse count N versus pulse height Q and phase angle φ diagram. Previous experimental results have adequately demonstrated that φ -Q-N distributions are strongly dependent upon PD sources, therefore, the 3-D patterns can be used to characterize insulation defects [4]. This provides the basis for pattern recognition techniques that can identify the different types of defects.

Previous efforts at PD pattern recognition have applied various identification techniques to make the problem solvable. Various pattern recognition techniques such as fuzzy clustering [5], expert system [6], extension theory [7], [8], and statistical analysis [9] have been proposed. These techniques have been successfully applied to PD pattern recognition. However, these conventional approaches not only require human experiences but also have some difficulties in acquiring knowledge and maintaining the database of decision rules. In recent years, a biologically artificial intelligence technique known as artificial neural network has emerged as a candidate for the pattern identification problem. The neural network can directly acquire experience from the training data to overcome the shortcomings of the conventional approaches. The neural network can quickly and stably learn to categorize input patterns and permit adaptive processes to access significant new information [10].

In this paper, a novel insulation diagnosis approach based on the PD and back-propagation neural network (BPNN) is proposed for epoxy-resin power transformers. Four cast-resin transformers with typical insulation defects, which were purposely made by a manufacturer, are used as the models of the PD examination. The DDX-7000 Digital Discharge Detector is then used to measure the 3-D PD signals of these transformers. So far, this work has established a database which contains 160 sets of 3-D PD patterns. Then, this database is used as the training data to train a BPNN. Finally, the training-accomplished neural network can be a good diagnosis system for the practical insulation diagnosis of epoxy-resin power transformers. Experimental results show that different types of insulation defects within power transformers are identified with rather high recognition rate.

2 Partial Discharge Experiments

2.1 Partial Discharge Measurement

PD is a forced phenomenon occurring in insulation parts in a power apparatus. When the intensity of electric field exceeds the breakdown threshold value of a defective dielectric, PD occurs and results in a partial breakdown in the surrounding dielectrics. PD is a symptom of insulation deterioration. Therefore, PD measurement and identification can be used as a good insulation diagnosis tool to optimize both maintenance and life-risk management for power apparatuses.

The new standard IEC60270 [11] for PD measurement has been published in 2001, which establishes an integral quality assurance system for PD measurement instead of the old standard IEC60060-2 published in 1994. The standard IEC60270 ensures accuracy of measuring results, comparability and consistency of different instruments

and measuring methods. Moreover, the new standard provides digital PD measuring recommendations as well as the analog measuring. In this work, all PD experiments are based on the new standard IEC60270.

A PD experiment laboratory, including a set of precious instrument (Hipotronics DDX-7000 Digital Discharge Detector), has been set up in the St. John's University

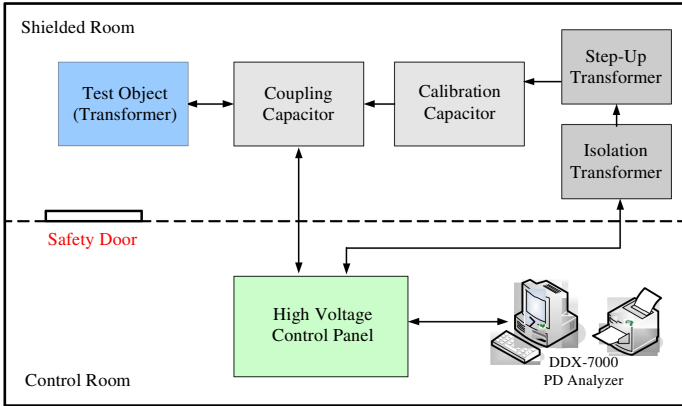


Fig. 1. Block diagram of PD experiment



Fig. 2. Practical PD field measurement



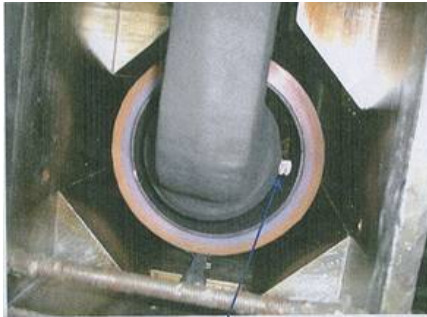
Fig. 3. PD analyzer and control panel

according to the standard IEC60270 recommendations. Fig. 1 shows the block diagram of the PD experiment laboratory. The constitution of the PD experiment laboratory includes a PD analyzer, a high-voltage control panel, an isolation transformer, a high-voltage generator (step-up transformer), a calibration capacitor, and a coupling capacitor. Fig. 2 shows a practical PD field measurement in the shielded room. Fig. 3 shows the PD analyzer and high-voltage control panel.

2.2 Experimental Models

In this work, the experimental objects are power transformers which use epoxy-resin as HV insulation materials. The rated voltage and capacity of the transformers are 12 kV and 2kVA, respectively. For testing purposes, four experimental models of power transformers with typical insulation defects were purposely manufactured by a power apparatuses manufacturer. These typical PD models include a low-voltage coil PD (Type A), a high-voltage coil PD (Type B), a high-voltage corona discharge (Type C), and a healthy transformer (Type D). Fig. 4 shows the pictures of these four experimental models.

The voltage step-up procedure of the PD experiment, according to the standard IEC60270 recommendations, is shown in Fig. 5. First, the high-voltage generator generates a rising-voltage from 0V to 18kV, which is the 1.5 times of the rated voltage, in 50 seconds. This high-voltage will be maintained for 1 minute to trigger discharges.



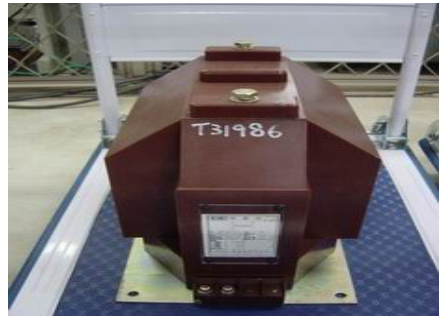
(a) Low-voltage coil PD (Type A).



(b) High-voltage coil PD (Type B).



(c) High-voltage corona discharge (Type C).



(d) Healthy transformer (Type D).

Fig. 4. Four experimental models with typical insulation defects

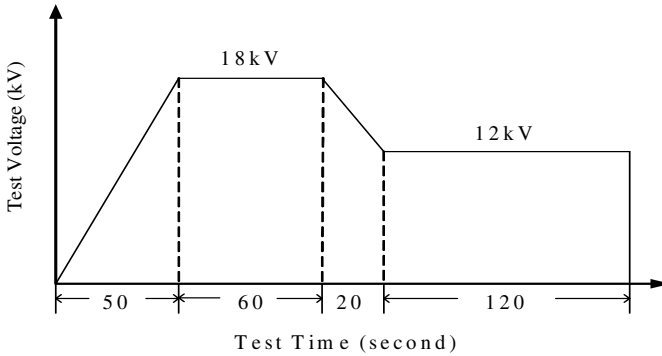


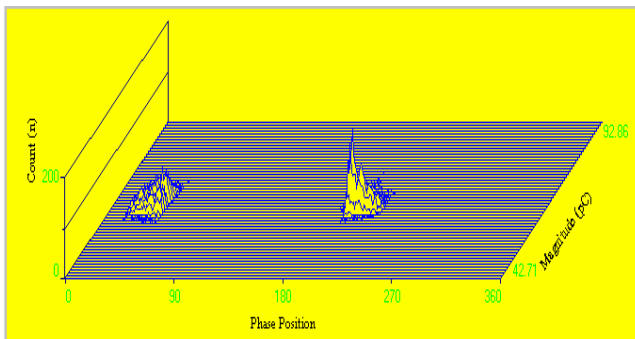
Fig. 5. Voltage step-up procedure

Then, the voltage will descend from 18kV to the rated voltage in 20 seconds. The 12kV rated voltage will be kept and the PD detector starts to measure and record the PD signals for 2 minutes. During the experimental process, all of the measuring analog data are converted to digital data in order to store them in a computer.

3 BPNN Solution Methodology

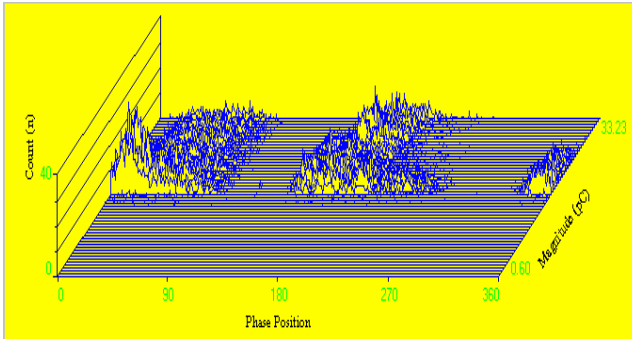
In recent years, a biologically artificial intelligence technique known as artificial neural network (ANN) has emerged as a candidate for the feature identification problems. The basic conception of ANN is intended to model the behavior of biological neural functions. An ANN is generally modeled as a massively parallel interconnected network of elementary neurons. The original desire for the development of ANN is intended to take advantage of parallel processors computing than traditional serial computation.

Fig. 6 shows the PD patterns of four typical defects. As shown in Fig. 6, obviously, the problem of transformer insulation diagnosis is essentially a PD patterns classification problem. Therefore, the ANN is an applicable solution tool to the problem of transformer insulation diagnosis.

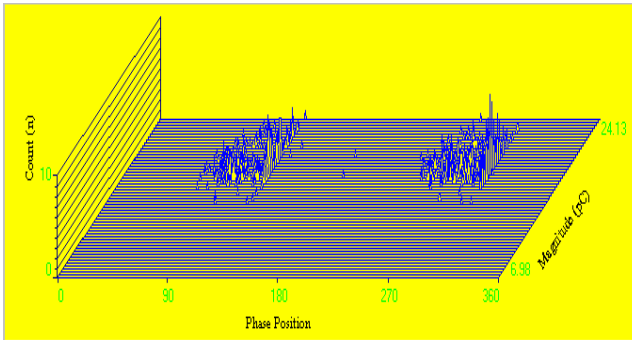


(a) Low-voltage coil PD (Type A).

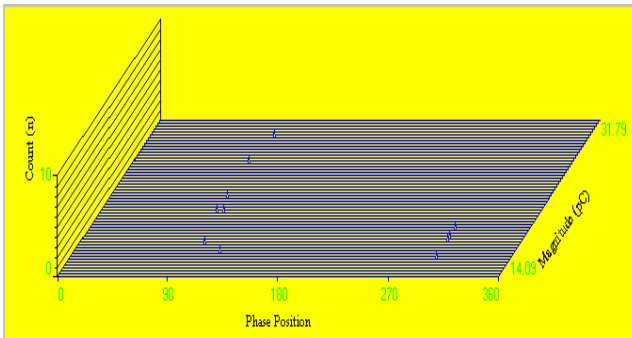
Fig. 6. Typical PD patterns of four defect types



(b) High-voltage coil PD (Type B).



(c) High-voltage corona discharge (Type C).



(d) Healthy transformer (Type D).

Fig. 6. (cont.): Typical PD patterns of four defect types

From the literature survey, several models and learning algorithms of ANN have been proposed for solving the patterns classification problems [12]. In this paper, we establish a triple-layer feed-forward BPNN, as shown in Fig. 7, for solving the PD patterns classification problem. The number of neurons in the output layer is set at the number of defect types. The input data for the BPNN is the field measuring 3-D PD pattern. To fit the form of the input layer and accelerate convergence, each original PD

pattern is pre-translated into a 1600x1 matrix. Then, the only control parameter is the number of neurons in the hidden layer.

In this work, the transfer function in the hidden layer is the hyperbolic tangent function, as shown in Fig. 8 [12]. The transfer function in the output layer is the sigmoid function, as shown in Fig. 9 [12]. In the training procedure, a faster back-propagation learning algorithm named “RPROP algorithm” is used as the learning rule. Riedmiller and Braun [13] showed that both convergence speed and memory requirement of the RPROP algorithm are better than traditional gradient-descent learning algorithms.

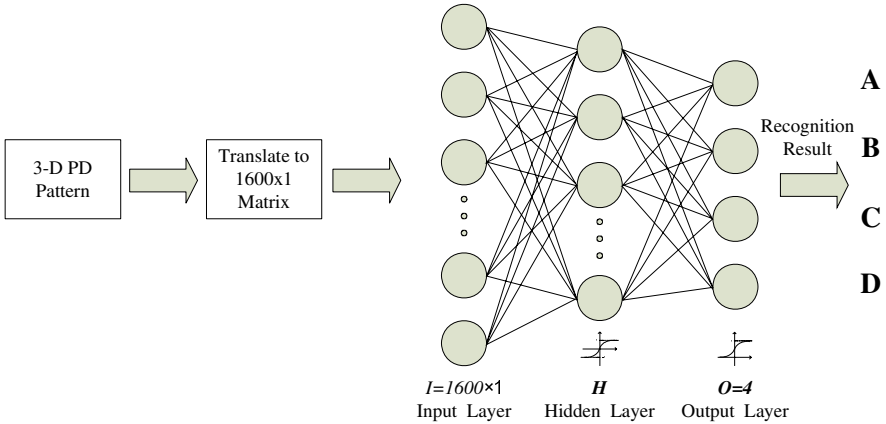


Fig. 7. Structure of the BPNN insulation diagnosis system

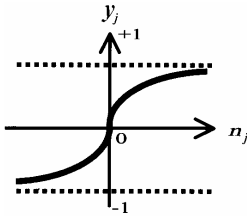


Fig. 8. Hyperbolic tangent transfer function

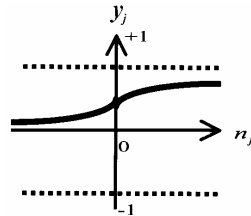


Fig. 9. Sigmoid transfer function

4 Experimental Results

The proposed BPNN approach was implemented on a MATLAB software and executed on a Pentium IV 2.8GHz personal computer. To illustrate the identification ability of the proposed approach, 160 sets of field measuring PD patterns are used to test the BPNN diagnosis system. The diagnosis system will randomly choose 80 sets of data as the training data, and the rest as the testing data. The structure of the proposed BPNN insulation diagnosis system is shown in Fig. 7. The number of neurons in the output layer (O) depends on the defect types to be identified, which is $O=4$ in this work. Since the original 3-D PD pattern is translated into a 1600x1 matrix in this work, the

number of neurons in the input layer (I) is set at $I=1600$. Then, for a multi-layer BPNN, the main control parameter is the number of neurons in the hidden layer (H).

Table 1 shows the test results of the proposed BPNN diagnosis system with different number of neurons in the hidden layer from $H=20$ to $H=100$. Test results show that $H=60$ has the highest recognition rate of 90%, 90%, 93%, and 80% for defect type A, B, C, and D, respectively. The proposed system achieves a high average recognition rate of 88.25%.

In a practical PD field measurement, the obtaining data would unavoidably contain some noise. The sources of noise include instrumental noise and environmental noise. This work takes noise into account in order to study the noise tolerance ability of the proposed approach. In this example, 240 sets of noise-contained testing data are generated by adding $\pm 10\%$, $\pm 20\%$, and $\pm 30\%$, respectively, of randomly distributed noise into the training data to test the BPNN diagnosis system. Table 2 shows the test results with different percentage of noise. As shown in Table 2, the descending of recognition rate generally follows the raising of noise percentage, a finding that is consistent with general expectations. However, experimental results show a good tolerance to noise interference of the proposed approach, which achieves a high recognition rate of 81.75% in noise of 20% and 75.75% in extreme noise of 30%.

Table 1. Recognition rate with different neurons in the hidden layer (H)

H	Defect Type				
	A	B	C	D	Average
20	80%	88%	85%	80%	83.25%
40	88%	88%	90%	80%	86.5%
60	90%	90%	93%	80%	88.25%
80	88%	80%	85%	77%	82.5%
100	88%	80%	77%	73%	79.5%

Table 2. Recognition rate with different percentage of noise

	Percentage of Noise			
	0%	$\pm 10\%$	$\pm 20\%$	$\pm 30\%$
Average Recognition Rate	88.25%	86.5%	81.75%	75.75%

5 Conclusions

This paper presents a new methodology based on the back-propagation neural network for solving the insulation diagnosis problem of epoxy-resin power transformers. The

proposed approach is considered a general tool because it can be easily implemented on the popular MATLAB software. Moreover, the approach is considered a flexible tool because it can be also applied to other high-voltage power apparatuses such as current transformer (CT), potential transformer (PT), cable, and rotation machine. Experimental results indicate that the proposed approach has a high degree of recognition accuracy and good tolerance of noise interference. We expect this work providing useful reference to electric power industry.

Acknowledgment

Financial support given to this research by the National Science Council of the Republic of China, Taiwan, under grant No. NSC 93-2213-E-129-015 is greatly appreciated.

References

1. Feinberg, R.: *Modern Power Transformer Practice*. John Wiley & Sons (1979)
2. Gulski, E., Burger, H.P., Vaillancourt, G.H., Brooks, R.: PD Pattern Analysis During Induced Test of Large Power Transformers. *IEEE Trans. Dielectrics and Electrical Insulation* **7** (2000) 95–101
3. Kim, C.S., Kondo, T., Mizutani, T.: Change in PD Pattern with Aging. *IEEE Trans. Dielectrics and Electrical Insulation* **11** (2004) 13–18
4. Krivda, A.: Automated Recognition of Partial Discharge. *IEEE Trans. on Dielectrics and Electrical Insulation* **2** (1995) 796–821
5. Tomsovic, K., Tapper, M., Ingvarsson, T.T.: A Fuzzy Information Approach to Integrating Different Transformer Diagnostic Methods. *IEEE Trans. on Power Delivery* **8** (1993) 1638–1643
6. Satish, L., Gururaj, B.I.: Application of Expert System to Partial Discharge Pattern Recognition. in *CIGRE Study Committee 33 Colloquium, Leningrad, Russia, (1991) Paper GIGRE SC 33.91*
7. Wang, M.H., Ho, C.Y.: Application of Extension Theory to PD Pattern Recognition in High-Voltage Current Transformers. *IEEE Trans. Power Delivery* **20** (2005) 1939–1946
8. Wang, M.H.: Partial Discharge Pattern Recognition of Current Transformers Using an ENN. *IEEE Trans. Power Delivery* **20** (2005) 1984–1990
9. Cavallini, A., Montanari, G.C., Puletti, F., Contin, A.: A New Methodology for the Identification of PD in Electrical Apparatus: Properties and Applications. *IEEE Trans. Dielectrics and Electrical Insulation* **12** (2005) 203–215
10. Salama, M.M.A., Bartnikas, R.: Determination of Neural Network Topology for Partial Discharge Pulse Pattern Recognition. *IEEE Trans. Neural Networks* **13** (2002) 446–456
11. IEC: *High-Voltage Test Techniques-Partial Discharge Measurements*. IEC 60270 (2001)
12. Wasserman, P.D.: *Neural Computing, Theory and Practice*. Van Nostrand Reinhold (1989)
13. Riedmiller, M., Braun, H.: A Direct Adaptive Method for Faster Back Propagation Learning- The RPROP Algorithm. *IEEE International Conference on Neural Networks* **1** (1993) 586–591

Momentum BP Neural Networks in Structural Damage Detection Based on Static Displacements and Natural Frequencies

Xudong Yuan, Chao Gao, and Shaoxia Gao

Civil Engineering Institute of Dalian Fisheries University, Postfach 116024,
Dalian, China
yxd0101@163.com

Abstract. Modeling error, measured noises and incomplete measured data are main difficulties for many structural damage processes being utilized. In this study, using static displacements and frequencies constitutes the input parameter vectors for neural networks. A damage numerical verification study on a five-bay truss was carried out by using an improved momentum BP neural network. Identification results indicate that the neural networks have excellent capability to identify structural damage location and extent under the conditions of limited noises and incomplete measured data.

1 Introduction

Structural health monitoring has become increasingly an important research topic in connection with damage assessment and safety evaluation of existing structures. At present, there are finite element method and non-linear mapped technique based on neural networks in this field [1]. Neural networks have unique capability to be trained to recognize given patterns and to classify other untrained patterns. Hence, the neural networks can be used for estimating the structural parameters with proper training.

Recently, many researchers have dealt with neural network approaches for damage estimation of many kinds structural models. Wu et al. applied the neural networks to recognize the locations and the extent of individual member damage of simple three story frame [2]. Szewczyk and Hajela proposed a modified counterpropagation neural network to estimate the stiffness of individual structural elements based on observable static displacements under prescribed loads [3]. Tsou and Shen identified damage of a spring-mass system with eight degrees-of-freedom [4]. Yoshimura proposed the Generalized Space Lattice transformation of training patterns to relax the illposedness of the problem [5]. Xu et al. proposed a structural dynamic updating method using neural networks techniques [6]. Lu et al. have used displacement mode and strain mode data as input-patterns to the neural network, and compared the identification accuracy of the location and extent of structural damage [7]. Wang et al. have used combined parameters, which be merged with natural frequencies and mode shape data at a few selected points, as input vectors for neural networks, and studied the location and quantification of frame connection damage [8].

The effects of structural damage identification depend on the accuracy and completeness of measured data. At present, due to the limit of test techniques and the influence of environments, measured data is usually incomplete, and included some disadvantage factors, such as measured noises and model errors [9]. How to overcome the influences of disadvantage factors and improve identification accuracy is a difficult issue in the area of structural damage identification. In order to identify the location and extent of structural damage, measured static displacements at partial points and several low natural frequencies have been used as input parameters vectors for neural networks in this study. An improved momentum BP neural network was adopted. A numerical analysis of structural damage identification on a five span truss was presented. Random noises were artificially added during training process, the robustness of the neural network is effectively increased.

2 Damage Theory Analysis

For health structure, the static response equation and the eigenvalue equation can be expressed by

$$Ku = p \quad (1)$$

$$(K + \omega_j^2 M)\varphi_j = 0 \quad (2)$$

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 mode shape vector. Let the change of stiffness matrix caused by structural damage be defined as ΔK , thus, Eq.(1) and (2) can be rewritten as

$$(K + \Delta K)u^* = p \quad (3)$$

$$(K + \Delta K + \omega_j^2 + \Delta\omega_j^2)M(\varphi_j + \Delta\varphi_j) = 0 \quad (4)$$

u^* is damaged displacement vector and defined as

$$u^* = (K + \Delta K)^{-1} p \approx (K^{-1} - K^{-1}\Delta K K^{-1})p \quad (5)$$

The change of displacement vector caused by damage is defined as

$$\Delta u = u - u^* \approx K^{-1}\Delta K K^{-1}p \quad (6)$$

The change of natural frequency is defined as

$$\Delta\omega_j^2 \approx \varphi_j^T \Delta K \varphi_j / \varphi_j^T M \varphi_j \quad (7)$$

When FEM model is used, the change of global stiffness matrix can be expressed by the change of element stiffness matrix

$$\Delta K = \sum_{i=1}^{ne} \alpha_i k_i \quad (8)$$

α_i is the damage parameter of element stiffness, $-1 \leq \alpha_i \leq 0$, k_i is element stiffness ($i=1,2,\dots,ne$), ne is the number of structural elements.

The indicator of damage location detection presented in literature [11] was used as input vector for the neural network, when individual element damage or several element damages with same extent were occurred,

$$DS = \frac{\Delta u}{\Delta \omega_j^2} = \frac{K^{-1} \Delta K K^{-1} p}{\varphi_j^T \Delta K \varphi_j / \varphi_j^T M \varphi_j} = \frac{K^{-1} \sum_{i=1}^{ne} k_i K^{-1} p}{\varphi_j^T \sum_{i=1}^{ne} k_i \varphi_j / \varphi_j^T M \varphi_j} \quad (9)$$

From Eq. (9), we can see that α_j is eliminated. Thus, DS is just related to damage location and independent of damage extent

$$\{\text{input-vector}\} = \{ DS_1, DS_2, \dots, DS_m \} \quad (j=1, 2, \dots, m) \quad (10)$$

j is the serial number of corresponding nodes.

For structural damage extent detection, the terms of damage extent must be introduced into the input vector of damage location detection, namely

$$RNF = \frac{\Delta \omega_j^2}{\omega_j^2} \approx \varphi_j^T \Delta K \varphi_j / \varphi_j^T K \varphi_j = \varphi_j^T \sum_{i=1}^{ne} \alpha_i k_i \varphi_j / \varphi_j^T K \varphi_j \quad (11)$$

Eq. (11) is related to damage extent, thus, DS and RNF together constitute the input vector for the neural networks

$$\{\text{input-vector}\} = \{ DS_1, DS_2, \dots, DS_m, RNF \} \quad (12)$$

3 Numerical Example Analysis

A five-span plane truss model in Fig.1 is adopted to demonstrate the effectiveness of the proposed neural network approach. Material properties with elastic modulus is $2.1 \times 10^2 \text{Gpa}$, density is $7.9 \times 10^3 \text{kg/m}^3$, cross-sectional area is $1.0 \times 10^{-4} \text{m}^2$.

In order to avoid the case that individual element is insensitive to structural deformation under the single load path, two load cases shown in Tab.1 were selected in training process of the neural networks $P_1=3.0 \times 10^3 \text{N}$, $P_2=3.0 \times 10^3 \text{N}$, $P_3=1.0 \times 10^3 \text{N}$,

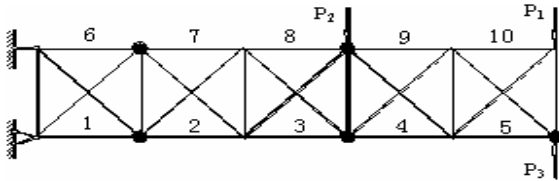


Fig. 1. Calculation analysis model of the truss

Table 1. Load cases of truss

Load case	Load combination
1	P_1
2	P_2, P_3

Since the neural network-based structural identification is highly dependent on the training patterns, the number of training patterns must be large enough to represent the structural system properly [12]. In order to reduce system identification scale, the damage detection of chord elements, which is the key member of the truss, were carried out in this study, the detailed cases are shown in Tab. 2.

X, Y direction displacements of five nodes, which are marked with black as shown in Fig.1, and the 3th natural frequency constitute the input vectors with 10 neurons. {output vector} = $\{o_1, o_2, \dots, o_{10}\}$, $0 \leq o_i \leq 1$, indicates damage probability of i th element.

Random noises are added to nodes static displacements used for combination of the training patterns. After noise added, the training patterns scales were expanded, and meanwhile the robust of the neural networks was enhanced. The neural network was trained by element damage with elastic modulus discount 50%. It was tested by element damage with elastic modulus discount 5% and 20%. The identification results indicate that although the random noises have effect on the identification accuracy, higher noises level may be existence. The identification errors were shown in Tab. 3. Fig.2 and 3 show the results of damage location detection on two elements with 10% noises. We can conclude that even if there are higher-level noises injections, the neural networks have the capability to identify damage location. Meanwhile, with the increase of damage extent; the identification accuracy is increased.

Identification parameters of damage extent were added to input vectors of damage location detection. In this study, the first three natural frequencies, which were selected, constitute identification parameters of damage extent. Thus, there are 13 neurons in input vectors of damage extent detection. {output} = $\{o_1, o_2, \dots, o_{10}\}$, $0 \leq o_i \leq 1$ indicates the i th element damage extent.

The neural networks were trained by element damage with elastic modulus discount 5%, 20%, 50%, and meanwhile, in order to extend the training patterns, there are 5% random noises added to the nodes displacements. The neural networks were tested with elastic modulus discount 30%, 70%. The results are shown in Tab.4, at the range of limited level noises, the noises have little effect on the identification errors, the identification accuracy depend on damage extent adopted during testing. When it's included in the range of damage extent adopted during training, identification accuracy is considerable higher; otherwise, identification errors will be very serious. Fig.4 and 5 indicate the identification results of damage extent with 10% noises. Although noises are existence, the damage extent identification for damaged member is quite accurate. For the condition of the serious identification errors on undamaged member, we can identify damage locations firstly, and then fixed on the damage location and extent by combination of the results of damage extent identification, finally the identification errors for undamaged member will be eliminated.

Table 2. Combination of train samples

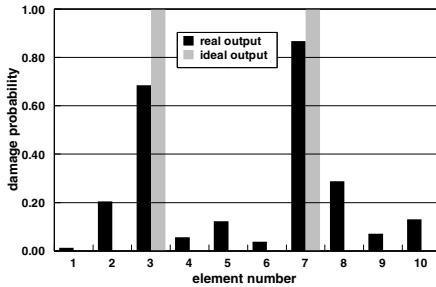
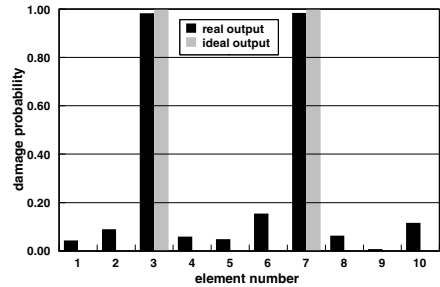
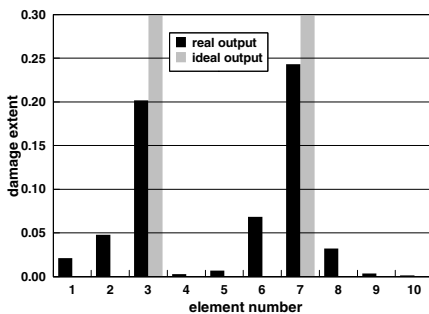
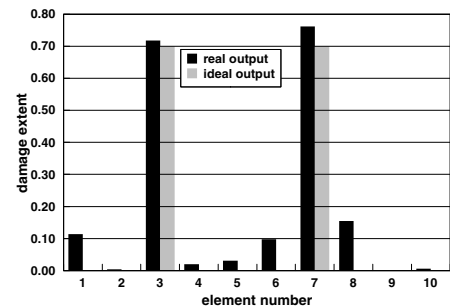
Load case	Damage element number	Damage extent
1	1, 2, 3, 6, 7, 8, 3&7, 2&8, 6&8	50%
2	4, 5, 9, 10	50%

Table 3. Identification tolerance of damage location of elements

Damage extent (%)	Noise level (%)	Identification error (%)
5	0	16.1
	5	17.3
	10	17.5
20	0	12.3
	5	12.7
	10	13.2

Table 4. Identification tolerance of damage extent of elements

Damage extent (%)	Noise level (%)	Identification error (%)
30	0	1.08
	5	1.15
	10	1.13
70	0	11.1
	5	11.4
	10	11.9

**Fig. 2.** Identification of damage location for the 3th and 7th element stiffness discount 5%**Fig. 3.** Identification of damage location for the 3th and 7th element stiffness discount 20%**Fig. 4.** Identification of damage extent for the 3th and 7th element stiffness discount 30%**Fig. 5.** Identification of damage extent for the 3th and 7th element stiffness discount 70%

4 Conclusion

In this study, one kind of the neural networks input parameters for structural damage identification, which was consisted of measured static displacements at partial nodes and several low natural frequencies, was presented. By the damage identification on a five-span truss, the identification results indicate that the measured noises have little effect on the neural networks. The robust of the neural networks is improved evidently. There are perfect effects on the damage location and extent identification for the truss structures.

If we identify structural damage extent directly, although the damage member can be identified, there will be serious identification errors, and several undamaged member are thought of as damaged; therefore, it's necessary to test the neural networks for damage location and extent identification simultaneity, thus the damaged member and its damage extent will be determined accurately. The formulations presented in this study were derived from continuum structural FEM, and have universality; so it should be also used for the other structural modes, and it remains to be validated for the future.

References

1. Qin, Q.: Health Monitoring of Long Span Bridges [J]. *China Journal of Highway and Transport*. **13**(2) (2000) 37-42
2. Wu, X., Ghaboussi, J., Garrett, J. J.: Use of Neural Networks in Detection of Structural Damage. *Comput Struct* **42**(4) (1992) 649-659
3. Szewczyk, Z., Hajela, P.: Neural Network Based Damage Detection in Structure. *ASCE Journal of Computing and Civil Engineering* **8**(2) (1994) 163-178
4. Tsou, P., Shen, M-HH.: Structural Damage Detection and Identification Using Neural Networks. *AIAA Journal* **32**(1) (1994) 176-183
5. Yoshimura, S., Matsuda, A.: New Regularization by Transformation for Neural Network Based Inverse Analyses and Its Application to Structure Identification. *International Journal of Numerical Methods in Engineering* **39** (1996) 3953-3968
6. Ma, H. W., Yang, G. T.: Basic Methods and Study Advances of Structural Damage Detection. *Advances in Mechanics* **29**(4) (1999) 513-525
7. Lu, Q. H., Li, D. B.: Neural Network Method in Damage Detection of Structures by Using Parameters from Modal Test. *Engineering Mechanics* **16**(1) (1999) 35-42
8. Wang, B. S., Ni, Y. Q., Gao, Z. M.: Input Parameters for Artificial Neural Networks in Frame Connection Damage Identification. *Journal of Vibration Engineering* **13**(1) (2000) 138-142
9. Wang, B. S., Ding, H. J.: Influence of Modeling Errors on Structural Damage Identification Using Artificial Neural Networks. *China Civil Engineering Journal* **33**(1) (2000) 50-55
10. Wen, X., Zhou, L.: *Neural Networks Application Design Based on MATLAB*. Beijing: Science Publishing Company (2000) 207-232
11. Wang, X., Hu, N., Fukunaga, Hisao., Yao, Z.H.: Structure Damage Identification Using Static Test Data and Changes in Frequencies. *Engineering Structures* **23** (2001) 610-621
12. YUN, C. B., Bahng, E. Y.: Substructure Identification Using Neural Networks. *Computers and Structures* **77** (2000) 41-52

Deformation Measurement of the Large Flexible Surface by Improved RBFNN Algorithm and BPNN Algorithm

Xiyuan Chen

School of Instrument Science and Engineering
Southeast University, Nanjing City, 210096, P.R. China
chxiyuan@seu.edu.cn, chxiyuan@263.net

Abstract. The Radial Basis Function (RBF) Neural Network (NN) is one of the approaches which has shown a great promise in this sort of problems because of its faster learning capacity. This paper presents the information fusion method based on improved RBFNN to deduce the deformation information of the whole flexible surface considering the complexity of the deformation of the large flexible structure. A distributed Strapdown Inertial Units (SIU) information fusion model for deformation measurement of the large flexible structure is presented. Comparing with the modeling results by improved RBFNN and back propagation (BP) NN, the simulation on a simple thin plate model shows that the information fusion based on improved RBFNN is effective and has higher precision than based on BPNN.

1 Introduction

In the field of deformation measurement for large workpiece surfaces or the large flexible structure, such as ship deck, building or railway etc., a common static deformation measurement technology for detecting global and local deformation is measuring the distance in a right angle between the workpiece and a reference plate [1]. This measurement strategy has some disadvantages such as low efficiency, artificial errors, low redundancy of the samples etc., so it is only fit for railway production to control quality. Another method is the application of a coordinate measurement system, but even more important is that it puts restrictions to the maximum size of the workpiece [2]. Applying a rotating laser beam and a light sensitive receiver to measure the complete envelope curve of the workpiece is a method fit for static deformation when the surface of workpiece has no obstacle [3], but it is restricted to measure static deformation. It is difficult to detect the dynamic deformation of large flexible surfaces such as ship decks when a ship is sailing. A high accuracy, high speed sampling deformation measurement sensor system is required by the user [4].

The advantages of the strapdown inertial units (SIUs, including three single-freedom-degree gyros and three accelerators) are well known and have been fully praised in the research literature. For example, the advantages of the SIUs are light weight, low cost, high sensitivity [5]. Therefore they have been widely used in

numerous engineering measurement of angular deformation. Recent research reports showed that two three-axis laser gyro units are installed near the peripheral device, such as radar antennas and optical systems on the ship deck to measure local angular deformation [6]. Many SIUs must be installed on the large flexible surfaces for angular deformation measurement of the whole surfaces according to the general situation. It is obvious that this is not an economic method and also is not necessary in the actual conditions.

This paper presents the information fusion method based on improved RBFNN algorithm to deduce the deformation information of the whole flexible surface considering finite SIUs on the large surface. At first, an overview of Information fusion based on RBFNN is demonstrated; secondly, a distributed SIUs information fusion model is introduced; the third, the consistent property test of local SIU's information is analyzed. In conclusion, comparison is presented between the modeling results on a simple thin plate model by information fusion based on improved RBFNN and based on BPNN.

2 Information Fusion Based on RBFNN

2.1 The Topology Structure of RBFNN

RBFNN shares features of the BPNN for pattern recognition. It is being extensively used for on-line and off-line nonlinear adaptive modeling and control applications. It stores information locally, whereas the conventional BPNN stores the information globally. RBFNN is one of the approaches which has shown a great promise in this sort of problems because of its faster learning capacity. Its typical topology structure (figure. 1) has input, hidden and output layer. The input space (total neurons number is I) can be either normalized or an actual representation can be used. This is then fed to the associative cells of the hidden layer which acts as a transfer function. Representing a bias is optional. The neurons of the hidden layer (total neurons number is H) are composed of the radiant basis function. Each hidden neuron receives as net

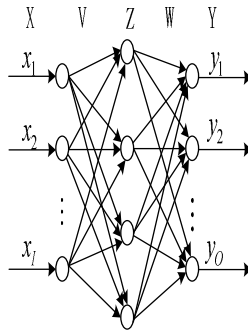


Fig. 1. Typical RBF network structure

input the distance between its weight vector and the input vector. The RBFNN uses a Gaussian transfer function in the hidden layer. The Output of the j th node of the hidden layer is given as:

$$Z_j = k(\|X - V_j\|) = \exp(-\Sigma(x_i - v_{ij})^2 / (2\sigma_j^2)), \quad (1 \leq j \leq H), \quad (1)$$

where, $\|\bullet\|$ denotes the Euclid Norm, $X = (x_1, x_2, \dots, x_l)$ is input pattern vector, $V_j = (v_{1j}, v_{2j}, \dots, v_{lj})^T$ is the center of the RBF of j th node of the hidden layer, σ_j is spread of the j th RBF, ie. the standard deviation of the Gaussian function.

Each neuron in the RBFNN outputs a value depending on its weight from the center of the RBF. The RBFNN uses a linear function in the output layer. The output of the k th node of the RBFNN linear layer is given by:

$$Y_k = \sum_{j=1}^H w_{jk} z_j - \theta_k = W_k Z_j - \theta_k, \quad (2)$$

where, $1 \leq k \leq O$ (O is the number of output nodes), Y_k is output of the k th node, $W_k = (w_{1k}, w_{2k}, \dots, w_{Hk})^T$ is the weight vector for node k , Z_j is the output vector from the j th hidden layer (can be augmented with a bias vector), θ_k is the threshold of the k th node of the output layer.

2.2 Improved Learning Algorithm of RBFNN

According to the different ways for selection of RBF center, RBFNN is usually applied to the methods such as random selection, self-organization learning (SOL) or supervisory learning, etc. This paper applies improved learning algorithm, namely, applies non-supervisory self-organization learning algorithm to choose the center of RBF and applies supervisory learning algorithm to choose the weights of the output layer. It is known that essentially the SOL algorithm is a mixed learning algorithm. The function of SOL is to adjust the center of RBF to the key area of input space. According to SOL algorithm, center vector $V_j(\tau+1)$ of the j th node of hidden layer at $(\tau+1)$ th learning moment is given by:

$$V_j(\tau+1) = \begin{cases} V_j(\tau) & d(V_j(\tau) - X(\tau+a)) \geq \gamma_j, \\ \beta V_j(\tau) + \alpha X(\tau+a) & d(V_j(\tau) - X(\tau+a)) < \gamma_j, \end{cases} \quad (3)$$

where, $\alpha > 0, \beta > 0, 1 \leq j \leq H$; center vector $V_j(\tau)$ is the j th node of the hidden layer at τ th learning moment; γ_j is clustering radius of j th node center vector $V_j(\tau)$ of the hidden layer; α and β are given respectively by:

$$\alpha = 1/N_j(\tau+1), \quad \beta = N_j(\tau)/N_j(\tau+1), \quad (4)$$

$N_j(\tau)$ is the clustering input samples of center vector $V_j(\tau)$ of the j th node of the hidden layer at τ th learning moment, $d(V_j(\tau) - X(\tau+1))$ is the Euclidean distance

between the center vector $V_j(\tau)$ of the j th node of the hidden layer and input vector $X(\tau+1)$ at $\tau+1$ moment and is given by:

$$d(V_j(\tau) - X(\tau+1)) = \sqrt{\sum_{i=1}^I (V_{ij}(\tau) - X_i(\tau+1))^2}. \quad (5)$$

The supervisory learning algorithm is applied to calculate the linear weights between the hidden layer and the output layer based on the steepest decent method. Suppose that the desired output of the k th neuron of the output layer of r th epoch s th learning sample at τ learning moment is $Y_{d,k}(\tau)$ or $Y_{d,k}(r,s)$, the actual output is $Y_k(\tau)$ or $Y_k(r,s)$ and suppose a weights matrix is $W(r)$ after the r th learning, then the weights matrix $W(r+1)$ is given by:

$$W(r+1) = W(r) + \eta \sum_{s=1}^N (Y_{d,k}(r,s) - Y_k(r,s)) Z_j(r,s), \quad (6)$$

where, $j = 1, 2, \dots, H$, $k = 1, 2, \dots, O$; η is learning rate; $Z_j(r,s)$ is the output vector from the j th hidden layer.

3 A Distributed SIU Information Fusion Model

Suppose the output attitude angular information of the i th local SIU and platform gyrocompass are Φ_s^i and Φ_p respectively in distributed SIU, angular deformation of the i th local part of installing SIU is Ψ_s^i , then

$$\Phi_s^i - \Phi_p = \Psi_s^i + \delta^i, \quad (7)$$

where, δ is total error (including system error, misalignment error and calculation error) of the i th local part of installing SIU, n is total number of SIU.

Then attitude information of the j th local part is deduced:

$$\Phi_w^j - \Phi_p = \Psi_w^j, \quad (8)$$

where, Φ_w^j , Ψ_w^j are attitude angle and flexible angle of the j th local part respectively, m is the total number of the local part angular information required induced.

According to equation (7) and equation (8), the attitude information of the j th local part is given by:

$$\Phi_w^j = \Phi_s^i + \Psi_w^j - \Psi_s^i - \delta^i, \quad i = 1, 2, \dots, n, j = 1, 2, \dots, m. \quad (9)$$

So, for the j th local part, one result of angular information Φ_w^{j0} and n results of angular information Φ_w^{ji} ($i = 1, 2, \dots, n$) are obtained from equation (8) and equation (9) respectively. But the $n+1$ results aren't accurate because of some inevitable errors and some fault of local SIU, the information fusion method based on RBFNN to deduce the deformation information of the whole flexible surface considering n finite SIUs on the the large surface is necessary to improve angular information precision.

A distributed SIU information fusion model is shown in fig. 2. First, the consistent property test is analyzed to determine the output information validity of every SIU. Then, $n+1$ results calculated by valid output information of SIU, flexible information and platform gyrocompass information is treated as input of RBFNN to deduce angular information of any part.

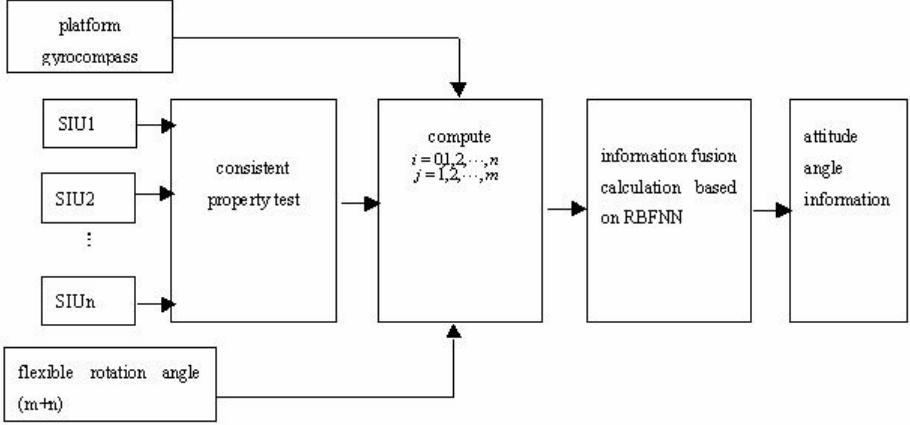


Fig. 2. A distributed SIU information fusion model

4 Consistent Property Test of Output Information of Local SIUs

Consistent property test is necessary for an information fusion system to reject non-consistent information and to reserve consistent information to calculate fusion value.

Suppose desired three attitude angles θ , φ , γ are given respectively by:

$$\begin{cases} \theta = \theta_0 + A_\theta \sin(\omega_\theta t + \phi_\theta) \\ \gamma = \gamma_0 + A_\gamma \sin(\omega_\gamma t + \phi_\gamma) \\ \varphi = \varphi_0 + A_\varphi \sin(\omega_\varphi t + \phi_\varphi) \end{cases}, \quad (10)$$

where, A_θ , A_γ , A_φ are amplitude respectively, ω_θ , ω_γ , ω_φ are angular frequency respectively, ϕ_θ , ϕ_γ , ϕ_φ are initial phase angle respectively, θ_0 , γ_0 , φ_0 are initial angle.

Then, maximum angular rate are obtained from the equation (10) respectively:

$$\dot{\theta}_{\max} = A_\theta \omega_\theta, \dot{\gamma}_{\max} = A_\gamma \omega_\gamma, \dot{\varphi}_{\max} = A_\varphi \omega_\varphi. \quad (11)$$

Actual attitude angles are calculated by SIU information, Suppose sampling period is t , Then consistent property test of output information of local SIUs is given by:

$$|\theta_{t+1} - \theta_t| \leq \dot{\theta}_{\max} t, |\gamma_{t+1} - \gamma_t| \leq \dot{\gamma}_{\max} t, |\varphi_{t+1} - \varphi_t| \leq \dot{\varphi}_{\max} t. \quad (12)$$

Output information of local SIUs is treated as valid information if they meet the demand of the equation (12).

4 Simulation

Usually the large flexible structure such as ship deck, flexible wing etc. is too complex to analyze precisely. A simple thin plate model (Figure. 3) assisted reinforcing steel rib is set up to verify the information fusion method based on RBFNN. Simulation based on the finite element method (FEM) is done for the thin plate on limited actual condition. Suppose Poisson rate $\sigma = 0.3$, modulus of elasticity $E = 2.01e11$ Mpa, density of mass $\rho = 7850 \text{ kg/m}^3$, the 15 orders vibration model of the thin plate are analyzed by FEM and treated as a monitoring object. (Here only angle verification is done along x direction). Simulation is done based on the static base (i.e. $\Phi_p = 0$) because of limited condition. Suppose four SIUs are installed on the large thin plate.

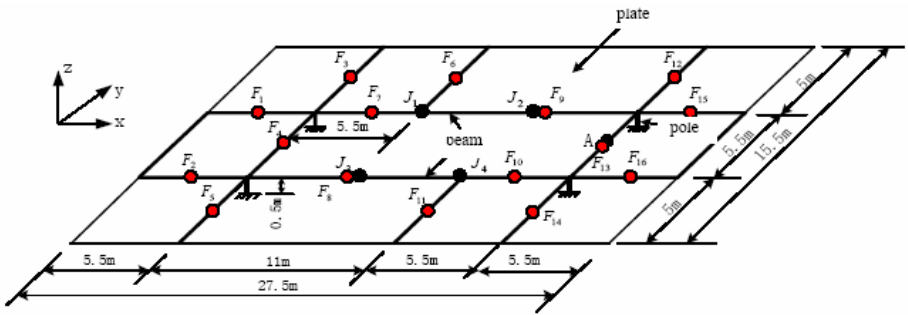


Fig. 3. The thin plate model (grid form is a rectangular form because of higher efficiency than a triangle form. Considering the plate shape, a suitable grid size is selected to be divided evenly according to line length (0.5 m), considering tiny flexible deformations. Four SIUs are located on J_1, J_2, J_3, J_4 respectively, we suppose that deformation of point A is necessary to be deduced and external loads are located F_1, F_1, \dots, F_{16} respectively).

In addition, we suppose that the output of a SIU is kept at high precision during some conditions, $2.5^\circ \leq A_\theta \leq 4^\circ$, $4s \leq T_\theta \leq 8s$, $12^\circ \leq A_\gamma \leq 18^\circ$, $8s \leq T_\gamma \leq 13s$, $t = 0.02s$, according to the equation (11), then the maximum angle increment are given respectively by:

$$\Delta\theta_{\max} = A_\theta \omega_\theta t = (4 * 2\pi / 4) * 0.02 = 0.1237^\circ,$$

$$\Delta\gamma_{\max} = A_\gamma \omega_\gamma t = (18 * 2\pi / 8) * 0.02 = 0.2827^\circ.$$

Considering an actual situation, suppose $\Delta\theta_{\max} = 0.2^\circ$, $\Delta\gamma_{\max} = 0.4^\circ$.

In this paper, in the Fig. 1, a RBFNN with 5 neurons in the input layer, 11 neurons in the hidden layer and one neuron in the output layer is employed. After consistent property of SIUs output information is tested, 50 group of results of FEA for point A at different combination states acting on 16 external load simultaneously are used as training input of ANN, the training time for the improved RBFNN model is about 5×10^3 epochs (about 10 minutes or so) for 1×10^{-4} precision, it is much less than that for

the training of the BPNN for the same precision. The learning speed for the improved RBFNN is much faster than that for training of the BPNN (see fig. 4 and fig. 5).

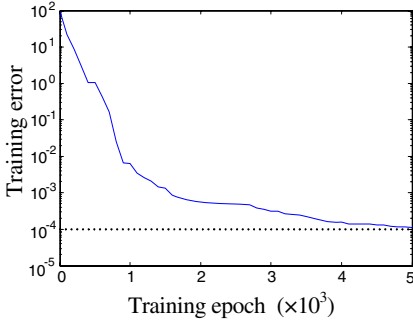


Fig. 4. Learning speed of the improved RBFNN

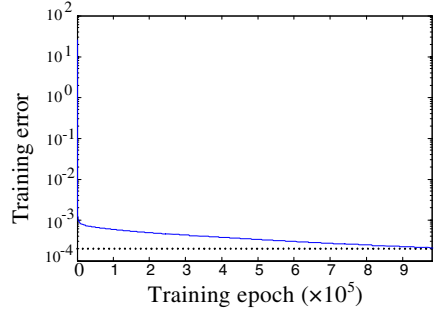


Fig. 5. Learning speed of the BPNN

10 groups of training samples selected from 50 groups of training samples of FEA are used as desired flexible attitude information. These 10 groups of training samples added to 10% random noise are used as the input of improved RBFNN and BPNN respectively to fusion according to chapter 2.2. Output results of attitude information fusion at point A are shown as table 1.

Table 1. output results of attitude information fusion at point A and modeling error (unit: degree)

Sample number	1	2	3	4	5	6	7	8	9	10
Desired value	0.05943	0.06437	0.06913	0.07528	0.08218	0.08905	0.09317	0.10993	0.12085	0.13645
Improved RBFNN	0.06074	0.06384	0.06815	0.07726	0.08127	0.09012	0.09415	0.10835	0.12426	0.13497
Error (%)	2.2	-0.98	-1.42	2.63	-1.107	1.2	1.05	-1.44	2.82	-1.08
BPNN	0.05716	0.06553	0.06758	0.07795	0.08367	0.08725	0.09529	0.11215	0.11539	0.13227
Error (%)	-3.82	1.802	-2.24	3.547	1.81	-2.02	2.28	2.02	-4.52	-3.06

Comparing with the results by improved RBFNN and BPNN, modeling precision of improved RBFNN is higher than BPNN at the same training time, at the same time, a information fusion method based on improved RBFNN has good anti-disturbance property for application of deformation measurements of large flexible surface.

6 Conclusions

The simulation on a simple thin plate model shows that the information fusion based on improved RBFNN is effective and has higher precision than based on back propagation (BP) NN. The application of improved RBFNN for the deformation measurement of a large flexible surface is feasible.

Acknowledgements. The work was supported by Program for New Century Excellent Talents in University (NCET) and the Southeast University Excellent Young Teacher Foundation (4022001002). The author would like to thank Dr. Fujun Pei and Prof. Dejun Wan of the School of Instrument Science and Engineering at Southeast University for helpful suggestions.

References

1. Drews, P., Galyga, B.: Workpiece Supervision and Quality Control. APS Mechatronik, Aachen, Germany (1993)
2. N.N.: Koordinatenme ß Technik im Betrieb. Scope Nr. 6, Sonderheft f r **Verfahrenstechnik**, Verlag Hoppenstedt GmbH, Darmstadt, Germany (1994)
3. Drews, P., Fromm, P.: LASSCA- A Sensor System for Straightness Measurement of Large Surfaces. IEEE (1996) 150-154
4. Chen, X.-Y., Fang, L., Wan, D.-J.: Deformation Measurement Sensor System for Large Surfaces. Proc. of the 2nd Internat. Symp. On Instrument. Sc. and Technol. Harbin Institute of Technology (2002) 186-190
5. Lebedev, D.V., Tkachenko, A.I.: Alignment of Strapdown Inertial System with Full Uncertainty of Initial Sensor Unit Attitude. Journal of Automation and Information Sciences **34** (2002) 31-38
6. Andrei, V. M., Andrey, V. K.: Use of Ring Laser Units for Measurement of Moving Object Deformation. Proc. SPIE Int. Soc. Opt. Eng. On Second International Conference on Lasers for Measurements and Information Transfer, **4680**. Bellingham WA (2002) 85-92
7. Ruiz, A., Lopezde, T., Pedro, E.: Nonlinear Kernel-based Statistical Pattern Analysis. IEEE Transactions on Neural Networks **12** (1) (2001) 16-32

Evaluation of the Growth of Real Estate Financial System Based on BP Neural Network*

Hu Nai-peng and Tian Jin-xin

School of Management, Harbin Institute of Technology, Harbin, 150001, China
np_hu@yahoo.com.cn, jxam@sina.com

Abstract. Currently, there is little quantitative research on macroscopic real estate finance at home and abroad. Seen from the whole system of real estate finance, this paper chooses 14 main indexes to compose an evaluation index system. Based on the evaluation index system, an error-back-propagation BP network model is built to evaluate the growth of real estate finance. Data of real estate financial system from 1997-2005 are used as train and test samples of BP neural network. After training, the BP neural network is used to evaluate and forecast by simulation. Through the good accuracy of evaluation and forecasting, the model is proved to be very efficient. By comparing the growing difference of two adjacent years and analyzing the related macro financial policies in related years, the running effect of related real estate financial policies in related years is gained. So by using the evaluation model of this paper, decision makers can decide to use what kind of macro adjusting and controlling policies to gain anticipated aim of real estate finance in the future.

1 Introduction

In recent years, with the course of urbanization and reform of urban housing system deepening, China debuted many policies which support the development of real estate industry and housing credit. Accompanying with the fast increase of real estate industry, real estate finance has risen to eminence in the field of national financial industry. So how to evaluate the growth of Chinese real estate finance scientifically has become a preliminary problem to instruct the real estate financial system to develop in a healthy way.

Currently, in foreign countries although there have been much research on real estate finance and the quantitative research focus mainly on real estate financial instruments and price, research on macroscopic real estate finance have been little. In our country research on real estate finance are also very much, these research are of related laws, policies, financial instruments, financial market and financial intermediaries, but they are almost of actual practice and quantitative research are little especially in macroscopic aspect. The main quantitative research involving in real estate finance at home and abroad sum up as follows:

* This paper is supported by funds of Ministry of Construction and numbered 05-72-18.

Allen and Gorton (1993) made a model under a continuous time limit. Then Allen (1998) developed a simple model. In the model the intermediation of bank sectors led to proxy problems which produced capital bubbles. Yuan Zhi-gang (2003) started from economic bubble theory and built a sectional equilibrium model of real estate market. The model showed the important roles of anticipation of actors, credit of banks and government policies in forming real estate bubble. Wong (1998) explained the collapse of real estate market before economic crisis in 1997 in Thailand. The continuous economic growth created not only high demand of real estate, but also increased optimized anticipation of future market. So the overmuch house property and bubbles were formed. In an international viewpoint Herring (1998) studied on the relationship between real estate prosperity and bank crisis. Through building credit market model together with Carey's model that bank's centering credit led to real estate prosperity, the real estate prosperity fermented bank crisis. Wu Kang-ping and Pi Shun (2004) made a general equilibrium analysis of the co-growth of real estate market and financial market. Pi Shun and Wu Kang-ping (2006) proved that there was double linear cause relationship between the development of Chinese real estate market and that of Chinese financial market.

Based on developmental economics which discusses on the economic increase and economic development, a scholar at home puts forward the concept of financial growth (II). The concept of real estate financial growth in this paper roots in that of financial growth. Real estate financial growth is an organic whole composed by real estate financial increase and real estate financial development. Real estate financial increase means the relative scale of inventory and flux of real estate finance. Being the given quantity of real estate financial growth, it can be expressed as the scale of real estate financial asset expanding compared with the expansion of national wealth. Real estate financial development means the high or low manned efficiency and use efficiency of real estate finance. Being the given quality of real estate financial growth, it can be expressed as the change of real estate financial structure and the perfection of real estate financial market. It also includes the spatial difference of financial structure, turn of financial concept and change of financial institutions.

In short, this paper discusses the problem of real estate financial growth by BP neural network.

2 Modeling Evaluation System of Real Estate Financial Growth by BP Neural Network

2.1 BP Neural Network Theory

Currently multilayer BP neural network is a most widespread artificial neural network. This network uses error-back-propagation algorithm which is called error back propagation artificial network or BP network for short by people. The learning process of BP network is composed of four processes: the learning mode is a mode-propagation-in-turn process from input layer visa medium layer to

output layer; the error signal coming from expected output of network and actual output is an error-back-propagation process revising link weight value from output layer visa medium layer to input layer; the alternation of mode-back-propagation and mode-propagation-in-turn forms the process of "memory training"; the network tends to converge, which is a "acquiring proficiency process" ([2][3]) making the composite error tend to minimize.

2.2 Building Evaluation Model of BP Neural Network

Although the error back propagation artificial network is almost the most simple form, this paper aims to solve a problem in a new field by it, not to gain a complex computing process or a result with good mathematical form. The most important thing lies in whether or not the model can solve our problems.

BP neural network is a multilayer network using nonlinear differentiable function to train weight value. Because it has simple structure and strong plasticity, and has absolute predominance ([4][5][6]) in dealing with problems of multi-dimension, complex and nonlinear, so this paper use three layers BP neural network to evaluate Chinese real estate financial growth.

The action function reflects the intensity of stimulated pulse from input of lower layer to upper node in the neural network. The function is also called stimulated function. This paper chooses Sigmoid function with continuous value $x \in (0, 1)$.

$$f(x) = \frac{1}{1 + e^{-x}} \quad (1)$$

BP neural network has direct relation with the number of input and output unit. If the units of hidden layer are too short, the network may train without any result; if the network is not strong enough, it may not recognize the samples which were not be seen before in the network, and its fault tolerance is not good enough; if the number of the hidden layers is too large, the network may take too much time to learn and its error may not reduce, so in general the unit number of hidden layer can be computed as the given empirical formula.

$$n_H = \sqrt{n_I + n_0} + l \quad (2)$$

where, n_H : neural number of hidden layer n_I neural number of input layer n_0 neural number of output layer l : an integral number from 1 to 10.

3 Empirical Analyzing

3.1 Building the Evaluation Index System of Chinese Real Estate Financial Growth

The evaluation index system involves in four aspects which influence the real estate financial system directly or indirectly: macroeconomic factor, resident factor, developmental level of real estate economy and real estate finance([7]).

The four aspects include 14 indexes according to state statistical bureau caliber classification, seen table 1. These chosen indexes presuppose some rules which are real, complete, scientific, fair and operable. The macroeconomic factor includes GDP, social commodity retail price index and urbanization level, these indexes may influence the structure and scale of real estate finance indirectly by influencing the supply and demand anticipation of real estate finance; resident factor includes urban resident per average controllable income and urban resident per average floor space. The two indexes may influence the supply of real estate finance directly; developmental level of real estate economy includes commercial housing sales, land developmental area and commercial housing construction area. These indexes directly influence the demand of real estate finance; the real estate finance factor includes domestic loans, other source of funds, self-financing, foreign investments, bonds and state budget funds. These indexes directly show the supply scale and structure of real estate finance and the prosperous degree of real estate financial market, and show the change of related financial system and participation degree of financial agencies at the same time.

3.2 Zero Dimensional Processing of Evaluation Index

According to the following formula (3), the dimensional difference of different index can be smoothed away. A real index x_{ij} can be converted into an evaluation value x'_{ij} ($\in (0, 1)$), where j is the j th index, i is the i th evaluation value of the j th index, $i = 1, 2, \dots, 9$, $j = 1, 2, \dots, 14$.

$$x'_{ij} = \frac{x_{ij} - \min(x_{ij})}{\max(x_{ij}) - \min(x_{ij})} \quad (3)$$

3.3 Training Mode Pair of Neural Network

The premise of training network is to get suitable quantity of training units. This paper makes index data of 1997-2005 be empirical analyzing object. The objects are the so called training units, where the 14 indexes of real estate financial system (as shown in table 1) are input data of training units, while the corresponding evaluation values of predominance analyzing model are output data of the network.

3.4 Training and Examining BP Network by Matlab

In the evaluation process of real estate financial growth by BP neural network, the network which has been trained completely supplies mainly much information not including subjective factors for decision makers. The indexes to be trained are very important for a neural network model, so these indexes must be overall and not repeated, and have strong maneuverability and comparability at the same time.

The 14 index data of real estate finance originating from standardized process and the evaluation values¹ coming from predominance analyzing theory of grey system compose of nine input and output data mode pair of the network. The mode pairs numbered 1-6 (1997-2002) are training units. The BP neural network imports these training units to train. When a BP neural network with a hidden layer has suitable quantity of units of hidden layers, it can simulate any function precisely. This can be showed in many advanced researches ([10],[11],[12],[13]) of BP neural network. The unit number of hidden layer can be determined according to formula (2). After computing, this paper gets 12 units of hidden layer. So a network with a structure 14-12-1 is built, where the learning accuracy is given $\varepsilon = 1e-06$, the learning speed is given $lr = 0.05$, each growth evaluation value of training units is output of anticipation. By computing with software of Matlab, the train of BP network completes with 100 training epochs and the total error is $0.617377e - 06$. The training result is shown in fig.1. Being test units, the training mode pairs numbered 7-9 (2003-2005) are imported into the network to evaluate and forecast the network. The evaluation result of data simulation is very close to that of predominance analyzing. The result of comparative analysis is shown in table 2.

3.5 Result Analyzing

Seen from table 2, the growth value of real estate finance of each year has been increasing year by year from 1997 to 2005. It proves that Chinese real estate financial system has been in a growing situation since 1997. While this situation exactly fits the whole prosperous situation of Chinese real estate industry in these years, so just as gotten in reference [7], the conclusion is gotten another time that the development of real estate market and that of financial market are in co-growth to some extent.

The difference between simulation result and anticipation output is very small, which means this trained network has good accuracy and objective evaluation result.

The difference percent values of two adjacent years obviously show the adjusting and controlling effect of the related industrial policies on real estate finance. For example, China implemented tighter fiscal and monetary policy in 1997, in 1998 China monetized housing distribution, promoted the development of real estate industry, so the difference value is a relatively large number 36.6%; in 1999 China implemented active fiscal policy, started housing consumption (removed personal income tax) and deepened monetizing housing distribution inform, so the difference value is also a large number 24.7%; in 2001 China encouraged to buy overstocking housing, so the difference value is a small number 1.5%; in 2002 China lowered the deposit and loan interest rates of public accumulation fund

¹ The 14 indexes in table 1 are classified into two categories, one is character vectors including FSUA, CHS, CHA, the other is relative factor vectors including GDP, IRP, LU, DIUR, DL, OSF, SF, FI, B, SBF. After standardizing these indexes, according to predominance analyzing method in ([8],[9]), the evaluation values can be obtained.

Table 1. Evaluation indexes of the growth of real estate finance

NUM	GDP	IRP	LU	M2	RCI	RFS	CHS
1	0	0.65517	0	0	0	0	0
2	0.035744	0.06897	0.12996	0.06499	0.04969	0.1083	0.0609
3	0.07001	0	0.25903	0.13912	0.13013	0.19254	0.10139
4	0.13814	0.25862	0.38899	0.20993	0.21001	0.30084	0.18224
5	0.21038	0.57895	0.51895	0.32396	0.31877	0.36101	0.26136
6	0.28273	0.37931	0.64801	0.4525	0.47684	0.60168	0.36115
7	0.39393	0.2931	0.77798	0.62682	0.62104	0.70999	0.52525
8	0.57127	1	0.88899	0.7851	0.79917	0.86643	0.73175
9	1	0.65517	1	1	1	1	1
NUM	CHA	DL	OSF	SF	FI	B	SBF
1	0	0	0	0	1	0.48283	0.69427
2	0.04842	0.04857	0.04162	0.032	0.69522	0.62376	1
3	0.09938	0.06854	0.07091	0.06128	0.37182	1	0.39298
4	0.17505	0.16209	0.15903	0.10572	0.10149	0.33877	0
5	0.28818	0.26715	0.25825	0.19965	0.0662	0.01474	0.83369
6	0.41117	0.4478	0.36889	0.29105	0	0.21136	0.61705
7	0.60724	0.76178	0.5421	0.43122	0.10549		
8	0.79915	0.76867	0.82842	0.69809	0.28447	0	0.6102
9	1	1	1	1	0.35767	0.55841	0.69688

Remark 1. NUM is number; GDP is gross domestic product; IRP is index of social commodity retail price; LU is level of urbanization; DIUR is annual per capita disposable income of urban residents; FSUA is per capita residence floor space in urban areas; CHS is commercial housing sales; CHA is commercial housing construction area; DL is domestic loans; OSF other source of funds; SF is self-financing; FI is foreign investment; B is bonds; SBF is state budget funds.

Table 2. Evaluation result of the growth of real estate finance

NUM	1	2	3	4	5	6	7	8	9
Y	1997	1998	1999	2000	2001	2002	2003	2004	2005
AO	0.17	0.23	0.28	0.32	0.33	0.4	0.46	0.48	0.52
SR	0.1665	0.2275	0.2837	0.3230	0.3279	0.4094	0.4778	0.4913	0.5284
DPY(%)		36.6	24.7	13.9	01.5	24.9	16.7	2.8	7.6
EDT	-0.0035	-0.0025	0.0037	0.003	-0.0021	0.0094	0.0178	0.0113	0.0084
EP(%)	-2.1	-1.1	1.3	0.9	0.6	2.4	3.8	2.3	1.6

Remark 2. NUM is number; Y is year; AO is anticipation output; SR is simulation result; DPY is difference percent of two adjacent years; EDT is errors of data test; EP is error percent.

for housing construction and the interest rate of loan over 5 years, so the difference value is large (24.9%) again; the difference value of 2003 is also relatively large (16.7%), because the effect of policies in 2002 continued; the difference value (2.8%) of 2004 goes down sharply, because since 2003, China has begun to adjust and control the developmental speed of real estate industry. Some tighter policies were implemented, such as controlling loans of real estate, levying tax and so on; although in 2004 China still made a further step to implement more tighter policies, in 2005 the difference value is 7.6% which is still above zero. There are two main reasons about 7.6%, one is because Chinese macroeconomy has been developing with a fast speed for many years, which speeds the process of urbanization. Urbanization brings large and real demands; the other is because the whole economic situation is optimized, but Chinese stock market has not matured and has great risk, so people have more optimized anticipation for Chinese real estate market.

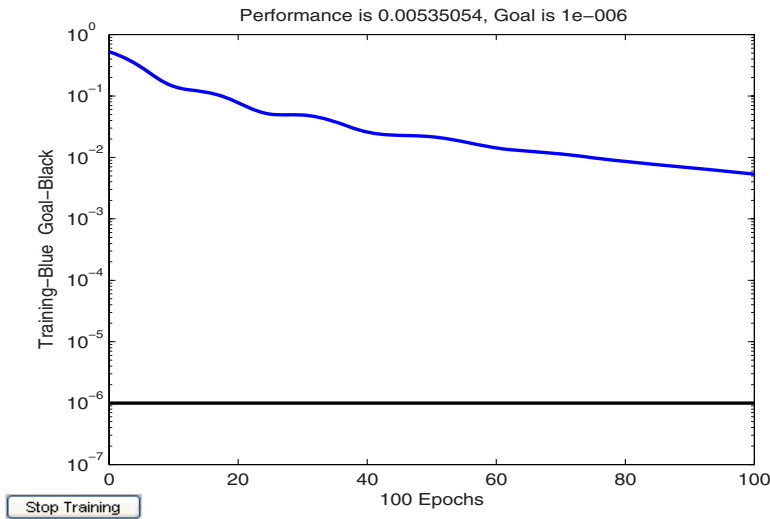


Fig. 1. Accuracy and effect of network training

4 Conclusion

This paper builds the evaluation model of the growth of real estate finance based on BP neural network. Being analyzing samples, data of 1997-2005 are introduced to evaluate the growth of real estate financial system in a macroscopic view. The network which has been trained may forecast the growth of real estate finance. Through test the accuracy of the network is relatively high and the evaluation result is objective and accurate basically.

BP neural network avoids the influence of determining standard values and weights subjectively in the process of evaluation. Because neural network has

strong self-learning ability, self-adapted ability and fault tolerance, so it can build more objective and accurate evaluation system.

By comparing with the growth value of each two adjacent years, we can gain the influence on the whole real estate finance system. The influence comes from the effect of industrial policies in each year. So the result has some reference value for decision makers to determine what kind of policy should be implemented in macro adjusting and controlling and what kind of effect should be.

References

1. Zhang J.: Economic Analysis of Chinese Financial Growth. 1st edn. Beijing, Chinese Economy Publishing House (1995) 13–18
2. Yuan Z.: Artificial Neural Network and Applications. 1st edn. Beijing, Tsinghua University Publishing House (1998)
3. Liu C., Li X.: The Improvement of BP Algorithm and Self-adjustment Architectural Parameters. *Journal of Information Management* **1** (2000) 86-92
4. Yang T.: Research on Comprehensive Evaluation Method and Applications of High-tech Enterprise Base on BP Neural Network. *Chinese Soft Science* **5** (2004)96-98
5. Miao L.: Intelligent Comprehensive Evaluation of the Ability of Making Profits in the Future of an Enterprise. *Mathematical Practice and Understanding* **5** (2004) 54-59
6. Dai W.: Comprehensive Evaluation Method and Applications of Multi Indexes Based on BP neural network with Three Layers. *Systematical Engineering Theory and Practice* **5** (1999) 29-40
7. Pi S., Wu K.: Research on the Relationship between Chinese Real Estate Market and Financial Market. *Journal of Management Engineering* **2** (2006) 1-6
8. Luo X., Yang H.: Grey Comprehensive Evaluation Model. *Systematical Engineering and Electronic Technology* **9** (1994) 18-25
9. Liu S., Guo T., Dang y.: Grey Systematical Theory and Applications. 1st edn. Beijing, Scientific Publishing House (1999) 49-70
10. Cybenko, G.: Approximation by Superposition of Sigmoid Function. *Math Control Signals Syst.* **2** (1999) 303-380
11. Funahashi, K.: On the Approximate Realization of Continuous Mappings by Neural Networks. *Neural Networks* **2** (1998) 183-192
12. Hornik, K.: Approximation Capabilities of Multiplier Feed for Ward Networks. *Neural networks* **6** (1993) 1069-1072
13. Hornik, K., Stinchcombe, M., White, H.: Multilayer Feed-forward Networks Are Universal Approximations. *Neural Networks* **2** (1998) 359-366

GA-Based Neural Network to Identification of Nonlinear Structural Systems

Grace S. Wang¹ and Fu-Kuo Huang²

¹ Department of Construction Engineering, Chaoyang University of Technology,
No. 168 Jifong E. Rd., Wufeng Township, Taichung County 41349, Taiwan
grace@cyut.edu.tw

² Department of Construction Engineering, Tamkang University,
No. 5, Lane 199, King-Hwa St., Taipei, Taiwan
fkhuang@mail.tku.edu.tw

Abstract. The initial weights of neural network (NN) are randomly selected and thus the optimization algorithm used in the training of NN may get stuck in the local minimal. Genetic algorithm (GA) is a parallel and global search technique that searches multiple points, so it is more likely to obtain a global solution. In this regard, a new algorithm of combining GA and NN is proposed here. The GA is employed to exploit the initial weights and the NN is to obtain the network topology. Through the iterative process of selection, reproduction, cross over and mutation, the optimal weights can then be obtained. The proposed new algorithm is applied to the Duffing's oscillator and Wen's degrading nonlinear systems. Finally, the accuracy of this method is illustrated by comparing the results of the predicted response with the measured one.

1 Introduction

Field of system identification has become important discipline due to the increasing need to estimate the behavior of a system with partially known dynamics. Identification is basically a process of developing or improving a mathematical model of a dynamic system through the use of measured experimental data. In addition to updating the structural parameters for better response prediction, system identification techniques made possible to monitor the current state or damage state of the structures. As for structural control problem, the system of interest also needs to be known to some extent. Structural identification can be categorized into classical and non-classical methods. Most of the classical methods are calculus-based search method. They are performed by point-to-point search strategy and normally require gradient or higher-order derivatives of the objective function. There is a possibility to fall into a local minimum rather than the global minimum. Therefore, these methods generally do not function well for structural identification problem involving a large number of unknowns. For such problems, the newly developed non-classical methods provide another alternative. Among the methods, the artificial neural network and genetic algorithm are the most common techniques for system identification. Some works of non-classical methods in the context of system identification are review as follows.

Jovanović [1] proposed a neural network approach for structural dynamic model identification, using the responses recorded in a real frame during earthquakes. A typical three-layer back propagation neural network was used for the purpose of identification and a five-story steel frame was chosen to demonstrate the performance of the neural network. Two earthquakes used for the dynamic model identification were recorded for the frame. They are the Petrovac 1979, component N-S and El Centro 1940, component N-S. The displacement and acceleration time histories were recorded for the sets of earthquakes on each floor. The data set, used for training of the neural network dynamic model, is the first 500 points taken from 1,000 points record of the Petrovac 1979 earthquake and the rest of response histories were used for verification of the trained neural network model. The results showed the great potential of using neural networks in structural dynamic model identification.

Loh and Huang [2] proposed a neural-network-based method to the modeling and identification of discrete-time nonlinear hysteretic system during strong ground motion. The learning or modeling capability of multilayer neural network was explained from the mathematical point of view. The main idea of the proposed neural approach was explained, and it was shown that multilayer neural network is a general type of NARMAX model and is suitable for the extreme nonlinear input-output mapping problem. Numerical simulation and real structure cases are used to demonstrate the proposed method.

The author [3] applied the real-coded GA to structural identification problems. The GA provides a stochastic search in the designate ranges of parameters. The system parameters associated with the minimal error index were then exploited after successive evolution of generations. The validity and the efficiency of the proposed GA strategy were explored for the cases of both SDOF linear/nonlinear dynamic systems and MDOF linear/nonlinear dynamic systems with simulated input/output measurements. The identified parameters are very close to the true one and the error index is extremely small in each case. As a result, the efficacy of the proposed algorithm was verified.

The results of neural network training may be sensitive to the choice of initial weights of network. To attain a best network topology, this paper proposes a method that merge Genetic algorithm to neural network. This proposed algorithm is applied to the Duffing's oscillator and Wen's degrading nonlinear systems.

2 Neural Network and Genetic Algorithm

Neural networks are data analysis methods and algorithms, which imitate the process of nervous systems of humans and animals. In general terms, an artificial neural network consists of a large number of simple processing units linked by weighted connections. By analogy to human brain, the processing units may be called neurons. Each unit receives inputs from many other units and generates a single output. The output acts as an input to other processing units. Unlike traditional linear algorithms, artificial neural networks using highly distributed representations and transformations that operate in parallel, have distributed control through many highly interconnected neurons, and stored their information in variable strength connections called synapses

- just like a human brain. The network is nonlinear in nature and thus is an exceptionally powerful method of analyzing real-world data that allows modeling extremely difficult dependencies. A certain network may be tuned to solve a particular problem, such as the modeling or prediction of the behavior of a complex system, by varying the connection topology and values of the connecting weights between units.

To bring proper results the neural networks require correct data preprocessing, correct architecture selection and correct network training. The most common type of artificial neural network, called the multi-layer feedforward network with the back-propagation training algorithm, consists of three groups, or layers, of units: a layer of "input" nodes is connected to a layer of "hidden" nodes, which is connected to a layer of "output" nodes:

- The activity of the input nodes represents the raw information that is fed into the network.
- The activity of each hidden node is determined by the activities of the input nodes and the weights on the connections between the input and the hidden nodes.
- The behaviour of the output nodes depends on the activity of the hidden nodes and the weights between the hidden and output nodes.

Feedforward NNs allow signals to travel one way only, from input to output. There is no feedback (loops) i.e. the output of any layer does not affect that same layer. In the standard back-propagation algorithm, the relation between A_j^n , the output in the j th node of the n th layer, and A_i^{n-1} , the outputs of the nodes in the $(n-1)$ th layer, is defined as:

$$A_j^n = f(\text{net}_j^n) \quad (1)$$

$$\text{net}_j^n = \text{summation output} = \sum_i W_{ij} A_i^{n-1} + \theta_j \quad (2)$$

$$f = \text{transfer function} = \frac{1 - e^{-x}}{1 + e^{-x}} \quad (3)$$

where W_{ij} is the connecting weight between nodes in the n th layer and those in the $(n-1)$ th layer; and θ_j is the bias term. The transfer function can be linear or nonlinear.

Identification procedure entails a matching between the system outputs and the identified outputs. During training stage, a system error or objective function is defined and used to monitor the performance of the network. In order to achieve the best performance of the network, adjusting the connecting weights through optimization techniques minimizes this function.

On the other hand, genetic algorithm is a stochastic search technique based on natural selection and genetics, developed by Holland [4]. Genetic algorithms model natural processes, such as selection, recombination, mutation, migration, and competition. The algorithms work on populations of individuals instead of a single solution. In this way, the search is performed in a parallel manner. However, better results can be obtained by introducing multiple subpopulations. Every subpopulation evolves over a few generation isolated (like the single population GA) before one or more

individuals are exchanged between subpopulation using the mechanisms of migration and competition. The multipopulation GA models the evolution of a species in a way more similar to nature than single population. Fig. 1 show the structure of a multipopulation GA.

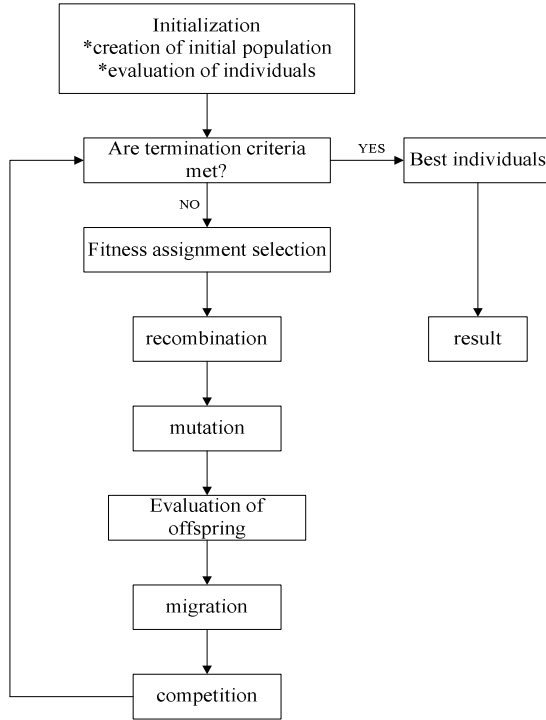


Fig. 1. Structure of a multipopulation genetic algorithm

3 GA-Based Neural Network Identification Algorithm

The network topologies for nonlinear systems are studied here. The network model is designed to construct the relationship between the structural responses, for example, displacement and velocity as input, and acceleration as output. Linear or nonlinear network with delay time may be assumed and then identified. However, the initial weights of the network are selected randomly and the identified result may be dependent on them. In this regard, a new algorithm is proposed here. A multipopulation GA is suggested to be used to exploit the best initial weights of the network here. At the beginning of the computation, a number of individuals for the initial weights are randomly generated. The network training is preceded using each individual as the initial weights. The objective function and the fitness function are then evaluated for the network associated with each individual. If the termination criteria are not met, the creation of a new generation starts. Individuals for initial weights are selected according to their fitness for the production of offspring. Parents are recombined to produce

offspring. All offspring will be muted with a certain probability. The offspring are then inserted into the population replacing the parents, producing a new generation. Every subpopulation evolves over a few generation isolated (like the single population GA) before one or more individuals are exchanged between subpopulation using the mechanisms of migration and competition. This cycle is performed until the optimization criteria for GA are reached. This new algorithm is called the ‘‘GA based neural network identification algorithm’’. The fitness function used in this paper is

$$Fitness = 2 - SP + 2\{SP - 1\} \times \frac{\{Pos - 1\}}{\{Nind - 1\}} \quad (4)$$

Where $Nind$ the number of individuals in the population, Pos the position of an individual in this population (least fit individual has $Pos=1$, the fittest individual $Pos=Nind$) and SP the selective pressure.

4 Application to System with Duffing’s Hysteretic Model

In this section, we attempt to find the neural network topology suitable for system with Duffing’s hysteretic model using the new algorithm aforementioned. The motion equation for such a system when excited by a uni-directional earthquake ground acceleration is

$$m\ddot{u} + c\dot{u} + k_1u + k_2u^3 = -m\ddot{u}_g \quad (5)$$

where $m = 1$, $c = 3.77$, $k_1 = 355.3$, $k_2 = 700.0$ and $\ddot{u}_g =$ ground acceleration. The measured response is the absolute acceleration and can be represented as

$$y = \ddot{u}_t = \ddot{u} + \ddot{u}_g = \frac{c}{m}\dot{u} + \frac{k_1}{m}u + \frac{k_2}{m}u^3 \quad (6)$$

The system is subjected to different levels of ground excitations. The peak ground acceleration of El Centro earthquake is scaled to 10 gal, 196 gal and 210 gal for this purpose. The response of the system is computed accordingly. The network architecture used here is the feedforward back-propagation network trained by Levenberg-Marquardt algorithm. The objective function or network performance is defined as

$$E.I. = \left[\frac{\sum_{i=1}^N (y_i - a_i)^2}{\sum_{i=1}^N (y_i)^2} \right]^{1/2} \quad (7)$$

where N is the number of measurement sequence; y_i is the measured response of the system; and a_i is the identified response. The first step is to find out the proper order of the models. The value of delay time for each input can be determined from value of the objective function, i.e., the value of delay time associated with the best performance of the network is adopted. Due to space limitation, the process is omitted. The value of delay time for model A and model B adopted here is 3 for both models.

Table 1 shows the associated linear and nonlinear models. Several neural network models are identified according to various network topology and the results are listed in Table 2. It is obvious that model A and model B perform better than model C. In order to demonstrate the effect of the new algorithm, the results using traditional neural network are also listed in the same table. Fig. 2 illustrate the comparison of the target output with the network output for model A with PGA=196gal.

Table 1. NN topology for Duffing’s hysteretic models

Model A (Linear)	$u(k) = \sum_{i=1}^3 a_i u(k-i) + \sum_{j=1}^3 b_j \dot{u}(k-j) + \sum_{s=1}^3 c_s \ddot{u}_i(k-s)$
Model B (Nonlinear)	$u(k) = \sum_{m=1}^3 (w_m \times f(\sum_{i=1}^3 a_{mi} u(k-i) + \sum_{j=1}^3 b_{mj} \dot{u}(k-j) + \sum_{s=1}^3 c_{ms} \ddot{u}_i(k-s)))$
Model C (Linear)	$-\ddot{u}_i(k) = au(k) + bu(k)$

Table 2. Network performance (E.I.) for Duffing’s hysteretic models

	Testing case	GA+NN	NN
Model A (Linear)	El ns 210gal	0.0084	0.0085
	El ns 196gal	0.0083	0.0083
Model B (Nonlinear)	El ns 210gal	0.0077	0.0078
	El ns 196gal	0.0076	0.0080
Model C (Linear)	El ns 10gal	0.0369	0.0369

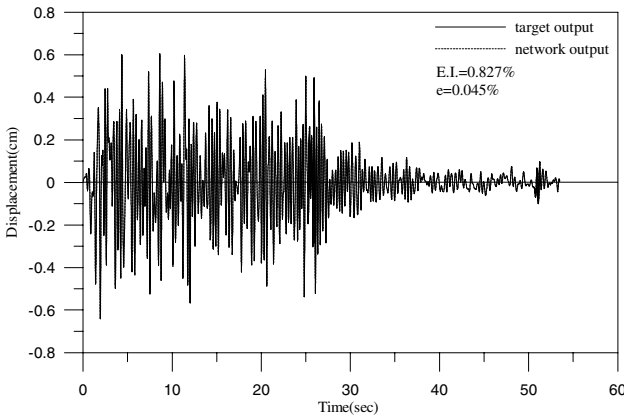


Fig. 2. Comparison of the target output with the network output for model A with PGA=196gal

5 Application to System with Wen's Hysteretic Model

Wen's model is a versatile nonlinear hysteretic model since it can capture a wide range of shapes of hysteresis loops that represent the properties of real nonlinear structural systems in a continuous function. Herein, we try to find the neural network topology for system with Wen's hysteretic model by using the new algorithm. The motion equation for such a system when excited by a uni-directional earthquake ground acceleration is

$$m\ddot{u} + c\dot{u} + \alpha ku + (1 - \alpha)kz = -m\ddot{u}_g \quad (8)$$

where z is the hysteretic component of the restoring force defined by $\dot{z} = \frac{A\dot{u} - \nu(\beta \cdot |\dot{u}| \cdot |z|^{(n-1)} \cdot z + \gamma \cdot \dot{u} \cdot |z|^n)}{\eta}$, $A(\varepsilon_T) = A_0 - \delta_A \varepsilon_T$, $\nu(\varepsilon_T) = \nu_0 + \delta_\nu \varepsilon_T$,

$\eta(\varepsilon_T) = \eta_0 + \delta_\eta \varepsilon_T$ and ε_T is the dissipating energy expressed as $\dot{\varepsilon}_T = (1 - \alpha)kz\dot{u}$. Parameters η, ν are related to the degradation characteristics of stiffness and strength. The shape of the hysteretic loop is controlled by parameters $A, \alpha, \beta, \gamma, \eta, \nu$. The measured response is the absolute acceleration and can be represented as

$$y = \ddot{u}_t = \ddot{u} + \ddot{u}_g = -2\xi\omega x_2 - \alpha\omega^2 x_1 - (1 - \alpha)\omega^2 x_3 \quad (9)$$

In this paper, results for model without stiffness and strength degradation are not shown owing to space limitation. Only model considering the stiffness and strength degradation characteristics are considered and the parameters are set as $m = 1, \xi = 39.47, n = 1, \alpha = \beta = \gamma = 0.5, A_0 = \nu_0 = \eta_0 = 1, \delta_A = \delta_\nu = \delta_\eta = 0.002$. The system is subjected to the El Centro earthquake with peak ground acceleration scaled to 331gal. Table 3 shows the linear and nonlinear models for either displacement output or acceleration output considered here. The value of delay time, which is determined by the same process as that in the previous section, is either 2 or 4 depending on the model. Therefore, neural network models in Table 3 are identified. Five neurons in the hidden layer with nonlinear activation function f are used in the nonlinear model for displacement predictor. The performance of model B is no more less than

Table 3. NN topology for Wen's hysteretic models ($\delta_A = \delta_\nu = \delta_\eta = 0.002$)

Model A (Linear)	$u(k) = \sum_{i=1}^2 a_i u(k-i) + \sum_{j=1}^2 b_j \dot{u}(k-j)$
Model B (Nonlinear)	$u(k) = \sum_{m=1}^5 (w_m \times f(\sum_{i=1}^2 a_{mi} u(k-i) + \sum_{j=1}^2 b_{mj} \dot{u}(k-j)))$
Model C (Linear)	$\ddot{u}_t(k) = \sum_{i=1}^2 a_i u(k-i) + \sum_{j=1}^4 b_j \dot{u}(k-j) + \sum_{s=1}^4 c_s \ddot{u}_t(k-s)$
Model D (Nonlinear)	$\ddot{u}_t(k) = w_1 \times f(\sum_{i=1}^2 a_i u(k-i) + \sum_{j=1}^4 b_j \dot{u}(k-j) + \sum_{s=1}^4 c_s \ddot{u}_t(k-s))$

that of model A. This means that nonlinear activation function is useless in the displacement prediction task. One neuron in the hidden layer with nonlinear activation function is used in the nonlinear model for acceleration predictor. It can be observed that simple linear model is sufficient to the acceleration prediction problem. The results using traditional neural network are also tabulated in the same table. Fig. 3 illustrates the comparison of the target output with the network output for model A with PGA=331gal.

Table 4. Network performance (E.I.) for Wen’s hysteretic models ($\delta_A = \delta_v = \delta_\eta = 0.002$)

	Testing case	GA+NN	NN
Model A (Linear)	EI ns 331gal	0.0012	0.0013
Model B (Nonlinear)	EI ns 331gal	0.0012	0.0021
Model C (Linear)	EI ns 331gal	0.0031	0.0031
Model D (Nonlinear)	EI ns 331gal	0.0031	0.0031

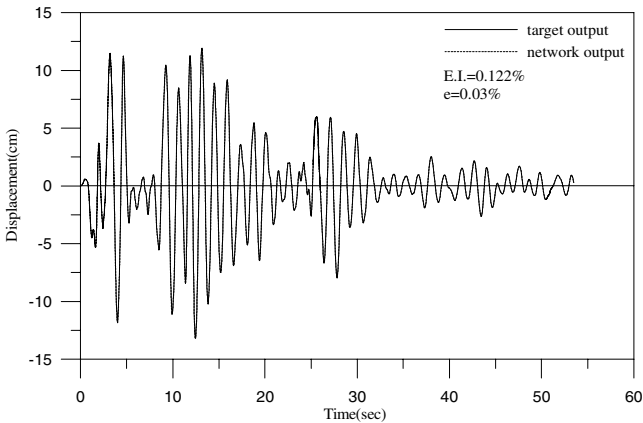


Fig. 3. Comparison of the target output with the network output for model A with PGA=331ga ($\delta_A = \delta_v = \delta_\eta = 0.002$)

6 Conclusions

Artificial neural networks, with their remarkable ability to gain information from complicated or imprecise data, can be used to derive model and extract parameters that are too complex to be noticed by either humans or other computer techniques. On the hand, genetic algorithms provide a very attractive computation method, as its implementation is relatively straightforward. Unlike many classical methods, there is no need to compute the derivatives with respect to the parameters. No initial guess is required. Furthermore, the fitness function can be defined in terms of the measurement

quantities directly. To avoid get in stuck in local optima when applying NN, merging GA to NN is very promising. Based on study of numerical examples in this paper, the following conclusion can be made:

- This paper presents an automatic procedure for pursuing the best initial weights of the NN without trial and error procedure.
- A set of new neural network topologies is presented to predict the system response for nonlinear structural systems such as Duffing's and Wen's nonlinear systems. Thus, the nonlinear input-output mapping ability is demonstrated. This is especially useful when the mathematical expression of the structural dynamic system is complex.
- In the future, the set of neural network topologies developed in this paper can be employed to replace the procedure for solving the governing (differential) equation when GA is used to identify the system dynamic parameters. As a result, an efficient identification technique combining GA and ANN can be developed.

Acknowledgments. This paper is the partial result of research project sponsored by National Science Council, Taiwan, through Grant No. NSC 93-2211-E-324-016. The support is appreciated by authors.

References

1. Jovanovi , O.: **Identification of Dynamic System Using Neural Network.** The Scientific Journal FACTA UNIVERSITATIS Series : Architecture and Civil Engineering **31** (1997) 525-53
2. Loh, C.H., and Huang, C.C.: **Nonlinear Identification of Dynamic Systems Using Neural Networks.** Computer-Aided Civil and Infrastructure Engineering **16** (1) (2001) 28-41
3. Wang, G.S., and Lin, H.H.: **Application of Genetic Algorithm to Structural Dynamic Parameter Identification.** Journal of the Chinese Institute of Civil Engineering and Hydraulic Engineering **17** (2) (2005) 281 -291
4. Holland, J.H.: **Outline for a Logical Theory of Adaptive Systems.** Journal of the Association for Computing Machinery **3** (1962) 297-314

Approximation Capability Analysis of Parallel Process Neural Network with Application to Aircraft Engine Health Condition Monitoring

Gang Ding and Shisheng Zhong

School of Mechatronics Engineering, Harbin Institute of Technology,
Harbin 150001, P.R. China
dingganghit@163.com

Abstract. Parallel process neural network (PPNN) is a novel spatio-temporal artificial neural network. The approximation capability analysis is very important for the PPNN to enhance its adaptability to time series prediction. The approximation capability of the PPNN is analyzed in this paper, and it can be proved that the PPNN can approximate any continuous functional to any degree of accuracy. Finally, the PPNN is utilized to predict the iron concentration of the lubricating oil in the aircraft engine health condition monitoring to highlight the approximation capability of the PPNN, and the application test results also indicate that the PPNN can be used as a well predictive maintenance tool in the aircraft engine condition monitoring.

1 Introduction

Parallel process neural network (PPNN) is a novel spatio-temporal artificial neural network. From a point view of architecture, PPNN is similar to the traditional parallel neural network. The major difference is that the inputs and the corresponding weights of the PPNN are time-varying functions. The major characteristics which distinguish the PPNN from the traditional multilayer feedforward process neural network (MFPNN) lies in the fact that there has an added link from the input layer direct to the output layer of the PPNN. This link is parallel to the link from the hidden layer to the output layer of the PPNN. It has been shown that the PPNN has a faster convergence speed and higher accuracy than the MFPNN [1,2].

In 1989, Hornik and Funahashi proved respectively that multilayer feedforward neural networks can approximate any continuous function to any degree of accuracy [3,4]. It seems that artificial neural networks have great potential to high nonlinear and uncertain systems. It also indicates that the approximation capability analysis is very important for the PPNN to enhance its adaptability to practical engineering applications. Thus, the approximation capability of the PPNN is analyzed in this paper. It can be proved that the PPNN can approximate any continuous functional.

The plan of this paper is as follows: In section 2, the PPNN model and its learning algorithm are reviewed. In section 3, the approximation capability of the PPNN is analyzed and proved. In section 4, the PPNN is utilized to predict the iron concentration

in the aircraft engine lubricating oil monitoring, and the test results highlight the approximation capability of the PPNN. Conclusions are given in section 5.

2 The PPNN Model and Its Learning Algorithm

2.1 Process Neuron

The process neuron as depicted in Fig (1) is composed of three sections: inputs, an activation unit and output [5].

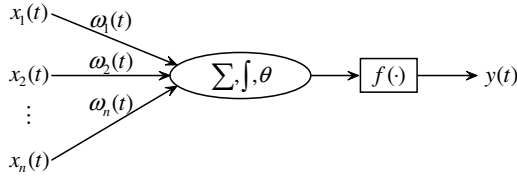


Fig. 1. Sketch diagram of process neuron model

The output of the process neuron model can be expressed as

$$y(t) = f\left(\sum_{i=1}^n \int_0^t \omega_i(t) x_i(t) dt - \theta\right) \tag{1}$$

Where $x_i(t) \in C[0, T]$ is the i -th input function, $\omega_i(t)$ is the i -th weight function. θ is the threshold. $f(\cdot)$ is the activation function.

2.2 The PPNN Model

The PPNN employed in this paper is comprised of three layers. The topological structure of the PPNN is $n - m - 1$, which is depicted in Fig (2).

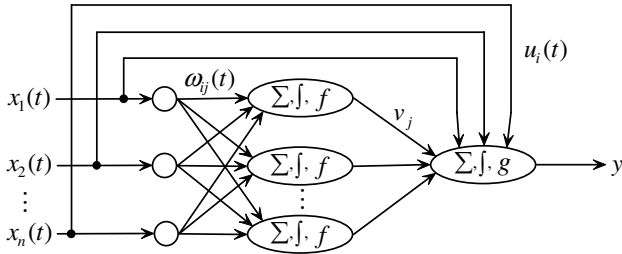


Fig. 2. The topological structure of the PPNN

Suppose that the activation function of the output layer is a linear function, thus the output of the PPNN model can be expressed as

$$y = \sum_{j=1}^m v_j f \left(\int_0^T \sum_{i=1}^n \omega_{ij}(t) x_i(t) dt - \theta_j^{(1)} \right) + \int_0^T \sum_{i=1}^n x_i(t) u_i(t) dt - \theta^{(2)} \quad (2)$$

Where $\omega_{ij}(t)$ is the connection weight function between the j -th process neuron in the hidden layer and the i -th unit in the input layer, $u_i(t)$ is the connection weight function between the i -th unit in the input layer and the unit in the output layer, $\theta_j^{(1)}$ is the threshold of the j -th process neuron in the hidden layer, θ is the threshold of the unit in the output layer.

2.3 Learning Algorithm

According to Weierstrass approximation theorem [6], $x_i(t)$, $\omega_{ij}(t)$ and $u_i(t)$ can be expanded as $x_i(t) = \sum_{p=1}^P a_i^{(p)} b_p(t)$, $\omega_{ij}(t) = \sum_{l=1}^P \omega_{ij}^{(l)} b_l(t)$ and $u_i(t) = \sum_{l=1}^P u_i^{(l)} b_l(t)$, respectively,

where $a_i^{(p)}, \omega_{ij}^{(l)}, u_i^{(l)} \in R$, and $\int_0^T b_p(t) b_l(t) dt = \begin{cases} 1 & l = p \\ 0 & l \neq p \end{cases}$. Thus,

$$y = \sum_{j=1}^m v_j f \left(\sum_{i=1}^n \sum_{p=1}^P a_i^{(p)} \omega_{ij}^{(p)} - \theta_j^{(1)} \right) + \sum_{i=1}^n \sum_{p=1}^P a_i^{(p)} u_i^{(p)} - \theta^{(2)} \quad (3)$$

Suppose that d_s is the s -th desired output, and y_s is the corresponding actual output of the PPNN, $s = 1, \dots, S$. Then, the error of the PPNN can be written as

$$E = \frac{1}{2} \sum_{s=1}^S (y_s - d_s)^2 = \frac{1}{2} \sum_{s=1}^S \left(\sum_{j=1}^m v_j f \left(\sum_{i=1}^n \sum_{p=1}^P a_{is}^{(p)} \omega_{ij}^{(p)} - \theta_j^{(1)} \right) + \sum_{i=1}^n \sum_{p=1}^P a_i^{(p)} u_i^{(p)} - \theta^{(2)} - d_s \right)^2 \quad (4)$$

For analysis convenience, z_{js} is defined as: $z_{js} = \sum_{i=1}^n \sum_{p=1}^P a_{is}^{(p)} \omega_{ij}^{(p)} - \theta_j^{(1)}$. According to the gradient descent method, the learning rules are defined as follows

$$\begin{cases} \omega_{ij}^{(p)}(k+1) = \omega_{ij}^{(p)}(k) + \alpha \Delta \omega_{ij}^{(p)}, u_i^{(p)}(k+1) = u_i^{(p)}(k) + \gamma \Delta u_i^{(p)} \\ v_j(k+1) = v_j(k) + \beta \Delta v_j \\ \theta_j^{(1)}(k+1) = \theta_j^{(1)}(k) + \eta \Delta \theta_j^{(1)}, \theta^{(2)}(k+1) = \theta^{(2)}(k) + \lambda \Delta \theta^{(2)} \end{cases} \quad (5)$$

Where $\alpha, \beta, \gamma, \eta, \lambda$ are learning rates, k is iteration.

$\Delta \omega_{ij}^{(p)}, \Delta v_j, \Delta u_i^{(p)}, \Delta \theta_j^{(1)}, \Delta \theta^{(2)}$ can be calculated as follows

$$\begin{cases} \Delta \omega_{ij}^{(p)} = -\frac{\partial E}{\partial \omega_{ij}^{(p)}} = -\sum_{s=1}^S (y_s - d_s) f'(z_{js}) a_{is}^{(p)}, \Delta u_i^{(p)} = -\frac{\partial E}{\partial u_i^{(p)}} = -\sum_{s=1}^S (y_s - d_s) a_{is}^{(p)} \\ \Delta v_j = -\frac{\partial E}{\partial v_j} = -\sum_{s=1}^S (y_s - d_s) f(z_{js}) \\ \Delta \theta_j^{(1)} = -\frac{\partial E}{\partial \theta_j^{(1)}} = -\sum_{s=1}^S (y_s - d_s) f'(z_{js}) (-1), \Delta \theta^{(2)} = -\frac{\partial E}{\partial \theta^{(2)}} = -\sum_{s=1}^S (y_s - d_s) (-1) \end{cases} \quad (6)$$

3 Approximation Capability Analysis

In this section, the approximation capability of the PPNN is described in the following Theorem 1. In order to prove this theorem, we begin with a definition.

Definition 1. Suppose that K is a compact set in R^n , $F(\bullet)$ is a continuous function in $C(K)$. Suppose that $X(t) = (x_1(t), \dots, x_n(t))^T$, where $x_i(t) \in C[0, T]$, $i = 1, \dots, n$. For any $t_1, t_2 \in [0, T]$, if $|x_i(t_1) - x_i(t_2)| \leq L|t_1 - t_2|$, where $L \geq 0$, then $x_i(t)$ satisfies the Lipschitz condition, if $\|X(t_1) - X(t_2)\| \leq L_x|t_1 - t_2|$, where $L_x \geq 0$, then $X(t)$ satisfies the Lipschitz condition, If $|F(X(t_1)) - F(X(t_2))| \leq L_f|t_1 - t_2|$, where $L_f \geq 0$, then $F(\bullet)$ satisfies the Lipschitz condition.

Theorem 1. For any continuous functional $G(X(t))$ and any $\varepsilon > 0$, there exists a PPNN P such that $|G(X(t)) - P(X(t))| < \varepsilon$.

Proof. Let $\sum^n(F) = \{f : U \rightarrow V \mid f(X(t)) = \int_0^T F(X(t))dt, X(t) \in U \subset K, f(X(t)) \in V \subset R\}$, then $\sum^n(F)$ is a functional space. Without loss of generality, suppose that $G(X(t))$ is a continuous functional in $\sum^n(F)$, then $G(X(t)) = \int_0^T F(X(t))dt$, and satisfies the Lipschitz condition.

Let $T = 1$, suppose that $[0, 1]$ contains N equally space nodes $t_h = \frac{h}{N}$, $h = 1, \dots, N$, and $t_0 = 0$, then $G(X(t)) = \sum_{h=1}^N \int_{\frac{h-1}{N}}^{\frac{h}{N}} F(X(t))dt$. Applying the mean value theorem for integrals to $F(X(t))$, there exists a $\xi_h \in [\frac{h-1}{N}, \frac{h}{N}]$ such that $\int_{\frac{h-1}{N}}^{\frac{h}{N}} F(X(t))dt = \frac{1}{N} F(X(\xi_h))$. Thus, $G(X(t)) = \frac{1}{N} \sum_{h=1}^N F(X(\xi_h))$. Suppose that $\tilde{G}(X(t)) = \frac{1}{N} \sum_{h=1}^N F(X(t_h))$.

$$\left| G(X(t)) - \tilde{G}(X(t)) \right| \leq \frac{1}{N} \sum_{h=1}^N L_f \|X(\xi_h) - X(t_h)\| \leq \frac{1}{N} \sum_{h=1}^N L_f L_x |\xi_h - t_h| \leq \frac{L_f L_x}{N} \quad (7)$$

Then, $G(X(t))$ can be expressed as

$$G(X(t)) = \frac{1}{N} \sum_{h=0}^N F(X(t_h)) + O\left(\frac{1}{N}\right) \quad (8)$$

Because $F(\bullet) \in C(K)$, according to Hornik's and Funahashi's papers [3,4] and He's paper [7], $F(\bullet)$ can be approximated by a traditional parallel neural network. It is obvious that the traditional parallel neural network is a special case of the PPNN. Thus, there exists a PPNN P_h such that $|F(X(t_h)) - P_h(X(t_h))| < \varepsilon_h$ for any $\varepsilon_h > 0$.

For any $\varepsilon > 0$, let $\varepsilon_h = \frac{\varepsilon}{2N}$, there exists N_0 , when $N > N_0$, according to Equation (8), $\left| G(X(t)) - \frac{1}{N} \sum_{h=0}^N F(X(t_h)) \right| < \frac{\varepsilon}{2}$.

Let $P(X(t)) = \frac{1}{N} \sum_{h=0}^N P_h(X(t_h))$, then

$$\left| G(X(t)) - P(X(t)) \right| \leq \left| G(X(t)) - \frac{1}{N} \sum_{h=0}^N F(X(t_h)) \right| + \left| \frac{1}{N} \sum_{h=0}^N F(X(t_h)) - \frac{1}{N} \sum_{h=0}^N P_h(X(t_h)) \right| < \varepsilon$$

Q.E.D.

4 Application Test

In this section, the PPNN is utilized to predict the iron (Fe) concentration of the lubricating oil in the aircraft engine condition monitoring to highlight the approximation capability of the PPNN. Aircraft engine is a complicated nonlinear system, which operates under high temperature and speed conditions [8]. The lubrication system is an important working system of the aircraft engine. The lubricating oil monitoring is essential in terms of the flight safety and also for reduction of the preventive maintenance cost. The monitoring analysis of the lubricating oil taken from the aircraft engine gives an indication of its suitability for continued use and provides important information about the health condition of the lubricated components within the aircraft engine. The sampling interval of the data used in this paper is about 24 hours. Thus, we get a Fe concentration time series with 155 discrete points such as $\{Fe_j\}_{j=1}^{155}$, which is depicted in Fig (3).

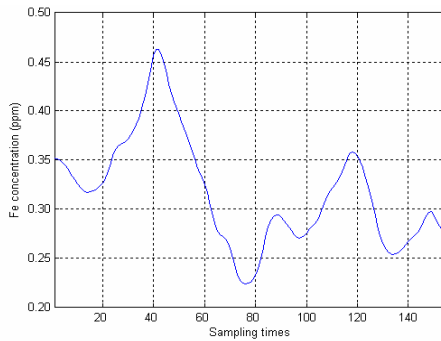


Fig. 3. Fe concentration time series

$(Fe_i, Fe_{i+1}, \dots, Fe_{i+4})$ can be used to generate an input function $IF_i, i = 1, \dots, 150$, and Fe_{i+5} can be used as the output corresponding to IF_i . Thus, we can get 150 couples of samples such as $\{IF_i, Fe_{i+5}\}_{i=1}^{150}$. $\{IF_i, Fe_{i+5}\}_{i=1}^{100}$ are selected to train the PPNN. The topological structure of the used PPNN is 1-10-1. The orthogonal Legendre basis

functions are selected to expand the input functions and the connection weight functions. The error goal is set to 10^{-6} , and the learning rate is set to 0.01, the max iteration number is set to 1000. After 94 iterations, the PPNN has converged. $\{IF_i, Fe_{i+5}\}_{i=101}^{150}$ are selected to test the PPNN. The test results as shown in Fig (4) indicate that the PPNN seems to perform well and appears suitable for using as a predictive maintenance tool.

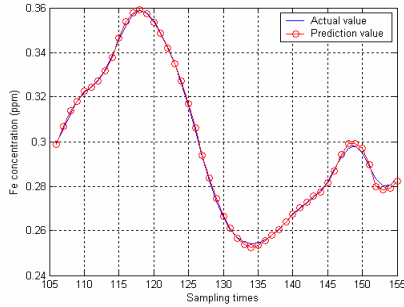


Fig. 4. Fe concentration time series prediction by the PPNN

5 Conclusion

The PPNN is a novel spatio-temporal artificial neural network. In this paper, the topological structure of the PPNN and its learning algorithm based on the expansion of the orthogonal basis functions are reviewed firstly. Subsequently, the approximation capability of the PPNN is analyzed, it has been proved that the PPNN can approximate any continuous functional to any degree of accuracy. Finally, the PPNN is utilized to predict the iron concentration of the lubricating oil in the aircraft engine condition monitoring to highlight the approximation capability of the PPNN, and the application test results also indicate that the PPNN can be used as a predictive maintenance tool for the aircraft engine condition monitoring.

Acknowledgement. This study was supported by the National Natural Science Foundation of China under Grant No.60572174.

References

1. Zhong S.S., Ding G.: Research on Double Parallel Feedforward Process Neural Networks and Its Application. *Control and Decision* **7** (2005) 764-768
2. Zhong S.S., Ding G., Su D.Z.: Parallel Feedforward Process Neural Network with Time-varying Input and Output Functions. *Lecture Notes in Computer Science* **3496** (2005) 473-478.
3. Hornik, K., Stinchcombe, M., White, H.: Multilayer Feedforward Networks Are Universal Approximators. *Neural Networks* **2** (1989) 359-366
4. Funahashi, K.: On the Approximate Realization of Continuous Mappings by Neural Networks, *Neural Networks* **2** (1989) 183-192

5. He X.G., Liang J.Z.: Some Theoretical Issues on Process Neural Networks. *Engineering Science* **2** (2000) 40-44
6. Jeffreys, H., Jeffreys, B.S.: *Methods of Mathematical Physics*. 3rd edn. Cambridge University Press, Cambridge (1988) 446-448
7. He M.Y., Bogner, R.E.: Algorithms and Experiments on Convergence for Double Feedforward Neural Networks. *Proc. of ICIIPS'92, Beijing* (1992) 109-112
8. Ding G., Zhong S.S.: Time Series Prediction by Parallel Feedforward Process Neural Network with Time-varied Input and Output Functions. *Neural Network World* **2** (2005) 137-147

Neural Network-Based Position Sensorless Control for Transverse Flux Linear SRM*

Zhongsong Chen¹, Baoming Ge¹, and Aníbal T. de Almeida²

¹ School of Electrical Engineering, Beijing Jiaotong University, Beijing 100044, China
bm-ge@263.net

² Institute of System and Robotics, University of Coimbra, 3030 Coimbra, Portugal
adealmeida@isr.uc.pt

Abstract. The purpose of this paper is to present a sensorless control method for transverse flux linear switched reluctance motor (TFLSRM) based on position estimation by employing a public back propagation neural network (BPNN). The system characterizes with that only one public BPNN is needed to transform the winding current and flux linkage of each phase into each segmental position signal, then final total position is obtained by combining each segmental position signal. The starting position is derived from the comparison of the calculated position values based on currents and flux linkages of all phases. A TFLSRM with three phases is used to verify the validity of the proposed method, the established position sensorless control system with a BPNN is simulated. The results illustrate the excellent performance of the BPNN-based position sensorless control system.

Keywords: linear reluctance motor, neural network, position estimation.

1 Introduction

Transverse flux linear switched reluctance motor (TFLSRM) has demonstrated huge potential in the application of railway vehicles. As an attractive alternative to rotary motor, TFLSRM is becoming preferable in linear motion [1].

The essential position signal of TFLSRM makes the position sensor be employed in traditional method. However, a physical position sensor will produce many problems in practical applications, such as increasing complexity of whole system, decreasing reliability etc. These factors justify the necessity for development of a high-grade position sensorless control for SRM drives [2]. Recently, many sensorless algorithms have been proposed and validated for control of rotary SRMs [3], but a paucity of literature introduces the position sensorless control of TFLSRM.

In this paper, a novel strategy for position sensorless control for TFLSRM is proposed. Position signal is estimated online by using an established back propagation

* The work was partially supported by the Key Project of Chinese Ministry of Education under Grant #2004104051, and in part by the Delta Science & Technology Educational Development Program Grant # DREG2005006.

neural network (BPNN), where the phase current and flux linkage are two inputs of the network. One major characteristic is that, only one BPNN is used in this strategy. Its starting operation and approach of position estimation are presented in detail. A three-phase TFLSRM is used in simulation to verify the proposed method.

2 Structure of TFLSRM

The 2-dimensional views of the proposed TFLSRM are provided in Fig.1, where the part including phase windings is referred as the primary side and another part without conductor or permanent magnets is called the secondary side. It can be noticed that the primary side is fixed in the movable bogie with equal pole pitches and the secondary side is fixed in the track with equal pole pitches. The machine supplies a thrust force in the Z direction. The three-phase windings located on the machine moving bogie are used to provide the required electromagnetic forces to the system. The primary side is composed of four sectors, and each sector includes three poles. Four windings of each phase respectively located on four sectors, can be connected in series or parallel to achieve high efficiency for whole ranges of different loads and speeds.

Its operating principle is similar to conventional rotary SRM. For example, for the continuous forward movement in the positive Z direction, the excitation sequence B-C-A is necessary for the motion in the increasing inductance region. On the contrary, the excitation sequence C-B-A will make the machine move in backward along the negative Z direction [1].

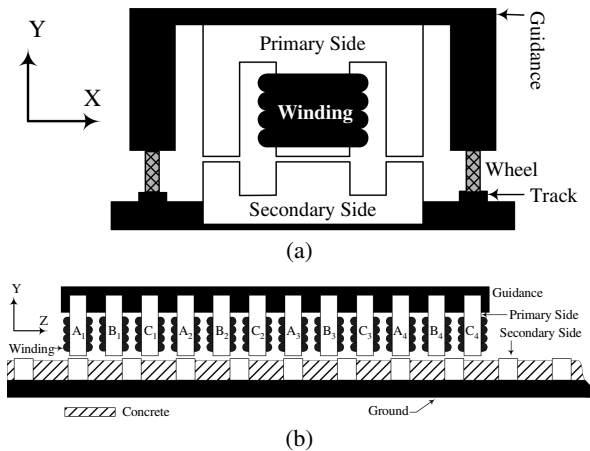


Fig. 1. 2-D views of the designed TFLSRM, (a) gives view from the Y-X plane, and (b) shows view from the Y-Z plane

3 Neural Network-Based Control System

Block diagram of the control system is illustrated in Fig. 2. Stator current and voltage of each phase are measured directly to calculate the flux linkage by

$$\lambda_j = \int (u_j - Ri_j) dt, \quad j \in (a, b, c) . \tag{1}$$

where λ_j , u_j , and i_j are respective flux linkage, voltage, and stator current of phase j , R is phase resistance.

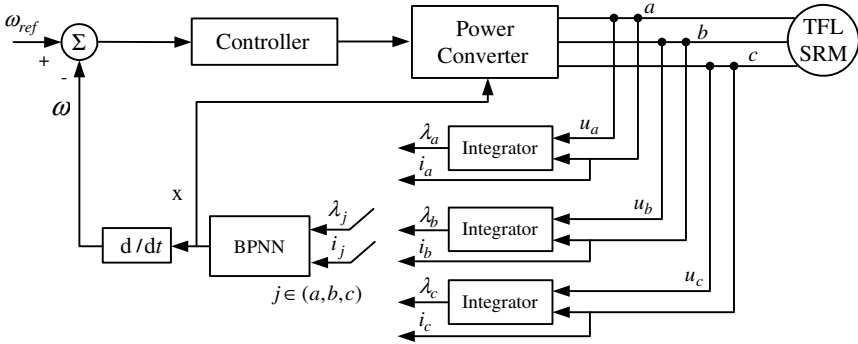


Fig. 2. Block diagram of the proposed control system

Two inputs of BPNN are stator current and flux linkage of each phase in turn. Switch is used to choose one phase from three phases so that the trained neural network estimates the position via only one phase any time. The calculated position signal is fed back to controller for the purpose of commutation and speed control.

3.1 Starting Operation

Trust force of TFLSRM is direct proportion with derivative of inductance with respect to position, and the starting operation should be based on the characteristic of inductance versus position, as shown in Fig. 3.

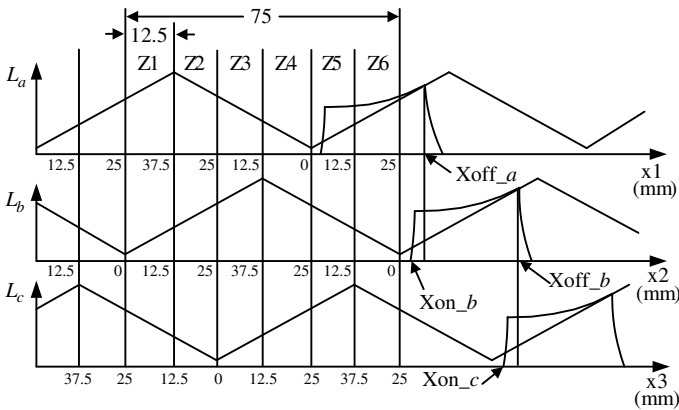


Fig. 3. The principle of starting operation

We define the zero-position at the minimum inductance, and 37.5 mm at the maximum inductance, according to the size of the designed TFLSRM. One periodical position of 75 mm includes six sections such as Z1, Z2, Z3, Z4, Z5, and Z6, which is useful to determine starting phases producing the desired thrust force in a given direction of operation. The position of each phase can be obtained by using the trained BPNN in turn. A tiny current is applied to the winding of each phase, and the measured current and the calculated flux linkage are used as inputs of the established network, the network output is the position corresponding to the presented phase. Three position signals of x_1 , x_2 , and x_3 can be obtained for a three-phase motor during starting operation. Table 1 shows a form to determine the operating phases for startup.

Table 1. A form to determine the operating phases for startup

Positions Sections	x_1 (mm)	x_2 (mm)	x_3 (mm)	Operating phases	
				Forward	Backward
Z5	(0, 12.5)	(12.5, 25)	(25, 37.5)	a, c	b
Z6	(12.5, 25)	(0, 12.5)	(25, 37.5)	a	b, c
Z1	(25, 37.5)	(0, 12.5)	(12.5, 25)	a, b	c
Z4	(0, 12.5)	(25, 37.5)	(12.5, 25)	c	a, b
Z3	(12.5, 25)	(25, 37.5)	(0, 12.5)	b, c	a
Z2	(25, 37.5)	(12.5, 25)	(0, 12.5)	b	a, c

3.2 Estimation of Position in Operation

The regular estimation of position in operation is finished by using the trained public BPNN. As shown in Fig. 3, we define X_{on_b} as turn-on position of phase b and X_{off_b} as turn-off position of phase b , and $(X_{off_b} - X_{on_b}) > 25$ mm. Therefore, the current and flux linkage of phase a are employed for estimating position before the phase a is turned off at X_{off_a} , although the phase b has been turned on at X_{on_b} . The task of estimating position is transferred to phase b at X_{off_a} up to X_{off_b} , which is achieved via a switch that ensures one effective phase any time during operation, shown in Fig. 2.

4 Position Model Based on BPNN

In the designed position model based on BPNN, two inputs are respective phase current and flux linkage, the estimated position is output, and one hidden layer is enough. The functions used in the hidden-layer are unipolar sigmoid functions, and in the output-layer is pure linear function.

With the training samples obtained from the characteristic of position versus phase current versus flux linkage, the BPNN-based position model is trained so that the sum squared network error reduces to 0.0001. Fig. 4 presents the characteristic of the trained BPNN when the current and flux linkage are inputs and position is output, which will be applied to control system for estimating position online in Fig. 2.

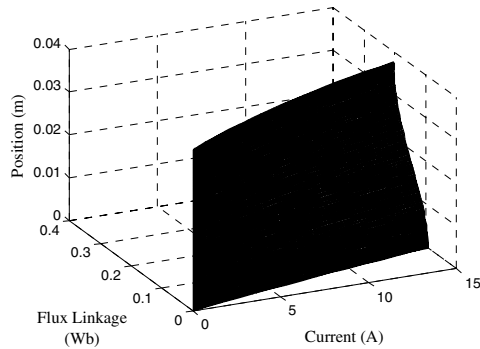


Fig. 4. Characteristic of the trained BPNN

5 Simulation Results

The simulation is carried out to verify the proposed control system in Fig. 2. The position signal is estimated by using the trained BPNN above. Fig. 5 shows the dynamic response of speed when the motor operates at a given speed reference of 1 m/s. The steady phase currents are given in Fig. 6. The BPNN output in Fig. 2 is shown in Fig. 7, where the maximum value of BPNN output is the position at turning off, and the minimum value corresponds to the starting point of the estimated position when using a new phase that is operating.

Fig. 8 is obtained by combining the estimated position signals of Fig. 7. The actual position is given in Fig. 8 for comparison with the estimated position. The magnified positions are presented in Fig. 9, which is obtained by employing position period of 75 mm to Fig. 8. It can be seen that one public neural network provides a perfect performance for the position detection of TFLSRM. The control system presents a good behavior when running in the position sensorless.

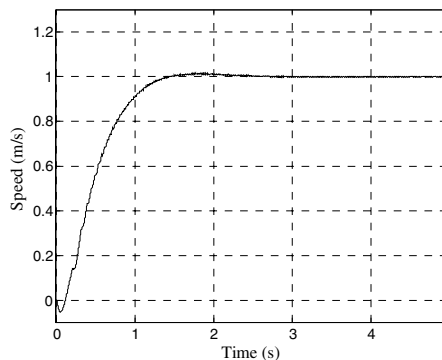


Fig. 5. Speed response

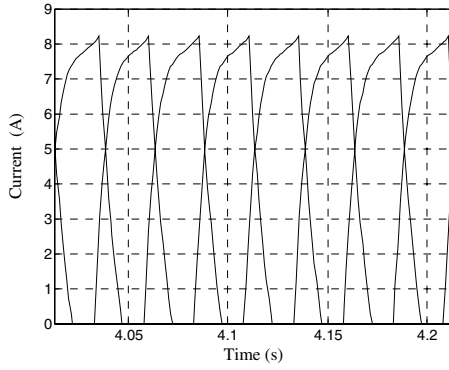


Fig. 6. Phase currents

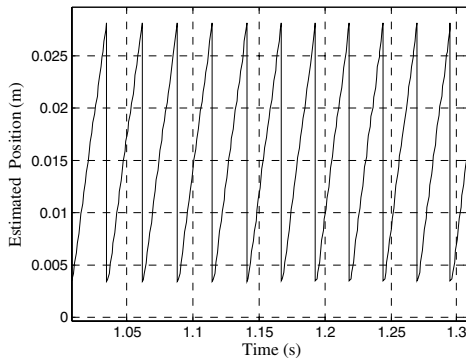


Fig. 7. Output of the trained BPNN in Fig. 2

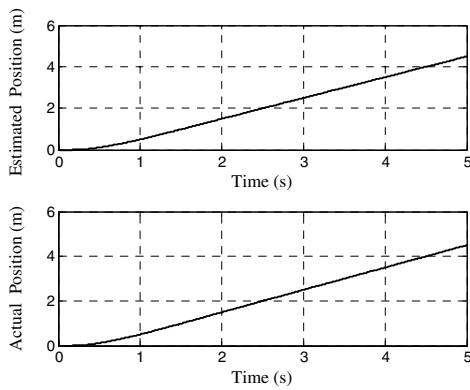


Fig. 8. Estimated position and actual position

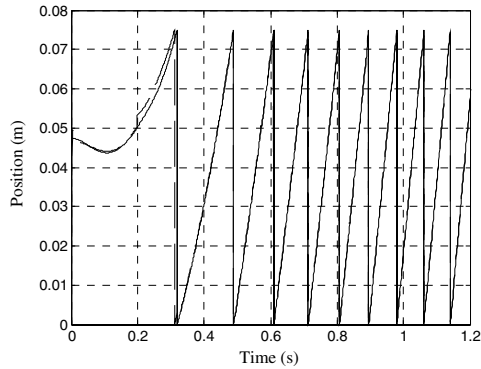


Fig. 9. Magnified positions for the estimated position (*dotted line*) and the actual position

6 Conclusions

Due to its simple construction, low cost, reliable operation, and high efficiency, the TFLSRM is suitable for many linear motions. The paper presented a sensorless control method for TFLSRM based on position estimation by employing a public BPNN that transformed the winding current and flux linkage of each phase into each segmental position signal, then final total position is obtained by combining each segmental position signal. A TFLSRM with three phases is employed in simulation. The simulated results showed that one public neural network have provided a perfect performance for the position detection of TFLSRM. The control system presented a good behavior when operating in the position sensorless.

References

1. Ge, B.M., Zhang, Y.H., Yu, X.H., Nan, Y.H.: Simulation Study of Transverse Flux Linear Switched Reluctance Drive. Proceedings of the Eighth International Conference on Electrical Machines and Systems **1** (2005) 608 – 613
2. Fahimi, B., Emadi, A., Sepe, R.B., Jr.: Four-Quadrant Position Sensorless Control in SRM Drives over the Entire Speed Range Power Electronics. IEEE Transactions Power Electronics **20** (2005) 154 - 163
3. Hudson, C., Lobo, N. S., Crishman, R.: Sensorless Control of Single Switch Based Switched Reluctance Motor Drive using Neural Network. Proceedings of 30th Annual Conference of IEEE Industrial Electronics Society **3** (2004) 2349 - 2354

Tourism Room Occupancy Rate Prediction Based on Neural Network

Junping Du¹, Wensheng Guo², and Ruijie Wang²

¹ Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia, School of Computer Science and Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
junpingdu@126.com

² School of Information Engineering, University of Science and Technology, Beijing 100083, China
Gws@126.com, sxwrj@126.com

Abstract. We studied how to use neural network in the tourism room occupancy rate prediction in Beijing. We gave the result of prediction on room occupancy rate. The results of the experiment showed that the prediction of the room occupancy rate made by neural network is superior to the two methods of regression and naïve extrapolation which are often used.

1 Introduction

In china, tourism industry and hotel industry keep to a rapid growth in the last few years, the hotel number and room number also grow very fast, making the necessity for the application of information technology in tourism and hotel industry. The research of room occupancy rate prediction in this paper can help hotel manger make high efficiency strategy plan, and provide support to the decision process of hotel manager. This paper studied how to apply neural network to the establishment of Beijing hotel room occupancy rate model, and gave the result of the prediction. The result done by neural network were compared with the result done by multiple regression and naïve extrapolation. The application of our research result to predict room occupancy rate of Beijing hotel industry was given.

Neural network is the intelligent calculation system that simulates the process ability of human brain. It is the information technology that reveals knowledge based on the parallel process and the pattern recognition of past experiences or examples. Its pattern recognition function makes it a very good tool for classification and prediction in the business application. Neural network can better recognize high level characteristic. It has been proved that if the training set is small and in the prediction high level white noise, it outperforms the normal statistic model.

This feature is especially useful in the prediction of room occupancy rate. Because the sample number is relatively small in the training set. Another advantage of using neural network in prediction is that it can catch nonlinear sample in the training set.

Unlike other time series model, neural network can reach lower percentage absolute error, accumulative relative error and square root error on the nonlinear tourist action prediction than the linear prediction.

2 Room Occupancy Rate Prediction

Room occupancy rate model is set up with neural network. Six samples were randomly selected in the ten samples, the rest were used to test the prediction. The input variables are: NoT: tourists' number; ASL: average stay time; NoH: hotels' number; TpR: tourists in each room; PHA: percentage of hotel occupancy rate. Input variables are correlative to the demand and supply of the hotel rooms. NoT, ASL, and PHA are requirement indexes, and also are the potential profit. PHA is the tourist percentage of the hotel. NoH and NoR are the provision indexes, TpR is the supply and demand proportion that directly correlative to room occupancy rate. ROR is the output variable, which represents room occupancy rate. Table 1 shows the experimental results of room occupancy rate.

Table 1. Experimental Results of Room Occupancy rate

Actual occupancy rate/%	Estimated occupancy rate/% (multiple regression)	Estimated occupancy rate/% (naïve extrapolation)	Estimated occupancy rate/% (neural network)
69.7	52.30	83.38	69.7
68.0	85.99	82.92	69.0
27.3	68.59	83.85	64.1
70.0	57.68	84.15	74.9

3 Experimental Results

Multiple regression and naïve extrapolation were employed to predict room occupancy rate based on the training set. Table1 shows the experimental results of the three different models. The output of the three prediction models is based on mean percentage error (\mathcal{E}), acceptable output percentage (Z), and normalized cross-correlation (R). \mathcal{E} is a relative measurement used for comparison across the testing data because it is easy to interpret, independent of scale, reliable and valid. Z is used as a relative measurement for acceptance level. Z was assigned a value of 5 percent. R is a measure of the closeness of the observed and estimated occupancy rates. Each of these measurements is defined as follows:

$$\epsilon = \frac{\sum_{i=1}^n \frac{|X_i - Y_i|}{Y_i}}{n} \times 100\%,$$

$$Z = \frac{\sum_{i=1}^n i}{n} \times 100\% \text{ for } \begin{cases} i = 1 \dots \text{if } \frac{|X_i - Y_i|}{Y_i} \leq 5\%, \\ i = 0 \dots \text{otherwise,} \end{cases}$$

$$R = \frac{\sum_{i=1}^n (X_i Y_i)}{\left[\sum_{i=1}^n (X_i)^2 \sum_{i=1}^n (Y_i)^2 \right]^{0.5}},$$

X_i and Y_i represent the estimated and actual occupancy rates for $i = 1, 2, 3, 4$. The values of Z and R are given in table 2. Data in table 3 are the relative percentage error of the three prediction models. Two Mann-Whitney U tests were conducted, and the values of U statistic are presented in Table 4.

Table 2. Comparison of Some Prediction Models

	ϵ	Z	R
Neural network	3.1	80	0.999
Multiple regression	12.8	50	0.996
Naïve extrapolation	6.7	50	0.997

Table 3. Relative Percentage Error of Test Samples

Samples	Neural network/%	Multiple regression/%	Naïve extrapolation/%
1(year 2001)	0.0	24.0	20.2
2(year 2002)	1.5	25.0	20.5
3(year 2003)	137.4	151.8	207.4
4(year 2004)	6	7.0	20.2
Average	3.75	18.67	20.3

Table 4. Tests For Differences in the Relative Percentage Error

Comparison	Mann-Whitney U value
Neural network vs multiple regression	23.5
Neural network vs naïve extrapolation	27

4 Results Analysis

It can be seen that the estimated room occupancy rates from a neural network are very close to the actual values, so the prediction output from a neural network is accurate with an acceptable amount of error. Low mean percentage error shows that the deviation between the estimated value by neural network and the actual value is very small. Therefore, a neural network succeeds in achieving 80 percent of output within the acceptable range, and the normalized cross-correlation is almost 1, this demonstrates the close relationship between the estimated results and the actual hotel data.

As shown in table 2, a neural network outperforms the multiple regression and naïve extrapolation models in terms of mean percentage error, acceptable output percentage and normalized cross-correlation. The non-parametric Mann-Whitney U statistic values in table 4 were both significant at the 0.05 level of a one-tailed test, meaning that a neural network outperforms both multiple regression and naïve extrapolation models in room occupancy rate prediction.

5 Conclusion

The procedures of room occupancy rate prediction for the Beijing hotel industry with neural network were presented. Data are divided into a training data set and a testing data set. By comparing with the actual data, prediction efficiency of neural network is demonstrated. Prediction efficiency of neural network outperforms those of multiple regression and naïve extrapolation, which shows the feasibility of applying neural network prediction model to Beijing tourism hotel industry. A future research is to include more dependent variables to determine the room occupancy rate prediction efficiency of a neural network. For instance, it is natural to believe that government policies, weather conditions and national wealth can play an important role in determining tourist arrivals, leading to a significant change in room occupancy rates. However, some factors such as government policies and weather conditions are dynamic in a continuous fashion. Hence it could be difficult to provide a commonly acceptable measurement for these factors.

Acknowledgement

This work was supported by Beijing Natural Science Foundation of China(4042012), Key Project of Beijing Educational Committee (KZ200510011009) and Project of National Science Foundation of China(60442003).

References

1. Mao, X., Xu, X.: Software Reuse based on Agent. *Computer Engineering and Science* 5 (2000) 88-91
2. Du, J.: The Application of Intelligent Agents. *Tsinghua-KAIST Joint Workshop Proceedings* (2002)

3. Guo, W., Du, J., Yin, Y.: Analysis Method for Holiday Tourism Information Based on Data Mining. The Third International Conference on Computational Intelligence, Robotics and Autonomous Systems, Singapore (2005)
4. Du, J.: The Study on the Distributed National Golden Week Holiday Tourism Forecast System. The 2003 International Conference on Information and Knowledge Engineering, Las Vegas, Nevada, USA (2003)

Recurrent Neural Networks on Duty of Anomaly Detection in Databases

Jaroslav Skaruz¹ and Franciszek Sereczynski^{1,2,3}

¹ Institute of Computer Science, University of Podlasie,
Sienkiewicza 51, 08-110 Siedlce, Poland
jaroslav.skaruz@ap.siedlce.pl

² Polish-Japanese Institute of Information Technology,
Koszykowa 86, 02-008 Warsaw

³ Institute of Computer Science, Polish Academy of Sciences,
Ordona 21, 01-237 Warsaw, Poland
serec@ipipan.waw.pl

Abstract. In the paper we present a new approach based on application of neural networks to detect SQL attacks. SQL attacks are those attacks that take advantage of using SQL statements to be performed. The problem of detection of this class of attacks is transformed to time series prediction problem. SQL queries are used as a source of events in a protected environment. To differentiate between normal SQL queries and those sent by an attacker, we divide SQL statements into tokens and pass them to our detection system, which predicts the next token, taking into account previously seen tokens. In the learning phase tokens are passed to recurrent neural network (RNN) trained by backpropagation through time (BPTT) algorithm. Teaching data are shifted by one token forward in time with relation to input. The purpose of the testing phase is to predict the next token in the sequence. All experiments were conducted on Jordan and Elman networks using data gathered from PHP Nuke portal. Experimental results show that the Jordan network outperforms the Elman network predicting correctly queries of the length up to ten.

1 Introduction

Large number of Web applications, especially those deployed for companies to e-business purpose involve data integrity and confidentiality. Such applications are written in script languages like PHP embedded in HTML allowing to establish connection to databases, retrieving data and putting them in WWW site. Security violations consist in not authorized access and modification of data in the database. SQL is one of languages used to manage data in databases. Its statements can be one of sources of events for potential attacks.

In the literature there are some approaches to intrusion detection in Web applications. In [1] the authors developed anomaly-based system that learns the profiles of the normal database access performed by web-based applications using a number of different models. A profile is a set of models, to which parts

of SQL statement are fed to in order to train the set of models or to generate an anomaly score. During training phase models are built based on training data and anomaly score is calculated. For each model, the maximum of anomaly score is stored and used to set an anomaly threshold. During detection phase, for each SQL query anomaly score is calculated. If it exceeds the maximum of anomaly score evaluated during training phase, the query is considered to be anomalous. The main disadvantage of the system is high rate of false positive alerts. Decreasing it involves creating models for custom data types for each application to which this system is applied.

Besides that work, there are some other works on detecting attacks on a Web server which constitutes a part of infrastructure for Web applications. In [2] detection system correlates the server-side programs referenced by clients queries with the parameters contained in these queries. It is similar approach to detection to the previous work. The system analyzes HTTP requests and builds data model based on attribute length of requests, attribute character distribution, structural inference and attribute order. In a detection phase built model is used for comparing requests of clients.

In [3] logs of Web server are analyzed to look for security violations. However, the proposed system is prone to high rates of false alarm. To decrease it, some site-specific available information should be taken into account which is not portable.

In this work we present a new approach to intrusion detection in Web application. Rather than building profiles of normal behavior we focus on a sequence of tokens within SQL statements observed during normal use of application. Two architectures of RNN are used to encode stream of such SQL statements.

The paper is organized as follows. The next section discusses SQL attacks. In section 3 we present two architectures of RNN. Section 4 shows training and testing data used for experiments. Next, section 5 contains experimental results. Last section summarizes results and shows possible future work.

2 SQL Attacks

2.1 SQL Injection

SQL injection attack consists in such a manipulation of an application communicating with a database, that it allows a user to gain access or to allow it to modify data for which it has not privileges. To perform an attack in the most cases Web forms are used to inject part of SQL query. Typing SQL keywords and control signs an intruder is able to change the structure of SQL query developed by a Web designer. If variables used in SQL query are under control of a user, he can modify SQL query which will cause change of its meaning. Consider an example of a poor quality code written in PHP presented below.

```
$connection=mysql_connect();  
mysql_select_db("test");  
$user=$HTTP_GET_VARS['username'];
```

```

$pass=$HTTP_GET_VARS['password'];
$query="select * from users where
    login='$user' and password='$pass'";
$result=mysql_query($query);
if(mysql_num_rows($result)==1)
    echo "authorization successful"
else
    echo "authorization failed";

```

The code is responsible for authorizing users. User data typed in a Web form are assigned to variables *user* and *pass* and then passed to the SQL statement. If retrieved data include one row it means that a user filled in the form login and password the same as stored in the database. Because data sent by a Web form are not analyzed, a user is free to inject any strings. For example, an intruder can type: " ' or 1=1 -" in the login field leaving the password field empty. The structure of SQL query will be changed as presented below.

```

$query="select * from users where login
=' ' or 1=1 --' and password=''";

```

Two dashes comments the following text. Boolean expression *1=1* is always true and as a result user will be logged with privileges of the first user stored in the table *users*.

2.2 Proposed Approach

The way we detect intruders can be easily transformed to time series prediction problem. According to [5] a time series is a sequence of data collected from some system by sampling a system property, usually at regular time intervals. One of the goal of the analysis of time series is to forecast the next value in the sequence based on values occurred in the past. The problem can be more precisely formulated as follows:

$$s_{t-2}, s_{t-1}, s_t \longrightarrow s_{t+1}, \quad (1)$$

where *s* is any signal, which is dependent on a solving problem and *t* is a current moment in time. Given s_{t-2}, s_{t-1}, s_t , we want to predict s_{t+1} . In the problem of detection SQL attacks, each SQL statement is divided into some signals, which we further call tokens. The idea of detecting SQL attacks is based on their key feature. SQL injection attacks involve modification of SQL statement, which lead to the fact, that the sequence of tokens extracted from a modified SQL statement is different than the sequence of tokens derived from a legal SQL statement. For example, let *S* means recorded SQL statement and T_1, T_2, T_3, T_4, T_5 tokens of this SQL statement. The original sequence of tokens is as follows:

$$T_1, T_2, T_3, T_4, T_5. \quad (2)$$

If an intruder performs an attack, the form of SQL statement changes. Transformation of the modified statement to tokens results in different tokens than

these shown in eq.(2). The example of a sequence of tokens related to modified SQL query is as follows:

$$T_1, T_2, T_{mod3}, T_{mod4}, T_{mod5}. \quad (3)$$

Tokens number 3, 4, 5 are modified due to an intruder activity. We assume that intrusion detection system trained on original SQL statements is able to predict the next token based on the tokens from the past. If the token T_1 occurs, the system should predict token T_2 , next token T_3 is expected. In case of attacks token T_{mod3} occurs which is different than T_3 , which means that an attack is performed.

Various techniques have been used to analyze time series [6,7]. Besides statistical methods, RNNs have been widely used for that problem. In our study presented in this paper we selected two RNNs, the Elman and the Jordan networks.

3 Recurrent Neural Networks

3.1 General Issues

Application of neural networks to solving any problem involves three steps. The first is training, during which weights of network connections are changed. Network output is compared to training data and the network error is evaluated. In the second step the network is verified. Values of connections weights are constant and the network is checked if its output is the same as in the training phase. The last step is generalization. The network output is evaluated for such data, which were not used for training the network. Good generalization is a desirable feature of all networks because it means that the network is prepared for processing data, which may occur in the future.

In comparison to feedforward neural networks RNN have feedback connections which provide dynamics. When they process information, output neurons signal depends on input and activation of neurons in the previous steps of training RNN.

3.2 RNN Architectures

There are some differences between the Elman and the Jordan networks. The first is that input signal for context layer neurons comes from different layers and the second is that Jordan network has additional feedback connection in the context layer. While in the Elman network the size of the context layer is the same as the size of the hidden layer, in the Jordan network the size of output layer and context layer is the same. In both networks recurrent connections have fixed weight equal to 1.0. Networks were trained by BPTT and the following equations are applied for the Elman network:

$$x(k) = [x_1(k), \dots, x_N(k), v_1(k-1), \dots, v_K(k-1)], \quad (4)$$

$$u_j(k) = \sum_{i=1}^{N+K} w_{ij}^{(1)} x_i(k), v_j(k) = f(u_j(k)), \quad (5)$$

$$g_j(k) = \sum_{i=1}^K w_{ij}^{(2)} v_i(k), y_j(k) = f(g_j(k)), \quad (6)$$

$$E(k) = 0.5 \sum_{i=1}^M [y_i(k) - d_i(k)]^2, \quad (7)$$

$$\delta_i^{(o)}(k) = [y_i(k) - d_i(k)] f'(g_i(k)), \delta_i^{(h)}(k) = f'(u_i(k)) \sum_{j=1}^K \delta_j^{(o)}(k) w_{ij}^{(2)}, \quad (8)$$

$$w_{ij}(k+1)^{(2)} = w_{ij}(k)^{(2)} + \sum_{k=1}^{sql-length} [v_i(k) \delta_j^{(o)}(k)], \quad (9)$$

$$w_{ij}(k+1)^{(1)} = w_{ij}(k)^{(1)} + \sum_{k=1}^{sql-length} [x_i(k) \delta_j^{(h)}(k)]. \quad (10)$$

In the equations (4)-(10), N , K , M stand for the size of the input, hidden and output layers, respectively. $x(k)$ is an input vector, $u_j(k)$ and $g_j(k)$ are input signals provided to the hidden and output layer neurons. Next, $v_j(k)$ and $y_j(k)$ stand for the activations of the neurons in the hidden and output layer at time k , respectively. The equation (7) shows how RNN error is computed, while neurons error in the output and hidden layers are evaluated according to (8). Finally, in the last step values of weights are changed using formulas (9) for the output layer and (10) for the hidden layer.

3.3 Training

The training process of RNN is performed as follows. The tokens of the SQL statement become input of a network. Activations of all neurons are computed. Next, an error of each neuron is calculated. These steps are repeated until last token has been presented to the network. Next, all weights are evaluated and activation of the context layer neurons is set to 0. For each input data, teaching data are shifted by one token forward in time with relation to input.

Training data consists of 276 SQL queries without repetition. The following tokens are considered: keywords of SQL language, numbers, strings and combinations of these elements. We used the collection of SQL statements to define 54 distinct tokens. Each token has a unique index. The table [11](#) shows selected tokens and their indexes. The indexes are used for preparation of input data for neural networks. The index e.g. of a keyword *WHERE* is 7. The index 28 points to a combination of keyword *FROM* and any string. The token with index 36 relates to a grammatical link between *SELECT* and any string. Finally, when any string is compared to any number within a SQL query, the index of a token

Table 1. A part of a list of tokens and their indexes

token	index
...	...
WHERE	7
...	...
FROM string	28
...	...
SELECT string	36
...	...
string=number	47
...	...
INSERT INTO	54

equals to 47. Figure 1 presents an example of SQL statement, its representation in the form of tokens and related binary four inputs of a network.

SQL statement is encoded as k vectors, where k is the number of tokens constituting the statement (see figure 1). The number of neurons on the input layer is the same as the number of defined tokens. Networks have 55 neurons in the output layer. 54 neurons correspond to each token similarly to the input layer but the neuron 55 is included to indicate that just processing input data is the last within a SQL query. Training data, which are compared to the output of the network have value either equals to 0.1 or 0.9. If a neuron number n in the output layer has small value then it means that the next processing token can not have index n . On the other hand, if output neuron number n has value of 0.9, then the next token in a sequence should have index equals to n .

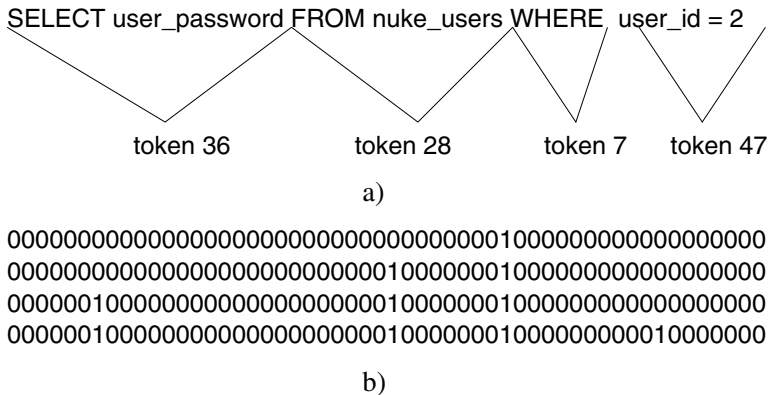


Fig. 1. Preparation of input data for a neural network: analysis of a statement in terms of tokens (a), input neural network data corresponding to the statement (b)

At the beginning, SQL statement is divided into tokens. The indexes of tokens are: 36, 28, 7 and 47. Each row is an input vector for RNN (see figure 1). In the

figure 1 the first token that has appeared is 36. As a consequence, in the first step of training output signal of all neurons in the input layer is 0 except neuron number 36, which has value of 1. Next input vectors indicate current indexes of tokens and the index of a token that has been processed by RNN. The next token in a sequence has index equals to 28. It follows that only neurons 36 and 28 have output signal equal to 1. The next index of a token is 7, which means that neurons: 36, 28 and 7 send 1 and all remaining neurons send 0. Finally, neurons 36, 28, 7, 47 have activation signal equal to 1. In that moment weights of RNN are updated and the next SQL statement is considered.

Training a network in such a way ensures that it will possess prediction capability. While network output depends on the previous input vectors, processing the set of tokens related to the next SQL query can not be dependent on tokens of the previous SQL statement.

4 Training and Testing Data

We evaluated our system using data collected from PHP Nuke portal [11]. Similarly to [1] we installed this portal in version 7.5, which is susceptible to some SQL injection attacks. A function of the portal related to executing SQL statements was modified. Besides its original purpose, each executed SQL query is written to a log file. Each time a Website is downloaded by a browser, SQL queries are sent to a database and logged to a file simultaneously. During operation of the portal we collected nearly 100000 SQL statements. Next, based on this collection we defined tokens, which are keywords of SQL and data types. The set of all SQL queries was divided into 12 subsets, each containing SQL statements of different length. 80% of each data set was used for training and remaining data used for examining generalization. Teaching data are shifted one time forward in time. Data with attacks are the same as reported in [1].

5 Experimental Results

Experimental study was divided into four stages. In the first one, we evaluated the best parameters of both RNNs and learning algorithm. We run experiments 10 times and averaged results. For the following values of parameters the error of the networks was minimal. For the Elman network all neurons in the hidden layer have sigmoidal activation function while all neurons in the output layer have *tanh* function. For the Jordan network *tanh* function was chosen for the hidden layer and sigmoidal function for the output layer. The number of neurons in the hidden layer equals to 58 neurons. In the most cases η (training coefficient) does not exceed 0.2 and α value (used in momentum) is less than 0.2.

In the second phase of the experimental study we trained 12 RNNs, one for each training data subset, using values from the first stage. In the most cases,

from the beginning of the training, the error of the Jordan network was much smaller than error of the Elman network. In the next a few epochs the error of both networks decreased quickly but the Jordan network error remained much smaller than the Elman network error. Figure 2 shows how error of networks changes for all subsets of SQL queries and how well the networks are verified. Here, a statement is considered as well predicted if for all input vectors, all neurons in the output layer have values according to training data. All values

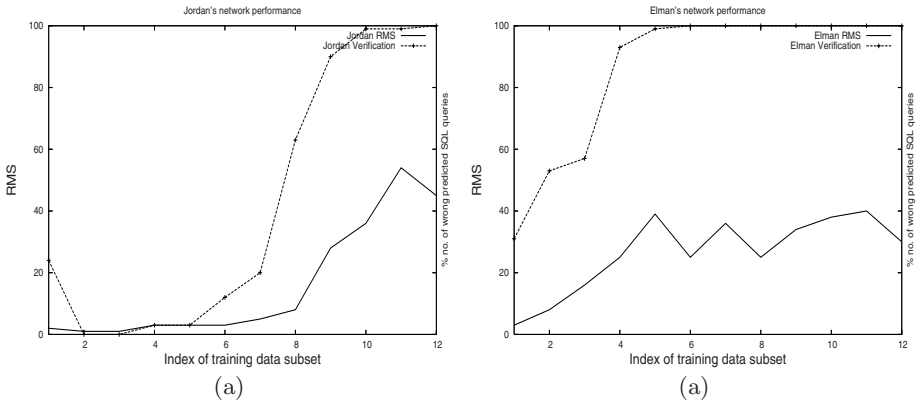


Fig. 2. Error and number of wrong predicted SQL queries for each subset of data. Jordan network (a), Elman network (b)

# SQL statement – attack	# legal SQL statement		
# index of input vector num. of errors	# index of input vector num. of errors	–ver	–ger
1	7	1	0
2	2	2	1
3	1	3	2
4	2	4	1
5	2	5	1
6	1	6	1
7	2	7	1
8	0	8	0

a) b)

Fig. 3. RNN output for an attack (a), RNN output for known and unknown SQL statement (b).

presented in figures are averaged on 10 runs of RNNs. One can see that nearly for all data subsets the Jordan network outperforms the Elman one. Only for data subsets 11 and 12 (see table 2) the error of the Jordan network is greater than the error of the Elman network. Despite of this the Jordan network is better then the Elman network in terms of percentage number of wrong predicted SQL queries. Verification states how good a network is trained. In the sense of the detecting of

attacks, it means that the better verification, the less false alarms of a system. The Jordan network predicts all tokens of 10 length statements (20.6% false alarms). In the third part of experiments we checked if RNNs correctly detect attacks. Each experiment was conducted using trained RNNs from the second stage. Figure 3a) presents the typical RNN output if an attack is performed. The left column depicts the number of input vector for RNN, while the right column shows the number of cases in which the index of the token indicated by network output is different than the index of the next processed by RNN token. What is common for each network is that nearly each output vector of a network has a few errors. This phenomenon is present for all attacks used in this work. Figure 3b) shows RNN output for verification (the 2nd column) and generalization (the 3rd column). It is easy to see that the number of errors in figures 3a) and 3b) strongly varies. Moreover, there is also more output vectors free of errors. Easily noticeable difference between an attack and normal activity allows us to re-evaluate obtained results presented in figure 2. To distinguish between an attack

Table 2. Results of verification and generalization of Elman and Jordan networks

Index of data subset	length of data subset	Elman ver	Elman gen	Jordan ver	Jordan gen
1	2-4	0	0	0	0
2	5	0	1.4	0	0
3	6	0	24.2	0	12.8
4	7	0	15.7	0	1.4
5	8	0	5	0	1.6
6	9	0	3.33	0	0
7	10	0	2.5	0	0
8	11	0	10	0	13.33
9	12	0	0	0	6.66
10	13-14	0	0	0	0
11	15-16	0	3.33	0	3.33
12	17-20	0	40	0	13.33

and a legitimate SQL statement we define the following rule for the Jordan network: an attack occurred if the average number of errors for each output vector is not less than 2.0 and 80% of output vectors include any error. For the Elman the threshold equals to 1.6 and the error coefficient equals to 90%. Applying these rules ensures that all attacks are detected by both RNN. The table 2 presents the percentage number of SQL statements wrongly predicted during verification and generalization if results were processed by the rules. For the most cases the Jordan network outperforms the Elman network. Only for data subsets containing statements made from 11 and 12 tokens, the Elman network is a little better than the Jordan network. The important outcome of defined rules is that both RNNs thought all statements and only few legitimate statements, which were not in the training set were detected as attacks.

6 Conclusions

In the paper we have presented a new approach to detecting SQL-based attacks. The problem of detection was transformed to time series prediction problem and two RNNs were examined to show their potential use for such a class of attacks. It turned out that the Jordan network is easily trained by BPTT algorithm. Despite the fact that large architecture of RNN was used, that network is able to predict sequences of up to ten length with acceptable error margin.

Deep analysis of the experimental results lead to the definition of rules used for distinguishing between an attack and legitimate statement. When these rules are applied, both networks are completely trained for all SQL queries included in the all training subsets. Accuracy of the results very strongly depends on the rules. The advisable part of experimental study is to apply defined rules to the other data set, which can confirm efficiency of the proposed approach to detecting SQL attacks.

References

1. Valeur, F., Mutz, D., Vigna, G.: A Learning-Based Approach to the Detection of SQL Attacks. Proceedings of the Conference on Detection of Intrusions and Malware and Vulnerability Assessment, Austria (2005)
2. Kruegel, C., Vigna, G.: Anomaly Detection of Web-based Attacks. Proceedings of the 10th ACM Conference on Computer and Communication Security (2003) 251-261
3. Almgren, M., Debar, H., Dacier, M.: A lightweight Tool for Detecting Web Server Attacks. In Proceedings of the ISOC Symposium on Network and Distributed Systems Security (2000)
4. Tan, K.M.C., Killourhy, K.S., Maxion, R.A.: Undermining an Anomaly-Based Intrusion Detection System Using Common Exploits. In Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (2002) 54-73
5. Nunn, I., White, T.: The Application of Antigenic Search Techniques to Time Series Forecasting. In Proceedings of the Genetic and Evolutionary Computation Conference USA (2005)
6. Kendall, M., Ord, J.: Time Series. Third Edition (1999)
7. Pollock, D.: A Handbook of Time-Series Analysis, Signal Processing and Dynamics. Academic Press, London (1999)
8. Lin, T., Horne, B.G., Tino, P., Giles, C.L.: Learning Long-Term Dependencies in NARX Recurrent Neural Networks. IEEE Transactions on Neural Networks (1996) 1329
9. Drake, P.R., Miller, K.A.: Improved Self-Feedback Gain in the Context Layer of a Modified Elman Neural Network. Mathematical and Computer Modelling of Dynamical Systems (2002) 307-311
10. http://www.insecure.org/sploits_all.html
11. <http://phpnuke.org/>

Solving Variational Inequality Problems with Linear Constraints Based on a Novel Recurrent Neural Network

Youshen Xia¹ and Jun Wang²

¹ College of Mathematics and Computer Science, Fuzhou University, China

² Department of Automation and Computer-Aided Engineering, The Chinese University of Hong Kong, China

Abstract. Variational inequalities with linear inequality constraints are widely used in constrained optimization and engineering problems. By extending a new recurrent neural network [14], this paper presents a recurrent neural network for solving variational inequalities with general linear constraints in real time. The proposed neural network has one-layer projection structure and is amenable to parallel implementation. As a special case, the proposed neural network can include two existing recurrent neural networks for solving convex optimization problems and monotone variational inequality problems with box constraints, respectively. The proposed neural network is stable in the sense of Lyapunov and globally convergent to the solution under a monotone condition of the nonlinear mapping without the Lipschitz condition. Illustrative examples show that the proposed neural network is effective for solving this class of variational inequality problems.

1 Introduction

Many engineering problems, including robot control, electrical networks control, and communications, can be formulated as optimization problems with linear constraints [1]. Because of the time-varying nature of these optimization problems, they have to be solved in real time. One application of real-time optimization in robotics is robot motion control [2]. Another suitable application of real-time optimization in signal processing is for adaptive beamforming [3]. Because of the nature of digital computers, conventional numerical optimization techniques are usually not competent. As parallel computational models, neural networks possess many desirable properties such as real-time information processing. Therefore, recurrent neural networks for optimization, control, and signal processing received tremendous interests [3-18]. These neural network models are theoretically analyzed to be globally convergent under various different conditions and extensively simulated to further demonstrate their operating characteristics in solving various optimization problems. For example, by the penalty method Kennedy and Chua [4] extended Hopfield and Tank's neural

network for solving nonlinear programming problems. To avoid penalty parameters Rodríguez-Vázquez et al. [6] proposed a switched-capacitor neural network for solving a class of nonlinear optimization problems. In term of Lagrange function methods, Zhang and Constantinides [8] developed a Lagrange programming neural network for solving a class of nonlinear optimization problems with equality constraints. Based on some projection formulations, Xia, and Xia and Wang [10-13] developed several recurrent neural networks for solving constrained optimization and related problems, respectively. Recently, based a normal mapping equation, Xia and Feng [14] developed a recurrent neural network for solving a class of projection equations and variational inequality problems with box constraints.

In this paper, we are concerned with the following variational inequality problem with general linear constraints:

$$(x - x^*)^T F(x^*) \geq 0, \quad x \in \Omega_0 \quad (1)$$

where $F : R^n \rightarrow R^n$ is differentiable,

$$\Omega_0 = \{x \in R^n \mid Bx = b, Ax \leq d, x \in X\},$$

where $B \in R^{r \times n}$, $A \in R^{m \times n}$, $b \in R^r$, $d \in R^m$, and $X = \{x \in R^n \mid l \leq x \leq h\}$. The problem (1) has been viewed as a natural framework for unifying the treatment of a large class of constrained optimization problems [1, 19,20]. The objective of this paper is to extend the existing recurrent neural network to solve (1). The proposed neural network is thus a significant extension for solving variational inequality problems from box constraints to linear constraints. Compared with existing projection neural network [12], the proposed neural network does not require the initial point in the feasible set X . Compared with the modified projection-type numerical method for solving (1) [20], which requires a varying step length, the proposed neural network not only has a lower complexity and but also has no requirement of the Lipschitz condition of the mapping F .

2 Neural Network Models

A. Reformulation

For convenience of discussion, we assume that the problem (1) has at least solution and denote the solution set by

$$X^* = \{x^* \in R^n \mid x^* \text{ solves (1)}\}.$$

From optimization literatures (Bertsekas, 1989) we know that the Karush-Kuhn-Tucker (KKT) condition for (1) can be written as the following

$$\begin{cases} y \geq 0, & Ax \leq d, \quad l \leq x \leq h \\ Bx = b, & y^T(Ax - d) = 0 \end{cases}$$

and

$$\begin{cases} (F(x) + A^T y - B^T z)_i \geq 0 & \text{if } x_i = h_i \\ (F(x) + A^T y - B^T z)_i = 0 & \text{if } l_i \leq x_i \leq h_i \\ (F(x) + A^T y - B^T z)_i \leq 0 & \text{if } x_i = l_i, \end{cases}$$

where $y \in R^m, z \in R^r$. According to the projection theorem (Kinderlehrer and Stampcchia, 1980) the above KKT condition can be equivalently represented as

$$\begin{cases} P_X(x - (F(x) + A^T y - B^T z)) = x \\ (y + Ax - d)^+ = y \\ Bx = b \end{cases} \quad (2)$$

where $(y)^+ = [(y_1)^+, \dots, (y_m)^+]^T$ with $(y_i)^+ = \max\{0, y_i\}$. Thus x^* is a solution to (1) if and only if there exists $y^* \in R^m$ and $z^* \in R^r$ such that (x^*, y^*, z^*) satisfies (2). Furthermore, let $\Omega = X \times R_+^m \times R^r$ and $P_\Omega(u) = [P_X(x), (y)^+, z]^T$, where $R_+^m = \{y \in R^m \mid y \geq 0\}$ and

$$P_X(x_i) = \begin{cases} l_i & x_i < l_i \\ x_i & l_i \leq x_i \leq h_i \\ h_i & x_i > h_i \end{cases},$$

and let

$$u = \begin{pmatrix} x \\ y \\ z \end{pmatrix}, \quad T(u) = \begin{pmatrix} F(x) + A^T y - B^T z \\ -(Ax - d) \\ Bx - b \end{pmatrix}.$$

Then the projection equations can be simply written as

$$P_\Omega[u - T(u)] = u. \quad (3)$$

Therefore, we have the following lemma.

Lemma 1. x^* is a solution to (1) if and only if there exists $y^* \in R^m$ and $z^* \in R^r$ such that $u^* = (x^*, y^*, z^*)$ is a solution of the projection equation (3).

B. Dynamical Equations

Based on the novel structure of the recurrent neural network developed in [14], we present a recurrent neural network for solving the problem (1) as follows

State equation

$$\frac{dw}{dt} = \lambda \{-T(P_\Omega(w)) + P_\Omega(w) - w\}, \quad (4)$$

Output equation

$$u = P_\Omega(w) \quad (5)$$

where $\lambda > 0$ is a designing constant, $w = (x, y, z) \in R^n \times R^m \times R^r$ is a state vector and u is an output vector. Substituting the representation of $T(u)$ into (4),

we can represent the state equation of the proposed neural network as follows

$$\frac{dw}{dt} = \lambda \begin{pmatrix} -F[P_X(x)] - A^T(y)^+ + B^T z + P_X(x) - x \\ (AP_X(x) - y - d) + (y)^+ \\ -B[P_X(x)] + b \end{pmatrix}. \quad (6)$$

It can be seen that the state equation (6) can be realized by a recurrent neural network with a one-layer projection structure, which consists of $n + m + r$ integrators, $m + n$ piecewise activation functions for $P_X(x)$ and $(y)^+$, n processors for $F(x)$, $2n(m + r)$ connection weights, and some summers. Therefore, the network complexity depends only on the mapping $F(x)$. It can be seen that the proposed neural network has same network complexity as the projection neural network given in [13]. Moreover, the proposed neural network is a significant generalization of existing neural networks.

C. Two Special Cases

Case 1: $A = O, B = O, b = 0$, and $d = 0$, the considered variational inequality becomes

$$(x - x^*)^T F(x^*) \geq 0, x \in X. \quad (7)$$

The corresponding neural network is then given by

State equation

$$\frac{d\hat{w}}{dt} = \lambda \{-F(P_X(\hat{w})) + P_X(\hat{w}) - \hat{w}\} \quad (8)$$

Output equation

$$x = P_\Omega(\hat{w}), \quad (9)$$

where $\hat{w} \in R^n$. The neural network model was presented in [14].

Case 2: $F(x) = \nabla f(x)$, where $f(x)$ is a continuously differentiable function. In the case, (1) becomes a well-known nonlinear programming problem with general linear constraints

$$\begin{aligned} &\text{minimize } f(x) \\ &\text{subject to } Bx = b, Ax \leq d, l \leq x \leq h \end{aligned} \quad (10)$$

The corresponding neural network is then given by

State equation

$$\frac{dw}{dt} = \lambda \begin{pmatrix} -\nabla f[P_X(x)] - A^T(y)^+ + B^T z + P_X(x) - x \\ (AP_X(x) - y - d) + (y)^+ \\ -B[P_X(x)] + b \end{pmatrix}. \quad (11)$$

Output equation

$$v = P_X(x).$$

In particular, when $A = O, B = O, b = 0$, and $d = 0$, the nonlinear programming (10) becomes a bounded nonlinear program

$$\begin{aligned} &\text{minimize } f(x) \\ &\text{subject to } l \leq x \leq h \end{aligned} \quad (12)$$

The corresponding neural network is then given by
State equation

$$\frac{dw}{dt} = \nabla f[P_X(w)] + P_X(w) - w \quad (13)$$

Output equation

$$x = P_X(w),$$

where $\hat{w} \in R^n$. This neural network model was presented in [10].

3 Convergence Results

As for the convergence of the proposed neural network in (4) and (5), by combining analysis techniques of papers [10, 11, 14] we can obtain the following main results.

Theorem 1. (i) For any initial point, there exists a unique solution trajectory for (4). (ii) The proposed neural network model in (4) and (5) has at least an equilibrium point. Moreover, if $w^* = (x^*, y^*, z^*)$ is an equilibrium point of the proposed neural network model in (4) and (5), then $P_X(x^*)$ is a solution of (1).

Theorem 2. Assume that the Jacobian matrix, $\nabla F(x)$, of F is positive semi-definite. If $\nabla F(x^*)$ is positive definite, then the proposed neural network in (4) and (5) is stable in the sense of Lyapunov and its output trajectory converge globally to a solution of (1).

The following result is an improvement on the existing one given in [14].

Theorem 3. Assume that $F(x)$ is pseudomonotone:

$$F(x^*)^T(x - x^*) \geq 0 \Rightarrow F(x)^T(x - x^*) \geq 0, \forall x \in X.$$

If $(x - x^*)^T F(x) = 0 \Rightarrow x \in X^*$, then the output trajectory of the neural network defined in (8) and (9) converges globally to a solution of (7).

Remark 1. The convergence condition of Theorem 3 is weaker than the one given in [14], where $F(x)$ is a strictly monotone mapping or $F(x)$ is a monotone gradient mapping. As pointed in paper [18], the pseudomonotonicity is a generalization of monotonicity, and the pseudomonotonicity of F implies the monotonicity of F .

As a direct corollary of Theorem 3, we have the following result.

Corollary 1. Assume that $F(x)$ is strictly pseudomonotone:

$$F(x^*)^T(x - x^*) > 0 \Rightarrow F(x)^T(x - x^*) > 0, \forall x \neq x^*, x \in X.$$

Then the input trajectory of the neural network defined in (8) and (9) converges globally to a solution of (7).

Remark 2. Since the nonlinear mapping $F(x)$ is only differentiable, the above results don't require the local Lipschitz condition of F .

4 Simulation Examples

Example 1. Let us consider the variational inequality problem (1), where

$$F(x) = \begin{bmatrix} 5(x_1)^+ + x_1^2 + x_2 + x_3 \\ 5x_1 + 3x_2^2 + 10(x_2)^+ + 3x_3 \\ 10x_1^2 + 8x_2^2 + 4(x_3)^+ + 3x_3^2 \end{bmatrix},$$

$X = \{x \in R^3 | x_1 + x_2 + x_3 \geq 6, x \geq 0\}$, $F(x)$ is monotone. This problem has a unique solution $x^* = [4.5, 1.5, 0]^T$. We use the proposed neural network in (4) and (5) to solve the above problem. All simulation results show that the neural network in (4) and (5) is always convergent to x^* . For example, let $\lambda = 5$ and let an initial point be zero. The obtained solution $\hat{x} = [4.499, 1.499, 0]^T$. Fig. 1

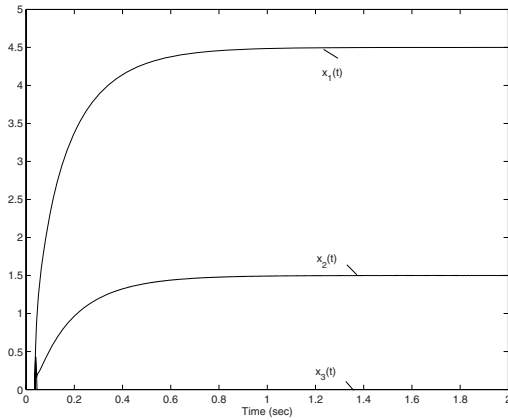


Fig. 1. The transient behavior of the output trajectory of the proposed neural network in (4) and (5) with a zero initial point in Example 1

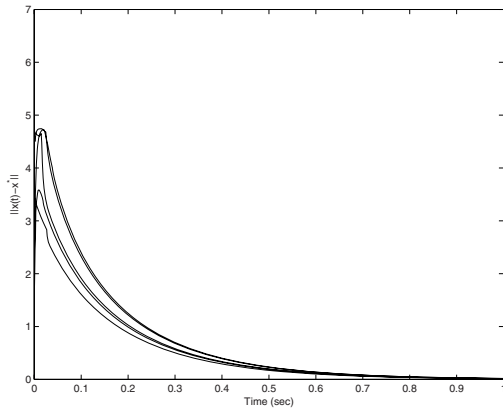


Fig. 2. The convergence behavior of the output trajectory error based on the neural network in (4) and (5) with 5 random initial points in Example 1

displays its transient behavior of the output trajectory of the neural network in (4) and (5). Fig. 2 displays the convergence behavior of the output trajectory error based on the proposed neural network in (4) and (5) with 5 random initial points.

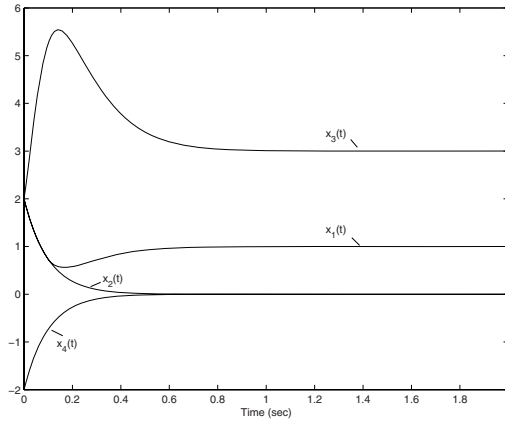


Fig. 3. The transient behavior of the neural network in (8) and (9) with initial point $x^0 = [-2, -2, -2, 2]^T$ in Example 2

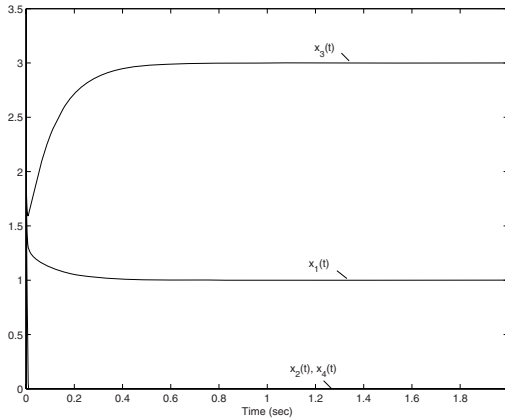


Fig. 4. The transient behavior of the neural network in (8) and (9) with initial point $x^0 = [-2, -2, -2, 2]^T$ in Example 2

Example 2. Let us consider a nonlinear complementarity problem (NCP)

$$x \geq 0, F(x) \geq 0, x^T F(x) = 0,$$

where

$$F(x) = \begin{pmatrix} 3x_1^2 + 2x_1x_2 + 2x_2^2 + x_3 + 3x_4 - 6 \\ 2x_1^2 + x_1 + x_2^2 + 10x_3 + 2x_4 - 2 \\ 3x_1 + x_1x_2 + 2x_2^2 + 2x_3 + 9x_4 - 9 \\ x_1^2 + 3x_2^2 + 2x_3 + 3x_4 \end{pmatrix}$$

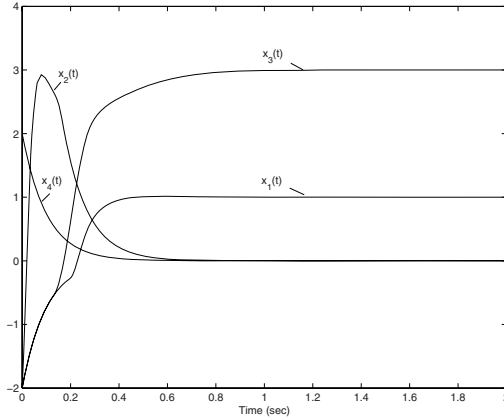


Fig. 5. The transient behavior of the neural network in (8) and (9) with initial point $x^0 = [2, 2, 2, -2]^T$ in Example 2

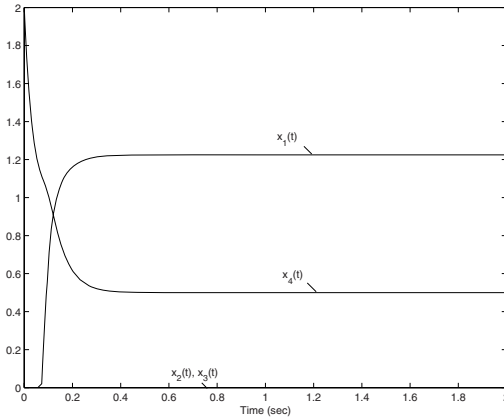


Fig. 6. The transient behavior of the projection neural network with initial point $x^0 = [2, 2, 2, -2]^T$ in Example 2

This problem has two solutions $x^1 = [1, 0, 3, 0]^T$ and $x^2 = [\sqrt{6}/2, 0, 0, 1/2]^T$. The NCP can be converted into the variational inequality problem (7), where $X = \{x \in R^4 | x \geq 0\}$ and $F(x)$ is not monotone on X . We perform the neural network in (8) and (9) and the projection neural network given in [12] to solve the above problem. Let the initial point be $x^0 = [-2, -2, -2, 2]^T$ and $\lambda = 10$. The neural network in (8) and (9) is convergent to x^1 , shown in Fig 3, and the projection neural network converges to x^1 also, shown in Fig.4. Again, let the initial point be $x^0 = [2, 2, 2, -2]^T$ and $\lambda = 10$. The neural network in (8) and (9) is convergent to x^2 , shown in Fig 5. While the projection neural network converges to x^1 still, shown in Fig 6.

5 Concluding Remarks

We have presented a recurrent neural network for solving variational inequality problems with general linear constraints. The proposed neural network has same network complexity as the existing projection neural network but no requirement that the initial point be in the feasible set. Moreover, the proposed neural network is a significant generalization of several existing neural networks for constrained optimization. The proposed neural network is stable in the sense of Lyapunov and globally convergent to the solution under a monotone condition of the nonlinear mapping without the Lipschitz condition. Further investigations will be aimed at the convergence rate of the proposed neural network and engineering applications

References

1. Bazaraa, M.S., Sherali, H.D., Shetty, C.M.: *Nonlinear Programming: Theory and Algorithms* (2nd Ed.), John Wiley, New York, 1993
2. Yoshikawa, T.: *Foundations of Robotics: Analysis and Control*, MIT Press, Cambridge, MA, 1990
3. Cichocki, A., Unbehauen, R.: *Neural Networks for Optimization and Signal Processing*, Wiley, England, 1993
4. Kennedy, M.P., Chua, L.O.: Neural Networks for Nonlinear Programming. *IEEE Transactions on Circuits and Systems* **35**(5) (1988) 554-562
5. Lillo, W.E., Loh, M.H., Hui, S., Zák, S.H.: On Solving Constrained Optimization Problems with Neural Networks: A Penalty Method Approach. *IEEE Transactions on Neural Networks* **4**(6) (1993) 931-939
6. Rodríguez-Vázquez, A., Domínguez-Castro, R., Rueda, A., Huertas, J.L., Sánchez-Sinencio, E.S: Nonlinear Switched-capacitor 'Neural Networks' for Optimization Problems. *IEEE Transactions on Circuits and Systems* **37** (1990) 384-397
7. Zák, S.H., Upatising, V., Hui, S.: Solving Linear Programming Problems with Neural Networks: A Comparative Study. *IEEE Transactions on Neural Networks* **6** (1995) 94-104
8. Zhang, S., Constantinides, A.G.: Lagrange Programming Neural Networks. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* **39** (1992) 441-452
9. Bouzerdoum, A., Pattison, T.R.: Neural Network for Quadratic Optimization with Bound Constraints. *IEEE Transactions on Neural Networks* **4**(2) (1993) 293-304
10. Xia, Y.S.: ODE Methods for Solving Convex Programming Problems with Bounded Variables. *Chinese Journal of Numerical Mathematics and Applications (English edition)* **18**(1) 1995
11. Xia, Y.S., Wang, J.: On the Stability of Globally Projected Dynamic Systems. *Journal of Optimization Theory and Applications* **106**(1) (2000) 129-150
12. Xia, Y.S., Leung, H., Wang, J.: A Projection Neural Network and its Application to Constrained Optimization Problems. *IEEE Transactions on Circuits and Systems - Part I* **49**(4) (2002) 447-458
13. Xia, Y.S.: An Extended Projection Neural Network for Constrained Optimization. *Neural Computation* **16**(4) (2004) 863-883
14. Xia, Y.S., Feng, G.: A New Neural Network for Solving Nonlinear Projection equations. Accepted in *Neural Networks*, 2007.

15. Sun, C.Y., Feng, C.B.: Neural Networks for Nonconvex Nonlinear Programming Problems: A Switching Control Approach. *Lecture Notes In Computer Science* **3496** (2005) 694-699
16. Hu, S.Q., Liu, D.R.: On the Global Output Convergence of A Class of Recurrent Neural Networks with Time-varying Inputs. *Neural Networks* **18** (2005) 171-178
17. Liao, X.X., Zeng, Z.G.: Global Exponential Stability in Lagrange Sense of Continuous-time Recurrent Neural Networks. *Lecture Notes In Computer Science* **3971** (2006) 115-121
18. Hu, X., Wang, J.: Solving Pseudomonotone Variational Inequalities and Pseudoconvex Optimization Problems Using the Projection Neural Network. **6** (2006) 1487-1499
19. Kinderlehrer, D., Stampcchia, G.: *An Introduction to Variational Inequalities and Their Applications*. Academic Press, New York, NY, 1980
20. Solodov, M.V., Tseng, P.: Modified Projection-type Methods for Monotone Variational Inequalities. *SIAM J. Control and Optimization* **2** (1996) 1814-1830

Equalization of 8PSK Signals with a Recurrent Neural Network

Dong-Chul Park, Young-Soo Song, and Dong-Min Woo

ICRL and Myongji IT Engineering Research Institute,
Myong Ji University, Korea
{parkd, twobasehit, dmwoo}@mju.ac.kr

Abstract. A novel equalization scheme for a wireless ATM communication channel using a recurrent neural network is proposed in this paper. The recurrent neural network used in this scheme is the Complex Bilinear Recurrent Neural Network (CBLRNN). A reduced version of the CBLRNN for faster and stable convergence is first proposed in this paper. The R-CBLRNN is then applied to equalization of a wireless ATM channel for 8PSK, which has severe nonlinearity and intersymbol interference due to multiple propagation paths. The experiments and results show that the proposed R-CBLRNN gives very favorable results in the Mean Square Error (MSE) criterion over conventional equalizers.

1 Introduction

Wireless technologies have received a great deal of attention due to the increasing popularity of portable computing applications in recent years. The Asynchronous Transfer Mode (ATM) has been used as a high-speed backbone network that provides a common infrastructure network to a diverse set of mobile technologies. One of the major problems which occur in wireless ATM network is the intersymbol interference leading to degradation in performance and capacity of the system.

One of the promising approaches to designing equalizers is the neural network (NN)-based approach [1,2]. As shown in a wide range of computing applications, NN has been successfully used for modelling complex nonlinear systems. The most distinguishable feature of NNs over conventional techniques is the ability of learning [3,4]. In addition, recent researches have shown that a recurrent type NN is more suitable than a feed-forward NN in predicting time series signals [5]. Recent researches on the use of recurrent type NN for designing nonlinear equalizer have shown promising results [1,5]. Among recurrent neuron networks, the Complex Bilinear Recurrent Neural network (CBLRNN)-based equalizer gives very favorable results over Volterra filter equalizers, DFEs, and Complex Multi-layered Perceptron type NN (CMLPNN) equalizers.

The Reduced-CBLRNN (R-CBLRNN) uses only a part of the feedback components for bilinear component calculation as introduced in Section 2. The channel model to be used as the target model for experiments of R-CBLRNN applications

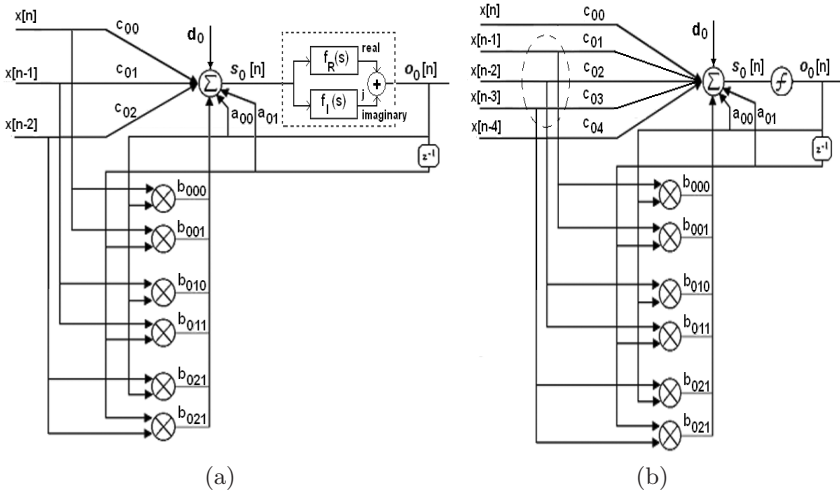


Fig. 1. Comparison of (a) complex bilinear recurrent neuron and (b) reduced complex bilinear recurrent neuron

is introduced in Section 3. Section 4 presents experiments performed on 8PSK signals and their results including performance comparisons of R-CBLRNN with some conventional equalizers. Concluding remarks are given in Section 5.

2 Complex Bilinear Recurrent Neural Network with a Reduced Architecture

BLRNN is a recurrent NN which has a robust ability in modelling nonlinear systems and was originally introduced in [6] and CBLRNN is a complex version of BLRNN. CBLRNN has been designed to deal with the problems with complex number operations found in equalizer design. Fig. 1(a) shows the output of a neuron in CBLRNN. More detailed explanation on CBLRNN can be found in [1].

Even though CBLRNN shows very promising results when applied to equalizer problems, it still suffers from slow convergence in practical use. For CBLRNN, most of the computational loads are due to the number of multiplications between complex values. Consider a CBLRNN with structure N_i - N_h - N_o using N_f feedback taps where N_i , N_h , and N_o are the numbers of input neuron, hidden neuron, and output neuron respectively. When $N_o = 1$, a simple case, the total number of multiplications for CBLRNN:

$$N_C = (N_i + N_f + N_f \cdot N_i) \cdot N_h + N_h. \tag{1}$$

From Eq. (1), we infer that the bilinear component (corresponding to $N_f \cdot N_i$ multiplications) contributes most to the total number of multiplications.

Choi *et.al* propose a simplified version of the bilinear DFE [7]. In this approach, only a part of feed-forward inputs are multiplied to the feedback portion for bilinear components without suffering from performance degradation. The results show that this simplified version gives an insight on the formation of bilinear components. By adopting this idea of reduced bilinear components to the CBLRNN, the output of the BLRNN is changed as follows:

$$s_p[n] = d_p + \sum_{k_1=0}^{N_f-1} a_p[k_1]o_p[n - k_1] + \sum_{k_2=0}^{N_i-1} c_p[k_2]x[n - k_2] \quad (2)$$

$$+ \sum_{k_1=0}^{N_f-1} \sum_{k_2=S}^E b_p[k_1, k_2]o_p[n - k_1]x[n - k_2],$$

where $E = \frac{N_i-1}{2} + p$ and $S = \frac{N_i-1}{2} - p$.

For further reduced scheme, common feedback tabs, $y[n]$, are used for bilinear components instead of individual feedback values from each hidden neuron, $(o_1[n], o_2[n], \dots, o_{N_h}[n])$.

By employing these ideas, the total number of multiplications in the R-CBLRNN becomes:

$$N_S = N_f \cdot N_i + (2N_i + N_f) \cdot N_h + N_h. \quad (3)$$

When N_S in Eq. (3) is compared with N_C in Eq. (1), the number of reduced multiplications, ΔN , is:

$$\Delta N = N_C - N_S = N_h \cdot N_f \cdot N_i - N_h \cdot N_i - N_f \cdot N_i. \quad (4)$$

Fig. 1(b) shows the output of a neuron in R-CBLRNN. In a CBLRNN architecture ($N_i = 10, N_h = 4, N_f = 5$), N_C and N_S are used in our experiments as 264 and 155 respectively. The number of reduced multiplications, ΔN , is 109 in this case and 41.3% of the multiplications can be reduced. Fig. 2 shows the comparison of convergence for CBLRNN and R-CBLRNN. Fig. 2

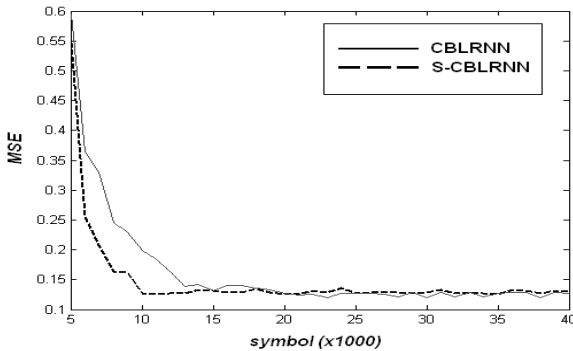


Fig. 2. Convergence for R-CBLRNN and CBLRNN (8PSK, SNR=15 dB)

confirms that the R-CBLRNN converges much faster than a CBLRNN and savings on the convergence speed of the R-CLRNN roughly matches with the savings on multiplication, ΔN .

3 Channel Model

Multipath channels with time-varying transmission characteristics have been modelled with linear time-varying filters [1]. The modulated signal $s_m(t)$ with symbol duration $T = 12.8 \text{ ns}$ (156 MBit/s signal) is transmitted over an ATM channel with P propagation paths modelled by the received signal $y(t)$ is then represented as follows:

$$y(t/T) = \sum_{i=1}^P a_i \cdot s_m(t/T - \tau_i) \cdot e^{-j(\omega_c + \varphi_i)(t/T)} + n(t/T), \quad (5)$$

where w_c is the carrier frequency.

From Eq. (5), we see that $y(t)$ is the sum of several multipath components with the scale a_i and a phase shift $\omega_c + \varphi_i$. In this paper, a 40GHz broadband channel which is introduced in [8] is used for experiments. Fig. 3 displays constellations of the transmitted and received 8 PSK signals through a wireless ATM channel.

Table 1. Channel gain and delay for each propagation path

path i	1	2	3	4	5	6
$\{a_i\}$	1	0.5	0.4	0.32	0.1	0.08
$\{\tau_i\}$	0	1/2	9/8	13/8	21/8	39/8

4 Experiments and Results

The ATM channel for experiments is modelled by the Eq. (5) with 6 propagation paths whose gains and delays are given in Table 1. The transmitted data symbols are randomly generated 8PSK signals. Transmitted signals over the ATM channel are corrupted with AWGN with various SNRs. Fig. 3 shows typical constellations for the original 8PSK signals and the received signals through the ATM channel with 20dB AWGN.

In experiments, randomly generated 30,000 data symbols are used for training the equalizers and another 100,000 data symbols are used for testing the performance of the equalizers. The proposed equalizer based on a R-CBLRNN is compared with a CDFE, a Volterra equalizer, a CMLPNN-based equalizer, and a CBLRNN-based equalizer.

Fig. 4 shows the constellation after equalization for the 8PSK system by the CDFE, the Volterra filter, the CMLPNN, and the R-CBLRNN when an AWGN (SNR=15 dB) is presented. In the result shown in Fig. 4, the R-CBLRNN yields much less MSE than any other equalizer. This result is very acceptable when

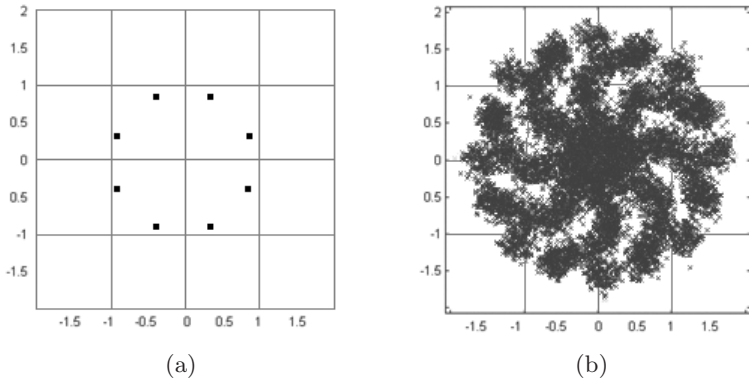


Fig. 3. Constellation of 8PSK signals: (a) the transmitted (b) the received

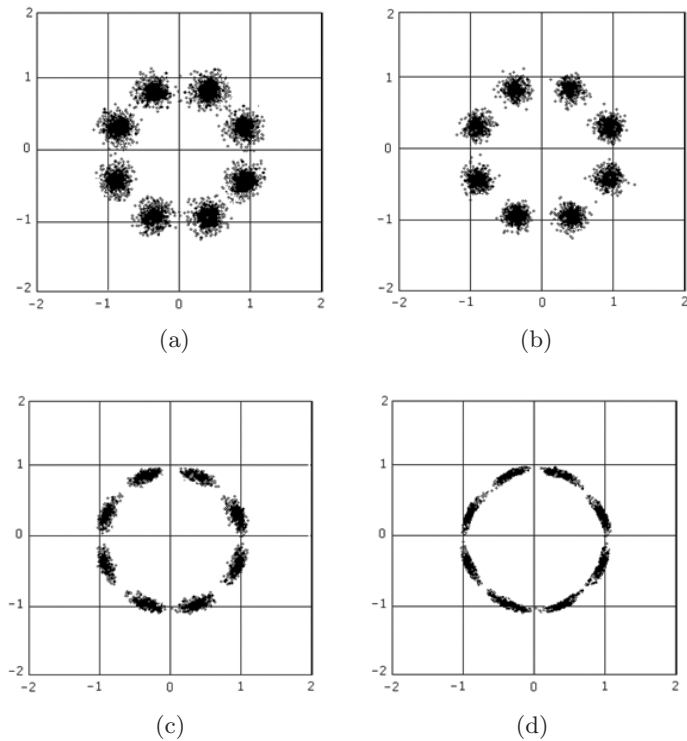


Fig. 4. Constellations after equalization, SNR=15 dB. (a) CDFE (MSE=0.197), (b) Volterra (MSE=0.170), (c) CMLPNN (MSE=0.151), (d) R-CBLRNN (MSE=0.120)

compared with other equalizers reported in the literature and implies that the R-CBLRNN-based equalizer is suitable for 8PSK system over a wireless ATM channel.

5 Conclusion

A reduced scheme of the Complex Bilinear Recurrent Neural Network (R-CBLRNN) is proposed and applied to the problem of equalization of wireless ATM channels in this paper. The proposed R-CBLRNN reconfigures the bilinear components in the BLRNN and reduces a part of the bilinear components. By doing so, the resultant R-CBLRNN can reduce its convergence time about 40 % without suffering from any performance degradation. Experiments on the equalization of a wireless ATM channel for the 8PSK signals, which has severe non-linearity and intersymbol interference due to multiple propagation paths show very favorable results in the MSE criterion over the equalizers such as a Volterra filter equalizer, decision feedback equalizer (DFE), and a MLPNN equalizer.

References

1. Park, D. C., Jeong, T.K.: Complex Bilinear Recurrent Neural Network for Equalization of a Satellite Channel. *IEEE Trans. Neural Network* **13** (2002) 711-725
2. Gibson, G., Siu, S., Cowan, C.: Multilayer Perceptron Structures Applied to Adaptive Equalizer for Data Communication. *IEEE Trans. Acoustics, Speech, and Signal Processing* **39** (1991) 2101-2104
3. Park, D. C., El-Sharkawi, M. A., Marks II, R. J.: Adaptively Trained Neural Network. *IEEE Trans. Neural Networks* **2** (1991) 334-345.
4. You, C., Hong, C.: Nonlinear Blind Equalization Schemes using Complex-Valued Multilayer Feedforward Neural Networks. *IEEE Trans. Neural Networks* **9** (1998) 1442-1445
5. Zhang, X., Wang, H., Zhang, L., Zhang, X.: A Blind Equalization Algorithm based on Bilinear Recurrent Neural Network. *WCICA, Fifth World Congress on Intelligent Control and Automation* **3** (2004) 198 -1984
6. Park, D. C., Zhu, Y.: Bilinear Recurrent Neural Network. *IEEE ICNN* **3** (1994) 1459-1464
7. Choi, J., Ko, K., Hong, I.: Equalization Techniques using a Simplified Bilinear Recursive Polynomial Perceptron with Decision Feedback. *Proceedings. IJCNN '01. International Joint Conference on Neural Networks* **4** (2001) 2883-2888
8. Drewes, C., Hasholzner, R., Hammerschmidt, J.S.: Adaptive Equalization for Wireless ATM. *13th Int Conf. DSP' 97, Santorini, Greece* (1997) 57-60

Short-Term Load Forecasting Using BiLinear Recurrent Neural Network

Sung Hwan Shin and Dong-Chul Park

ICRL and Myongji IT Engineering Research Institute,
Myong Ji University, Korea
{sshwan, parkd}@mju.ac.kr

Abstract. A prediction scheme of short-term electric load forecasting using a BiLinear Recurrent Neural Network (BLRNN) is proposed in this paper. Since the BLRNN is based on the bilinear polynomial, it has been successfully used in modeling highly nonlinear systems with time-series characteristics and the BLRNN can be a natural choice in predicting electric load. The performance of the proposed BLRNN-based predictor is evaluated and compared with the conventional MultiLayer Perceptron Type Neural Network (MLPNN)-based predictor. Experiments are conducted on load data from the North-American Electric Utility (NAEU). The results show that the proposed BLRNN-based predictor outperforms the MLPNN-based one in terms of the Mean Absolute Percentage Error (MAPE).

1 Introduction

Forecasting of electricity load can be performed by approximating an unknown nonlinear function of load data and other exhaustive variables such as weather variables. Traditionally, statistical models such as the autoregressive model [1], the linear regression model [2], and the autoregressive moving average (ARMA) [3] have been widely used in practice because of their simplicity. However, these statistical models are based on linear analysis techniques. Therefore, these models may not be suitable for load forecasting since using linear-based models for approximating nonlinear function often lead to inaccurate forecasting.

In recent years, various nonlinear-based models have been proposed for load forecasting. Among these models, neural network (NN)-based models have been proven to be an efficient choice for load forecasting because of their universal approximation abilities. Neural networks have been shown to have the ability not only to model time series load curves but also to model an unspecified nonlinear relationship between a load series and weather variables [4,6,7,8]. A comprehensive review of the application of neural networks to load forecasting from most recent studies shows that neural network-based models give encouraging results and are well accepted in practice by many utilities [9]. However, due to the very high cost associated with errors in practice, developing efficient load forecasting models still receive great interests.

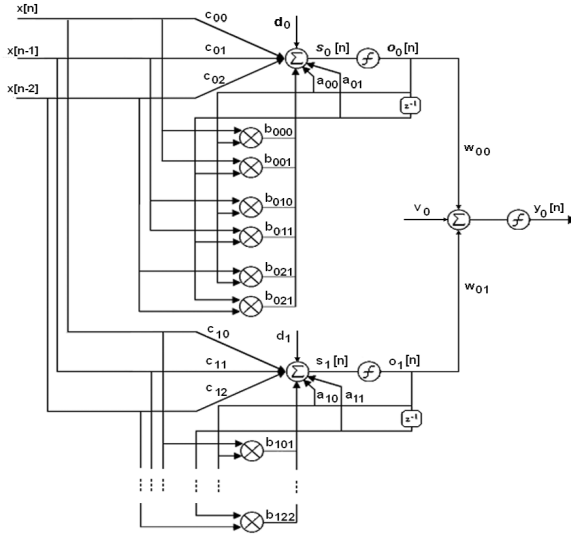


Fig. 1. Simple BLRNN with structure 3-2-1 and 2 recursion lines

Artificial neural networks of multi-layered perceptron type have shown encouraging results [4,5]. However, the Multi-Layered type Neural Network (MLPNN) does not provide to accurate results in several occasions when time-series tendency is stronger than the regression components in electric loads.

A recurrent type neural network called BLRNN has been introduced [10]. Since BLRNN is naturally capable to simulate the complex nonlinear system with the minimum number of parameters, it should be an alternative tool for electric load forecasting problem [11,12].

This paper is organized as follows: A brief summary of the BLRNN is presented in Section 2. Section 3 presents experiments and results including the performance comparison of conventional MLPNN model. Conclusions and some remarks are given in Section 4.

2 BiLinear Recurrent Neural Networks

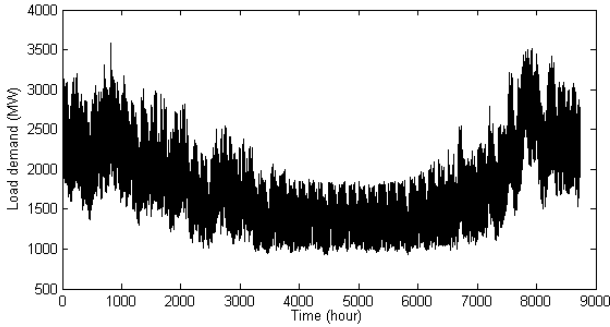
The BLRNN is a simple recurrent neural network, which has a robust ability in modeling dynamically nonlinear systems and is especially suitable for time-series data. The model was initially proposed by Park and Zhu [4]. It has been successfully applied in modeling time-series data [4,6].

Fig. 1 shows an example of a BLRNN with one hidden layer. The output value of a bilinear recurrent neuron is computed by the equation:

$$s_p[n] = d_p + \sum_{k_2=0}^{N_f-1} a_{pk_2} o_p[n - k_2] + \sum_{k_1=0}^{N_i-1} c_{pk_1} x[n - k_2] \quad (1)$$

Table 1. List of input variables for load forecasting models

Input	Variable name	Lagged value
1-5	Hourly load	1,2,3,24,168
6-10	Hourly temperature	1,2,3,24,168
11	Calendar variable	$\sin(2\pi t/24)$
12	Calendar variable	$\cos(2\pi t/24)$

**Fig. 2.** Hourly load from January 1, 1985 to December 31, 1986

$$\begin{aligned}
 & + \sum_{k_1=0}^{N_i-1} \sum_{k_2=0}^{N_f-1} b_{pk_1k_2} x[n-k_1] o_p[n-k_2] \\
 & = w_p + A_p^T A_p^T [n] + Z_p[n] B_p^T X[n] + C_p^T X[n],
 \end{aligned}$$

where d_p is the weight of bias neuron for the p -th hidden neuron, $p = 1, 2, \dots, N_h$. N_h , N_i , and N_f are the number of hidden neurons, input neurons, and feedback lines, respectively. A_p is the weight vector for recurrent portion, B_p is the weight matrix for the bilinear recurrent portion, and C_p is the weight vector for the feed-forward portion. T represents the transpose of a matrix. More detailed information on BLRNN can be found in [10,12].

3 Experiments and Results

The performance of BLRNN load forecasting model is evaluated and compared with the conventional MultiLayer Perceptron Type Neural Network (MLPNN) model on the North-American Electric Utility (NAEU) data set. The NAEU data set consists of load and temperature data and is available for download at the following web site:

<http://www.ee.washington.edu/class/559/2002spr>.

The temperature and load data were recorded at every hour of the day from January 1, 1985 to October 12, 1992, rendering 2,834 days of load and temperature data. Fig. 2 shows the hourly load demands from January 1, 1985 to

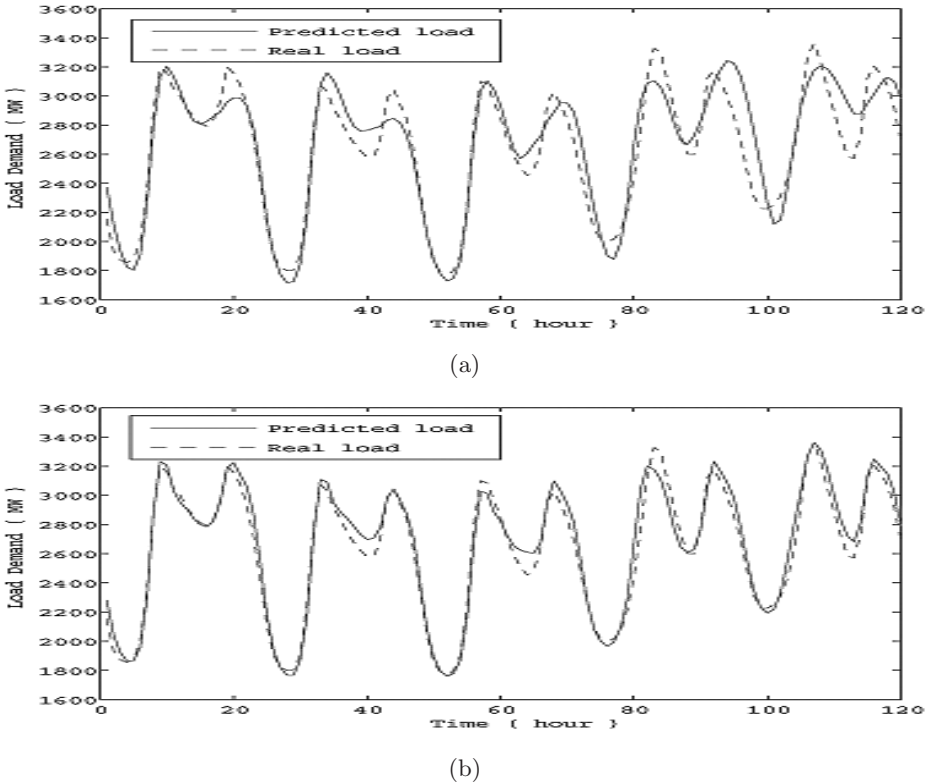


Fig. 3. 24 steps ahead hourly forecasting and actual hourly load demand form December 16, 1991 to December 21, 1991. (a) MLPNN model (b) BLRNN model.

December 31, 1985. All of data are normalized to the range of (0,1) to make them suitable for inputs of neural networks.

One of the most important steps in designing neural networks might be choosing the input variables. Selecting appropriate input variables for neural networks can be performed based on an analysis of input data. As can be seen from Fig. 2, the load demand have multiple seasonal patterns such as daily and weekly periodicity. That is high demand on day time and low demand at night time or high demand on weekday and low demand at weekend. The load demand also has a strong correlation with the temperature. Low temperature results in high demand and high temperature results in low demand. Based on these seasonal patterns and correlation analysis, the input variables for load forecasting models are selected as shown in Table 1.

The load forecasting for 1-24 hours ahead is performed by using the recursive forecasting method. The forecasted output is fed back as the input for the next time-unit forecasting and all other network inputs are shifted back one time unit. However, the future temperature is not available in practice when the recursive forecasting was performed. Therefore, it was necessary to estimate the

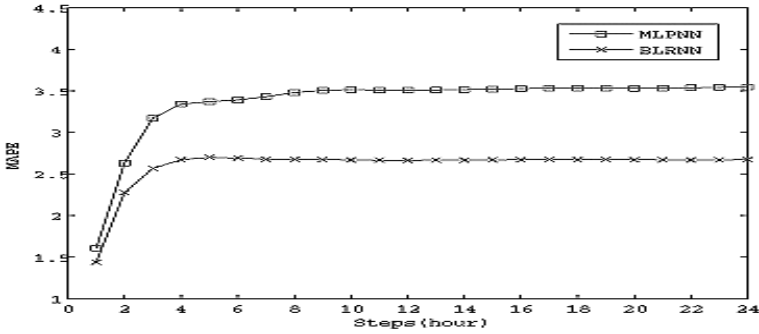


Fig. 4. 1-24steps ahead hourly forecasting performance in terms of MAPE

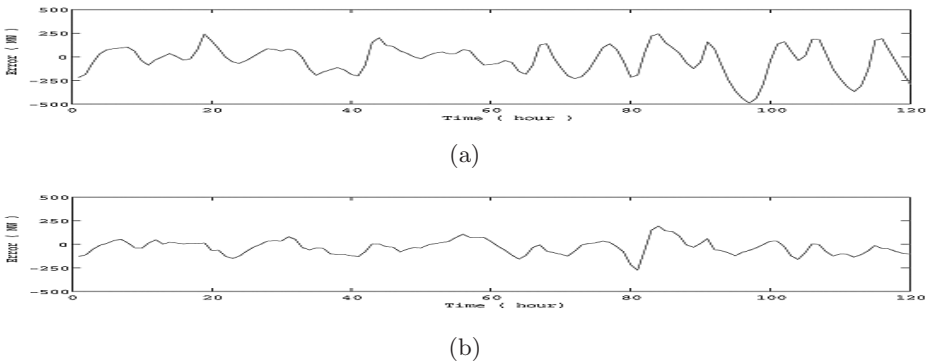


Fig. 5. 24 steps ahead hourly forecasting load error from December 16, 1991 to December 21, 1991 (a) MLPNN model (b) BLRNN model

temperature. In the experiments for this, the temperature was estimated from an average of the past temperature data.

The MLPNN model and the BLRNN model used the input variables as shown in Table 1. The above two models are trained with 3,000 iterations. they are re-trained at the end of each day to incorporate the most recent load information. All of the data used in our experiments were treated as normal working days. Holidays and anomalous days were not considered in this paper. The performance of all load forecasting models was evaluated in terms of the mean absolute percentage error (MAPE).

Fig. 3 shows the forecasting for 24 hour ahead and the real hourly load demand from December 16, 1991 to December 21, 1991 which had a typical load profile in Seattle during the winter season. Fig. 4 shows the performance over 1-24 hours ahead for the short-term load forecasting using different models during the month of December 1991. As can be seen from Fig. 4, the BLRNN model achieves a significant improvement when compared with the traditional MLPNN model. Fig. 5 plots the corresponding errors of forecasting results of each model.

4 Conclusions

A short-term load forecasting model using BiLinear Recurrent Neural Network (BLRNN) is proposed in this paper. It is important to have accurate forecasting of electric loads for several reasons including economy and security. When applying to load data from the NAEU, the proposed BLRNN model show a significant improvement about 28% decrease in MAPE of performance in comparison with the traditional MLPNN. Thus, the BLRNN model is an efficient for practical load forecasting problems.

References

1. Papalexopoulos, A.D., Hesterberg, T.C.: A Regression-Based Approach to Short-Term System Load Forecasting. *IEEE Trans. Power System* **5** (1990) 1535-1547
2. Hyde, O., Hodnett, P.F.: An Adaptable Automated Procedure for Short-Term Electricity Load Forecasting. *IEEE Trans. Power System* **12** (1997) 84-94
3. Huang, S.J., Shih, K.R.: Short-Term Load Forecasting via ARMA Model Identification Including Non-Gaussian Process Considerations. *IEEE Trans. Power System* **18** (2003) 673-679
4. Park, D.C., El-Sharkawi, M.A., Marks II, R.J., Atlas, L.E., Damborg, M.J.: Electric Load Forecasting using an Artificial Neural Network. *IEEE Trans. Power System* **6** (1991) 442-449
5. Peng, T.M., Hubele, N.F., Karady, G.G.: Advancement in the Application of Neural Networks for Short-Term Load Forecasting. *IEEE Trans.* **7** (1992) 250-257
6. Park, D.C., Park, T., Choi, S.: Short-Term Electric Load Forecasting using Recurrent Neural Network. *Proc. of ISAP'97* (1997) 367-371
7. Taylor, J.W., Buizza, R.: Neural Network Load Forecasting with Weather Ensemble Predictions. *IEEE Trans. Power System* (2002) **17** 626-632
8. Mohan, S.L., Kumar, S.M.: Artificial Neural Network-Based Peak Load Forecasting using Conjugate Gradient Methods. *IEEE Trans. Power System* **17** (2002) 907-912
9. Hippert, H.S., Pedreira, C.E., Souza, R.C.: Neural Networks for Short-Term Load Forecasting: A Review and Evaluation. *IEEE Trans. Power System* **16** (2001) 44-55
10. Park, D.C., Zhu, Y.: Bilinear Recurrent Neural Network. *IEEE ICNN* **3** (1994) 1459-1464
11. Park, D.C., Park, T.H.: DPCM with a Recurrent Neural Network Predictor for Image Compression. *IEEE IJCNN* **2** (1988) 826 -831
12. Park, D.C., Jeong, T.K.: Complex Bilinear Recurrent Neural Network for Equalization of a Satellite Channel. *IEEE Trans. Neural Networks* **13** (2002) 711-725

Convergence of Gradient Descent Algorithm for a Recurrent Neuron

Dongpo Xu, Zhengxue Li, Wei Wu*, Xiaoshuai Ding, and Di Qu

Dept. Appl. Math., Dalian University of Technology, Dalian 116023, P.R. China
wuwei@dlut.edu.cn

Abstract. Probabilistic convergence results of online gradient descent algorithm have been obtained by many authors for the training of recurrent neural networks with infinitely many training samples. This paper proves deterministic convergence of online gradient descent algorithm for a recurrent neural network with finite number of training samples. Our results can be hopefully extended to more complicated recurrent neural networks, and serve as a complementary result to the existing probability convergence results.

1 Introduction

Recurrent neural networks (RNN) are proposed by Williams and Zipser [5], and online and offline gradient learning algorithms are suggested for training RNN. Experiments have shown the effectiveness of the algorithms [6]. Probabilistic convergence properties of such learning methods have been analyzed by Kuan et al. [3] for the RNN training with infinitely many training samples. The purpose of this paper is to investigate the deterministic convergence of offline gradient descent algorithm for an RNN with finite number of training samples. For simplicity, we concentrate our attention to a simple recurrent neuron. The monotonicity of the error function in the training iteration and the convergence results are proved. Our results can be hopefully extended to more complicated RNN, and serve as complementary of the existing probabilistic convergence results.

The rest of this paper is organized as follows. The network architecture and the learning algorithm are described in the next section. Section 3 presents our main theorem on the convergence and the monotonicity of the error function sequence. The detailed proof of the theorem is given as an Appendix.

2 The Basic Algorithm

As shown in Fig.1, we consider a recurrent neuron with N external input nodes. Denote the weight vector of the network by $w = (w_0, w_1 \cdots, w_N)^T \in R^{N+1}$. Let $\xi^j = (\xi_1^j, \cdots, \xi_N^j)^T \in R^N$ denote external input signals to the network at time j

* Corresponding author, supported by the National Natural Science Foundation of China (10471017).

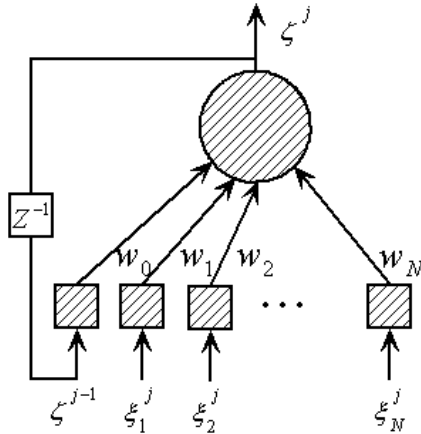


Fig. 1. A recurrent neural network with N-1-1 structure

($1 \leq j \leq J$), and $\zeta^j \in R$ the output of the network at time j . For convenience, we concatenate ζ^{j-1} and ξ^j to form an $N + 1$ dimensional vector u^j as follows:

$$u_n^j = \begin{cases} \zeta^{j-1}, & n = 0 \\ \xi_n^j, & n = 1, 2, \dots, N \end{cases} \tag{1}$$

Let

$$S^j = w \cdot u^j = \sum_{n=1}^N w_n \xi_n^j + w_0 \zeta^{j-1}, \tag{2}$$

be the input to the output node at time j , and let the output of the network be

$$\zeta^j = g(S^j), \quad j = 1, \dots, J \tag{3}$$

where g is given activation function. The initial condition is

$$\zeta^0 = 0. \tag{4}$$

We now describe the gradient descent algorithm for training this network. Let O^j denote the target output of the network at time j . The error function is defined as

$$E(w) = \frac{1}{2} \sum_{j=1}^J [O^j - g(S^j)]^2 \equiv \sum_{j=1}^J g_j(S^j). \tag{5}$$

We minimize this error function by the following gradient descent rule.

$$w^{m+1} = w^m - \eta E_w(w^m) = w^m - \eta \sum_{j=1}^J g'_j(S^{m,j}) [u^{m,j} + w_0^m p^{m,j-1}], \tag{6}$$

$$m = 1, 2, \dots$$

where w^0 is arbitrarily chosen initial guess, and

$$\begin{aligned} S^{m,j} &= S^j|_{w=w^m}, & u^{m,j} &= u^j|_{w=w^m}, \\ p^{m,j} &= p^j|_{w=w^m}, & p^j &= \left(\frac{\partial \zeta^j}{\partial w_0}, \frac{\partial \zeta^j}{\partial w_1}, \dots, \frac{\partial \zeta^j}{\partial w_N}\right)^T, \end{aligned} \tag{7}$$

satisfy

$$p^{m,j} = g'(S^{m,j})[u^{m,j} + w_0^m p^{m,j-1}], \tag{8}$$

with initial conditions

$$p^{m,0} = 0. \tag{9}$$

3 Main Results

The following assumptions will be used in our discussion.

- (A1) $|g(t)|, |g'(t)|, |g''(t)|$ are uniformly bounded for $t \in \mathbb{R}$.
- (A2) $|w_0^m|$ ($m = 0, 1, 2, \dots$) are uniformly bounded.
- (A3) There exists a closed bounded region Φ such that $\{w^n\} \subset \Phi$, and the set $\Phi_0 = \{w \in \Phi : E_w(w) = 0\}$ contains only finite points.

Remark 1. we note that from (5) and Assumption (A1) that $|g_j(t)|, |g'_j(t)|$ and $|g''_j(t)|$ are also uniformly bounded for any $t \in R$. An assumption like (A2) is often used in literature (see e.g. [1]) for a nonlinear iteration procedure to guarantee the convergence.

Theorem 1. *Suppose that the error function is given by (5), that the weight sequence $\{w^m\}$ is generated by the algorithm (6) for any initial value w^0 , and that Assumptions (A1)–(A2) are valid. Then for small enough η (cf. (22) below) we have*

- (a) $E(w^{m+1}) \leq E(w^m), \quad m = 0, 1, 2, \dots;$
- (b) *There is $E^* \geq 0$ such that $\lim_{m \rightarrow \infty} E(w^m) = E^*$;*
- (c) $\lim_{m \rightarrow \infty} \|\Delta w^m\| = 0, \quad \lim_{m \rightarrow \infty} \|E_w(w^m)\| = 0.$

Moreover, if Assumption (A3) is also valid, then we have the strong convergence:

- (d) *There exists $w^* \in \Phi_0$ such that $\lim_{m \rightarrow \infty} w^m = w^*$.*

References

1. Gori, M., Maggini, M.: Optimal convergence of on-line backpropagation. IEEE Transaction on Neural Networks **7** (1996) 251–254.
2. Haykin, S.: Neural Networks: A Comprehensive Foundation. 2nd edition. Beijing: Tsinghua University Press and Prentice Hall (2001)
3. Kuan, C.M., Hornik, K., White, H.: A Convergence Results for Learning in Recurrent Neural Networks. Neural Computation **6** (1994) 420–440

4. Ortega, J., Rheinboldt, W.: Iterative Solution of Nonlinear Equations in Several Variables. New York: Academic Press (1970)
5. Williams, R.J., Zisper, J.: A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural computation* **1** (1989) 270-280
6. Williams, R.J., Zisper, J.: Experimental Analysis of the Real-Time Recurrent Learning Algorithm. *Connection Science* **1** (1989) 81-111
7. Wu, W.: Computation of Neural Networks. Beijing: Higher Education Press (2003)
8. Yuan, Y.X., Sun, W.Y.: Optimization Theory and Methods. Beijing: Science Press (2001)
9. Zhang, X.S.: Neural Networks in Optimization. Boston: Kluwer Academic Publishers (2000)

Appendix

Let us introduce a few symbols to be used in our proof later on:

$$\Delta w^m = w^{m+1} - w^m, \Delta S^{m,j} = S^{m+1,j} - S^{m,j}.$$

To begin with, let us present four lemmas. In the following argument, we use C for a generic positive constant which may be different in different places.

The next two lemmas can be proved by induction arguments, of which the details are omitted to save the space.

Lemma 1. *It follows from (8) and (9) that*

$$p^{m,j} = \sum_{k=0}^{j-1} \left[\prod_{l=j-k}^j g'(S^{m,l}) \right] (w_0^m)^k u^{m,j-k}. \tag{10}$$

Lemma 2. *Suppose that (A1) and (A2) are satisfied, then*

$$\Delta S^{m,j} = \Delta w^m \cdot [u^{m,j} + w_0^m p^{m,j-1}] + \delta_1^{m,j} + \delta_2^{m,j}, \tag{11}$$

where

$$\delta_1^{m,j} = \Delta w^m \cdot \sum_{k=1}^{j-1} \left[\prod_{l=j-k}^{j-1} g'(S^{m,l}) \right] [(w_0^{m+1})^k - (w_0^m)^k] u^{m,j-k}, \tag{12}$$

$$\delta_2^{m,j} = \frac{1}{2} \sum_{k=1}^{j-1} \left[\prod_{l=j-k+1}^{j-1} g'(S^{m,l}) \right] (w_0^{m+1})^k g''(\theta^{m,j-k}) |\Delta S^{m,j-k}|^2, \tag{13}$$

and $\theta^{m,j-k}$ is a real number between $S^{m+1,j-1}$ and $S^{m,j-1}$.

Lemma 3. *Suppose that (A1) and (A2) are satisfied, then*

$$|\Delta S^{m,j}|^2 \leq C \|\Delta w^m\|^2, \quad m = 0, 1, \dots; j = 1, 2, \dots, J \tag{14}$$

Proof. Using the Taylor expansion to the first order, we have

$$\begin{aligned}
\Delta S^{m,j} &= S^{m+1,j} - S^{m,j} \\
&= w^{m+1} \cdot u^{m+1,j} - w^m \cdot u^{m,j} \\
&= \Delta w^m \cdot u^{m,j} + w^{m+1} \cdot [u^{m+1,j} - u^{m,j}] \\
&= \Delta w^m \cdot u^{m,j} + w_0^{m+1} [\zeta^{m+1,j-1} - \zeta^{m,j-1}] \\
&= \Delta w^m \cdot u^{m,j} + w_0^{m+1} [g'(t_1) \Delta S^{m,j-1}],
\end{aligned}$$

where t_1 lies between $S^{m+1,j-1}$ and $S^{m,j-1}$. Using (A1), (A2) and the Cauchy-Schwarz inequality, we get the recursion formula of $|\Delta S^{m,j}|^2$ about j

$$\begin{aligned}
|\Delta S^{m,j}|^2 &= |S^{m+1,j} - S^{m,j}|^2 \\
&= |\Delta w^m \cdot u^{m,j} + w_0^{m+1} [g'(t_1) \Delta S^{m,j-1}]|^2 \\
&\leq C \|\Delta w^m\|^2 + C |\Delta S^{m,j-1}|^2.
\end{aligned} \tag{15}$$

Noting

$$|\Delta S^{m,1}|^2 = |\Delta w^m \cdot u^{m,1}|^2 \leq C \|\Delta w^m\|^2,$$

and using an induction argument, we can prove that

$$|\Delta S^{m,j}|^2 \leq C \|\Delta w^m\|^2, \quad m = 0, 1, \dots; \quad j = 1, 2, \dots, J$$

This completes the proof. \square

The next lemma is basically the same as Theorem 14.1.5 in [4] (also see [8]). Its proof is thus omitted.

Lemma 4. *Let $F : \Theta \subset \mathbb{R}^m \rightarrow \mathbb{R}^m$ ($m \geq 1$) be continuous for a bounded closed region Θ , and $\Theta_0 = \{x \in \Theta : F(x) = 0\}$ contains finite points. Let the sequence $\{x^k\} \subset \Theta$ satisfy $\lim_{k \rightarrow \infty} F(x^k) = 0$ and $\lim_{k \rightarrow \infty} \|x^{k+1} - x^k\| = 0$. Then, there exists $x^* \in \Theta_0$ such that $\lim_{k \rightarrow \infty} x^k = x^*$.*

Now, we are ready for the proof of our main result.

Proof (of Theorem 1). Using the Taylor expansion, we have

$$\begin{aligned}
E(w^{m+1}) - E(w^m) &= \sum_{j=1}^J [g_j(S^{m+1,j}) - g_j(S^{m,j})] \\
&= \sum_{j=1}^J [g'_j(S^{m,j}) (\Delta S^{m,j}) + \frac{1}{2} g''_j(t_2) |\Delta S^{m,j}|^2],
\end{aligned}$$

where t_2 lies between $S^{m+1,j}$ and $S^{m,j}$. It follows from (11) that

$$\begin{aligned}
E(w^{m+1}) - E(w^m) &= \Delta w^m \cdot \sum_{j=1}^J g'_j(S^{m,j}) [u^{m,j} + w_0^m p^{m,j-1}] \\
&+ \sum_{j=1}^J (\delta_1^{m,j} + \delta_2^{m,j}) g'_j(S^{m,j}) + \delta_3^m,
\end{aligned} \tag{16}$$

where

$$\delta_3^m = \frac{1}{2} \sum_{j=1}^J g_j''(t_2) |\Delta S^{m,j}|^2. \quad (17)$$

From (6), we have

$$\sum_{j=1}^J g_j'(S^{m,j}) [u^{m,j} + w_0^m p^{m,j-1}] = -\frac{1}{\eta} \Delta w^m. \quad (18)$$

Thus

$$E(w^{m+1}) - E(w^m) = -\frac{1}{\eta} \|\Delta w^m\|^2 + \sum_{j=1}^J (\delta_1^{m,j} + \delta_2^{m,j}) g_j'(S^{m,j}) + \delta_3^m. \quad (19)$$

It follows from (A1), (A2), (12), (13) and (17) that

$$\sum_{j=1}^J (\delta_1^{m,j} + \delta_2^{m,j}) g_j'(S^{m,j}) + \delta_3^m \leq C \|\Delta w^m\|^2. \quad (20)$$

A combination of (19) and (20) leads to

$$E(w^{m+1}) - E(w^m) \leq -\left(\frac{1}{\eta} - C\right) \|\Delta w^m\|^2. \quad (21)$$

Hence, Conclusion (a) is valid if the learning rate is small enough such that

$$0 < \eta < \frac{1}{C}, \quad (22)$$

where C is the constant in (21).

Since the nonnegative sequence $\{E(w^n)\}$ is monotone and bounded below, there must be a limit value $E^* \geq 0$ such that $\lim_{m \rightarrow \infty} E(w^m) = E^*$. So Conclusion (b) is proved.

Next, we prove the weak convergence (c). Let $\beta = (\frac{1}{\eta} - C)$. By (21) we have

$$E(w^{M+1}) \leq E(w^M) - \beta \|\Delta w^M\|^2 \leq \dots \leq E(w^0) - \beta \sum_{m=0}^M \|\Delta w^m\|^2. \quad (23)$$

Since $E(w^{M+1}) \geq 0$ for any $M \geq 0$, we see that

$$\sum_{m=0}^M \|\Delta w^m\|^2 \leq \frac{1}{\beta} E(w^0) < \infty. \quad (24)$$

This together with (6) and (25) leads to

$$\lim_{m \rightarrow \infty} \|\Delta w^m\|^2 = 0, \quad \lim_{m \rightarrow \infty} \|E_w(w^m)\| = 0. \quad (25)$$

Finally, we prove the strong convergence (d). , Let us take $x = w$ and $f(x) = f(w) = E_w(w)$. Then, Conclusion (d) immediately results from a combination of Conclusion (c), the finiteness of Φ_0 (cf. Assumption (A3)), Equation (25) and Lemma 4. This completes the proof. \square

Periodicity of Recurrent Neural Networks with Reaction-Diffusion and Dirichlet Boundary Conditions

Chaojin Fu^{1,2}, Chongjun Zhu¹, and Boshan Chen¹

¹ Department of Mathematics, Hubei Normal University,
Huangshi, Hubei, 435002, China
chaojinfu@126.com

² Hubei Province Key Laboratory of Bioanalytical Technique,
Hubei Normal University, Huangshi, Hubei, 435002, China

Abstract. In this paper, a class of reaction-diffusion recurrent neural networks with time-varying delays and Dirichlet boundary conditions are considered by using an approach based on the delay differential inequality and the fixed-point theorem. Some sufficient conditions are obtained to guarantee that the reaction-diffusion recurrent neural networks have a periodic orbit and this periodic orbit is globally attractive. The results presented in this paper are the improvement and extension of the existed ones in some existing works.

1 Introduction

Recurrent neural networks (RNNs) have been found useful in areas of signal processing, image processing, associative memories, pattern classification [1]. As dynamic systems, RNNs frequently need to be analyzed for stability. The stability criteria of equilibrium points are established in a series of papers; e.g., [2]-[4]. In many applications, the properties of periodic oscillatory solutions are of great interest. For example, the human brain is in periodic oscillatory or chaos. Hence, the existence of periodic orbits of RNNs and time-varying delayed neural networks (DRNNs) is an interesting dynamic behavior. It has been found applications in learning theory, which is motivated by the fact that learning process usually requires repetition.

In addition, an equilibrium point can be viewed as a special periodic orbit of neural networks with arbitrary period. In this sense the analysis of periodic orbits of neural networks could be more general than that of equilibrium points. Recently, stability analysis and existence of periodic solutions have been widely researched for recurrent neural networks with and without delays in [5]-[9].

Moreover, both the biological neural networks and the artificial neural networks, strictly speaking, diffusion effects cannot be avoided when electrons are moving in asymmetric electromagnetic fields. So we must consider that the activations vary in space as well as in time. The stability of the neural networks with diffusion terms has been considered in [10] and [11], which are expressed

by partial differential equations. The boundary conditions of the investigated reaction-diffusion neural networks in [10] and [11] are all the Neumann boundary conditions.

Motivated by the above discussions, our aim in this paper is to consider the globally attractive periodic state of the recurrent neural networks with reaction-diffusion and Dirichlet boundary conditions.

This paper consists of the following sections. Section 2 describes some preliminaries. The main results are stated in Sections 3. Finally, concluding remarks are made in Section 4.

2 Preliminaries

Throughout of this paper, let $C([-\tau, 0] \times \mathbb{R}^m, \mathbb{R}^n)$ be the Banach space of continuous functions which map $[-\tau, 0] \times \mathbb{R}^m$ into \mathbb{R}^n with the topology of uniform converge, where τ is a constant. Let $\Omega = \{(x_1, x_2, \dots, x_m)^T \mid |x_i| < l_i, i = 1, 2, \dots, m\}$ be an open bounded domain in \mathbb{R}^m with smooth boundary $\partial\Omega$. Denote $mes\Omega > 0$ as the measure of Ω . $L^2(\Omega)$ is the space of real functions on Ω which are L^2 in the Lebesgue measure. It is a Banach space for the norm

$$\|u(t)\|_2 = \left[\sum_{i=1}^n \|u_i(t)\|_2^r \right]^{1/r},$$

where $u(t) = (u_1(t), \dots, u_n(t))^T$, $\|u_i(t)\|_2 = (\int_{\Omega} |u_i(t, x)|^2 dx)^{1/2}$, and $r \geq 1$. For any $\varphi(t, x) \in C([-\tau, 0] \times \Omega, \mathbb{R}^n)$, we define

$$\|\varphi\|_2 = \left[\sum_{i=1}^n \|\varphi_i\|_2^r \right]^{1/r},$$

where $\varphi(t, x) = (\varphi_1(t, x), \dots, \varphi_n(t, x))^T$, $\|\varphi_i\|_2 = \left(\int_{\Omega} |\varphi_i(x)|^2 dx \right)^{1/2}$,

$|\varphi_i(x)|_{\tau} = \sup_{-\tau \leq s \leq 0} |\varphi_i(s, x)|$, $|\varphi(t, x)|^{(\tau)} = \max_{1 \leq i \leq n} |\varphi_i(x)|_{\tau}$.

Consider the following reaction-diffusion delayed recurrent neural networks with the Dirichlet boundary conditions:

$$\begin{cases} \frac{\partial u_i(t, x)}{\partial t} = \sum_{k=1}^m \frac{\partial}{\partial x_k} (a_{ik} \frac{\partial u_i}{\partial x_k}) - b_i u_i(t, x) \\ \quad + \sum_{j=1}^n c_{ij} f_j(u_j(t, x)) \\ \quad + \sum_{j=1}^n d_{ij} g_j(u_j(t - \tau_j(t), x)) + I_i(t), & (x, t) \in \Omega \times [0, +\infty), \\ u_i(t, x) = 0, & (x, t) \in \partial\Omega \times [-\tau, +\infty), \\ u_i(t, x) = \phi_i(t, x), & (x, t) \in \partial\Omega \times [-\tau, 0], \end{cases} \tag{1}$$

where $i = 1, 2, \dots, n$, n is the number of neurons in the networks; $x = (x_1, x_2, \dots, x_m)^T \in \Omega \subset \mathbb{R}^m$, $u(t, x) = (u_1(t, x), u_2(t, x), \dots, u_n(t, x))^T \in \mathbb{R}^n$ and $u_i(t, x)$ is the state of the i -th neurons at time t and in point x , smooth function $a_{ik} > 0$ represents the transmission diffusion operator along the i -th unit, $b_i > 0$

represents the rate with which the i -th the unit will reset its potential to the resting state in isolation when disconnected from the networks and external inputs, c_{ij} denotes the strength of the j -th unit on the i -th unit at time t and in point x , d_{ij} denotes the strength of the j -th unit on the i -th unit at time $t - \tau_j(t)$ and in point x , $\tau_j(t)$ corresponds to time-varying transmission delay along the axon of the j -th unit and satisfies $0 \leq \tau_j(t) \leq \tau$ and $\tau_j(t + \omega) = \tau_j(t)$, where $\omega \geq 0$ is a constant. In addition, $I(t + \omega) = I(t)$. $f_j(u_j(t, x))$ denotes the activation function of the j -th unit at time t and in point x , $g_j(u_j(t - \tau_j(t), x))$ denotes the activation function of the j -th unit at time $t - \tau_j(t)$ and in point x , $\phi(t, x) = (\phi_1(t, x), \phi_2(t, x), \dots, \phi_n(t, x))^T$ and $\phi_i(t, x)$ are continuous functions.

In the following discussions, we always assume that the activation functions f_j and g_j ($j = 1, 2, \dots, n$) are globally Lipschitz continuous; i.e., $\forall j \in \{1, 2, \dots, n\}$, $\forall r_1, r_2, r_3, r_4 \in \mathbb{R}$, there exist real number ℓ_j and μ_j such that

$$|f_j(r_1) - f_j(r_2)| \leq \ell_j |r_1 - r_2|, \quad |g_j(r_3) - g_j(r_4)| \leq \mu_j |r_3 - r_4|.$$

It is easy to find that $f_j(\theta) = (1 - e^{\lambda\theta})/(1 + e^{\lambda\theta}), 1/(1 + e^{\lambda\theta}) (\lambda > 0)$, $\arctan(\theta), \max(0, \theta), (|\theta + 1| - |\theta - 1|)/2$ are all globally Lipschitz continuous.

Definition 1. An equilibrium point $u^* = (u_1^*, u_2^*, \dots, u_n^*)^T$ of the recurrent neural network (1) is said to be globally exponentially stable, if there exist constant $\varepsilon > 0$ and $\mathcal{Y} \geq 1$ such that for any initial value ϕ and $t \geq 0$,

$$\|u(t, x) - u^*\|_2 \leq \mathcal{Y} \|\phi - u^*\|_2 e^{-\varepsilon t}.$$

Definition 2. Let $f : \mathbb{R} \rightarrow \mathbb{R}$ be a continuous function. The upper right Dini-derivative $D^+ f$ is defined as

$$D^+ f(t) = \limsup_{h \rightarrow 0^+} \frac{f(t+h) - f(t)}{h}.$$

Lemma 1. Let $h(x)$ be a real-valued function belonging to $C^1(\Omega)$ which vanish on the boundary $\partial\Omega$ of Ω ; i.e., $h(x)|_{\partial\Omega} = 0$. Then

$$\int_{\Omega} h^2(x) dx \leq l_i^2 \int_{\Omega} \left| \frac{\partial h}{\partial x_i} \right|^2 dx. \tag{2}$$

Proof. If $x \in \Omega$, then

$$h(x) = \int_{-l_i}^{x_i} \frac{\partial}{\partial x_i} h(x_1, \dots, x_m) dx_i, \tag{3}$$

$$h(x) = - \int_{x_i}^{l_i} \frac{\partial}{\partial x_i} h(x_1, \dots, x_m) dx_i. \tag{4}$$

Form (3) and (4), we can obtain

$$2|h(x)| \leq \int_{-l_i}^{l_i} \left| \frac{\partial}{\partial x_i} h(x_1, \dots, x_m) \right| dx_i. \tag{5}$$

From (5) and the Schwarz's inequality,

$$|h(x)|^2 \leq \frac{l_i}{2} \int_{-l_i}^{l_i} \left| \frac{\partial}{\partial x_i} h(x_1, \dots, x_m) \right| dx_i. \tag{6}$$

Integrating both sides of (6) with respect to x_1, x_2, \dots, x_m , we get

$$\int_{\Omega} h^2(x) dx \leq l_i^2 \int_{\Omega} \left| \frac{\partial h}{\partial x_i} \right|^2 dx. \tag{7}$$

Lemma 2. Let H be a mapping on complete metric space $(C([-τ, 0] \times \mathbb{R}^m, \mathbb{R}^n), \|\cdot, \cdot\|^{(\tau)})$. If $H(C([-τ, 0] \times \mathbb{R}^m, \mathbb{R}^n)) \subset C([-τ, 0] \times \mathbb{R}^m, \mathbb{R}^n)$ and there exists a constant $\alpha < 1$ such that $\forall \phi, C([-τ, 0] \times \mathbb{R}^m, \mathbb{R}^n), \|H(\phi), \mathbb{R}^n\|, \|H(\phi), H(\varphi)\|^{(\tau)} \leq \alpha \|\phi, \varphi\|^{(\tau)}$, then there exists a unique fixed point $\phi^* \in C([-τ, 0] \times \mathbb{R}^m, \mathbb{R}^n)$ such

3 Main Results

Denote $\bar{A} = \text{diag}\{\sum_{k=1}^m \frac{a_{1k}}{l_k^2} + b_1, \sum_{k=1}^m \frac{a_{2k}}{l_k^2} + b_2, \dots, \sum_{k=1}^m \frac{a_{nk}}{l_k^2} + b_n\}$, $|C| = (|c_{ij}|)_{n \times n}$, $|D| = (|d_{ij}|)_{n \times n}$, $\ell = \text{diag}\{\ell_1, \ell_2, \dots, \ell_n\}$, $\mu = \text{diag}\{\mu_1, \mu_2, \dots, \mu_n\}$.

Theorem 1. If $\bar{A} - |C|\ell - |D|\mu$ is a nonsingular M matrix, then the neural network (1) has a periodic state which is globally attractive.

Proof. Suppose $u(t, x)$ and $v(t, x)$ are two arbitrary solutions of the neural network (1) with initial conditions $\varphi_u(t, x), \varphi_v(t, x) \in C([-τ, 0] \times \Omega, \mathbb{R}^n)$. Let $z(t, x) = (u(t, x) - v(t, x))/T$, $\varphi_z(t, x) = (\varphi_u(t, x) - \varphi_v(t, x))/T$, where the constant $T \neq 0$. Then from (1), for $i = 1, 2, \dots, n$,

$$\begin{aligned} \frac{\partial z_i(t, x)}{\partial t} &= \sum_{k=1}^m \frac{\partial}{\partial x_k} \left(a_{ik} \frac{\partial z_i(t, x)}{\partial x_k} \right) - b_i z_i(t, x) \\ &\quad + \frac{1}{T} \sum_{j=1}^n c_{ij} (f_j(u_j(t, x)) - f_j(v_j(t, x))) \\ &\quad + \frac{1}{T} \sum_{j=1}^n d_{ij} (g_j(u_j(t - \tau_j(t), x)) - g_j(v_j(t - \tau_j(t), x))). \end{aligned} \tag{8}$$

Multiplying both sides of the above equation (8) by $z_i(t, x)$ and integrating with respect to x over the domain Ω , for $i = 1, 2, \dots, n$,

$$\begin{aligned} \frac{1}{2} \frac{d}{dt} \int_{\Omega} (z_j(t, x))^2 dx &= \sum_{k=1}^m \int_{\Omega} z_i(t, x) \frac{\partial}{\partial x_k} \left(a_{ik} \frac{\partial z_i(t, x)}{\partial x_k} \right) dx - b_i \int_{\Omega} (z_i(t, x))^2 dx \\ &\quad + \frac{1}{T} \sum_{j=1}^n \int_{\Omega} c_{ij} z_i(t, x) (f_j(u_j(t, x)) - f_j(v_j(t, x))) dx \\ &\quad + \frac{1}{T} \sum_{j=1}^n \int_{\Omega} d_{ij} z_i(t, x) (g_j(u_j(t - \tau_j(t), x)) \\ &\quad - g_j(v_j(t - \tau_j(t), x))) dx. \end{aligned} \tag{9}$$

From the Green's formula and the Dirichlet boundary condition, we have

$$\sum_{k=1}^m \int_{\Omega} z_i(t, x) \frac{\partial}{\partial x_k} \left(a_{ik} \frac{\partial z_i(t, x)}{\partial x_k} \right) dx = - \sum_{k=1}^m \int_{\Omega} a_{ik} \left(\frac{\partial z_i(t, x)}{\partial x_k} \right)^2 dx. \quad (10)$$

Furthermore, from Lemma 1,

$$\begin{aligned} \sum_{k=1}^m \int_{\Omega} z_i(t, x) \frac{\partial}{\partial x_k} \left(a_{ik} \frac{\partial z_i(t, x)}{\partial x_k} \right) dx &= - \sum_{k=1}^m \int_{\Omega} a_{ik} \left(\frac{\partial z_i(t, x)}{\partial x_k} \right)^2 dx \\ &\leq - \sum_{k=1}^m \int_{\Omega} \frac{a_{ik}}{l_k^2} (z_i(t, x))^2 dx. \end{aligned} \quad (11)$$

From (9), (11), and the Holder inequality, we have:

$$\begin{aligned} \frac{d}{dt} \|z_i(t, x)\|_2^2 &\leq - \sum_{k=1}^m \frac{2a_{ik}}{l_k^2} \|z_i(t, x)\|_2^2 - 2b_i \|z_i(t, x)\|_2^2 \\ &\quad + 2 \sum_{j=1}^n |c_{ij}| \ell_j \|z_i(t, x)\|_2 \|z_j(t, x)\|_2 \\ &\quad + 2 \sum_{j=1}^n |d_{ij}| \mu_j \|z_i(t, x)\|_2 \|z_j(t - \tau_j(t), x)\|_2; \end{aligned} \quad (12)$$

i.e.,

$$\begin{aligned} \frac{d \|z_i(t, x)\|_2}{dt} &\leq - \left(\sum_{k=1}^m \frac{a_{ik}}{l_k^2} + b_i \right) \|z_i(t, x)\|_2 + \sum_{j=1}^n |c_{ij}| \ell_j \|z_j(t, x)\|_2 \\ &\quad + \sum_{j=1}^n |d_{ij}| \mu_j \|z_j(t - \tau_j(t), x)\|_2. \end{aligned} \quad (13)$$

Since $\bar{A} - |C|\ell - |D|\mu$ is a nonsingular M matrix, there exist positive numbers $\gamma_1, \dots, \gamma_n$ such that

$$\gamma_i \left(\sum_{k=1}^m \frac{a_{ik}}{l_k^2} + b_i \right) - \sum_{j=1}^n \gamma_j (|c_{ij}| \ell_j + |d_{ij}| \mu_j) > 0. \quad (14)$$

Let $y_i(t, x) = \|z_i(t, x)\|_2 / \gamma_i$. From (13),

$$\begin{aligned} D^+ y_i(t, x) &\leq - \left(\sum_{k=1}^m \frac{a_{ik}}{l_k^2} + b_i \right) y_i(t, x) + \left(\sum_{j=1}^n \gamma_j |c_{ij}| \ell_j y_j(t, x) \right. \\ &\quad \left. + \sum_{j=1}^n \gamma_j |d_{ij}| \mu_j y_j(t - \tau_j(t), x) \right) / \gamma_i. \end{aligned} \quad (15)$$

From (14) there exists a constant $\theta > 0$ such that

$$\gamma_i \left(\sum_{k=1}^m \frac{a_{ik}}{l_k^2} + b_i \right) - \sum_{j=1}^n \gamma_j (|c_{ij}| \ell_j + |d_{ij}| \mu_j e^{\theta \tau}) \geq 0. \tag{16}$$

Let $\nu(0, x) = \max_{1 \leq i \leq n} \{ \sup_{-\tau \leq s \leq 0} \{ y_i(s, x) \} \}$. Then $\forall t \geq 0$,

$$\|y(t, x)\| \leq \nu(0, x) \exp\{-\theta t\}. \tag{17}$$

Otherwise, there exist $t_2 > t_1 > 0$, $q \in \{1, 2, \dots, n\}$ and sufficiently small $\varepsilon > 0$ such that $\forall s \in [-\tau, t_1]$, (17) holds, and

$$y_i(s, x) \leq \nu(0, x) \exp\{-\theta s\} + \varepsilon, \quad s \in (t_1, t_2], i \in \{1, 2, \dots, n\}, \tag{18}$$

$$D^+ y_q(t_2, x) + \theta \nu(0, x) \exp\{-\theta t_2\} > 0. \tag{19}$$

But from (15), (16) and (18),

$$D^+ y_q(t_2, x) + \theta \nu(0, x) \exp\{-\theta t_2\} \leq 0. \tag{20}$$

Hence, from this conclusion of absurdity, it shows that (17) holds.

Define $u_\phi^{(t)}(\theta) = u(t+\theta, x; \phi)$, $\theta \in [-\tau, 0]$. Define a mapping $H : C([-\tau, 0] \times \mathbb{R}^m, \mathbb{R}^m, \mathbb{R}^n) \rightarrow C([-\tau, 0] \times \mathbb{R}^m, \mathbb{R}^n)$ by $H(\phi) = u_\phi^{(\omega)}$. Then $H(C([-\tau, 0] \times \mathbb{R}^m, \mathbb{R}^n)) \subset C([-\tau, 0] \times \mathbb{R}^m, \mathbb{R}^n)$, and $H^m(\phi) = u_\phi^{(m\omega)}$.

From (17), $\|u(t, x; \phi) - u(t, x; \varphi)\|^{(\tau)} \leq \frac{\max_{1 \leq i \leq n} \{\gamma_i\}}{\min_{1 \leq i \leq n} \{\gamma_i\}} \|\phi, \varphi\|^{(\tau)} \exp\{-\theta t\}$.

Choose a positive integer m such that $\frac{\max_{1 \leq i \leq n} \{\gamma_i\}}{\min_{1 \leq i \leq n} \{\gamma_i\}} \exp\{-m\omega\} \leq \alpha < 1$. Hence,

$$\begin{aligned} \|H^m(\phi), H^m(\varphi)\|^{(\tau)} &= \|y(m\omega + \theta, x; \phi) - y(m\omega + \theta, x; \varphi)\|^{(\tau)} \\ &\leq \|\phi, \varphi\|^{(\tau)} \exp\{-m\omega\} \leq \alpha \|\phi, \varphi\|^{(\tau)}. \end{aligned}$$

According to Lemma 2, there exists a unique fixed point $\phi^* \in \phi^* \in C([-\tau, 0] \times \mathbb{R}^m, \mathbb{R}^n)$ such that that $H^m(\phi^*) = \phi^*$. In addition, for any integer $r \geq 1$, $H^m(H^r(\phi^*)) = H^r(H^m(\phi^*)) = H^r(\phi^*)$. This shows that $H^r(\phi^*)$ is also a fixed point of H^m . Hence, by the uniqueness of the fixed point of the mapping H^m , $H^r(\phi^*) = \phi^*$, that is, $u_{\phi^*}^{(r\omega)} = \phi^*$. Let $u(t, x; \phi^*)$ be a state of the neural network (1) with initial condition ϕ^* . Then from (1), $\forall i = 1, 2, \dots, n, \forall t \geq 0$,

$$\begin{aligned} \frac{\partial u_i(t, x; \phi^*)}{\partial t} &= \sum_{k=1}^m \frac{\partial}{\partial x_k} \left(a_{ik} \frac{\partial u_i}{\partial x_k} \right) - b_i u_i(t, x; \phi^*) + \sum_{j=1}^n c_{ij} f_j(u_j(t, x; \phi^*)) \\ &\quad + \sum_{j=1}^n d_{ij} g_j(u_j(t - \tau_j(t), x; \phi^*)) + I_i(t). \end{aligned}$$

Hence, $\forall i = 1, 2, \dots, n, \forall t + \omega \geq 0$,

$$\begin{aligned}
\frac{\partial u_i(t + \omega, x; \phi^*)}{\partial t} &= \sum_{k=1}^m \frac{\partial}{\partial x_k} \left(a_{ik} \frac{\partial u_i}{\partial x_k} \right) - b_i u_i(t + \omega, x; \phi^*) \\
&\quad + \sum_{j=1}^n c_{ij} f_j(u_j(t + \omega, x; \phi^*)) \\
&\quad + \sum_{j=1}^n d_{ij} g_j(u_j(t + \omega - \tau_j(t), x; \phi^*)) + I_i(t + \omega) \\
&= \sum_{k=1}^m \frac{\partial}{\partial x_k} \left(a_{ik} \frac{\partial u_i}{\partial x_k} \right) - b_i u_i(t + \omega, x; \phi^*) \\
&\quad + \sum_{j=1}^n c_{ij} f_j(u_j(t + \omega, x; \phi^*)) \\
&\quad + \sum_{j=1}^n d_{ij} g_j(u_j(t + \omega - \tau_j(t + \omega), x; \phi^*)) + I_i(t),
\end{aligned}$$

this implies $u(t + \omega, x; \phi^*)$ is also a state of the neural network (1) with initial condition ϕ^* . $u_{\phi^*}^{(r\omega)} = \phi^*$ implies that $u(r\omega + \theta, x; \phi^*) = y((r - 1)\omega + \theta, x; \phi^*)$. $\forall t \geq 0$, there exist \bar{r} and $\bar{\theta}$ such that $t = (\bar{r} - 1)\omega + \bar{\theta}$; i.e., $u(t + \omega, x; \phi^*) = u(t, x; \phi^*)$. Hence, $u(t, x; \phi^*)$ is a periodic orbit of the neural network (1) with period ω .

From (17), it is easy to see that all other states of (1) converge to this periodic orbit as $t \rightarrow +\infty$. Hence, the periodic orbit $u(t, x; \phi^*)$ is globally attractive.

If $a_{ik} \equiv 0$, consider recurrent neural networks with time-varying delays

$$\frac{\partial u_i(t, x)}{\partial t} = -b_i u_i(t, x) + \sum_{j=1}^n (c_{ij} f_j(u_j(t, x)) + d_{ij} g_j(u_j(t - \tau_j(t), x))) + I_i(t), \quad (21)$$

where $i = 1, \dots, n$.

Denote $B = \text{diag}\{b_1, b_2, \dots, b_n\}$.

Corollary 1. If $B - |C|\ell - |D|\mu$ is a nonsingular M matrix, then the neural network (21) has a periodic state which is globally attractive.

Proof. According to Theorem 1, Corollary 1 holds.

Since an equilibrium point can be viewed as a special periodic orbit of neural networks with arbitrary period, according to Theorems 1 and 2, we have the following corollaries:

Corollary 2. If $\bar{A} - |C|\ell - |D|\mu$ is a nonsingular M matrix, and $I(t) \equiv I$, where I is a constant, then (1) is globally exponentially stable.

Corollary 3. If $B - |C|\ell - |D|\mu$ is a nonsingular M matrix, and $I(t) \equiv I$, where I is a constant, then (21) is globally exponentially stable.

4 Concluding Remarks

In this paper, using the delay differential inequality and the fixed-point theorem, we have obtained some sufficient conditions to guarantee that the reaction-diffusion recurrent neural networks have a periodic orbit and this periodic orbit is globally attractive. The results presented in this paper are the improvement and extension of the existed ones in some existing works.

Acknowledgement

This work was supported by the Natural Science Foundation of China under Grant 60405002, and the Key Project of Hubei Provincial Education Department of China under Grant B20052201.

References

1. Chua, L.O., Yang, L.: Cellular Neural Networks: Theory. *IEEE Trans. Circuits and Systems* **35** (1988) 1257-1272
2. Forti, M., Tesi, A.: New Conditions for Global Stability of Neural Networks with Application to Linear and Quadratic Programming Problems. *IEEE Trans. Circuits and Systems* **I 42** (1995) 354-366
3. Liao, X.X., Wang, J.: Algebraic Criteria for Global Exponential Stability of Cellular Neural Networks with Multiple Time Delays. *IEEE Trans. Circuits and Systems* **I 50** (2003) 268-275
4. Zeng, Z.G., Wang, J.: Complete Stability of Cellular Neural Networks with Time-varying Delays. *IEEE Trans. Circuits and Systems* **I 53** (2006) 944-955
5. Yi, Z., Heng, P.A., Vadakkepat, P.: Absolute Periodicity and Absolute Stability of Delayed Neural Networks. *IEEE Trans. Circuits and Systems* **I 49** (2002) 256-261
6. Sun, C., Feng, C.: Global Robust Exponential Stability of Interval Neural Networks with Delays. *Neural Processing Letters* **17** (2003) 107-115
7. Zeng, Z.G., Huang, D.S., and Wang, Z.F.: Global Stability of a General Class of Discrete-time Recurrent Neural Networks. *Neural Processing Letters* **22** (2005) 33-47
8. Zeng, Z.G., Wang, J., Liao, X.X.: Global Asymptotic Stability and Global Exponential Stability of Neural Networks with Unbounded Time-varying Delays. *IEEE Trans. Circuits and Systems* **II 52** (2005) 168-173
9. Zeng, Z.G., Wang, J.: Multiperiodicity and Exponential Attractivity Evoked by Periodic External Inputs in Delayed Cellular Neural Networks. *Neural Computation* **18** (2006) 848-870
10. Song, Q., Cao, J.: Global Exponential Stability and Existence of Periodic Solutions in BAM Networks with Delays and Reaction Diffusion Terms. *Chaos, Solitons & Fractals* **23** (2005) 421-430
11. Song, Q., Cao, J., Zhao, Z.: Periodic Solutions and Its Exponential Stability of Reaction-Diffusion Recurrent Neural Networks with Continuously Distributed Delays. *Nonlinear Analysis: Real World Applications* **7** (2006) 65-80

New Critical Analysis on Global Convergence of Recurrent Neural Networks with Projection Mappings

Chen Qiao and Zong-Ben Xu

Institute for Information and System Science,
Faculty of Science,
Xi'an Jiaotong University, Xi'an, Shaan'xi, 710049, P.R. China
{qiaochen, zbxu}@mail.xjtu.edu.cn

Abstract. In this paper, we present the general analysis of global convergence for the recurrent neural networks (RNNs) with projection mappings in the critical case that $M(L, \Gamma)$, a matrix related with the weight matrix W and the activation mapping of the networks, is nonnegative for a positive diagonal matrix Γ . In contrast to the existing conclusion such as in [1], the present critical stability results do not require the condition that ΓW must be symmetric and can be applied to the general projection mappings other than nearest point projection mappings. An example has also been shown that the theoretical results obtained in the present paper have explicitly practical application.

1 Introduction

The recurrent neural networks (RNNs) with projection mappings are widely studied in recent years because of their immense potentials of application perspective. The analysis of dynamic behaviors of the networks is a first and necessary step for any practical design and application of such networks. One kind of the dynamical behaviors must be expected is the globally convergent dynamics. However, it is by no means easy to conduct a meaningful global convergence analysis. Recently, such analysis has been focused on a stability analysis of the system in the critical case when the discriminant matrix $M(L, \Gamma) = L^{-1}\Gamma - (\Gamma W + W^T\Gamma)/(2)$ is nonnegative, where W is the weight matrix of the network, $G(x) = (g_1(x_1), g_2(x_2), \dots, g_N(x_N))^T$ is the activation mapping and $L = \text{diag}\{L_1, L_2, \dots, L_N\}$ with each L_i being the minimum Lipschitz constant of g_i . There have been considerable efforts on the no critically global stability analysis of the RNNs (see, e.g., [2][5][4][6] and the references therein). In [1], a global convergence analysis in the critical case was conducted for the nearest point projection under some special conditions.

The purpose of this paper is to present a critical analysis on the global convergence of the RNNs with projection mappings (other than nearest point projection) and discriminate the symmetric requirement in [1].

2 Review of Some Existing Stability Results

We consider the RNN model as:

$$\tau \frac{dx(t)}{dt} = -x(t) + G(Wx(t) + q), \quad x(0) = x_0 \in \mathcal{R}^N \tag{1}$$

where $x(t) = (x_1(t), x_2(t), \dots, x_N(t))^T$ is the neural states, τ is a positive constant, $W = (\omega_{ij})_{N \times N}$ is the synaptic weight matrix and $G = (g_1, g_2, \dots, g_N)^T$ is the activation mapping.

For the nonlinear activation mapping $G : \mathcal{R}^N \rightarrow \mathcal{R}^N$, denote the *domain*, the *range* and the *fixed-point-set* of G respectively by $\mathbf{D}(G)$, $\mathbf{R}(G)$ and $\mathbf{F}(G)$. G is said to be *diagonally nonlinear* if G is defined componentwisely by $G(x) = (g_1(x_1), g_2(x_2), \dots, g_N(x_N))^T$, where each g_i is an one-dimensional nonlinear function; G is said to be a *projection mapping* if $G \circ G = G$, or equivalently, $\mathbf{R}(G) \subseteq \mathbf{F}(G)$; G is said to be a *diagonal projection* if it is diagonally nonlinear, and each component function g_i is an one-dimensional projection; G is said to be a *nearest point projection* if there is a bounded, closed and convex subset $\Omega \subset \mathbf{R}^N$ such that $G(x) = \underset{z \in \Omega}{\text{arg min}} \|x - z\|$. Such a mapping is denoted by P_Ω , i.e., $G(x) = P_\Omega(x)$. Obviously, nearest point projection is a special projection mapping, but the inverse is not necessarily true. L_i , the *minimum Lipschitz constant* of g_i , is defined as follows

$$L_i = \sup_{t \neq s} \frac{|g_i(t) - g_i(s)|}{|t - s|}. \tag{2}$$

In [1], the author proved the following theorem.

Theorem 1. *Let $G(\cdot) = P_\Omega(\cdot) = (g_1, g_2, \dots, g_N)^T$ be a diagonal nearest point projection with a bounded, closed and convex subset $\Omega \subset \mathcal{R}^N$. Assume that there is a positive diagonal matrix $\Gamma = \text{diag}\{\xi_1, \xi_2, \dots, \xi_N\}$ such that*

- (i) ΓW is symmetric and
- (ii) $\Gamma(I - W)$ is nonnegative definite.

Then RNN model (1) is globally convergent on Ω . That is, for any trajectory $x(t)$ of (1) starting from Ω , there corresponds an equilibrium state x^ of (1) such that $\lim_{t \rightarrow +\infty} x(t) = x^*$.*

At present, one case of the critical stability analysis of system (1) is refer to as the analysis on stability of (1) when $M(L, \Gamma) = L^{-1}\Gamma - (\Gamma W + W^T \Gamma)/(2) \geq 0$. In respect that $L_i = 1$ when $P_\Omega(\cdot)$ is a diagonal nearest point projection mapping [1, 7], so, in Theorem 1, the critical stability analysis of model (1) was conducted under the conditions that ΓW is symmetric, $M(L, \Gamma)$ is nonnegative definite and the activation mapping is a diagonal nearest point projection. In this paper, we will present some more general critical global convergence analysis of system (1) in the case when $M(L, \Gamma) \geq 0$ and the activation mapping is a generally diagonal projection mappings. The established results are without the assumption that ΓW is symmetric, and generalize most known stability results of the systems.

3 Critical Global Convergence Results

To begin the main results of the paper, we need two lemmas.

Lemma 1. (see [3][1]). *If $G : \mathcal{R}^N \rightarrow \Theta$ is a projection mapping with Θ being a bounded, closed and convex subset of \mathcal{R}^N , then the solution $x(t, x_0)$ of (1) satisfies $x(t, x_0) \in \Theta$ for all $t \geq 0$ when $x_0 \in \Theta$.*

Lemma 2. *For any continuous, real-valued and monotonically increasing function g defined on \mathcal{R} , we have*

$$\int_s^t (g(\theta) - g(s))d\theta \geq L^{-1}(g(t) - g(s))^2/2, \quad \forall t, s \in \mathcal{R} \tag{3}$$

where L is the minimum Lipschitz constant of g . Specially, $L^{-1} = 0$ when $L = +\infty$.

Proof. Let

$$f(t) = \int_s^t (g(\theta) - g(s))d\theta - L^{-1}(g(t) - g(s))^2/2.$$

If g is continuously differentiable, then it can be shown that

$$f'(t) = (1 - L^{-1}g'(t))(g(t) - g(s)).$$

It is clear that $|L^{-1}g'(t)| \leq 1$. Since g is monotonically increasing, then, $f'(t) \geq 0$ when $t \geq s$ and $f'(t) \leq 0$ when $t \leq s$. Thus, $f(t)$ attains its unique minimum at $t = s$. In view of the fact that $f(s) = 0$, then the required estimate (3) follows when g is continuously differentiable.

Since any continuous function can be arbitrarily approximated by continuously differentiable functions, then (3) also holds when g is continuous. The proof is thus completed.

We now state and prove the main results of this paper.

Theorem 2. *Assume that $G = (g_1, g_2, \dots, g_N)^T$ is a diagonal projection with each g_i being monotonically increasing and continuous. Suppose that $\mathbf{R}(G) \subset \mathcal{R}^N$ is a bounded, closed and convex set. Let $L = \text{diag}\{L_1, L_2, \dots, L_N\}$ with each L_i being the minimum Lipschitz constant of g_i . If there is a positive diagonal matrix $\Gamma = \text{diag}\{\xi_1, \xi_2, \dots, \xi_N\}$ such that*

- (i) $M(L, \Gamma) = L^{-1}\Gamma - \frac{\Gamma W + W^T \Gamma}{2}$ is nonnegative definite and
- (ii) $\langle \Gamma(L^{-1}z(t) + x(t)), x(t) - z(t) \rangle \geq 0$, where $z(t) = G(Wx(t) + q)$,

then RNN model (1) is globally convergent on $\mathbf{R}(G)$. That is, for any trajectory $x(t)$ of (1) starting from $\mathbf{R}(G)$, there corresponds an equilibrium state x^* of (1) such that $\lim_{t \rightarrow +\infty} x(t) = x^*$.

Proof. For any trajectory $x(t)$ of (III) starting from $x_0 \in \mathbf{R}(G)$, let $y_0 = Wx_0 + q$ and $y(t) = Wx(t) + q$. Define

$$\begin{aligned} E_1(x(t)) &= \tau x^T(t)((L^{-1} + I)\Gamma - (\Gamma W + W^T \Gamma))x(t) - 2\tau x^T(t)\Gamma q \\ &\quad + \tau \sum_{i=1}^N \xi_i \int_{(y_0)_i}^{y_i(t)} g_i(s) ds - \frac{1}{2} \tau \{ (y(t) - x(t))^T \Gamma (y(t) - x(t)) \\ &\quad - y^T(t)\Gamma y(t) - x^T(t)(I - L^{-1})\Gamma x(t) \}. \end{aligned}$$

The proof will be broken down into four steps.

Step 1) We want to show that $\lim_{t \rightarrow +\infty} \frac{dE_1(x(t))}{dt} = 0$.

Note first that

$$\begin{aligned} &\frac{1}{2} \tau \frac{d}{dt} [(y(t) - x(t))^T \Gamma (y(t) - x(t)) - y^T(t)\Gamma y(t) - x^T(t)(I - L^{-1})\Gamma x(t)] \\ &= \langle \Gamma x(t), (L^{-1} - W)(-x(t) + z(t)) \rangle - \langle \Gamma y(t), -x(t) + z(t) \rangle \end{aligned} \quad (4)$$

and

$$\begin{aligned} &\tau \frac{d}{dt} \left(\sum_{i=1}^N \xi_i \int_{(y_0)_i}^{y_i(t)} g_i(s) ds \right) \\ &= \langle L^{-1} \Gamma z(t), -x(t) + z(t) \rangle - \langle \Gamma z(t), (L^{-1} - W)(-x(t) + z(t)) \rangle. \end{aligned} \quad (5)$$

Since $((L^{-1} + I)\Gamma - (\Gamma W + W^T \Gamma))$ is symmetric, let $u(t) = -x(t) + z(t)$, then a direct calculation by using (4) and (5) is

$$\begin{aligned} \frac{dE_1(x(t))}{dt} &= 2\langle ((L^{-1} + I)\Gamma - (\Gamma W + W^T \Gamma))x(t), u(t) \rangle - 2\langle \Gamma q, u(t) \rangle \\ &\quad + \langle L^{-1} \Gamma z(t), u(t) \rangle - \langle \Gamma z(t), (L^{-1} - W)u(t) \rangle \\ &\quad - \langle \Gamma x(t), (L^{-1} - W)u(t) \rangle + \langle \Gamma y(t), u(t) \rangle \\ &= -2\langle \Gamma(y(t) - x(t)), u(t) \rangle + 2\langle L^{-1} \Gamma x(t), u(t) \rangle - 2\langle \Gamma x(t), Wu(t) \rangle \\ &\quad + \langle L^{-1} \Gamma z(t), u(t) \rangle - \langle \Gamma z(t), (L^{-1} - W)u(t) \rangle \\ &\quad - \langle \Gamma x(t), (L^{-1} - W)u(t) \rangle + \langle \Gamma y(t), u(t) \rangle \\ &= -2\langle \Gamma(y(t) - x(t)), u(t) \rangle + \langle \Gamma(x(t) - z(t)), (L^{-1} - W)u(t) \rangle \\ &\quad + \langle L^{-1} \Gamma z(t), u(t) \rangle + \langle \Gamma y(t), u(t) \rangle \\ &= -\langle \Gamma(y(t) - x(t)), -x(t) + z(t) \rangle - \langle \Gamma(y(t) - x(t)), -x(t) + z(t) \rangle \\ &\quad - (x(t) - z(t))^T (L^{-1} \Gamma - \frac{\Gamma W + W^T \Gamma}{2})(x(t) - z(t)) \\ &\quad + \langle \Gamma(L^{-1} z(t) + y(t)), -x(t) + z(t) \rangle \\ &= -\langle \Gamma(y(t) - x(t)), -x(t) + z(t) \rangle - (x(t) - z(t))^T (L^{-1} \Gamma \\ &\quad - \frac{\Gamma W + W^T \Gamma}{2})(x(t) - z(t)) + \langle \Gamma(L^{-1} z(t) + x(t)), -x(t) + z(t) \rangle. \end{aligned}$$

It follows from Lemma 1 that $x(t) \in \mathbf{R}(G)$, so $x(t) \in \mathbf{F}(G)$ since G is a projection. Meanwhile, on noting that $L^{-1} \Gamma - \frac{\Gamma W + W^T \Gamma}{2}$ is nonnegative and

$\langle \Gamma(L^{-1}z(t) + x(t)), x(t) - z(t) \rangle \geq 0$, we get

$$\frac{dE_1(x(t))}{dt} \leq - \sum_{i=1}^N \xi_i [(Wx(t) + q)_i - x_i(t)] \times [g_i((Wx(t) + q)_i) - g_i(x_i(t))] \quad (6)$$

Note that $\frac{dE_1(x(t))}{dt}$ is continuous and $x(t) \in \mathbf{F}(G)$ with $\mathbf{F}(G)$ being bounded and closed, it follows that $\frac{dE_1(x(t))}{dt}$ is a uniformly continuous function of t in $[0, +\infty)$. Further, we have $\frac{dE_1(x(t))}{dt} \leq 0$ since g_i is monotonically increasing, which implies that the limit $\lim_{t \rightarrow +\infty} E_1(x(t))$ exists. Thus, applying the well-known Barbalat Lemma ([8], p. 123), we obtain that $\lim_{t \rightarrow +\infty} \frac{dE_1(x(t))}{dt} = 0$.

Step 2) We show that any limit point of $x(t)$ is an equilibrium state of (II).

Let x^* be any limit point of $x(t)$, i.e., $\lim_{n \rightarrow +\infty} x(t_n) = x^*$ for some positive sequence $\{t_n\}$ with $t_n \rightarrow +\infty$ as $n \rightarrow +\infty$. (Notice that x^* exists since $x(t)$ is bounded).

From (6) and the monotonically increasing property of each g_i , we have

$$\begin{aligned} 0 &= \lim_{t \rightarrow +\infty} \frac{dE_1(x(t))}{dt} \\ &\leq \liminf_{t \rightarrow +\infty} \left\{ - \sum_{i=1}^N \xi_i [(Wx(t) + q)_i - x_i(t)] \times [g_i((Wx(t) + q)_i) - g_i(x_i(t))] \right\} \\ &\leq 0. \end{aligned}$$

Consequently,

$$\lim_{t \rightarrow +\infty} \sum_{i=1}^N \xi_i [(Wx(t) + q)_i - x_i(t)] \times [g_i((Wx(t) + q)_i) - g_i(x_i(t))] = 0. \quad (7)$$

Then, it can be deduced that

$$\sum_{i=1}^N \xi_i [(Wx^* + q)_i - x_i^*] \times [g_i((Wx^* + q)_i) - g_i(x_i^*)] = 0. \quad (8)$$

Note that $R(G)$ is closed, which combined with the projection property of each g_i , implies that $g_i(x_i^*) = x_i^*$. Thus, whether $(Wx^* + q)_i - x_i^* = 0$ or $g_i((Wx^* + q)_i) - g_i(x_i^*) = 0$, it is easy to see that $g_i((Wx^* + q)_i) = x_i^*$ for all $i = 1, 2, \dots, N$. That means x^* is an equilibrium state of (II).

Step 3) Define $y(t_n) = Wx(t_n) + q$ and $y^* = Wx^* + q$, we want to show $\lim_{t \rightarrow +\infty} (G(y(t)) - G(y^*)) = 0$.

Consider the follow system:

$$\tau \frac{dy(t)}{dt} = -y(t) + WG(y(t)) + q, \quad y(0) = y_0 \in \mathcal{R}^N \quad (9)$$

it is clear that y^* is an equilibrium state of system (9) and $\lim_{n \rightarrow +\infty} y(t_n) = y^*$.

Let

$$E_2(x(t)) = \tau \sum_{i=1}^N \xi_i \int_{y_i^*}^{y_i(t)} (g_i(r) - g_i(y_i^*)) dr .$$

Then, it is obviously that $E_2(x(t)) \geq 0$ by Lemma 2. Meanwhile, we have

$$\begin{aligned} \frac{dE_2(x(t))}{dt} &= \sum_{i=1}^N \xi_i (g_i(y_i(t)) - g_i(y_i^*)) \cdot \left(\tau \frac{dy_i(t)}{dt} \right) \\ &= (G(y(t)) - G(y^*))^T \Gamma W (-x(t) + G(y(t))) \\ &= (G(y(t)) - G(y^*))^T \Gamma W (-x(t) + x^* - G(y^*) + G(y(t))) \\ &= -(G(y(t)) - G(y^*))^T \Gamma (y(t) - y^*) \\ &\quad + (G(y(t)) - G(y^*))^T \frac{\Gamma W + W^T \Gamma}{2} (G(y(t)) - G(y^*)) \end{aligned} \tag{10}$$

On noting that g_i is monotonically increasing and has the minimum Lipschitz constant L_i , we obtain that

$$\begin{aligned} (G(y(t)) - G(y^*))^T \Gamma (y(t) - y^*) &= \sum_{i=1}^N \xi_i (g_i(y_i) - g_i(y_i^*)) (y_i - y_i^*) \\ &\geq \sum_{i=1}^N \xi_i L_i^{-1} (g_i(y_i) - g_i(y_i^*))^2 \end{aligned}$$

whenever $y_i \geq y_i^*$ or $y_i < y_i^*$. Then,

$$\frac{dE_2(x(t))}{dt} \leq -(G(y(t)) - G(y^*))^T \left(L^{-1} \Gamma - \frac{\Gamma W + W^T \Gamma}{2} \right) (G(y(t)) - G(y^*)) .$$

Thus, the nonnegative definiteness of $(L^{-1} \Gamma - \frac{\Gamma W + W^T \Gamma}{2})$ implies $\frac{dE_2(x(t))}{dt} \leq 0$ for all $t \geq 0$, and furthermore, $\lim_{t \rightarrow +\infty} E_2(x(t))$ exists. This, together with the fact that $\lim_{n \rightarrow +\infty} y(t_n) = y^*$, implies that $\lim_{t \rightarrow +\infty} E_2(x(t)) = 0$. As a result, we obtain by applying Lemma 2 to each component g_i of G that

$$\lim_{t \rightarrow +\infty} (G(y(t)) - G(y^*)) = 0 . \tag{11}$$

Step 4) We finally prove $\lim_{t \rightarrow +\infty} x(t) = x^*$.

By the differential equation theory, $x(t)$ solves the following integral equation:

$$x(t) - x^* = e^{-\frac{1}{\tau}(t-t_0)I} (x_0 - x^*) + \int_{t_0}^t e^{-\frac{1}{\tau}(t-s)I} \cdot \frac{1}{\tau} \cdot (G(y(s)) - G(y^*)) ds .$$

Obviously, it holds that

$$\|x(t) - x^*\| \leq e^{-\frac{1}{\tau}(t-t_0)} \|x_0 - x^*\| + \int_{t_0}^t e^{-\frac{1}{\tau}(t-s)} \cdot \frac{1}{\tau} \cdot \|G(y(s)) - G(y^*)\| ds . \tag{12}$$

By (10), for any $\varepsilon > 0$, there is a $T_\varepsilon > 0$ such that, whenever $t > t_0 \geq T_\varepsilon$,

$$\|G(y(t)) - G(y^*)\| \leq \varepsilon.$$

Therefore, we conclude from (12) that, when $t > t_0 \geq T_\varepsilon$,

$$\begin{aligned} \|x(t) - x^*\| &\leq e^{-\frac{1}{\tau}(t-t_0)} \|x_0 - x^*\| + \frac{\varepsilon}{\tau} \int_{t_0}^t e^{-\frac{1}{\tau}(t-s)} ds \\ &< e^{-\frac{1}{\tau}(t-t_0)} \|x_0 - x^*\| + \varepsilon. \end{aligned}$$

Letting $t \rightarrow +\infty$ in the above inequality yields $\lim_{t \rightarrow +\infty} \|x(t) - x^*\| \leq \varepsilon$, which implies $\lim_{t \rightarrow +\infty} x(t) = x^*$ since ε is arbitrary. This completes the proof of the theorem.

Applying Theorem 2 to the RNNs with nearest point projection, we obtain the following results.

Corollary 1. (Global convergence of model (4) with nearest point projection). Assume that $G = P_\Omega = (g_1(x_1), g_2(x_2), \dots, g_N(x_N))^T$ with each g_i being an 1-D nearest point projection and $\Omega \subset \mathcal{R}^N$ being a bounded, closed and convex set. If there is a positive diagonal matrix $\Gamma = \text{diag}\{\xi_1, \xi_2, \dots, \xi_N\}$ such that

- (i) $M(L, \Gamma) = \Gamma - \frac{\Gamma W + W^T \Gamma}{2}$ is nonnegative definite and
- (ii) $x^T(t) \Gamma x(t) - P_\Omega^T(Wx(t) + q) \Gamma P_\Omega(Wx(t) + q) \geq 0$,

then RNN model (4) with nearest point projection is globally convergent on Ω . That is, for any trajectory $x(t)$ of (4) starting from Ω , there corresponds an equilibrium state x^* of (4) such that $\lim_{t \rightarrow +\infty} x(t) = x^*$.

Proof. It has been proved in [7] that the nearest point projection $G : \mathcal{R}^N \rightarrow \Omega$ satisfy the inequality

$$\langle P_\Omega(x) - P_\Omega(y), x - y \rangle \geq \|P_\Omega(x) - P_\Omega(y)\|^2, \quad \forall x, y \in \mathcal{R}^N.$$

It is clear that each $L_i = 1$ and g_i is monotonically increasing and continuous when g_i is an 1-D nearest point projection. Since $\langle \Gamma(z(t) + x(t)), x(t) - z(t) \rangle = x^T(t) \Gamma x(t) - z^T(t) \Gamma z(t)$, where $z(t) = G(Wx(t) + q)$, we can easily get the result of the Corollary.

Remark 1. Comparing with Theorem 1, Theorem 2 not only eliminates the request on W that ΓW must be symmetric, but also applies to a generally diagonal projection other than diagonal nearest point projection. Corollary 1 is new for system (1) with nearest point projection. An example will present in the next section to show the availability of the convergent results obtained.

4 An Example

Example 1: Consider the neural networks defined by

$$\begin{cases} \frac{dx_1(t)}{dt} = -x_1(t) + g_1(x_1(t) + 2x_2(t) - 2) \\ \frac{dx_2(t)}{dt} = -x_2(t) + g_2(-3x_1(t) + x_2(t)) \end{cases} \quad (13)$$

where $g_i(s)$ is defined as follows:

$$g_i(s) = \begin{cases} 1, & s > 1 \\ s, & s \in [0, 1] \\ 0, & s < 0 \end{cases} \tag{14}$$

In this example, $W = \begin{pmatrix} 1 & 2 \\ -3 & 1 \end{pmatrix}$, $q = \begin{pmatrix} -2 \\ 0 \end{pmatrix}$ and the equilibrium set of (13) is

$$\Omega_e = \{(0, s)^T \mid \forall s \in [0, 1]\} .$$

It is easy to verify that each $L_i = 1$. Note first that for any positive diagonal matrix Γ , ΓW is not symmetric, so Theorem 1 can not be applied here. We now show the convergent results established in this paper will work well.

By setting $\Gamma = \text{diag}(3, 2)$, we get that $M(L, \Gamma) = L^{-1}\Gamma - \frac{\Gamma W + W^T \Gamma}{2}$ is nonnegative definite. Meanwhile, it is easy to verify that for all trajectory $x(t) = (x_1(t), x_2(t))^T$ of (13) starting from $x_0 \in \Omega$ (here $\Omega = [0, 1] \times [0, 1]$),

$$\begin{cases} x_1(t) \geq x_1(t) + 2x_2(t) - 2 \\ x_2(t) \geq -3x_1(t) + x_2(t), \end{cases} \tag{15}$$

so $x^T(t)\Gamma x(t) = G^T(x(t))\Gamma G(x(t)) \geq G^T(Wx(t) + q)\Gamma G(Wx(t) + q)$ since each $g_i(s)$ is monotonically increasing and nonnegative. Thus, by Corollary 1, system (13) is globally convergence on Ω .

5 Conclusion

In this paper, the global convergence of RNNs with general projection mappings has been studied under the critical condition that $M(L, \Gamma)$ is nonnegative definite for any positive definite diagonal matrix Γ . The obtained results of RNN model (11) exploited new globally convergent analysis. An example has been presented to demonstrate both theoretical importance and practical significance of the critical results obtained.

Acknowledgment. This research was supported by the National Nature Science Foundation of China under contract Nos.10371097 and 70531030.

References

1. Peng, J., Xu, Z.B., Qiao, H., Zhang, B.: A Critical Analysis on Global Convergence of Hopfield-type Neural Networks. *IEEE Trans. Circuits Syst.* **52** (2005) 804-814
2. Forti, M., Tesi, A.: New Conditions for Global Stability of Neural Networks with Applications to Linear and Quadratic Programming Problems. *IEEE Trans. Circuits Syst.* **42** (1995) 354-366

3. Xia, Y.S., Wang, J.: On the Stability of Globally Projected Dynamical Systems. *J. Optim. Theory Appl.* **106** (2000) 129-150
4. Zhang, Y.P., Heng, A., Fu, A.W.C.: Estimate of Exponential Convergence Rate and Exponential Stability for Neural Networks. *IEEE Trans. Neural Networks* **10** (1999) 1487-1493
5. Liang, X.B., Si, J.: Global Exponential Stability of Neural Networks with Globally Lipschitz Continuous Activation and Its Application to Linear Variational Inequality Problem. *IEEE Trans. Neural Networks* **12** (2001) 349-359
6. Chen, T.P., Amari, S.: New Theorems on Global Convergence of Some Dynamical Systems. *Neural Networks* **14** (2001) 251-255
7. Qiao, H., Peng, J., Xu, Z.B., Zhang, B.: A Reference Model Approach to Stability Analysis of Neural Networks. *IEEE Trans. Syst., Man, Cybern. B* **33** (2003) 925-936
8. Slotine, J.J.E., Li, W.: *Applied Nonlinear Control*. Englewood Cliffs, NJ: Prentice-Hall (1991)

A Study on Digital Media Security by Hopfield Neural Network*

Minseong Ju¹, Seoksoo Kim¹, and Tai-hoon Kim²

Hannam University, Department of Multimedia Engineering, Postfach, 306 791
133 Ojeong-Dong, Daedeok-Gu, Daejeon, South Korea
nimpe2@naver.com

Ewha Womans University, Postfach, 120 750
Dai Heoyon-Dong, Seodamun-Gu, Seoul, South Korea
taihoonn@paran.com

Abstract. Recently, the distribution and using of the digital multimedia contents are easy by developing the internet application program and related technology. However, the digital signal is easily duplicated and the duplicates have the same quality compare with original digital signal. To solve this problem, there is the multimedia fingerprint which is studied for the protection of copyright. Fingerprinting scheme is a technique which supports copyright protection to track redistributors of electronic information using cryptographic techniques. Only regular user can know the inserted fingerprint data in fingerprinting schemes differ from a symmetric/asymmetric scheme and the scheme guarantee an anonymous before re-contributed data. In this paper, we present a new scheme which is the detection of colluded multimedia fingerprint by neural network. This proposed scheme is consists of the anti-collusion code generation and the neural network for the error correction. Anti-collusion code based on BIBD(Balanced Incomplete Block Design) was made 100% collusion code detection rate about the average linear collusion attack, and the Hopfield neural network using (n,k) code designing for the error bits correction confirmed that can correct error within 2bits.

1 Introduction

The development of Internet has made the sharing and distribution of information possible all over the world. As a result, digital media has penetrated into our lives over years so that the uses of various digital contents, such as images, videos and audios have increased rapidly. But digital contents are easy to duplicate and the duplicated digital contents are difficult to distinguish from originals, so illegal duplication and distribution are also prevalent. It results in the economic losses of digital creative workers. Therefore, there has been growing needs for the reliable and effective method of digital media protection.

* This work was supported by a grant from Security Engineering Research Center of Ministry of Commerce, Industry and Energy.

To counter above, digital contents protection technology was developed. The digital contents protection technology can be defined as a technology that imbed the information related to the copyright of contents producer, invulnerably to the attack from outside, and classified roughly into watermarking technology and fingerprinting technology. Watermarking technology transforms the information related to the copyright of contents producer into watermarking and inserts them into contents invisibly. This technology can certificates the ownership of digital contents producer but cannot trace the attackers when the inserted watermarks are damaged and destroyed by many attacks. Briefly, its fault is the incompetence to find out the illegal distribution process of digital contents. For this, multimedia fingerprinting technology is being studied. Fingerprinting technology is a measure to protect the intellectual property rights of original producer and to prevent the illegal duplication and distribution of digital brainchild. This content protection technology imbeds the information about users into contents so that it can trace and extract the attack colluders when some collusive attack was occurred to duplicate the content. It was originated from the transactional watermarking [1] that was used to protect algorithm table from illegal duplication. Digital fingerprinting technology is divided into two techniques. One is the dual watermarking/fingerprinting technology that is proposed by Malvar [2] and others. The other is the collusion secure code techniques [3-6] that designs the inserted codes to be free from collusive attack.

Dual watermarking/fingerprinting technique, which is realized in the media player platform of Micro-soft, uses watermarking module to protect copyrights and fingerprinting module to identify original buyer's information. Collusion secure code techniques, as a code that designed fingerprinting code difficult to be colluded, is invulnerable to the collusive attack and includes c-secure (proposed by Bonehand Shaw), c-frame proof code [3], d-detecting code [4] (proposed by Dittmann), 3-secure code [5,6] (pro-posed by Domingo-Ferrer) and Anti- Collusion code (proposed by Trappe), etc.

These fingerprinting techniques would have different fingerprinting codes from each user. So, if some-one would take advantage of such character and compare many contents, he could analogize fingerprinting information and prepare collusive attack. Major collusive attacks include Averaging Attack, Max-Min Attack, Negative-Correlation Attack, Zero-Correlation Attack and Mosaic Attack.

The algorithm proposed in this study designed BIBD (ACC: Anti-Collusion Codes) based code, used it as fingerprint, designed Hopfield neural network with the manner of feedback-type associative memory and extract accurately the colluded fingerprints and users. Also, this study analyzed the dispersion of BIBD code and measured the tenacity against the illegal collusive attack of proposed algorithm and error correcting capacity. For this, chapter II explains the theoretical background of Hopfield model for BIBD code and error correcting, chapter III describes the algorithm proposed in this paper for the illegal collusive code extraction and error correcting, chapter IV reviews the measurement of proposed algorithm's capacity and its result, and final Chapter V addresses the conclusion and the course of future study.

2 Related Works

Collusive attackers would perform Averaging Attack, Max-Min Attack, Negative-Correlation Attack, Zero-Correlation Attack and Mosaic Attack on contents to increase the vagueness of inserted identification information or, the information of fingerprints' removals and extractions, ultimately incapacitating all tracing against collusive attackers. Figure 1 indicates the basic methods of collusive attack.

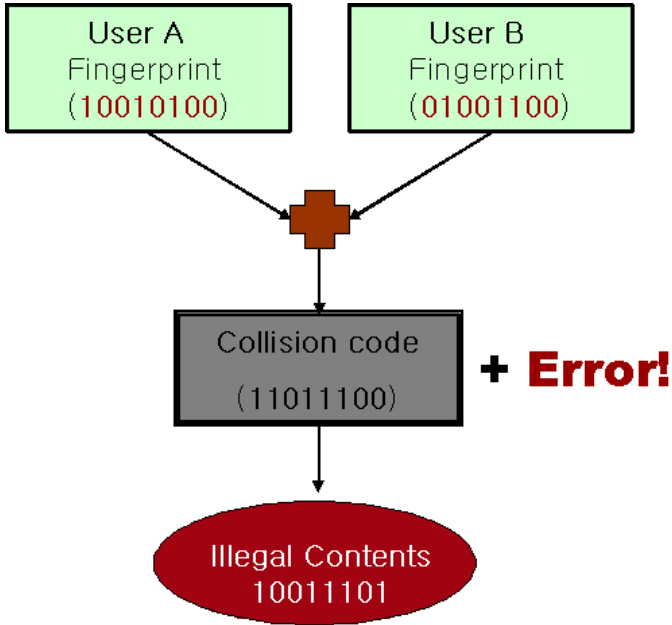


Fig. 1. Basic methods of collusive attack

2.1 BIBD Code

For combination, matrix model can create a matrix meeting the conditions. As BIBD code creates incidence matrix that meets the conditions of anti-collision code, the symmetry of matrix can be partly broke down. This code is an anti-collision code invulnerable to collusive attack and all the combinations of less than (n-1) code vectors from n code vectors have different combinations each other, so it can extract colluders of less than (n-1).

2.2 Hopfield Network

Hopfield network is a mutual combinative nerve network model. It assumes that the operation of neuron is only the operation of critical values therefore the data by training would be expressed by the intensity of connections. So, it proposed the associative memory system and applied it to solve the optimization problems.

Hopfield network can secure global optimization through many numbers of asynchronous and local calculations and store certain general-purpose patterns then seek out the most similar pattern when given unknown input pattern, especially in associative memory. In Hopfield network, as a circuit network inter-actively associated with all units other than each oneself, like figure 2, the existing line becomes input pattern and y line becomes out pattern in which circuit network converges. Hopfield network system can recollect many related parts from small amount of information like human memory system. As it finds the information stored in w by the data input into X , it is called as CAM (content addressable memory) or associative memory.

As shown in figure 2, Hopfield network as a circuit network interactively associated with all units (neuron, $w_0, w_1, w_2, \dots, w_{n-1}$) other than each oneself, $x_0, x_1, x_2, \dots,$ and x_{n-1} are input patterns and $y_0, y_1, y_2, \dots,$ and y_{n-1} are output patterns in which circuit network is converging.

Input vector is input to x and output y is feed backed to all y units, then the output of each unit is determined as below formula [7].

As the content memorized in unit can make global optimization with similar vector having error, it performs error correction function.

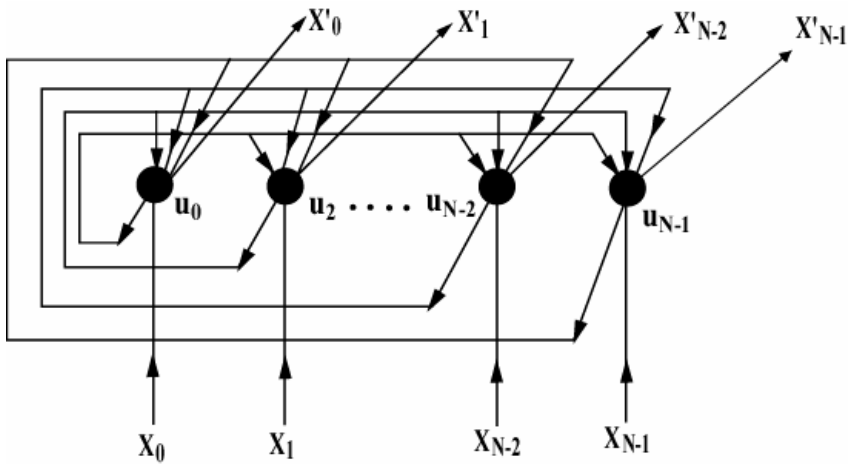


Fig. 2. Basic Structure of Hopfield Network

3 The Multimedia Fingerprint Detection

This study used BIBD based ACC, which is invulnerable to collusive attack and Hopfield error correcting circuit to give tenacity against external noise attack. The algorithm that can extract illegal colluders using the neural circuit network proposed in this paper is like figure 3. The proposed algorithm is expanded with based on (n,k) code in order to enhance the reliability of fingerprint created by BIBD basis. That is, the created fingerprint id expanded to maintain the peculiarities of each unit against

external attacks when establishing Hopfield neural network for error correcting. The (n,k) code language is expressed like formula (9).

$$C(x) = D(x) \cdot G(x). \quad (9)$$

Here, $C(x)$ is an expanded polynomial expression less than $n-1$ degrees, $D(x)$, fingerprint code, is an information polynomial expression less than $k-1$ degrees, and $G(x)$ is a check bit added with considering minimum Hamming distance.

$$\text{Check_bit} = 2 \cdot \text{error_bit} + 1 \quad (10)$$

In the error correcting block of proposed algorithm, when an error is added by some causes to fingerprint $C(x)$ that is created by formula (9), the feedback-type associative memory of Hopfield model will correct the error, calculate $D(x)$ and extract colluders finally by code book reference.

Hopfield neural network error correcting circuit designed by this paper. It can determine the illegal collusion by correcting two bits from the 12 bit of fingerprint code. Overall circuit was embodied in the MOSFET of N and P type. As the state of MOSFET is controlled to the excited and restrained states according to the changes of Channel-widths, Channel-widths and gate connecting input values of MOSFET, the error of input data will be corrected.

4 Error Correction Circuit Using Neural Network

To measure the performance of proposed algorithm, simulation was set by Matlab and IBM PC of Intel Pentium IV 3.0GHz CPU & 4.0GB RAM. This paper created the code that had the condition in which ACC creation parameter $\{v,k,\lambda\}$ is $\{7,3,1\}$, $\{15,7,3\}$, $\{23,11,5\}$, $\{31,15,7\}$ then carried out test. Table 2 indicates the number of combinable cases, in collusive attack, according to the number of colluders. This paper set the number of colluders to 6 and performed the colluder extraction test. Test was performed on the tenacity against the Averaging Attack and bit error correction transformed by Gaussian noise attack on the fingerprint attacked collusively.

4.1 Tenacity Against the Collusive Averaging Attack

Table 5 explains the process in which two collusive attackers from seven users are distinguished by $\{7,3,1\}$ BIBD code. When the correlation coefficient evaluated between colluded code and codebook is higher than critical value, it shall be assumed as colluder. In figure 5, as the correlation coefficient between a colluded code, b_1 and b_6 is higher than critical value, it is a colluded code. The correlation coefficient was evaluated by formula (12).

Figure 6 indicates the correlation coefficient between colluded code and codebook. If correlation coefficient(r) was higher than 0, it was determined to be a colluder. If, lower higher than 0, not a colluder.

Figure 3 is the result of extraction of Averaging Attack colluders using the algorithm proposed by this paper. The algorithm extracted 100% of colluders for Collusive Averaging Attack only.

4.2 Bit Error Correction Using Neural Circuit

To avoid tracing, colluders could attack the colluded code with noise and intentional bit invention. This paper arranged 2bit error correction using Hopfield neural circuit to have the tenacity against collusive attack. To measure the performance of such neural circuit, this study added error correction bit of 5bit to the created fingerprint code, for 2bit error correction, by formula (10).

The error correction rate of neural circuit that was designed when AWGN was higher than 12dB showed 100%. That means that there can be less than 2 error bit in a code. Table 4 is the result of colluder extraction according to the change of AWGN, after the creation of 1,000 codes from each code. AWGN can extract the number of colluders exactly up to 12dB, but under 10dB, the number of colluders decreases in proportion to code's length.

In conclusion, the fingerprint extraction algorithm using the neural circuit of this paper can extract 100% of Averaging Attack colluders by the code based on BIBD and extract colluders of the bit transformation attack using Hopfield neural circuit, within 2bit code change.

5 Conclusion

The This study designed BIBD based illegal collusion preventive code invulnerable to collusive attacks to protect the copyright of digital contents from illegal duplications and collusive attacks. Fingerprint information can be damaged by external attack and noise during transmitting, so, this study proposed finger-print algorithm that can correct the damaged code using Hopfield neural circuit. The proposed algorithm consists of BIBD based illegal collusion preventive code invulnerable to linear collusive attacks and Hop-field neural circuit of feedback-type associative memory to correct the code damaged by external attack. As a result of test, BIBD based illegal collusion preventive code could extract 100% of collusive codes in linear collusive Averaging Attack and Hopfield neural circuit using (n,k) code can correct the error bit less than 2 bit. In conclusion, the proposed algorithm can extract colluders exactly when Averaging At-tack is occurred and less than 2 error bit is created in collusive code. In the future, the study should be progressed on how to develop an effective algorithm to insert the proposed multimedia fingerprint algorithm into real multimedia and how to get tenacity against the nonlinear collusive attack such as Zero-Correlation Attack.

Acknowledgment. This work was supported by a grant from Security Engineering Research Center of Ministry of Commerce, Industry and Energy.

References

1. Dittmann, J.: Combining Digital Watermarks and Collusion Secure Fingerprints for Customer Copy Monitoring. Proc. IEE Seminar Sec. Image &Image Auth. (2000) 128-132
2. Trappe, W., Wu, M., Wang, Z.J., Liu, K.J.R.: Anti-Collusion Fingerprinting for Multimedia. IEEE Trans. Signal Processing **51** (2003) 1069-1087

3. Ingemar, J.C., Miller, M.L., Bloom, J.A.: Watermarking Applications and Their Properties. International Conference Information technology'2000, Las Vegas (2000)
4. Kirovski, D., Malvar, H.S., Yacobi, Y.: Multimedia Content Screening using a Dual Watermarking and Fingerprinting System. ACM Multimedia (2002)
5. Boneh, D., Shaw, J.: Collusion-Secure Fingerprinting for Digital Data. IEEE Trans. Inf. Theory. **44** (1998) 1897-1905
6. Sebe, F., Ferrer, J.D.: Short 3-Secure Fingerprinting Codes for Copyright Protection. Lecture Notes in Computer Science **2384** (2002) 316- 327
7. Hopfield, J.J.: Artificial Neural Network. IEEE Circuits and Device Magazine (1986) 3-9

Two Theorems on the Robust Designs for Pattern Matching CNNs*

Bing Zhao, Weidong Li, Shu Jian, and Lequan Min

Applied Science School
University of Science and Technology Beijing
Beijing 100083, PR China
{wingardium,bjliweidong,jianshumvp}@126.com,
minlequan@sina.com

Abstract. The cellular neural/nonlinear network (CNN) has become a useful tool for image and signal processing, robotic and biological visions, and higher brain functions. Based on our previous research, this paper set up two new theorems of robust designs for Pattern Matching CNN in processing binary images, which provide parameter inequalities to determine parameter intervals for implementing the prescribed image processing function. Three numerical simulation examples are given.

1 Introduction

The CNN, first introduced by Chua & Yang ([1], [2]) as an implementable alternative to fully-connected Hopfield neural networks, is a large scale nonlinear circuits composed of locally connected cells. This property has allowed CNN theory models to be made of CNN universal chips whose theoretical computation speed can be at least a thousand times faster than the current digital processor. Now the CNN has been widely studied for theoretical foundations and practical applications in image and video signal processing, robotic and biological visions, and higher brain functions (see [3]-[11]).

Robust design is an important issue for the study in CNN. Practically, an engineer always hopes to design such a CNN that is able not only to perform its prescribed task for the “nominal (idea) model” but also to work well for a large set of perturbed models.

In [3] and [10], the robust analysis of a large kind of CNNs–uncoupled Boolean CNNs has been discussed, which provides optimal design schemes for CNNs with prescribed tasks. Recently, three uncoupled CNNs with linear and nonlinear \mathbf{B} -templates are introduced, respectively, based on robust parameter designs, which are able to detect edges, convex corners or counter of object in some gray-scale images ([12]- [14]). In our previous paper [15], two theorems are set up to design robust templates for a kind of uncoupled CNNs.

* This project is jointly supported by the National Natural Science Foundations of China (Grant Nos. 60674059, 70271068), the Research Fund for the Doctoral Program of Higher Education (Grant No. 20020008004) by the Ministry of Education of China.

In the CNN Library [16], many interesting CNNs are introduced. One of them is the Pattern Matching CNN for processing binary images. This paper first shows the Local Rules of Pattern Matching CNN, and then studies the issue for designing robust Pattern Matching CNNs. Two corresponding robustness theorems are set up. The theorems provide the parameter inequalities for determining parameter intervals to guarantee the Pattern Matching CNNs to perform prescribed functions. Three numerical simulation examples are given to verify the new theorems to be efficient in practical applications for computer digital image processing.

2 A Theorem on Pattern Matching CNN

The CNN architecture used in this paper is composed of a two-dimensional M by N array of cells. Each cell is denoted by $C(i, j)$ where $i = 1, 2, \dots, M; j = 1, 2, \dots, N$. The dynamics of each cell is given by the equation [3]

$$\begin{aligned} \dot{x}_{i,j} = & -x_{i,j} + \sum_{k=-1}^1 \sum_{l=-1}^1 a_{k,l} y_{i+k,j+l} \\ & + \sum_{k=-1}^1 \sum_{l=-1}^1 b_{k,l} u_{i+k,j+l} + z \end{aligned} \tag{1}$$

where $u_{i,j}, x_{i,j}$, and $y_{i,j}$ are the input, state, and output variables of the cell; $a_{i,j}$'s, $b_{k,l}$'s and z are the elements of the \mathbf{A} -template and the \mathbf{B} -template, and threshold, respectively. The output $y_{i,j}$ is the piece-wise linear function given by

$$y_{i+k,j+l} = \frac{1}{2}(|x_{i+k,j+l} + 1| - |x_{i+k,j+l} - 1|).$$

The standard Pattern Matching template has the following form

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} b_{-1,-1} & b_{-1,0} & b_{-1,1} \\ b_{0,-1} & b_{0,0} & b_{0,1} \\ b_{1,-1} & b_{1,0} & b_{1,1} \end{bmatrix}, \quad z = 0.5 - N. \tag{2}$$

where

$$b_{k,l} = \begin{cases} 1, & \text{if the corresponding pixel is required to be black} \\ 0, & \text{if the corresponding pixel is not cared} \\ -1, & \text{if the corresponding pixel is required to be white} \end{cases} \tag{3}$$

N = the number of pixels required to be either black or white, i.e. the number of non-zero values in the B template.

I. Global Task

Given : *static binary image \mathbf{P} possessing the 3×3 pattern prescribed by the template.*

Input : $\mathbf{U}(t) = \mathbf{P}$.

Initial State : $\mathbf{X}(0) = \text{Arbitrary}$.

Boundary Conditions : $[U] = 0$.

Output : $\mathbf{Y}(t) \Rightarrow \mathbf{Y}(\infty) = \text{Binary}$
image representing the locations of the 3×3 pattern prescribed by the template. The pattern having a black/white pixel where the template value is $+1/-1$, respectively, is detected.

II. Local Rules

- $u_{i,j}(0) \rightarrow y_{i,j}(\infty)$
1. arbitrary (-1 or 1) \rightarrow black, if $u_{i+k,j+l}(0) = \text{sgn}(b_{k,l})$ for all $b_{k,l} \neq 0$
 2. arbitrary (-1 or 1) \rightarrow black, if all $b_{k,l} = 0$
 3. arbitrary (-1 or 1) \rightarrow white, otherwise

III. Mathematical Analysis

State-output EQ. has a form:

$$\dot{x}_{i,j} = -x_{i,j} + y_{i,j} + w_{i,j} \quad (4)$$

$$w_{i,j} = \sum_{k=-1}^1 \sum_{l=-1}^1 b_{k,l} u_{i+k,j+l} - N + 0.5. \quad (5)$$

From Fig. [1](#), we can conclude that:

$$y_{i,j}(\infty) = \begin{cases} 1 & \text{if } w_{i,j} > 0 \\ -1 & \text{if } w_{i,j} < 0 \end{cases} \quad (6)$$

Case 1. If $u_{i+k,j+l}(0) = \text{sgn}((b_{k,l}))$ for all $b_{k,l} \neq 0$, then

$$\begin{aligned} w_{i,j} &= N - N + 0.5 \\ &= 0.5 > 0. \end{aligned} \quad (7)$$

From [\(6\)](#) and [\(7\)](#), we conclude that $y_{i,j}(\infty) = 1$.

Case 2. If all $b_{k,l} = 0$, then

$$\begin{aligned} w_{i,j} &= 0 - 0 + 0.5 \\ &= 0.5 > 0. \end{aligned} \quad (8)$$

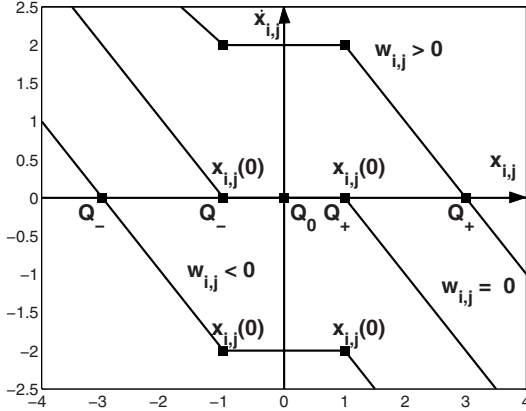


Fig. 1. The dynamic route of $(\dot{x}_{i,j} - x_{i,j})$

From (6) and (8), we conclude that $y_{i,j}(\infty) = 1$.

Case 3. If there exists at least one $u_{i+k,j+l}(0) = -sgn(b_{k,l})$ for all $b_{k,l} \neq 0$, then

$$w_{i,j} \leq (N - 1) - 1 + N - 0.5 = -1.5 < 0. \tag{9}$$

From (6) and (9), we conclude that $y_{i,j}(\infty) = -1$. In summary, we complete the proof. ■

3 Robust Design of Pattern Matching CNNs

Assume that the robust CNN template has the following form:

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & a & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} b_{-1,-1} & b_{-1,0} & b_{-1,1} \\ b_{0,-1} & b_{0,0} & b_{0,1} \\ b_{1,-1} & b_{1,0} & b_{1,1} \end{bmatrix}, \quad z = c - Nb \tag{10}$$

where

$$b_{k,l} = \begin{cases} b, & \text{if the corresponding pixel is required to be black} \\ 0, & \text{if the corresponding pixel is not cared} \\ -b, & \text{if the corresponding pixel is required to be white} \end{cases} \tag{11}$$

N = number of pixels required to be either black or white, i.e. the number of non-zero values in the B template.

Theorem 1. If the parameters a, b, c are determined by the following inequations, then the CNN can perform the global task and the local rules of Pattern

Matching CNN

$$\begin{cases} a > 1 \\ b > 0 \\ a - c < 1 \\ a - 2b + c < 1 \end{cases} \quad (12)$$

Proof. State-output EQ. has a form:

$$\dot{x}_{i,j} = -x_{i,j} + ay_{i,j} + w_{i,j} \quad (13)$$

$$w_{i,j} = \sum_{k=-1}^1 \sum_{l=-1}^1 b_{k,l} u_{i+k,j+l} - Nb + c. \quad (14)$$

From Fig. 2, we can conclude that:

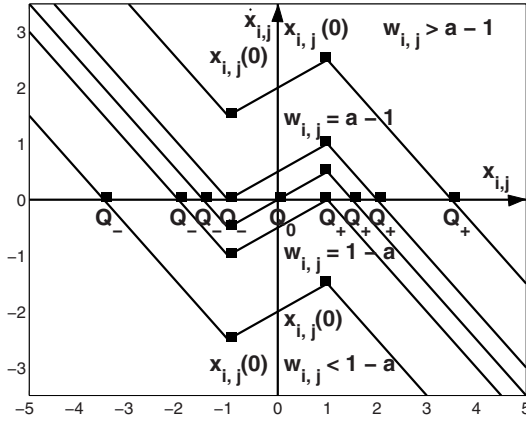


Fig. 2. The dynamic route of $(\dot{x}_{i,j}) - x_{i,j}$

$$y_{i,j}(\infty) = \begin{cases} 1 & \text{if } w_{i,j} > a - 1 \\ -1 & \text{if } w_{i,j} < 1 - a \end{cases} \quad (15)$$

Case 1. If $u_{i+k,j+l}(0) = \text{sgn}(b_{k,l})$ for all $b_{k,l} \neq 0$, then

$$\begin{aligned} w_{i,j} &= Nb - Nb + c \\ &= c > a - 1. \end{aligned} \quad (16)$$

From (15) and (16), we conclude that $y_{i,j}(\infty) = 1$.

Case 2. If all $b_{k,l} = 0$, then

$$\begin{aligned} w_{i,j} &= 0b - 0b + c \\ &= c > a - 1. \end{aligned} \quad (17)$$

From (15) and (17), we conclude that $y_{i,j}(\infty) = 1$.

Case 3. If there exists at least one $u_{i+k,j+l}(0) = -sgn(b_{k,l})$ for all $b_{k,l} \neq 0$, then

$$\begin{aligned} w_{i,j} &\leq (N - 1)b - b - Nb + c \\ &= c - 2b < 1 - a. \end{aligned} \tag{18}$$

From (15) and (18), we conclude that $y_{i,j}(\infty) = -1$. ■

Theorem 2. If the parameters a, b, c are determined by the following inequations, then the CNN can perform the global task and the local rules of Pattern Matching CNN

$$\begin{cases} a < 1 \\ b > 0 \\ a + c > 1 \\ a + 2b - c > 1 \end{cases} \tag{19}$$

Proof. State-output EQ. has a form:

$$\dot{x}_{i,j} = -x_{i,j} + ay_{i,j} + w_{i,j} \tag{20}$$

$$w_{i,j} = \sum_{k=-1}^1 \sum_{l=-1}^1 b_{k,l}u_{i+k,j+l} - Nb + c. \tag{21}$$

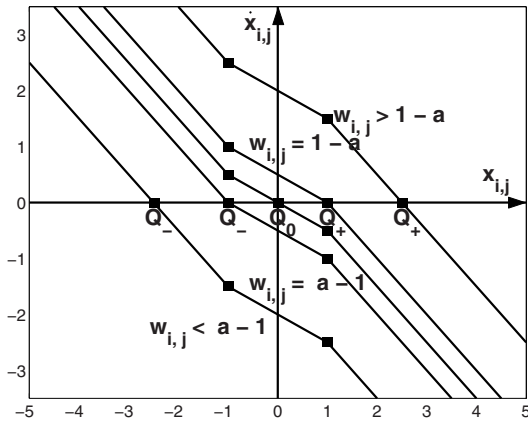


Fig. 3. The dynamic route of $(\dot{x}_{i,j}) - x_{i,j}$

From Fig. 3, we can conclude that:

$$y_{i,j}(\infty) = \begin{cases} 1 & \text{if } w_{i,j} > 1 - a \\ -1 & \text{if } w_{i,j} < a - 1 \end{cases} \tag{22}$$

Case 1. If $u_{i+k,j+l}(0) = \text{sgn}(b_{k,l})$ for all $b_{k,l} \neq 0$, then

$$\begin{aligned} w_{i,j} &= Nb - Nb + c \\ &= c > 1 - a. \end{aligned} \quad (23)$$

From (22) and (23), we conclude that $y_{i,j}(\infty) = 1$.

Case 2. If all $b_{k,l} = 0$, then

$$\begin{aligned} w_{i,j} &= 0b - 0b + c \\ &= c > 1 - a. \end{aligned} \quad (24)$$

From (22) and (24), we conclude that $y_{i,j}(\infty) = 1$.

Case 3. If there exists at least one $u_{i+k,j+l}(0) = -\text{sgn}(b_{k,l})$ for all $b_{k,l} \neq 0$, then

$$\begin{aligned} w_{i,j} &\leq (N-1)b - b - Nb + c \\ &= c - 2b < a - 1. \end{aligned} \quad (25)$$

From (22) and (25), we conclude that $y_{i,j}(\infty) = -1$. ■

4 Numerical Simulation

Now let us consider three CNNs with the parameters given in Table 1.

Table 1. Three CNNs satisfying the standard Pattern Matching CNN or satisfying the conditions in the Theorem 1 or Theorem 2.

No.	a	b	c
1	1	1	0.5
2	2	1.5	2
3	0.5	1.5	2

1. First let us use the parameters numbered by 1 listed in Table 1 to design the CNN as the following form:

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & -1 & 1 \\ 0 & 1 & 0 \\ 1 & -1 & 1 \end{bmatrix}, \quad z = -6.5 \quad (26)$$

Then we use the above CNN to process the image shown in Fig 4(a). The output image is shown in Fig 4(b). Only the 3×3 neighborhoods of the three pixels in Fig 4(a), which correspond to the three black pixels shown in Fig 4(b) match the template \mathbf{B} , respectively. It can be seen that the center of the matched 3×3 pattern in the image shown in Fig 4(a) are detected correctly by the CNN.

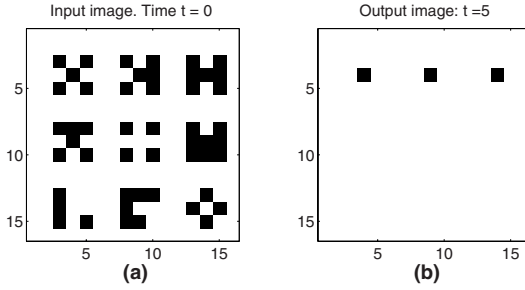


Fig. 4. (a) Input (b) Output

2. Second let us use the parameters numbered by 2 listed in Table 1 to design the CNN as the following form:

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad z = 2 \tag{27}$$

Template (27) is a special case of the Pattern Matching CNNs where $N = 0$. We use it to process the image shown in Fig 5(a). Since all $b_{k,l} = 0$, Local Rules 2 indicates that all the pixels turn to black, as that shown in Fig 5(b).

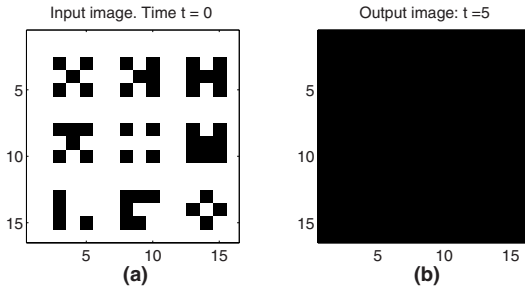


Fig. 5. (a) Input (b) Output

3. Finally let us use the parameters numbered by 3 listed in Table 1 to design the CNN as the following form:

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -1.5 & -1.5 & -1.5 \\ -1.5 & -1.5 & -1.5 \\ -1.5 & -1.5 & -1.5 \end{bmatrix}, \quad z = -11.5 \tag{28}$$

Template (28) is a special case of the Pattern Matching CNNs, where all $b_{k,l} = -b$, and then we use it to process the image shown in Fig 6(a). Only the 3×3 neighborhoods whose pixels are all white in Fig 6(a) match the template B . The output image is shown in Fig 6(b).

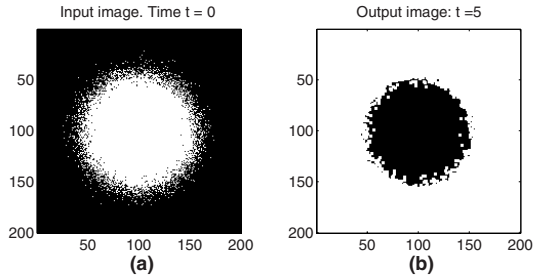


Fig. 6. (a) Input (b) Output

5 Conclusions

In summary, we can obtain the following conclusions.

(1) This paper gives an analytical proof for the performance (the Local Rules) of the standard Pattern Matching CNN [16].

(2) The two robustness design theorems provide general constrain conditions of template parameters for the CNNs, which guarantee the CNNs to implement the corresponding Local Rules.

(3) The two group inequalities given in the two theorems contain three parameters a, b , and c , respectively. It follows that each robust parameter region is surrounded by four planes in space \mathbb{R}^3 . The four planes are determined via the inequalities given by (12) or (19).

(4) Consequently the Pattern Matching CNN template parameters with the most robustness are able to be calculated via algorithms.

Three numerical simulation examples confirm that the theoretical analysis is efficient in practical applications for computer digital image processing.

To our knowledge, the limitation of the image processing of the CNN Universal is 176×144 pixels [17]. Practically, this means that a real image is needed to be divided in 176×144 pixels regions for the CNN image processing. This limits the speed of the CNN image processing.

It is expected that the CNN-UM technology break the 176×144 array barrier to obtain more resolution and expand the action field of the CNN processing. However results obtained here indicate that there are still some uncharted engineering and mathematical territories waiting us to explore and exploit for novel applications of the standard CNN.

References

1. Chua, L.O., Yang, L.: Cellular Neural Networks: Theory. IEEE Trans. Circuits Syst. **35** (1988) 1257-1272
2. Chua, L.O., Yang, L.: Cellular Neural Networks: Applications. IEEE Trans. Circuits Syst. **35** (1988) 1273-1290

3. Chua, L.O.: CNN: A Version of Complexity. *Int. J. Bifurcation and Chaos*. **7** (1997) 2219-2425
4. Chua, L.O., Roska, T.: *Cellular Neural Networks and Visual Computing*, Cambridge: Cambridge University Press (2000)
5. Weblin, F., Roska, T., Chua, L.O.: The Analogic Cellular Neural Network as Bionic Eye. *Int. J. Circuits Theor. Appl.* **23** (1994) 541-569
6. Weblin, F., Jacobs, A., Teeters, J.: The Computational Eye. *IEEE Spectrum*, May. (1996) 30-37
7. Roska, T., Vandewalle, J.: *Cellular Neural Networks*, New York: Wiley (1994)
8. Orzo, L., Vidnyanszky, Z., Hamori, J., Roska, T.: CNN Model of the Feature-linked Synchronized Activities in the Visual Thalamo-cortical System, in *Proc. 1996 Fourth IEEE Int. Workshop on Cellular Neural Networks and Their Application (CNNA-96)*, Seville, Spain. (1996) 291-296
9. Tetzlaff, R., edits, in *Proc. of the 7th IEEE Workshop on Cellular Neural Networks and Their Application*, Singapore: World Scientific (2002)
10. Dogaru, R., Chua, L.O.: Universal CNN Cells. *Int. J. Bifurcation and Chaos*, **9**. (1999) 1-48
11. Hänggi, M., Moschytz, G.S.: Genetic Optimization of Cellular Neural Networks. *Proc. IEEE Int. Conf. Evolutionary Computation*, Anchorage, AK (1988) 381-386
12. Li, G., Min, L., Zang, H.: Design for Robustness Edgegray Detection CNN. *Proc. 2004 Int. Conf. on Communications, Circuits and Systems* **2** (2004) 1061-1065
13. Lei, M., Min, L.: Robustness Design of Templates of CNN for Detecting Inner Corners of Objects in Gray-scale Images. *Proc. 2004 Int. Conf. on Communications, Circuit and Systems* **2** (2004) 1090-1093
14. Zang, H., Li, G., Min, L., et al.: Design for Robustness Counter Detection CNN with Applications. *Proc. 2005 Int. Conf. on Communications, Circuits and Systems* **2** (2005) 953-958
15. Min, L.: Robustness Desins of a Kind of Uncoupled CNNs with Applications. *Proc. of 9th IEEE Int. Workshop on Cellular Neural Networks and Applications* (2005) 98-101
16. CNN Software Library, <http://lab.analogic.sztaki.hu/Candy/csl.html>
17. Paasio, A., Kananen, A., Halonen, K., et al.: A QCIF Resolution Binary I/O CNN-UM Chip. *J. of VLSI Signal Processing* **23 2-3** (1999) 281-290

Iterative Learning Control Analysis Based on Hopfield Neural Networks

Jingli Kang^{1,2} and Wansheng Tang¹

¹ Institute of Systems Engineering, Tianjin University, Tianjin 300072, China
jlkang@eyou.com, tang@tju.edu.cn

² Department of Mathematics, Tianjin University of Finance and Economics,
Tianjin 300222, China

Abstract. Iterative learning control problem based on improved discrete-time Hopfield neural networks is considered in this paper. For the every process of iterative learning control, the neural networks execute a cycle that includes variable terms of learning time and training iterative number. The iterative learning control with improved Hopfield neural networks is formulated that can be described as a two-dimensional (2-D) Roesser model with variable coefficients. In terms of 2-D systems theory, sufficient conditions that iterative learning error approaches to zero are given. It has been shown that convergence of iterative learning control problem based on Hopfield neural networks is derived by 2-D systems theory instead of conventional algorithms that minimize a cost function.

1 Introduction

Iterative learning control has generated considerable interests in theoretical and practical aspects in the past two decades. Iterative learning control systems successfully use information from previous executions to improve the tracking output by iterative process in the sense that the error is sequentially reduced. Most of research has concentrated on providing new algorithms and analyzing their convergence for linear systems and nonlinear systems [1,7,11].

Iterative learning control rules based on 2-D systems theory have been widely introduced recently. In the iterative learning process, dynamics of the control system and the behavior of iterative process can be described by 2-D systems. Kurek and Zaremba [8] presented a fundamental learning control rule and an extended rule for linear discrete-time multivariable systems based on 2-D systems theory. The convergence of learning control rules was robust with respect to small perturbations of the system parameters. It implied that these learning control rules were easy to apply to linear systems. In [4], 2-D systems iterative learning control algorithm for continuous-time systems was proposed. Necessary and sufficient conditions were given for convergence of the proposed learning control rules. Three proposed learning rules were less restrictive and had wider applications in robot manipulators. Li X. D. et. al [9,10] applied 2-D systems theory to deal with iterative learning control problem for linear time-variant discrete systems and linear continuous systems with time delays. In terms of the convergent

property of 2-D linear time-variant discrete systems with only one independent variable, the learning law was presented and its convergence was proved in [9]. Extended 2-D systems theory iterative learning control techniques for linear systems, in [10] it was shown that 2-D linear continuous-discrete Roesser model can be described the iterative learning control process of continuous multivariable systems with time-delays in state and in inputs.

Since neural networks have advantageous to deal with nonlinear systems and complex dynamic systems, some results have been received in the use of neural networks for iterative learning control problem in recent years. For nonlinear systems, the purpose of conventional iterative learning algorithms is to present new iterative learning control laws. The gain matrixes of these learning control laws are variable in every iteration, thus there are some difficulties to obtain gain matrixes or it consume a large time in calculating the exact matrixes. Yamakita et al [12] combined an adaptive control with radial basis function (RBF) neural networks and an adaptive iterative learning control algorithm which can be approximated any continuous function to arbitrary accuracy by linear combination of Gaussian basis function. From the updated weights and the updated center of radial basis functions, the new adaptive iterative learning control algorithm with RBF of artificial neural networks was proposed and was used in a golf-swing robot. In [5], a neural network controller for nonlinear systems trajectory tracking was presented. In contrast to the adaptive neural network control scheme, a series of independent networks was assigned to the points along the desired trajectory. It implied that all local networks can be realized by a simple structure with relatively less neurons. Moreover, it guaranteed that the stability of the trajectory of nonlinear systems.

Feedback networks are better suited for descriptions of learning process in various kinds of neural networks. Chow W. T. S. et al [2,3] presented a newly continuous-time and discrete-time recurrent neural networks (RNN) working together with 2-D training algorithms which is capable of approximating any continuous-time or discrete-time trajectory to a very high degree of accuracy with a few iterations. In their works, two RNN's of the same network architecture were utilized. One RNN was used to approximate the nonlinear system, while another one was used to mimic the desired output. The 2-D systems theory was introduced to analysis the convergence of RNN training iterative learning control algorithms instead of minimizing the cost function of RNN.

Therefore, it is a new and effective way to train iterative learning control algorithm by neural networks and analysis convergence of new algorithm by 2-D systems theory. In this paper, the improved discrete-time Hopfield neural network is applied to iterative learning control. The Hopfield neural network executes a cycle that includes variable terms of learning time and iterative number for every learning iteration. The 2-D Roesser model with variable coefficients that describes iterative learning control based on Hopfield neural network is derived. Using 2-D systems theory, the convergence of iterative learning control problem is analyzed and the sufficient conditions are given.

2 Hopfield Neural Networks and Iterative Learning Control

Consider a general expression of improved discrete-time Hopfield neural networks

$$\begin{aligned} u_k(t+1) &= Hy_k(t) + Bu_k(t) - I, \\ y_k(t) &= S(u_k(t)), \end{aligned} \quad (1)$$

where $u_k(t)$ is the input of the neural network at time t , $y_k(t)$ is the corresponding output of the neural network at time t , I is the external input, H is the connection intensity of neurons, B is an appropriate dimensional coefficient matrix, $S(\cdot)$ denotes a sigmoid function.

In this paper, we consider the iterative learning control problem by improved discrete-time Hopfield neural networks.

Let t denote variable learning time, k denote the training iterative number, the iterative learning control based on improved discrete-time Hopfield neural networks can be rewritten as

$$\begin{aligned} u(t+1, k) &= Hy(t, k) + Bu(t, k) - I, \\ y(t, k) &= S(u(t, k)), \end{aligned} \quad (2)$$

where $u = (u_1, u_2, \dots, u_n)^T \in \mathbf{R}^n$, $y = (y_1, y_2, \dots, y_n)^T \in \mathbf{R}^n$, H , B , I are $n \times n$ matrices.

In every learning iteration, the neural network perform a cycle. If every cycle begins with the invariable nonzero initial condition, we get

$$u(0, k) = u(0), \quad k = 0, 1, 2, \dots \quad (3)$$

Let the learning error

$$e(t, k) = y_d(t) - y(t, k), \quad (4)$$

where $y_d(t)$ is the desired output vector, the iterative learning control problem based on Hopfield neural networks can be stated as follows. For the iterative learning control based on improved discrete-time Hopfield neural networks (2) with initial conditions (3) and desired output $y_d(t)$, it is important to reduce the error $e(t, k)$ step by step. When the iterative number increases, it can be assured that $e(t, k)$ approaches to zero.

3 2-D Systems Theory and Convergence Analysis

3.1 Preliminaries

In order to extend 2-D systems theory to iterative learning control based on Hopfield neural networks, some results are given as preliminaries.

Lemma 1 ([6]). Consider the Roesser model of 2-D linear discrete systems

$$\begin{bmatrix} \eta(t+1, k) \\ e(t, k+1) \end{bmatrix} = \begin{bmatrix} A_1(t, k) & A_2(t, k) \\ A_3(t, k) & A_4(t, k) \end{bmatrix} \begin{bmatrix} \eta(t, k) \\ e(t, k) \end{bmatrix}, \tag{5}$$

where $\eta(t, k) \in \mathbf{R}^{n_1}$, $e(t, k) \in \mathbf{R}^{n_2}$, $A_1(t, k) \in \mathbf{R}^{n_1 \times n_1}$, $A_2(t, k) \in \mathbf{R}^{n_1 \times n_2}$, $A_3(t, k) \in \mathbf{R}^{n_2 \times n_1}$, $A_4(t, k) \in \mathbf{R}^{n_2 \times n_2}$. If boundary conditions for (3) are given by

$$\eta(0, k) = 0, \quad k = 0, 1, 2, \dots \tag{6}$$

and finite $e(t, 0)$ for $t = 0, 1, 2, \dots$, then the solution of (5) with the boundary condition (6) is given by

$$\begin{bmatrix} \eta(t, k) \\ e(t, k) \end{bmatrix} = \sum_{i=0}^t T_{t,k}^{i,j} \begin{bmatrix} 0 \\ e(t-i, 0) \end{bmatrix}, \tag{7}$$

where the state transition matrix $T_{t,k}^{i,j}$ is defined as follows

$$T_{t,k}^{i,j} = \begin{cases} I, & i = j = 0, \\ A_{t-1,k}^{1,0} T_{t-1,k}^{i-1,j} + A_{t,k-1}^{0,1} T_{t,k-1}^{i,j-1}, & i \geq 0, j \geq 0, \\ 0, & i < 0 \text{ or } j < 0 \text{ or } t < 0 \text{ or } k < 0, \end{cases} \tag{8}$$

and

$$A_{t-1,k}^{1,0} = \begin{bmatrix} A_1(t, k) & A_2(t, k) \\ 0 & 0 \end{bmatrix}, \quad A_{t-1,k}^{0,1} = \begin{bmatrix} 0 & 0 \\ A_3(t, k) & A_4(t, k) \end{bmatrix}.$$

Lemma 2 ([2]). Suppose that $Sup_{t,k} \| A_{t,k}^{1,0} \|$ and $Sup_{t,k} \| A_{t,k}^{0,1} \|$ are finite, we have

$$\| T_{t,k}^{i,j} \| \leq 2^{i+j-1} \| A^{0,1} \|^i \| A^{0,1} \|^j,$$

where $\| A \|$ denotes the norm of matrix A , and

$$\| A^{1,0} \| = Sup_{t,k} \| A_{t,k}^{1,0} \|, \quad \| A^{0,1} \| = Sup_{t,k} \| A_{t,k}^{0,1} \|.$$

From lemma 1 and lemma 2, it is easy to see that the following result is held.

Lemma 3 ([2]). For 2-D systems (5) and (6), we have

$$\left\| \begin{bmatrix} \eta(t, k) \\ e(t, k) \end{bmatrix} \right\| \rightarrow 0,$$

for $k \rightarrow 0$ and any given t , if $\| A^{0,1} \| < \frac{1}{2}$.

3.2 Convergence Analysis

In this subsection, we consider the convergence of the iterative learning control problem based on Hopfield neural networks by using 2-D systems theory.

For $t > 0$, from (2), we have

$$\begin{aligned} & u(t + 1, k) - u(t, k) \\ &= (Hy(t, k) + Bu(t, k) + I) - (Hy(t - 1, k) + Bu(t - 1, k) + I) \\ &= H(y(t, k) - y(t - 1, k)) + B(u(t, k) - u(t - 1, k)). \end{aligned}$$

Let

$$\begin{aligned} \eta(t + 1, k) &= u(t + 1, k) - u(t, k), \\ y(t, k) - y(t - 1, k) &= D(t - 1, k)\eta(t, k), \end{aligned}$$

where $D(t - 1, k) = \text{diag}[S'(\xi_1), S'(\xi_2), \dots, S'(\xi_n)]$, $\xi = (\xi_1, \xi_2, \dots, \xi_n)^T$ has a value between $y(t, k)$ and $y(t - 1, k)$, then

$$\eta(t + 1, k) = (HD(t - 1, k) + B)\eta(t, k). \tag{9}$$

Moreover, From (4), we get

$$\begin{aligned} e(t, k + 1) - e(t, k) &= y(t, k) - y(t, k + 1) \\ &= y(t, k) - y(t - 1, k) + y(t - 1, k) - y(t, k + 1) \\ &= D(t - 1, k)\eta(t, k) - (y(t, k + 1) - y(t - 1, k)). \end{aligned}$$

Let

$$y(t, k + 1) - y(t - 1, k) = \Delta M(t - 1, k + 1)x(t - 1, k + 1),$$

where $\Delta M(t - 1, k + 1) = \text{diag}[S'(\phi_1), S'(\phi_2), \dots, S'(\phi_n)]$, $\phi = (\phi_1, \phi_2, \dots, \phi_n)^T$ has a value between $y(t, k + 1)$ and $y(t - 1, k)$, $x(t - 1, k + 1) = u(t, k + 1) - u(t - 1, k)$, then

$$e(t, k + 1) - e(t, k) = D(t - 1, k)\eta(t, k) - \Delta M(t - 1, k + 1)x(t - 1, k + 1). \tag{10}$$

Thus, in terms of the property of 2-D linear time-variant discrete systems, sufficient condition of convergence of iterative learning control problem based on Hopfield neural networks is given as follows.

Theorem 1. *If*

$$\begin{aligned} \Delta M(t, k) &= (M_1(t + 1, k - 1)e(t + 1, k - 1) - M_2(t + 1, k - 1)\eta(t + 1, k - 1)) \\ &\quad (x^T(t, k)x(t, k))^{-1}x^T(t, k), \end{aligned} \tag{11}$$

and $\|A^{1,0}\|$ is boundary, $\|A^{0,1}\| < \frac{1}{2}$, then for every given t and $k \rightarrow \infty$,

$$\|e(t, k)\| \rightarrow 0.$$

Proof. Instituting (11) to (10), we have

$$e(t, k + 1) - e(t, k) = D(t - 1, k)\eta(t, k) - (M_1(t, k)e(t, k) - M_2(t, k)\eta(t, k)). \tag{12}$$

From (9) and (12), we obtain

$$\begin{bmatrix} \eta(t + 1, k) \\ e(t, k + 1) \end{bmatrix} = \begin{bmatrix} HD(t - 1, k) + B & 0 \\ D(t - 1, k) + M_2(t, k) & I - M_1(t, k) \end{bmatrix} \begin{bmatrix} \eta(t, k) \\ e(t, k) \end{bmatrix}. \tag{13}$$

It is a Roesser model of 2-D systems with variable coefficients. In terms of Lemma 1, we get

$$A_{t,k}^{1,0} = \begin{bmatrix} HD(t - 1, k) + B & 0 \\ 0 & 0 \end{bmatrix},$$

$$A_{t,k}^{0,1} = \begin{bmatrix} 0 & 0 \\ D(t - 1, k) + M_2(t, k) & I - M_1(t, k) \end{bmatrix}.$$

From (3) and (4), it is obtained that $\eta(1, k) = 0, k = 0, 1, 2, \dots$ and $e(t, 0)$ ($t = 1, 2, \dots, T$) is boundary. If $\| A^{1,0} \|$ is boundary and $\| A^{0,1} \| < \frac{1}{2}$, from lemma 1, lemma 2 and lemma 3, this theorem is held.

From theorem 1, it is easy to see that the following result is held.

Corollary 1. *If $\Delta M(t, k)$ satisfies (17), and*

$$M_2(t, k) = -D(t - 1, k),$$

$$\| M_1(t, k) \| < \frac{1}{2},$$

then for every given t and $k \rightarrow \infty$,

$$\| e(t, k) \| \rightarrow 0.$$

4 Conclusion

In this paper, iterative learning control problem based on improved discrete-time Hopfield neural networks is discussed. Hopfield neural network is a feedback network that can be better reflect learning process of dynamic systems. Applied improved discrete-time Hopfield neural networks to iterative learning control problem, learning process can be described as a 2-D Roesser model with variable coefficients. In terms of 2-D systems theory, sufficient conditions of convergence of iterative learning control problem based on Hopfield neural networks are derived.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 70471049 and China Postdoctoral Science Foundation under Grant 20060400190.

References

1. Chew, W., Choi, C.H., Ahn, H.S.: Technical Note on Iterative Learning Control for Constrained Nonlinear Systems. *International Journal of Systems Science* **30** (1999) 659-664
2. Chow, T.W.S., Fang, Y.: A Recurrent Neural-Network-Based Real-Time Learning Control Strategy Applying to Nonlinear Systems with Unknown Dynamics. *IEEE Transactions on Industrial Electronics* **45** (1998) 151-161
3. Chow, T.W.S., Li, X.D.: A Real-Time Learning Control Approach for Nonlinear Continuous-Time System Using Recurrent Neural Network. *IEEE Transactions on Industrial Electronics* **47** (2000) 478-486
4. Chow, T.W.S., Fang, Y.: An Iterative Learning Control Method for Continuous-Time Systems Based on 2-D System Theory. *IEEE Transactions on Circuits and Systems—I Fundamental Theory and Applications* **45** (1998) 683-689
5. Jiang, P., Unbehauen, R.: Iterative Learning Neural Network Control for Nonlinear Systems Trajectory Tracking. *Neucomputing* **48** (2002) 141-153
6. Kaczorek, T., Klamka, J.: Minimum Energy Control of 2-D Linear Systems with Variable Coefficients. *International Journal of Control* **44** (1986) 645-650
7. Kang, J.L., Tang, W.S., Mao, Y.Y.: A New Iterative Learning Control Algorithm for Output Tracking of Nonlinear Systems. *Proceedings of the Forth International Conference on Machine Learning and Cybernetics* **3** (2005) 1240-1243
8. Kurek, J.E., Zaremba, M.B.: Iterative Learning Control Synthesis Based on 2-D System Theory. *IEEE Transactions on Automatic Control* **38** (1993) 121-125
9. Li, X.D., Ho, J.L.K., Chow, T.W.S.: Iterative Learning Control for Linear Time-Variant Discrete Systems Based on 2-D System Theory. *IEEE Proceedings of Control Theory and Applications* **152** (2005) 13-18
10. Li, X.D., Chow, T.W.S.: 2-D System Theory Based on Iterative Learning Control for Linear Continuous Systems with Time Delays. *IEEE Transactions on Circuits and Systems—I Regular Papers* **52** (2005) 1421-1430
11. Xu, J.X., Tian, Y.: *Linear and Nonlinear Iterative Learning Control*. Springer-Verlag, Berlin Heidelberg New York (2003)
12. Yamakita, M., Ueno, M., Sadahiro, T.: Trajectory Tracking Control by an Adaptive Iterative Learning Control with Artificial Neural Network. *Proceedings of the American Control Conference* **48** (2001) 1253-1255

Stochastic Stabilization of Delayed Neural Networks

Wudai Liao¹, Jinhuan Chen¹, Yulin Xu², and Xiaoxin Liao³

¹ School of Electrical and Information Engineering, Zhongyuan University of Technology, 450007, Zhengzhou, Henan, China
{wdliao, jhchen}@zzti.edu.cn

² School of Mechatronics and Automation, Shanghai University, Shanghai 200072
xuyulin@shu.edu.cn

³ Department of Control Science and Engineering, Huazhong University of Science and Technology, 430074, Wuhan, Hubei, China
xiaoxin-liao@hotmail.com

Abstract. By introducing appropriate stochastic factors into the neural networks, there were results showing that the neural networks can be stabilized. In this paper, stochastic stabilization of delayed neural networks is studied. First, a new type Razumikhin-type theorem about stochastic functional differential equations is proposed and the rigid proof is given by using Itô formula, Borel-Contelli lemma etc.. As a corollary of the theorem, a new type Razumikhin-type theorem of delayed stochastic differential equation is obtained. Next, taking the results obtained in the first section as the theoretic basis, the stabilization of the delayed deterministic neural networks is examined. The result obtained in the paper shows that the neural networks can be stabilized so long as the intensity of the random perturbation is large enough. The expression of the random intensity is presented which is convenient to networks' design.

1 Introduction

Razumikhin-type theorem [2] plays a very important role in the stability analysis of deterministic functional differential equations as follows

$$\frac{dx(t)}{dt} = f(x_t, t), x_{t_0}(s) = \xi(s) \in C, -\tau \leq s \leq 0, \quad (1)$$

where $x(\cdot) \in \mathbb{R}^n$, $\tau \geq 0$, $C = C([-\tau, 0]; \mathbb{R}^n)$, $x_t(s) = x(t+s)$, $s \in [-\tau, 0]$. And it also works to the case of stochastic functional differential equations [1] as follows

$$dx(t) = f(x_t, t)dt + \sigma(x_t, t)dw(t), x_{t_0}(s) = \xi(s) \in L_{F_{t_0}}^2, -\tau \leq s \leq 0, \quad (2)$$

where w is a standard m -dimension Brownian motion defined on the completed probability space $(\Omega, F, \{F_t\}_{t \geq t_0}, P)$, the functionals f, σ satisfy Lipschitz conditions and the linear growth conditions and $f(0, t) = 0, \sigma(0, t) = 0$ for all $t \geq t_0$, $L_{F_{t_0}}^2 := L_{F_{t_0}}^2([-\tau, 0]; \mathbb{R}^n)$.

But the Razumikhin-type theorem in [1] deals with the case of p -moment stability of Equation (2) it can't solve the problem of stochastic stabilization of general time-varying delayed neural networks. For the purpose of solving the problem above, a new type Razumikhin theorem is proposed which guarantees the trivial solution of Equation (2) almost surely exponential stability and includes the deterministic one as a special case. A strategy to stabilize the Hopfield neural networks with non-delays is introduced in [17], and concludes that it can be stabilized by noises considering that the random strength is large enough. Are there some similar conclusions for the general time-varying delayed neural networks ? On the basis of the results reached previously, the positive answer is given to the question in this section. And the computational presentation to the random strength is also done for the neural networks stated above.

2 Notations

$\mathbb{R}^n, \mathbb{R}^{n \times n}, \mathbb{R}_+$, the n -dimensional Euclidean space, $n \times n$ matrix space and positive half line respectively; $|x|, |A|$, the Euclidean norm of a vector x and a matrix A respectively; $C([-\tau, 0]; \mathbb{R}^n)$, the space of all continuous functions ϕ from $[-\tau, 0]$ to \mathbb{R}^n with a norm $\|\phi\| = \sup_{-\tau \leq \theta \leq 0} |\phi(\theta)|$; $L^2_{F_{t_0}}([-\tau, 0]; \mathbb{R}^n)$, the family of all F_{t_0} -measurable $C([-\tau, 0]; \mathbb{R}^n)$ -valued random variables ϕ such that $E\|\phi\|^2 < \infty$; $C^{2,1}(\mathbb{R}^n \times \mathbb{R}_+; \mathbb{R}_+)$, the family of all positive real-valued functions $V(x, t)$ defined on $\mathbb{R}^n \times \mathbb{R}_+$ which are continuously twice differentiable in $x \in \mathbb{R}^n$ and once differentiable in $t \in \mathbb{R}_+$.

3 A New Razumikhin-Type Theorem of Stochastic Functional Equations

Let $V \in C^{2,1}(\mathbb{R}^n \times \mathbb{R}_+; \mathbb{R}_+)$ be positive definite Lyapunov function, and define the differential operator L with respect to Equation (2) as follows

$$LV(\phi, t) = V_t(\phi(0), t) + V_x^T(\phi(0), t)f(\phi, t) + \frac{1}{2}\text{trace}[\sigma^T(\phi, t)V_{xx}(\phi(0), t)\sigma(\phi, t)] . \tag{3}$$

We give the main results of this section.

Theorem 1. *If there exist positive constants $p, c_1, c_2, \lambda, q > 1$ and a positive definite Lyapunov function $V \in C^{2,1}(\mathbb{R}^n \times \mathbb{R}_+; \mathbb{R}_+)$ such that*

$$c_1|x|^p \leq V(x, t) \leq c_2|x|^p, \quad \forall(x, t) \in \mathbb{R}^n \times [t_0 - \tau, +\infty) \tag{4}$$

and

$$LV(\phi, t) - \frac{|\sigma^T(\phi, t)V_x(\phi(0), t)|^2}{2V(\phi(0), t)} := \Delta V(\phi, t) \leq -\lambda V(\phi(0), t), \quad t \geq t_0 \tag{5}$$

holds for those $\phi \in C([-\tau, 0]; \mathbb{R}^n)$ satisfying

$$V(\phi(t + \theta), t + \theta) \leq qV(\phi(t), t), \quad -\tau \leq \theta \leq 0 , \tag{6}$$

then, for any initial condition $x_{t_0} = \xi \in L^2_{F_{t_0}}$, the corresponding solution $x(t) := x(t; t_0, \xi)$ of Equation (2) has the following property

$$\limsup_{t \rightarrow \infty} \frac{1}{t} \log |x(t)| \leq -\frac{\alpha}{p} \quad a.s. \tag{7}$$

where $\alpha = \min\{\lambda, \log(q)/\tau\}$.

That is, under the conditions of the theorem, the trivial solution of Equation (2) is almost surely exponential stability.

Proof. From the assumption of f, σ in the Equation (2), we know that the trivial solution $x = 0$ of Equation (2) is not reachable, that is, for the solution $x(t)$ of Equation (2) with respect to the initial condition $x_{t_0} = \xi \in L^2_{F_{t_0}}$, we have $x(t) \neq 0$ almost surely. For these almost all $\omega \in \Omega$, we define

$$U(t) = \max_{-\tau \leq \theta \leq 0} \{\alpha(t + \theta) + \log V(x(t + \theta), t + \theta)\}, \quad t \geq t_0 .$$

In the following, we prove the following inequality

$$D^+U(t) = \limsup_{h \rightarrow 0^+} \frac{U(t + h) - U(t)}{h} \leq 0 \tag{8}$$

holds for arbitrarily fixed $t \geq t_0$ and almost all $\omega \in \Omega$. To show this, define

$$\bar{\theta} = \max\{\theta \in [-\tau, 0] : \alpha(t + \theta) + \log V(x(t + \theta), t + \theta) = U(t)\} .$$

Obviously, $\bar{\theta} \in [-\tau, 0]$ and

$$U(t) = \alpha(t + \bar{\theta}) + \log V(x(t + \bar{\theta}), t + \bar{\theta}) .$$

If $\bar{\theta} < 0$, then for any $\theta: \bar{\theta} < \theta \leq 0$, we have

$$\alpha(t + \theta) + \log V(x(t + \theta), t + \theta) < \alpha(t + \bar{\theta}) + \log V(x(t + \bar{\theta}), t + \bar{\theta}) .$$

Therefore, for sufficiently small $h > 0$, we have

$$\alpha(t + h) + \log V(x(t + h), t + h) \leq \alpha(t + \bar{\theta}) + \log V(x(t + \bar{\theta}), t + \bar{\theta}) .$$

That is $U(t + h) \leq U(t)$ and this implies $D^+U(t) \leq 0$.

If $\bar{\theta} = 0$, then for any $\theta: -\tau \leq \theta \leq 0$, we have

$$\alpha \cdot (t + \theta) + \log V(x(t + \theta), t + \theta) \leq U(t) = \alpha t + \log V(x(t), t) .$$

That is

$$V(x(t + \theta), t + \theta) \leq e^{-\alpha\theta} V(x(t), t) \leq qV(x(t), t) .$$

From the condition (5) of Theorem 1, we have

$$LV(x_t, t) - \frac{|\sigma^T(x_t, t)V_x(x(t), t)|^2}{2V(x(t), t)} \leq -\lambda V(x(t), t) . \tag{9}$$

On the other hand, for any $h > 0$ by using Itô formula, we have

$$\log V(x(t+h), t+h) - \log V(x(t), t) = \int_t^{t+h} \frac{LV(x_s, s)}{V(x(s), s)} ds + M(t) - \frac{1}{2}N(t) ,$$

where

$$M(t) = \int_t^{t+h} \frac{V_x^T(x(s), s)\sigma(x_s, s)}{V(x(s), s)} dw(s), N(t) = \int_t^{t+h} \frac{|\sigma^T(x_s, s)V_x(x(s), s)|^2}{V^2(x(s), s)} ds .$$

By the exponential martingale inequality,

$$P\left(\sup_{t_0 \leq t \leq n} [M(t) - \frac{\epsilon}{2}N(t)] > \frac{2}{\epsilon} \log n\right) \leq \frac{1}{n^2}$$

holds for any positive numbers ϵ, n , from Borel-Contelli Lemma, for almost all $\omega \in \Omega$, there exists a positive number $n_0(\omega)$, when $n > \max\{n_0, t\}$, one can easily deduce the following inequality

$$M(t) \leq \frac{\epsilon}{2}N(t) + \frac{2}{\epsilon} \log n$$

holds, and furthermore,

$$\begin{aligned} & \log V(x(t+h), t+h) - \log V(x(t), t) \\ & \leq \int_t^{t+h} \frac{LV(x_s, s)}{V(x(s), s)} ds - \frac{1}{2}(1-\epsilon)N(t) + \frac{2}{\epsilon} \log n \\ & \leq \int_t^{t+h} \frac{1}{V(x(s), s)} [LV(x_s, s) - \frac{1}{2}(1-\epsilon) \frac{|\sigma^T(x_s, s)V_x(x(s), s)|^2}{V(x(s), s)}] ds + \frac{2}{\epsilon} \log n \end{aligned}$$

holds for some fixed $n, n > \max\{n_0, t\}$ and sufficiently small $\epsilon > 0$. From the condition (9), we have

$$\log V(x(t+h), t+h) - \log V(x(t), t) \leq -\lambda h \leq -\alpha h .$$

That is,

$$\alpha(t+h) + \log V(x(t+h), t+h) \leq \alpha t + \log V(x(t), t) = U(t) \tag{10}$$

holds for any $h > 0$.

By the definition of $U(t)$ and the inequality (10), for sufficiently small $h > 0$, we have

$$U(t+h) \leq U(t) ,$$

and therefore $D^+U(t) \leq 0$. The inequality (8) is proved.

From the condition (4) and inequality (8), we have

$$\begin{aligned} U(t_0) &= \max_{-\tau \leq \theta \leq 0} \{\alpha(t_0 + \theta) + \log V(x(t_0 + \theta), t_0 + \theta)\} \\ &\leq \alpha t_0 + \log c_2 + \max_{-\tau \leq \theta \leq 0} \log |x(t_0 + \theta)|^p \\ &= \alpha t_0 + \log c_2 + p \log |\xi| < \infty , \end{aligned}$$

and $U(t) \geq \alpha t + \log c_1 |x(t)|^p$, we have

$$\alpha t + \log c_1 |x(t)|^p < \infty ,$$

it's easy to obtain that

$$\limsup_{t \rightarrow \infty} \frac{1}{t} \log |x(t)| \leq -\frac{\alpha}{p} .$$

The proof of Theorem [1](#) is complete.

For a positive definite Lyapunov function $V \in C^{2,1}(\mathbb{R}^n \times \mathbb{R}_+; \mathbb{R}_+)$, define the differential operator D with respect to deterministic Equation [\(1\)](#) as follows

$$DV(x_t, t) = V_t(x(t), t) + V_x(x(t), t)f(x_t, t) . \tag{11}$$

And define

$$\Delta_\sigma V(x_t, t) = \frac{|V_x(x(t), t)\sigma(x_t, t)|^2}{V(x(t), t)} - \text{trace}[\sigma^T(x_t, t)V_{xx}(x(t), t)\sigma(x_t, t)] . \tag{12}$$

Corollary 1. *If there exist positive numbers $p, c_1, c_2, q > 1$, real numbers c_3, c_4 , and positive definite Lyapunov function $V \in C^{2,1}(\mathbb{R}^n \times \mathbb{R}_+; \mathbb{R}_+)$ such that*

$$c_1|x|^p \leq V(x, t) \leq |x|^p, \forall(x, t) \in \mathbb{R}^n \times [t_0 - \tau, +\infty) , \tag{13}$$

and

$$DV(\phi, t) \leq c_3V(\phi(0), t) , \tag{14}$$

and

$$\Delta_\sigma V(\phi, t) \geq c_4V(\phi(0), t) \tag{15}$$

hold for $t \geq t_0$ and only for those $\phi \in C([-\tau, 0]; \mathbb{R}^n)$ satisfying

$$V(\phi(t + \theta), t + \theta) \leq qV(\phi(t), t), -\tau \leq \theta \leq 0 , \tag{16}$$

then, for any initial condition $x_{t_0} = \xi \in L^2_{F_{t_0}}$, the corresponding solution $x(t)$ of Equation [\(2\)](#) has the following property

$$\limsup_{t \rightarrow \infty} \frac{1}{t} \log |x(t)| \leq -\frac{\alpha}{p} \text{ a.s.} \tag{17}$$

where $\alpha = \min\{\frac{c_4}{2} - c_3, \log(q)/\tau\}$.

In particular, if $c_4 > 2c_3$, then under the conditions of the corollary, the trivial solution of Equation [\(2\)](#) (it can be treated as a perturbed equation of determinate Equation [\(1\)](#)) is almost surely exponential stability.

Proof. Because $\Delta V(\phi, t) = DV(\phi, t) - \frac{1}{2}\Delta_\sigma V(\phi, t)$, and by using Condition [\(14\)](#), [\(15\)](#) of Corollary [1](#), the inequality

$$\Delta V(\phi, t) \leq (c_3 - \frac{1}{2}c_4)V(\phi(0), t)$$

holds for $t \geq t_0$ and for those $\phi \in C([-\tau, 0]; \mathbb{R}^n)$ satisfying

$$V(\phi(t + \theta), t + \theta) \leq qV(\phi(t), t), -\tau \leq \theta \leq 0 .$$

By using Theorem 1, the corollary is true. The proof is complete.

Remark 1. Corollary [1](#) includes Razumikhin-type theorem of the deterministic Equation [\(1\)](#) as a special case, that is the following corollary holds.

Corollary 2. *If there exist positive numbers $p, c_1, c_2, q > 1$, and a positive definite Lyapunov function V such that*

$$c_1|x|^p \leq V(x, t) \leq c_2|x|^p, \forall(x, t) \in \mathbb{R}^n \times [t_0 - \tau, +\infty)$$

and

$$DV(\phi, t) \leq -\lambda V(\phi(0), t)$$

holds for $t \geq t_0$ and for those $\phi \in C([-\tau, 0]; \mathbb{R}^n)$ satisfying

$$V(\phi(t + \theta), t + \theta) \leq qV(\phi(t), t), -\tau \leq \theta \leq 0 ,$$

then, the solution $x(t) := x(t; t_0, \xi)$ of Equation [\(1\)](#) has the following property

$$\limsup_{t \rightarrow \infty} \frac{1}{t} \log |x(t)| \leq -\frac{\alpha}{p} .$$

This shows that the trivial solution of Equation [\(1\)](#) is exponential stability, where $\alpha = \min\{\lambda, \log(q)/\tau\}$.

The corresponding versions in delayed stochastic differential equation can be easily deduced from the Razumikhin theorem obtained above, and they are the basis of the researches of the stochastic stabilization of time-delayed neural networks to be discussed in the next section.

Delayed stochastic differential equation can be written as follows

$$\begin{aligned} dx(t) &= f(x(t), x(t - \tau), t)dt + \sigma(x(t), x(t - \tau), t)dw(t) \\ x(s) &= \xi(s) \in C([-\tau, 0]; \mathbb{R}^n) , \end{aligned} \tag{18}$$

where $x(t - \tau) = (x_1(t - \tau_1), \dots, x_n(t - \tau_n))^T$, $\tau_i \geq 0$ is the delay of $x_i, i = 1, 2, \dots, n$, and $\tau = \max_{1 \leq i \leq n} \{\tau_i\}$. The other variables and parameters are the same as Equation [\(2\)](#).

For a positive definite Lyapunov function $V \in C^{2,1}(\mathbb{R}^n \times \mathbb{R}_+; \mathbb{R}_+)$ and $\forall(x, y) \in \mathbb{R}^n \times \mathbb{R}^n$, define an operator L as follows

$$LV(x, y) = V_t(x, t) + V_x(x, t)f(x, y, t) + \frac{1}{2}\text{trace}[\sigma^T(x, y, t)V_{xx}(x, t)\sigma(x, y, t) .$$

The following two corollaries are the direct conclusions in the case of delayed stochastic differential equations of Theorem 1 and Corollary [1](#) respectively.

Corollary 3. *If there exist positive numbers p, c_1, c_2, λ and a positive definite Lyapunov function $V(x, t)$, such that*

$$c_1|x|^p \leq V(x, t) \leq c_2|x|^p$$

and

$$LV(x, y, t) - \frac{|\sigma^T(x, y, t)V_x(x, t)|^2}{2V(x, t)} \leq -\lambda V(x, t)$$

holds for $t \geq t_0$ and those $x, y \in \mathbb{R}^n$ satisfying: there exists $q > 1$, for any $-\tau \leq \theta \leq 0$ such that

$$V(y, t + \theta) \leq qV(x, t) ,$$

then, for any initial function $x(s) = \xi(s)$, $-\tau \leq s \leq 0$, the corresponding solution $x(t)$ of Equation (2) has the following property:

$$\limsup_{t \rightarrow \infty} \frac{1}{t} \log |x(t)| \leq -\frac{\alpha}{p} ,$$

where $\alpha = \min\{\lambda, \log(q)/\tau\}$.

For a positive definite function $V(x, t) \in R_+$, and any $x, y \in \mathbb{R}^n$, define two operators D, Δ_σ related to Equation (18) as following

$$DV(x, y, t) = V_t(x, t) + V_x^T(x, t)f(x, y, t) ,$$

$$\Delta_\sigma(x, y, t) = \frac{|\sigma^T(x, y, t)V_x(x, t)|^2}{V(x, t)} - \text{trace}[\sigma^T(x, y, t)V_{xx}(x, t)\sigma(x, y, t)] .$$

Corollary 4. *If there exist positive numbers $p, c_1, c_2, q > 1$, real numbers $c_3, c_4 \in \mathbb{R}$ and a positive definite Lyapunov function $V(x, t)$ such that*

$$c_1|x|^p \leq V(x, t) \leq c_2|x|^p, \forall (x, t) \in \mathbb{R}^n \times [t_0 - \tau, +\infty)$$

and

$$\begin{aligned} DV(x, y, t) &\leq c_3V(x, t) \\ \Delta_\sigma V(x, y, t) &\geq c_4V(x, t) \end{aligned}$$

hold for $t \geq t_0$ and those $x, y \in \mathbb{R}^n$ satisfying

$$V(y, t + \theta) \leq qV(x, t), -\tau \leq \theta \leq 0 ,$$

then, for any initial value $x_{t_0} = \xi$, the corresponding solution $x(t)$ of Equation (18) has the following property:

$$\limsup_{t \rightarrow \infty} \frac{1}{t} \log |x(t)| \leq -\frac{\alpha}{p} .$$

In particular, if $c_4 > 2c_3$, then the trivial solution $x = 0$ of Equation (18) is almost surely exponential stability, where $\alpha = \min\{\frac{c_4}{2} - c_3, \log(q)/\tau\}$.

4 Stochastic Stabilization of Delayed Neural Networks

In the following, we will solve the problem of stochastic stabilization of delayed neural networks as follows

$$\frac{dx(t)}{dt} = -Bx(t) + Af(x(t - \tau)) , \tag{19}$$

where $x = (x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n$ is the state vector, $A = (a_{ij})_{n \times n}$ is the connected matrix among the neurons of the neural networks, $B = \text{diag.}(b_1, b_2, \dots, b_n)$ is the positive definite diagonal matrix, $y_i(t) = x_i(t - \tau_i)$, $\tau_i \geq 0$ is the delay of the i -th neuron, $f(y(t)) = (f_1(y_1(t)), \dots, f_n(y_n(t)))^T$ is the output vector of the neurons and assume that there exist positive numbers α_i such that

$$|f_i(u)| \leq 1 \wedge \alpha_i |u|, -\infty < u < +\infty, i = 1, 2, \dots, n . \tag{20}$$

For neural networks (19) with properties (20), we will introduce stochastic perturbation in some way into it to make it stabilization. That is, the following stochastic neural networks

$$dx(t) = [-Bx(t) + Af(x(t - \tau))]dt + \sigma(x(t), x(t - \tau))dw(t) \tag{21}$$

is exponential stability, where σ is $n \times m$ perturbed intensity matrix and $\sigma(0, 0) = 0$, w is m -dimension standard Brownian motion defined on the completed probability space $(\Omega, F, \{F_t\}_{t \geq 0}, P)$.

Theorem 2. For stochastic neural networks (21), if the perturbed intensity matrix σ takes the form

$$\sigma(x, y) = (\theta_1 x + \delta_1 y, \dots, \theta_m x + \delta_m y), \forall x, y \in \mathbb{R}^n , \tag{22}$$

and satisfies

$$(1 - \epsilon) \sum_{i=1}^m \theta_i^2 > (1 + \frac{1}{\epsilon})q \sum_{i=1}^m \delta_i^2 - 2b + \alpha(1 + q)|A| \tag{23}$$

with $0 < \epsilon < 1, q > 1$, and $b = \min\{b_i, 1 \leq i \leq n\}$, $\alpha = \max\{\alpha_i, 1 \leq i \leq n\}$, then the stochastic neural networks (21) is almost surely exponential stability, that is, the deterministic neural networks (19) is stabilized by the stochastic perturbation satisfying (22) and (23).

Proof. Taking Lyapunov function $V(x) = |x|^2$, according to Corollary 4, the operator D related to the neural networks (21) is as the following

$$D(x, y) = 2x^T[-Bx + Af(y)] = -2x^T Bx + 2x^T Af(y), x, y \in \mathbb{R}^n .$$

By using Properties (20) and the basic inequality $2ab \leq a^2 + b^2$, we have

$$D(x, y) \leq -2b|x|^2 + 2\alpha|A| \cdot |x| \cdot |y| \leq (-2b + \alpha|A|)|x|^2 + \alpha|A||y|^2 .$$

For those $x, y \in \mathbb{R}^n$ satisfying $V(y) \leq qV(x)$, that is $|y|^2 \leq q|x|^2$, we have

$$DV(x, y) \leq (-2b + \alpha|A|(1 + q))|x|^2 .$$

Let $c_3 = -2b + \alpha|A|(1 + q)$, then

$$D(x, y) \leq c_3 V(x) . \tag{24}$$

Furthermore, the operator Δ_σ related to the neural networks (21) is as the following

$$\Delta_\sigma(x, y) = \frac{|2\sigma^T(x, y)x|^2}{|x|^2} - 2\text{trace}[\sigma^T(x, y)\sigma(x, y)] .$$

The second term of right hand is easily dealt with considering that (22):

$$2\text{trace}[\sigma^T(x, y)\sigma(x, y)] = 2 \sum_{i=1}^m (\theta_i x^T + \delta_i y^T)(\theta_i x + \delta_i y) = 2 \sum_{i=1}^m (\theta_i^2 |x|^2 + \delta_i^2 |y|^2 + 2\theta_i \delta_i y^T x) .$$

The first term of right hand can be estimated as follows:

$$\begin{aligned} |2x^T \sigma(x, y)|^2 / |x|^2 &= 4x^T \sigma(x, y) \sigma^T(x, y) x / |x|^2 \\ &= 4x^T \left[\sum_{i=1}^m (\theta_i x + \delta_i y)(\theta_i x^T + \delta_i y^T) \right] x / |x|^2 \\ &= 4 \sum_{i=1}^m (\theta_i^2 x^T x x^T x + \delta_i^2 x^T y y^T x + 2\theta_i \delta_i x^T x y^T x) / |x|^2 \\ &= 4 \sum_{i=1}^m (\theta_i^2 |x|^4 + \delta_i^2 |x^T y|^2 + 2\theta_i \delta_i |x|^2 y^T x) / |x|^2 \\ &\geq 4 \sum_{i=1}^m \theta_i^2 |x|^2 + 8 \sum_{i=1}^m \theta_i \delta_i y^T x . \end{aligned}$$

So, we have

$$\begin{aligned} \Delta_\sigma(x, y) &\geq 2 \sum_{i=1}^m \theta_i^2 |x|^2 - 2 \sum_{i=1}^m \delta_i^2 |y|^2 + 4 \sum_{i=1}^m \theta_i \delta_i y^T x \\ &\geq 2 \sum_{i=1}^m \theta_i^2 |x|^2 - 2 \sum_{i=1}^m \delta_i^2 |y|^2 - 4 \sum_{i=1}^m |\theta_i \delta_i| \cdot |y| \cdot |x| . \end{aligned}$$

Choosing $0 < \epsilon < 1$, from the basic inequality $2ab \leq a^2 + b^2$, we have

$$2|\theta_i \delta_i| \cdot |y| \cdot |x| \leq \epsilon \theta_i^2 |x|^2 + \frac{\delta_i^2}{\epsilon} |y|^2 ,$$

so, we have

$$\Delta_\sigma(x, y) \geq 2(1 - \epsilon) \sum_{i=1}^m \theta_i^2 |x|^2 - 2(1 + \frac{1}{\epsilon}) \sum_{i=1}^m \delta_i^2 |y|^2 .$$

For those $x, y \in \mathbb{R}^n$ satisfying $V(y) \leq qV(x)$, that is $|y|^2 \leq q|x|^2$, we have

$$\Delta_\sigma(x, y) \geq [2(1 - \epsilon) \sum_{i=1}^m \theta_i^2 - 2(1 + \frac{1}{\epsilon})q \sum_{i=1}^m \delta_i^2] |x|^2 .$$

Let $c_4 = 2(1 - \epsilon) \sum_{i=1}^m \theta_i^2 - 2(1 + \frac{1}{\epsilon})q \sum_{i=1}^m \delta_i^2$, we have the following inequality

$$\Delta_\sigma(x, y) \geq c_4 V(x) . \quad (25)$$

From inequality (24), (25) and Corollary 4, if $c_4 > 2c_3$, that is, (23) holds, then the trivial solution of the neural networks (21) is almost surely exponential stability. The proof is completed.

Acknowledgement

The work is supported by National Natural Science Foundation of China (60474001) and Natural Science Foundation of Henan Province of China (0611054500).

References

1. Mao, X.: Stochastic Differential Equations and Applications. Horwood Pub., Chichester (1997)
2. Hale, J.K.: Theory of Functional Differential Equations. Springer-Verlag (1977)
3. Liao, W., Liao, X.: Robust Stability of Time-Delayed Interval CNN in Noisy Environment. Acta Automatica Sinica **30** (2) (2004) 300-305
4. Liao, X.: Stability Theory and Applications on Power Systems. Beijing: Press of Defence Industry (2000)
5. Liu, Y., Feng, Z.: Theory and Applications on Large-scale Dynamic Systems-Randomicity, Stability and Control. Guangzhou: Press of South China University of Technology (1992)
6. Zeng, Z., Wang, J., Liao, X.: Global Asymptotic Stability and Global Exponential Stability of Neural Networks with Unbounded Time-Varying delays. IEEE Trans. on Circuits and Systems II, Express Briefs **52** (3) (2005) 168-173
7. Mao, X.: Stochastic Stability and Stabilization. Proceedings of the 2002 International Conference on Control and Automation, Xiamen, China (2002) 1208-1212
8. Mao, X.: Razumikhin-Type Theorems on Exponential Stability of Stochastic Functional Differential Equations. Stochastic Processes and Their Applications **65** (1996) 233-250
9. Hopfield, J.J.: Neural Networks and Physical Systems with Emergent Collective Computational Abilities. Proc. Natl. Acad. Sci. USA **79** (1982) 2554-2558
10. Hopfield, J.J.: Neurons with Graded Response Have Collective Computational Properties Like Those of Two-state Neurons. Proc. Natl. Acad. Sci. USA **81** (1984) 3088-3092
11. Deng, F., Feng, Z., Liu, Y.: Stability and Feedback-stabilization of General Linear Delayed Systems. Control Theory and Application **15**(2) (1998) 299-303
12. Shen, Y., Liao, X.: Exponential Stability of Delayed Hopfield Neural Networks. Acta Mathematica Scientia **19** (2) (1999) 211-218
13. Zeng, Z., Wang, J., Liao, X.: Global Exponential Stability of Neural Networks with Time-Varying Delays. IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications **50** (10) (2003) 1353-1358

An Evolutionary Multi-objective Neural Network Optimizer with Bias-Based Pruning Heuristic

Prospero C. Naval Jr.¹ and John Paul T. Yusiong²

¹ Department of Computer Science, University of the Philippines-Diliman, Diliman, Quezon City, Philippines

`pnaval@up.edu.ph`

² Division of Natural Sciences and Mathematics, University of the Philippines-Visayas, Tacloban City, Leyte, Philippines

`jtyusiong@up.edu.ph`

Abstract. Neural network design aims for high classification accuracy and low network architecture complexity. It is also known that simultaneous optimization of both model accuracy and complexity improves generalization while avoiding overfitting on data. We describe a neural network training procedure that uses multi-objective optimization to evolve networks which are optimal both with respect to classification accuracy and architectural complexity. The NSGA-II algorithm is employed to evolve a population of neural networks that are minimal in both training error and a Minimum Description Length-based network complexity measure. We further propose a pruning rule based on the following heuristic: connections to or from a node may be severed if their weight values are smaller than the network's smallest bias. Experiments on benchmark datasets show that the proposed evolutionary multi-objective approach to neural network design employing the bias-based pruning heuristic yields networks that have far fewer connections without seriously compromising generalization performance when compared to other existing evolutionary optimization algorithms.

1 Introduction

The architecture of a neural network greatly influences the success of the training process. A network that is too small will not be capable of learning the problem effectively whereas a network that is too large will overfit and exhibit low generalization performance. When one aims for an architecturally simple neural network with a good generalization performance then designing the neural network architecture becomes a challenging task. Manually designing such a neural network is not straightforward since classifier complexity and good generalization performance are conflicting goals.

In classical optimization, an inherently multi-objective problem is usually reformulated as one having a single objective through a weighted summation of

objectives prior to optimization leading to the production of a single optimal solution for each run. However, if the solution cannot be accepted for some reasons mentioned in [4], then these methods have to be applied multiple times until a suitable solution is found.

On the other hand, multi-objective optimization [2] searches for Pareto optimal solutions to problems that have multiple performance criteria which are often conflicting. This approach can concurrently optimize multiple, often conflicting objectives and then generate a Pareto set from which a suitable solution can be chosen.

Evolutionary Algorithms (EAs) have been increasing in popularity for the design of neural network architectures since they are less prone to getting stuck in local optima compared to gradient-based algorithms [13]. Furthermore, over the past decade, EAs were extended to handle multi-objective optimization problems because of their population-based feature which allow them to maintain a diverse set of solutions. In fact, some authors [4] seem to suggest that multi-objective optimization is a suitable problem area for EAs. NSGA-II [3], MOGA [5] and NPGA [8] are among the various multi-objective evolutionary algorithms (MOEAs).

The rest of the paper is organized as follows. We describe in detail our proposed procedure for training neural networks (which we call MNNO) in the next section. Section 3 outlines the experiments conducted to test the performance of MNNO as well as the results obtained from these experiments while the final section contains the conclusions.

2 Description of the Multi-objective Neural Network Optimizer (MNNO)

In this section, a new evolutionary multi-objective approach to neural network training is described which is based on NSGA-II [3]. The proposed algorithm called Multi-objective Neural Network Optimizer (MNNO) will simultaneously

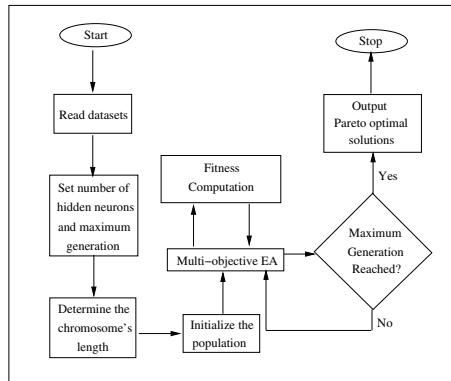


Fig. 1. The MNNO Procedure for optimizing Neural Networks

optimize the set of connection weights and its corresponding architecture by treating the problem as a multi-objective minimization problem. Figure 1 shows the procedure for training neural networks with MNNO.

The MNNO Procedure begins by reading the dataset from which the number of input and output nodes are determined. The number of hidden nodes and maximum number of generations are then set. Next, the chromosome length L , as shown in eq. (1), is computed. A population of neural networks is then generated and initialized according to the pertinent settings of the underlying NSGA-II algorithm.

The two fitness functions compute the training error and MDL-based network complexity for each neural network in the population for the given dataset

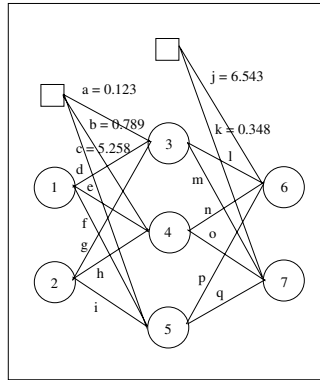


Fig. 2. A fully-connected feedforward neural network

Node	3			4			5			6			7				
Link	a	d	g	b	e	h	c	f	i	j	l	n	p	k	m	o	q
Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Fig. 3. MNNO’s chromosome representation of the fully-connected feedforward neural network above. We note that this representation makes the application of the Bias-based Pruning Heuristic straightforward.

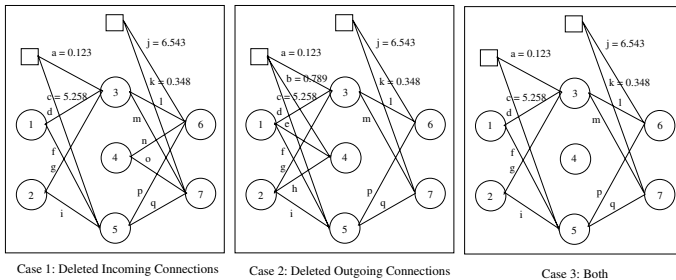


Fig. 4. Possible ways a node is deleted when Bias-Based Pruning Heuristic is applied

and return the values to NSGA-II. Bias-based pruning is applied during the forward computation when performing training error calculations. The procedure stops and outputs a set of non-dominated neural networks after the maximum generation is reached.

2.1 Chromosome Representation of a Neural Network

A population is composed of chromosomes and a chromosome string represents a one-hidden layer neural network with fields that correspond to the weight values of the connections whose length L is:

$$L = (I + 1) \cdot H + (H + 1) \cdot O \tag{1}$$

where I , H and O respectively refer to the number of input, hidden and output nodes. The +1 in the equation indicates that each hidden and output node has a bias. Figure 3 is a chromosome representation of the neural network shown in Figure 2. In addition, the chromosomes in the population are initialized with random values from a uniform distribution in the range of

$$\left[\frac{-1}{\sqrt{fan-in}}, \frac{1}{\sqrt{fan-in}} \right] \tag{2}$$

where the value of $fan-in$ is the number of incoming connections to a node.

Bias-based Pruning Method. We adopt a top-down approach in evolving the set of Pareto-optimal neural networks. During its early exploratory phase, MNNO attempts to evolve a population of networks that are larger than necessary and as the training process progresses components that are not needed are removed. Large initial sizes permit the networks to learn reasonably quickly with less sensitivity to initial conditions while the reduced complexity of the trimmed system favors improved generalization [11].

We prune connections according to the following heuristic: *a connection may be severed if its weight value is smaller than the smallest bias of the entire network. Accordingly, a node is removed if all its incoming or outgoing connections have weight values that are smaller than the smallest bias of the network.*

The logic behind this heuristic is that the absolute value of the connection weight that is smaller than the absolute value of the smallest bias of the entire network will contribute the least to the node it is connected to and the effects of its initial removal can be absorbed by other connection weights after some retraining.

Furthermore, this pruning technique incorporated into MNNO allows the algorithm to simultaneously perform structure optimization and weight adaptation. Figure 4 shows the three possible node deletion scenarios when pruning is applied: deleted incoming connections (when $e, h < a$), deleted outgoing connections (when $n, o < a$) and deleted incoming and outgoing connections (when $e, h, n, o < a$) given that a is the smallest bias of the network.

Moreover, MNNO’s neural network chromosome representation facilitates the application of this heuristic. It is straightforward to compute for the smallest bias of the network in this representation and to skip deleted nodes.

2.2 Fitness Computation

The chromosome's fitness is assessed on the basis of the mean-squared error (MSE) on the training set and the Minimum Description Length (MDL) principle.

Mean-Squared Error. This is the first fitness function which determines the neural network's training error. We define the network's mean-squared error (MSE) as:

$$MSE = \frac{1}{nQ} \sum_{q=1}^Q \sum_{i=1}^n (D_i(q) - O_i(q))^2 \quad (3)$$

where $D_i(q)$ and $O_i(q)$ are the desired and actual outputs of node i for pattern q while Q and n are the number of patterns and number of output nodes respectively.

The Bias-based Pruning Heuristic is applied during network forward computation prior to the calculation of the training error.

Minimum Description Length. This second fitness function minimizes the neural network's complexity by applying the Minimum Description Length principle described in [16]. MDL is an elegant method to determine the appropriate complexity size of a model because the MDL principle, as explained in [7] asserts that the best model of some data is the one that minimizes the combined cost of describing the model and describing the misfit between the model and the data. MDL effectively minimizes the complexity of the neural network structure by minimizing the number of active connections with respect to the neural network's performance on the training set as shown in the equations below:

$$MDL = Class\ Error + Complexity \quad (4)$$

$$Class\ Error = 1.0 - \frac{numberofCorrectClassification}{numberofExamples} \quad (5)$$

$$Complexity = \frac{numberofActiveConnections}{TotalPossibleConnections} \quad (6)$$

where: *Class Error* is the error in classification while *Complexity* is the complexity measure in terms of the ratio between the active connections and the total number of possible connections.

3 Implementation and Results

Seven datasets from the UCI machine learning repository [9] were used to determine the effectiveness of MNNO. The characteristics of these datasets are summarized in Table 1 and the parameters of the NSGA-II algorithm are shown in Table 2.

Input nodes correspond to the attributes of the examples in the dataset while output nodes correspond to the output classes wherein the output classes are encoded with the usual 1-of- m classes using 1, 0 output if there are $m > 2$ values (i.e. one of the m outputs is set to 1 and the rest to 0). If there are only two output classes, then one class is set to 1 while the other is set to 0.

For each dataset, the experiments were repeated thirty (30) times and each experiment uses a different randomly generated initial population. In addition, the number of input and output nodes are problem-dependent but there is no exact method to determine the best number of hidden nodes. Fortunately, there are rules of thumb [12] to obtain this value. In this study, the number of hidden nodes is set to 10. Also, a training example with missing attribute values is deleted from the dataset.

Table 1. Description of the datasets used in the experiments

Dataset	Train Set	Test Set	Features	Class	MaxGen
Monks-1	124	432	6	2	500
Vote	300	135	16	2	300
Breast	457	226	9	2	300
Iris	100	50	4	3	500
Heart	180	90	13	2	500
Thyroid	3772	3428	21	3	200
Wine	118	60	13	3	500

Table 2. Parameters and Values for NSGA-II used in our Experiments

Parameters	NSGA-II
Optimization Type	Minimization
Population Size	100
Archive Size	100
Fitness Functions	2
Lower Limit of Variable	-100.0
Upper Limit of Variable	100.0
Probability of Mutation	$1.0/\text{number of variables}$
Probability of Crossover	0.9
Distribution Index of Mutation (η_{m})	20
Distribution Index of Crossover (η_{c})	20

3.1 Performance of the Multi-objective Neural Network Optimizer

Tables 3-6 show the results obtained from our experiments. The average values shown were obtained by averaging on the entire Pareto set over 30 trials. The number of neural networks produced by MNNO is problem-dependent and exhibits a large variation. The test set and training sets' mean-squared error and

Table 3. Average number of neural networks generated

Dataset	Average	Median	Std. Dev.
Monks-1	8.300	7.000	5.688
Vote	36.733	34.000	13.946
Breast	35.267	31.500	17.312
Iris	22.133	18.000	9.247
Heart	35.200	30.000	16.614
Thyroid	29.700	29.000	9.535
Wine	19.667	18.500	9.260

Table 4. Average Mean-Squared Error on the training and test sets

Dataset	Training Set			Test Set		
	Average	Median	Std. Dev.	Average	Median	Std. Dev.
Monks-1	1.56%	0.93%	0.020	2.16%	1.13%	0.025
Vote	1.87%	1.76%	0.004	1.92%	1.84%	0.005
Breast	1.33%	1.06%	0.009	1.63%	1.40%	0.008
Iris	1.81%	1.44%	0.010	3.08%	3.22%	0.010
Heart	5.97%	6.02%	0.004	6.64%	6.26%	0.011
Thyroid	4.57%	4.65%	0.013	4.66%	4.75%	0.012
Wine	2.26%	2.16%	0.011	5.36%	5.29%	0.020

Table 5. Average percentage of misclassification on the training and test sets

Dataset	Training Set			Test Set		
	Average	Median	Std. Dev.	Average	Median	Std. Dev.
Monks-1	2.00%	0.00%	0.034	3.97%	0.87%	0.053
Vote	3.95%	3.95%	0.006	4.35%	4.42%	0.011
Breast	2.35%	2.32%	0.004	3.42%	3.43%	0.004
Iris	2.82%	2.59%	0.013	6.18%	6.19%	0.016
Heart	14.30%	14.56%	0.014	16.96%	17.36%	0.031
Thyroid	6.41%	6.31%	0.005	6.60%	6.54%	0.004
Wine	4.20%	3.81%	0.022	10.87%	10.61%	0.041

Table 6. Average number of connections and hidden nodes used

Dataset	Connections			Hidden nodes		
	Average	Median	Std. Dev.	Average	Median	Std. Dev.
Monks-1	21.31	21.20	3.684	4.23	4.30	0.496
Vote	23.51	21.76	6.795	2.90	2.82	0.454
Breast	14.58	14.35	3.896	2.31	2.43	0.458
Iris	26.59	27.57	5.585	4.40	4.19	0.973
Heart	22.58	22.35	4.269	2.58	2.50	0.431
Thyroid	17.29	17.25	4.925	2.64	2.68	0.787
Wine	47.23	47.89	7.930	5.33	5.38	0.941

misclassification disparity are small for most datasets indicating that optimal networks have been obtained. The relatively high standard deviations indicate that the networks produced vary in quality between trials. We observe however that the variations in mean-squared error, number of connections and number of hidden nodes within the Pareto set are small.

Moreover, network complexity as measured by the number of connections and number of hidden nodes is problem-dependent. Variability in complexity is observed mainly within the Pareto set while inter-trial variations are small.

Best Neural Networks. Among the neural networks in the Pareto set, two neural networks were considered, the minimum-error neural networks and the least-complex neural networks. Tables 7-9 compare these two neural networks. As expected, the minimum-error neural networks perform much better than the least-complex neural networks but require a much larger number of connections and hidden nodes.

Table 7. Minimum-Error and Least-Complex Networks' Mean-Squared Error on the training and test sets

Dataset	Training Set		Test Set	
	Minimum-Error	Least-Complex	Minimum-Error	Least-Complex
Monks-1	0.97%	2.27%	1.56%	2.82%
Vote	1.44%	3.19%	1.72%	2.90%
Breast	0.84%	2.55%	1.34%	2.64%
Iris	0.98%	3.75%	2.39%	4.64%
Heart	5.43%	7.12%	6.55%	7.29%
Thyroid	3.03%	10.37%	3.19%	10.36%
Wine	1.64%	3.47%	5.36%	5.72%

Table 8. Minimum-Error and Least-Complex Networks' Classification Error on the training and test sets

Dataset	Training Set		Test Set	
	Minimum-Error	Least-Complex	Minimum-Error	Least-Complex
Monks-1	1.45%	2.37%	3.34%	4.27%
Vote	3.27%	4.84%	4.25%	4.49%
Breast	1.88%	2.96%	3.27%	3.95%
Iris	2.30%	3.63%	5.60%	6.87%
Heart	13.83%	15.59%	17.30%	17.37%
Thyroid	6.26%	6.91%	6.61%	6.86%
Wine	3.90%	4.94%	11.17%	10.56%

Comparison with Existing Methods. The performance of a neural network is characterized by its generalization error, which is one of the most important

criteria to determine the effectiveness of neural network learning. Tables 10 and 11 compare MNNO with MGNN [10] and EPNet [14] respectively. Results show that MNNO produces far fewer connections compared to the two algorithms without seriously compromising generalization performance.

Table 9. Average Number of Connections and Hidden nodes for Minimum-Error and Least-Complex Neural Networks

Dataset	Connections		Hidden nodes	
	Minimum-Error	Least-Complex	Minimum-Error	Least-Complex
Monks-1	29.13	14.63	5.30	3.20
Vote	54.37	9.17	4.77	1.57
Breast	31.87	6.10	4.20	1.10
Iris	40.50	17.90	6.53	2.93
Heart	39.73	10.83	3.93	1.30
Thyroid	37.03	9.30	4.17	1.73
Wine	64.37	35.77	6.53	4.33

Table 10. Performance comparison between MGNN and MNNO

Algorithm	MSE on Test set			Number of Connections		
	Breast	Iris	Wine	Breast	Iris	Wine
MGNN-ep	3.28%	6.17%	5.12%	80.87	56.38	132.83
MGNN-rank	3.33%	7.28%	6.23%	68.46	47.06	111.53
MGNN-roul	3.05%	8.43%	8.46%	76.40	55.13	120.36
MNNO	1.63%	3.08%	5.36%	14.58	26.59	47.23

Table 11. Performance comparison between EPNet and MNNO

Algorithm	MSE on Test set			Number of Connections		
	Breast	Heart	Thyroid	Breast	Heart	Thyroid
EPNet	1.42%	12.27%	1.13%	41.00	92.60	208.70
MNNO	1.63%	6.64%	4.66%	14.58	22.58	17.29

4 Conclusion

This paper introduces an evolutionary-based multi-objective approach to neural network design called Multi-objective Neural Network Optimizer. Experiments on benchmark datasets as well as comparisons with EPNet and MGNN are promising. Results showed that the incorporation of our proposed bias-based pruning heuristic into MNNO effectively optimizes the neural network architecture producing far fewer connections than both algorithms without seriously sacrificing generalization for the datasets tested.

References

1. Barron, A., Rissanen, J., Yu, B.: The Minimum Description Length Principle in Coding and Modeling. *IEEE Trans. Information Theory* **44** (1998) 2743-2760
2. Deb, K.: *Multi-Objective Optimization Using Evolutionary Algorithms*. John Wiley & Sons (2001)
3. Deb K., Pratap A., Agarwal S., Meyarivan, T.: A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Trans. Evolutionary Computation* **6**(2) (2002) 182-197
4. Fonseca, C.M., Fleming, P.J.: An Overview of Evolutionary Algorithms in Multi-objective Optimization. *Evolutionary Computation* **3**(1) (1995) 1-16
5. Fonseca, C.M., Fleming, P.J.: Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In *Genetic Algorithms: Proceedings of the Fifth International Conference* (S. Forrest, ed.), San Mateo, CA: Morgan Kaufmann (1993) 416-423
6. Grunwald, P.: A Tutorial Introduction to the Minimum Description Length Principle. *Advances in Minimum Description Length: Theory and Applications*. MIT Press (2004)
7. Hinton, G., van Camp, D.: Keeping Neural Networks Simple by Minimizing the Description Length of the Weights. *Proceedings of COLT-93* (1993)
8. Horn, J., Nafpliotis, N., Goldberg, D.E.: A Niche Pareto Genetic Algorithm for Multiobjective Optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, Piscataway, New Jersey. *IEEE Service Center* **1** (1994) 82-87
9. Newman, D., Hettich, S., Blake, C., Merz, C.: *UCI Repository of Machine Learning Databases*. Irvine, CA: University of California, Department of Information and Computer Science (1998)
10. Palmes, P., Hayasaka, T., Usui, S.: Mutation-based Genetic Neural Network. *IEEE Trans. Neural Networks* **6**(3) (2005) 587-600
11. Reed, R.: Pruning algorithms - A Survey. *IEEE Trans. Neural Networks* **4**(5) (1993) 740-747
12. Shahin, M., Jaksza, M., Maier, H.: Application of Neural Networks in Foundation Engineering. Theme paper to the International e-Conference on Modern Trends in Foundation Engineering: Geotechnical Challenges and Solutions, Theme No. 5: Numerical Modelling and Analysis, Chennai, India (2004)
13. Yao, X.: Evolving Artificial Neural Networks. *Proceedings of the IEEE* **87** (1999) 1423-1447
14. Yao, X., Liu, Y.: Evolving Artificial Neural Networks through Evolutionary Programming, Presented at the Fifth Annual Conference on Evolutionary Programming, 29 February-2 March 1996, San Diego, CA, USA MIT Press (1996) 257-266

A Hybrid of Particle Swarm Optimization and Hopfield Networks for Bipartite Subgraph Problems

Jiahai Wang

Department of Computer Science, Sun Yat-sen University,
No.135, Xingang West Road, Guangzhou 510275, P.R. China
wjiahai@hotmail.com

Abstract. The bipartite subgraph problem is a classical problem in combinatorial optimization. In this paper, we incorporate a chaotic discrete Hopfield neural network (CDHNN), as a local search scheme, into the discrete particle swarm optimization (DPSO) and develop a hybrid algorithm DPSO-CDHNN for the bipartite subgraph problem. The proposed algorithm not only performs exploration by using the population-based evolutionary search ability of the DPSO, but also performs exploitation by using the CDHNN. Simulation results show that the proposed algorithm has superior ability for bipartite subgraph problem.

1 Introduction

The bipartite subgraph problem is a classical problem in combinatorial optimization. The goal of this problem is finding a bipartite subgraph with maximum number of edges of a given graph [1]. The problem is known to be NP-complete [1]. Therefore, it is computationally intractable, even to be approximated with certain absolute performance bounds. It is important to develop methods for finding reasonable solutions in reasonable computation time.

In 1989, Johnson et al. proposed simulated annealing (SA) [2] for such NP-complete problems. Later, Lee et al. [3] proposed the first parallel algorithm using a maximum neural model (MNN) [4] for the bipartite subgraph problem, and illustrated that the solution quality was superior to that of the best existing algorithms. Galán-Marín et al. [5] [6] analyzed the maximum neural model and proposed a novel optimal competitive Hopfield model (OCHOM) for combinatorial optimization problems. Their simulation results for the bipartite subgraph problem illustrated that the OCHOM was much superior to the MNN in terms of the solution quality and the computation time [6]. By introducing a hill-climbing dynamics into the OCHOM, we proposed a stochastic optimal competitive Hopfield algorithm (SOCHOM) which permits temporary energy increases and therefore can help the OCHOM escape from local minima [7]. The solution quality found by the SOCHOM was superior to that of OCHOM and SA algorithms [7].

Wu et al. [8] gave new convergence conditions for discrete Hopfield neural networks (DHNN) [9]. In order to prevent the DHNN to be trapped in a stable

state corresponding to a local minimum, we proposed chaotic DHNN (CDHNN), which helps the DHNN escape from local minima [10]. During the past decade, a novel evolutionary computation technique, particle swarm optimization (PSO), was proposed by Kennedy and Eberhart [11] and it has attracted much attention. Most of the applications have been concentrated on solving continuous optimization problems. To solve discrete (combinatorial) optimization problems, Kennedy and Eberhart [12] also developed a discrete version of PSO (DPSO), which however has seldom been utilized.

In this paper, a DPSO algorithm is developed. Further, the CDHNN, as a local search scheme, is incorporated into the proposed DPSO to improve its performance and a hybrid particle swarm optimization algorithm, called DPSO-CDHNN is proposed for the bipartite subgraph problem. A number of instances have been simulated, and the simulation results show that the proposed algorithm can get better results than SOCHOM, CDHNN, and DPSO algorithms.

2 Description of Bipartite Subgraph Problems

Let $G = (V, E)$ be an undirected graph, where V is a set of vertices and E is a set of edges. If the vertex set V of graph G can be partitioned into 2-disjoint subsets and no edge exists between two vertices in the same subset, then the graph G is called a bipartite graph. Given a graph $G = (V, E)$, the goal of the bipartite subgraph problem is to remove the minimum number of the edges from a given graph such that the remaining graph is a bipartite graph. One case in point is a simple undirected graph composed of five vertices and seven edges as shown in Fig.1 (a) [7]. The graph becomes bipartite for the case where one edge is removed. Figure 1 (b) shows the resulting bipartite graph of the graph Fig.1 (a).

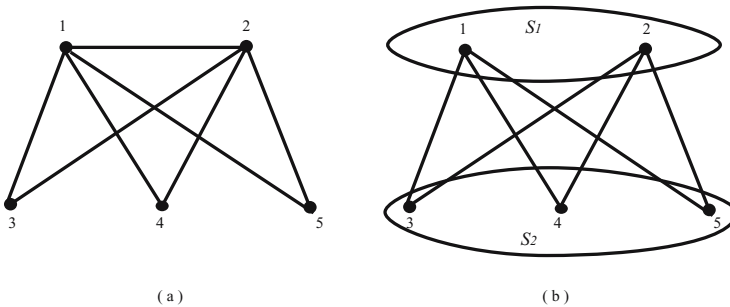


Fig. 1. (a) A simple undirected graph composed of five vertices and seven edges, (b) one of its bipartite graphs

3 DPSO-CDHNN for Bipartite Subgraph Problems

In this Section, we incorporate the chaotic discrete Hopfield neural network (CDHNN), as a local search scheme, into the discrete particle swarm optimization (DPSO) and develop a hybrid algorithm DPSO-CDHNN for the bipartite subgraph problem.

3.1 Discrete Particle Swarm Optimization (DPSO)

The PSO is inspired by observing the bird flocking or fish school [10]. A large number of birds/fishes flock synchronously, change direction suddenly, and scatter and regroup together. Each individual, called a particle, benefits from the experience of its own and that of the other members of the swarm during the search for food. Due to the simple concept, easy implementation and quick convergence, the PSO has been applied successfully to continuous nonlinear function [10], neural network [13], nonlinear constrained optimization problems [14], etc. Most of the applications have been concentrated on solving continuous optimization problems. To solve discrete (combinatorial) optimization problems, Kennedy and Eberhart [12] also developed a discrete version of PSO (DPSO), which however has seldom been utilized.

Denote by N the number of particles in the swarm. Let $X_i(t) = (x_{i1}(t), x_{i2}(t), \dots, x_{iD}(t))$, $x_{id}(t) \in \{-1, 1\}$, be particle i with D bits at iteration t , where $X_i(t)$ being treated as a potential solution has a rate of change called velocity. Denote the velocity as $V_i(t) = (v_{i1}(t), v_{i2}(t), \dots, v_{iD}(t))$, $v_{id}(t) \in R$. Let $P_i(t) = (p_{i1}(t), p_{i2}(t), \dots, p_{iD}(t))$ be the best solution that particle i has obtained until iteration t , and $P_g(t) = (p_{g1}(t), p_{g2}(t), \dots, p_{gD}(t))$ be the best solution obtained from in the whole swarm at iteration t . As in continuous PSO, each particle adjusts its velocity according to previous velocity of the particle, the cognition part and the social part:

$$v_{id}(t+1) = v_{id}(t) + c_1 \cdot r_1 \cdot (p_{id}(t) - x_{id}(t)) + c_2 \cdot r_2 \cdot (p_{gd}(t) - x_{id}(t)), \quad (1)$$

where c_1 is the cognition learning factor and c_2 is the social learning factor; r_1 and r_2 are the random numbers uniformly distributed in $[0,1]$; The velocity value is constrained to the interval $[-1, 1]$ using the following function:

$$s(v_{id}) = \tanh(v_{id}), \quad (2)$$

where $s(v_{id})$ denotes the probability of bit x_{id} taking 1. Then the particle changes its bit value by

$$v_{id} = \begin{cases} 1 & \text{if } \text{rand}() \leq s(v_{id}) \\ -1 & \text{otherwise} \end{cases}, \quad (3)$$

where $\text{rand}()$ is a random number selected from a uniform distribution in $[-1,1]$. To avoid $s(v_{id})$ approaching 1 or -1, a constant V_{\max} is used to limit the range of v_{id} , that is, $v_{id} \in [-V_{\max}, V_{\max}]$.

r_1 and r_2 cannot guarantee the optimization's ergodicity entirely in phase space because they are absolutely random. Therefore, chaotic mapping with certainty, ergodicity and the stochastic property is introduced into particle swarm optimization to improve the global convergence [14]. r_1 and r_2 are chosen as follows:

$$r_i(t + 1) = 4.0 \cdot r_i(t) \cdot (1 - r_i(t)), \tag{4}$$

where $r_i(t) \in (0, 1)$, $i = 1, 2$.

3.2 Chaotic Discrete Hopfield Neural Network (CDHNN)

Let M be a discrete Hopfield neural network (DHNN) with n neurons, where each neuron is connected to all the other neurons. The input, output state and threshold of neuron i is denoted by u_i, v_i and t_i , respectively, for $i = 1, \dots, n$; $w_{i,j}$ is the interconnection weight between neurons i and j , where symmetric weights are considered, for $i, j = 1, \dots, n$. Therefore, $W = (w_{ij})$ is an $n \times n$ weight matrix and $T = (t_1, \dots, t_n)^T$ is a threshold vector. The Lyapunov function of the DHNN is given by [8] [10]:

$$E = -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n w_{ij} v_i v_j + \sum_{x=1}^n t_x v_x. \tag{5}$$

The inputs of the neurons are computed by

$$u_i(t + 1) = \sum_{j=1}^n w_{ij} v_j + t_i, \tag{6}$$

and the output states of the neurons are computed by

$$v_i(t + 1) = \text{sgn}(u_i(t + 1)), \tag{7}$$

where sgn is the signum function (or bipolar binary function) defined by

$$\text{sgn}(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ -1 & \text{otherwise} \end{cases}. \tag{8}$$

Given initial input values for $u_i(0)(t = 0)$, the DHNN, which runs in the asynchronous mode, will always converge to a stable state if its diagonal elements of weight matrix W are nonnegative (that is $w_{ii} > 0$) [8].

Wu et al. [8] gave new convergence conditions for the DHNN. They proved that the DHNN can converge to a stable state or a stable cycle by rearranging the weight matrix W to even negative diagonal element. That is: Let $d_i = \min\{\sum_{j \neq i}^n w_{ij} v_j + t_i \mid \sum_{j \neq i}^n w_{ij} v_j + t_i \neq 0, v_i \in \{-1, 1\}\}$, if $w_{ii} > -d_i, i = 1, \dots, n$, then the DHNN can converge to a stable state or a stable cycle. Furthermore, they pointed out that smaller diagonal elements may result in fewer stable states of a network, then reduce the possibility to be trapped in a state corresponding to a local minimum or poor quality solution and may approach a better solution.

In order to prevent the network to be trapped in a stable state corresponding to a local minimum, we introduce a nonlinear self-feedback term to the motion equation (6) of the DHNN as defined below [10]:

$$u_i(t+1) = \sum_{j=1}^n w_{ij}v_j + t_i + g(u_i(t) - u_i(t-1)). \quad (9)$$

The self-feedback term in Eq.(9) is switched on at $t = 1$, so that $u_i(0)$ is sufficient to initialize the system. In the Eq.(9), g is a nonlinear function as defined below:

$$g(x) = p_1 x \exp(-p_2 |x|), \quad (10)$$

where p_1 and p_2 are adjustable parameters, which determine the chaotic dynamics of the function $g(x)$. When p_2 is fixed, the network can transfer from one minimum to another in a chaotic fashion at large p_1 . Generally, chaotic transition among local minima can be obtained provided that p_1 is not too small and p_2 not too large [10].

For the nonlinear function disturbance, we apply a simulated annealing strategy to the parameter p_1 and the effect of the disturbance is gradually eliminated with the annealing process and finally the nonlinear function disturbance does not affect the evolution of DHNN, and then the neural network will converge to a stable state and be expected to get the optimal solution. The simulated annealing strategy is defined below:

$$p_1(t+1) = p_1(t) \times (1 - \beta), \quad (11)$$

where β is damping factor for p_1 .

DHNN with the nonlinear self-feedback is called chaotic DHNN (CDHNN). The n vertices bipartite subgraph problem can be mapped onto the CDHNN with n neuron representation. The output $v_i = 1$ means vertex i is assigned to one set, the $v_i = -1$ means vertex i is assigned to the other set. The energy function for the bipartition subgraph problem is given by

$$E_{BSP} = -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n d_{ij} \frac{1 - v_i v_j}{2}, \quad (12)$$

where d_{ij} is edge whose endpoints are vertex i and j . A stable state of the CDHNN corresponds to a good solution of the bipartition subgraph problem.

3.3 Hybrid Algorithm (DPSO-CDHNN)

In this section, we attempt to incorporate the CDHNN, as a local search scheme into the proposed the CDPSO algorithm (called DPSO-CDHNN) to improve its performance. The basic idea is as follows. Given the current solution S_1 , the mechanism of the DPSO leads the solution to an intermediate solution S_2 . Then a CDHNN is applied to S_2 to reach a solution S_3 . We represent a particle as

$X = (v_1, v, \dots, v_n)$, where $v_i = 1$ means that vertex i is assigned to one set, the $v_i = -1$ means vertex i is assigned to the other set. Therefore, we propose a reasonable procedure for DPSO-CDHNN as follows:

1. Initialize.
 - 1.1 Generate 1 particle using the CDHNN, and generate the other $N-1$ particles at random.
2. Repeat until a given maximal number of iterations ($MaxIter$) is achieved.
 - 2.1 Evaluate the fitness of each particle using: $F = -E_{BSP}$.
 - 2.2 Determine the best solution obtained so far by each particle.
 - 2.3 Determine the best solution obtained so far by the whole swarm.
 - 2.4 Update velocities using Eq.(1) restricted by a maximum threshold V_{\max} .
 - 2.5 Update the potential solution using Eq.(2).
 - 2.6 Improve the solution quality of each particle using the CDHNN for every fixed number (say 50) of iterations.

At the beginning of the algorithm, we incorporate the CDHNN into the random initialization of the DPSO. Therefore the proposed algorithm can guarantee to obtain a solution no worse than that of the CDHNN because the CDHNN can generate a suboptimal solution rapidly. Moreover, the diversity of the initial population can be maintained to a certain extent because the other solutions are still generated randomly. At the same time, the solutions generated randomly can share information with the suboptimal solution generated by the CDHNN. In order to allow each particle to arrive at an intermediate solution, the local search should be implemented occasionally. Therefore, in the loop of the DPSO procedure, the CDHNN is carried out once for every fixed number (say 50) of iterations.

The CDHNN is a gradient-based neural network and therefore is a powerful local search method. Further, the chaotic dynamics of the CDHNN can help the neural network escape from local minima as described in previous section. In the hybrid algorithm, the DPSO is applied to perform global exploration and the CDHNN is employed to perform locally oriented search (exploitation) for the solutions resulted by the DPSO. In this way, the hybrid algorithm has more powerful search ability than the CDHNN or DPSO alone.

When the algorithm is terminated, the best solution obtained by the whole swarm, and the corresponding fitness value are output and considered as the bipartite subgraph problem solution.

4 Simulation Results

The proposed algorithm was implemented in C on a DELL-PC (Pentium 4 2.80GHz). The parameters, $N = 10$, $c_1 = c_2 = 1.2$, $V_{\max} = 4$, $p_1 = 15$, $p_2 = 5$, $\beta = 0.02$ and $MaxIter = 200$ were used in the simulations. The CDHNN is applied to all particles once for every 50 iterations.

For comparison, SOCHOM [7], CDHNN [10], and DPSO for this problem were also implemented. All the tested data were randomly generated graphs;

Table 1. Simulation results of 15 test problems.

n	Edges	p	SOCHOM		CDHNN		DPSO		DPSO-CDHNN	
			Max.	Av.	Max.	Av.	Max.	Av.	Max.	Av.
100	1485	0.3	911	906.38	911	909.1	904	888.04	911	909.46
	2970	0.6	1673	1671.2	1673	1671.3	1669	1646.36	1673	1672.8
	3465	0.8	1916	1913.16	1916	1915.8	1910	1887.64	1916	1916
200	5970	0.3	3485	3479.42	3485	3481.1	3396	3347.88	3485	3482.06
	11940	0.6	6510	6497.54	6510	6507	6405	6373.92	6510	6507.38
	15920	0.8	8412	8402.4	8412	8411.2	8324	8293.14	8412	8411.26
300	13455	0.3	7645	7621.22	7627	7624.26	7373	7315.84	7645	7625.16
	26910	0.6	14451	14435.1	14446	14443.2	14175	14102.7	14453	14446.2
	35880	0.8	18770	18754	18757	18746.9	18521	18471.2	18770	18764.6
400	23940	0.3	13376	13347.3	13336	13335.2	12883	12782.2	13382	13368.8
	47880	0.6	25466	25436.5	25468	25467	24925	24825.3	25474	25459.6
	63840	0.8	33195	33169.5	33161	33160	32751	32656.3	33197	33181.5
500	37425	0.3	20680	20622.6	20607	20592	19839	19746	20680	20660.4
	74850	0.6	39583	39549.5	39473	39455.7	38627	38542.2	39593	39551.8
	99800	0.8	51660	51618.1	51648	51634.6	50944	50839.9	51668	51641.9

the graphs were defined in terms of two parameters, n and p . The parameter n specified the number of vertices in the graph; the parameter p , $0 < p < 1$, specified the probability that any given pair of vertices constituted an edge. The information of all tested graphs is shown in Table 1.

Table 1 shows simulation results. The maximum number of remaining edges (“Max.”) and the average number of remaining edges (“Av.”) in the solutions produced by SOCHOM, CDHNN, DPSO and the proposed algorithm respectively within 50 simulation runs. Simulation results show that the proposed algorithm combining the CDHNN and the DPSO algorithm can obtain better solutions than the CDHNN and the DPSO. We also can find that the proposed algorithm can obtain better solutions than the SOCHOM. The better average performance of the proposed algorithm shows that the hybrid algorithm is of a certain robustness for the initial solutions. The reasons why the proposed algorithm is better than other algorithms are: (1) the stochastic nature of the DPSO enables the proposed algorithm to escape from local minima, (2) the local search algorithm, CDHNN with chaotic search dynamics, also has ability of escaping from local minima, and especially, (3) the proposed algorithm combines the local search method CDHNN into the global search method DPSO; therefore the proposed algorithm has good performance of exploration and exploitation.

Table 2 shows the comparison of computation time which is the average of 50 simulations. The proposed algorithm requires more CPU computational time to perform evolutionary computations in DPSO phase and chaotic searching in CDHNN scheme. Although the hybrid algorithm requires much more

Table 2. Computation time of the algorithm of SOCHOM, CDHNN, DPSO and the proposed algorithm (seconds).

n	Edges	p	SOCHOM	CDHNN	DPSO	DPSO-CDHNN
100	1485	0.3	0.10	0.10	0.35	1.24
	2970	0.6	0.10	0.10	0.35	1.27
	3465	0.8	0.10	0.10	0.36	1.27
200	5970	0.3	0.33	0.23	1.18	4.31
	11940	0.6	0.33	0.23	1.18	4.31
	15920	0.8	0.33	0.23	1.20	4.31
300	13455	0.3	0.68	0.48	2.59	9.20
	26910	0.6	0.67	0.47	2.70	9.30
	35880	0.8	0.68	0.48	2.70	9.37
400	23940	0.3	1.16	0.96	4.40	9.37
	47880	0.6	1.17	0.97	4.51	16.07
	63840	0.8	1.18	0.98	4.51	16.05
500	37425	0.3	1.78	1.38	7.45	26.60
	74850	0.6	1.83	1.33	7.46	25.20
	99800	0.8	1.76	1.36	7.47	25.67

computational time than other algorithms, we think the moderate increase in computational time should not discourage practitioners from considering the method because it is possible to carry out the computations using high-speed computers. Therefore we can conclude that the proposed algorithm has superior ability of searching the globally optimal or near-optimum solution within reasonable computation time.

The proposed algorithm not only performs exploration by using the population based evolutionary search ability of the DPSO, but also performs exploitation by using the CDHNN. The proposed hybrid algorithm with global exploration and local exploitation search can be seen as a kind of memetic algorithm. Recently, Zhang et al. proposed an improved DPSO (named MPPSO) for the bipartite subgraph problem [15]. In their algorithm, a mutation operator and a personality factor were introduced into original DPSO to improve the original DPSO performance. Further, symmetry of solution space of the bipartite subgraph problem, as a special prior knowledge of problem domain, is considered. But there is no local search to perform exploitation in MPPSO. Wang et al. proposed a hill-shift learning algorithm of Hopfield network for the bipartite subgraph problem [16]. In their algorithm, the hill-shift learning method was introduced to help the original discrete Hopfield neural network escape from local minima. Essentially, the algorithm of Wang [16] is a local search method based on neural network, just like the SOCHOM or CDHNN. Therefore, the improved

DPSO, improved neural network-based local search and prior knowledge of the bipartite subgraph problem also can be incorporated into the framework of our PSO-based memetic algorithm to further improve the proposed algorithm for the bipartite subgraph problem.

5 Conclusions

In this paper, we incorporate a chaotic discrete Hopfield neural network (CDHNN), as a local search scheme, into the discrete particle swarm optimization (DPSO) and develop a hybrid algorithm DPSO-CDHNN for the bipartite subgraph problem. The proposed algorithm not only performs exploration by using the population based evolutionary search ability of the DPSO, but also performs exploitation by using the CDHNN. Simulation results show that the proposed algorithm has superior ability for bipartite subgraph problem. Further, the proposed algorithm can be seen as a PSO-based memetic algorithm, and therefore the proposed algorithm can be further improved by incorporating other improved method and can be applied other had combinatorial optimization problems.

References

1. Karp, R.M.: Reducibility among Combinatorial Problems. Complexity of Computer Computations, Plenum Press, Miller, R. and Thatcher, J., eds., New York, (1972) 85–104
2. Johnson, D. S., Aragon, C.R., McGeoch, L.A. and Schevon, C.: Optimization by Simulated Annealing: An Experimental Evaluation, Part 1, Graph Partitioning. Operations Research, **37**(6) (1989) 865–892
3. Lee, K.C., Funabiki, N. and Takefuji, Y.: A Parallel Improvement Algorithm for The Bipartite Subgraph Problem. IEEE Trans. Neural Networks, **3**(1) 1992 139–145
4. Takefuji, Y., Lee, K. and Aiso, H.: An Artificial Maximum Neural Network: A Winner-take-all Neuron Model Forcing The State of The System in A Solution Domain. Biological Cybernetics, **67**(3) (1992) 243–251
5. Galán-Marín, G., Mérida-Casermeyro E., and Muñoz-Pérez, J.: Modelling Competitive Hopfield Networks for The Maximum Clique Problem. Computer & Operations Research, **30**(4) (2003) 603–624
6. Galán-Marín, G. and Muñoz-Pérez, J.: Design and Analysis of Maximum Hopfield Networks. IEEE Trans. Neural Networks, **12**(2) (2001) 329–339
7. Wang, J., Tang, Z.: An Improved Optimal Competitive Hopfield Network for Bipartite Subgraph Problems. Neurocomputing, **61** (2004) 413–419
8. Wu, L.Y., Zhang X.S. and Zhang, J.L.: Application of Discrete Hopfield-type Neural Network for Max-cut Problem. Proceedings of 8th International Conference on Neural Information Processing, Fudan University Press, Shanghai (2001) 1439–1444
9. Hopfield J.J. and Tank, D.W.: Neural Networks and Physical Systems with Emergent Collective Computational Abilities. Proceedings of the National Academy of Sciences, USA, **79**(8) (1982) 2554–2558

10. Wang, J.: An Improved Discrete Hopfield Neural Network for Max-cut Problems. *Neurocomputing*, **69** (2006) 1665–1669
11. Kennedy, J. Eberhart, R.C.: Particle Swarm Optimization. Proceedings of IEEE International Conference on Neural Networks. NJ: Piscataway, (1995) 1942–1948
12. Kennedy, J., Eberhart, R.C.: A Discrete Binary Version of The Particle Swarm Algorithm. Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics. NJ: Piscataway, (1997) 4104–4109
13. Van den Bergh, F., Engelbrecht, A.P.: Cooperative Learning in Neural Network Using Particle Swarm Optimizers. *South African Computer Journal*, **26** (2000) 84–90
14. El-Galland, AI., El-Hawary, ME., Sallam, AA.: Swarming of Intelligent Particles for Solving the Nonlinear Constrained Optimization Problem. *Engineering Intelligent Systems for Electrical Engineering and Communications*, **9** (2001) 155–163
15. Zhang, D., Li, Z., Song, H. and Zhan, T.: Particle Swarm Optimization for Bipartite Subgraph Problem: A Case Study. *Lecture Notes in Computer Science*, **3612** (2005) 602–611
16. Wang, R.-L. and Okazaki, K.: A Hill-shift Learning Algorithm of Hopfield Network for Bipartite Subgraph Problem. *IEICE Trans. Fundamentals*, E89-A(1) (2006) 345–358

Convergence of a Recurrent Neural Network for Nonconvex Optimization Based on an Augmented Lagrangian Function

Xiaolin Hu and Jun Wang

Department of Mechanical and Automation Engineering
The Chinese University of Hong Kong, Shatin, New Territories, Hong Kong, China
{xlhu, jwang}@mae.cuhk.edu.hk

Abstract. In the paper, a recurrent neural network based on an augmented Lagrangian function is proposed for seeking local minima of nonconvex optimization problems with inequality constraints. First, each equilibrium point of the neural network corresponds to a Karush-Kuhn-Tucker (KKT) point of the problem. Second, by appropriately choosing a control parameter, the neural network is asymptotically stable at those local minima satisfying some mild conditions. The latter property of the neural network is ensured by the convexification capability of the augmented Lagrangian function. The proposed scheme is inspired by many existing neural networks in the literature and can be regarded as an extension or improved version of them. A simulation example is discussed to illustrate the results.

1 Introduction

In the past two decades, recurrent neural networks for solving optimization problems have attracted much attention since the early work of Hopfield and Tank [1], [2]. The theory, methodology, and applications of these neural networks have been widely investigated. Now many elegant neural networks have been proposed for solving various optimization problems and related problems. For example, Rodríguez-Vázquez *et al.* proposed the switched-capacitor neural network for nonlinear programming [3]; Wang proposed the deterministic annealing neural network [4,5] for convex optimization; Bouzerdoum and Pattison invented a neural network for quadratic programming with bound constraints [6]; Liang and Si studied a neural network for solving linear variational inequalities [7]; Xia and Wang *et al.* developed several neural networks for solving optimization problems and variational inequalities [8,9,10,11,12,13,14]. However, most of these neural networks can solve convex programs only. In contrast, little progress has been made on nonconvex optimization in the neural network community. This is mainly due to the difficulty in characterizing global optimality of nonconvex optimization problems by means of explicit equations. From the optimization context, it is known that under fairly mild conditions an optimum of the problem must be a Karush-Kuhn-Tucker (KKT) point, while the KKT points are

easier to characterize. In terms of developing neural networks for global optimization, it seems far-reaching to find global optima at the very beginning; and a more attainable goal at present is to design neural networks for seeking local optima first with the aid of KKT conditions.

In [15] a recurrent neural network based on an augmented Lagrangian function was proposed for solving nonlinear optimization problems with equality constraints. The neural network was pointed out to be (locally) asymptotically stable at KKT points that correspond to local optima under mild conditions. Recently, a neural network based on another augmented Lagrangian function was proposed in [16] for seeking local optima of nonconvex optimization problems with inequality constraints, and it was also proved asymptotically stable at local optima. Unfortunately, the equilibrium set of the neural network does not coincide with the KKT point set. In other words, the neural network may converge to some non-KKT points. In the paper, a new recurrent neural network is proposed for solving inequality constrained optimization problems based on a similar augmented Lagrangian function to that in [16]. It will be shown that each equilibrium corresponds to a KKT point, and the neural network is locally convergent to local optima under some mild conditions.

2 Problem Formulation and Preliminaries

Consider the following constrained optimization problem:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0, \end{aligned} \tag{1}$$

where $f : R^n \rightarrow R$, $g(x) = [g_1(x), \dots, g_m(x)]^T$ is an m -dimensional vector-valued function of n variables. In the paper, the functions $f, g_1(x), \dots, g_m(x)$ are assumed to be twice differentiable. If all functions $f(x)$ and $g_j(x)$ are convex over R^n , the problem is called a *convex optimization problem*; otherwise, it is called a *nonconvex optimization problem*. Equation (1) represents a wide variety of optimization problems. For example, it is well known that if a problem has equality constraints $h(x) = 0$, then this constraint can be expressed as $h(x) \leq 0$ and $-h(x) \leq 0$.

Throughout the paper, the following notations are used. R_+^n stands for the nonnegative quadrant of the n -dimensional real space R^n . $I = \{1, \dots, n\}$, $J = \{1, \dots, m\}$. If $u \in R^n$, then $u^+ = (u_1^+, u_2^+, \dots, u_n^+)^T$ where $u_i^+ = \max(u_i, 0)$; $u^2 = (u_1^2, \dots, u_n^2)^T$; $\Gamma(u) = \text{diag}(u_1, \dots, u_n)$. A square matrix $A > 0$ ($A \geq 0$) means that A is positive definite (positive semidefinite).

Definition 1. A solution x satisfying the constraints in (1) is called a *feasible solution*. A feasible solution x is said to be a *regular point* if the gradients of $g_j(x)$, $\nabla g_j(x)$, $\forall j \in J | g_j(x) = 0$, are linearly independent.

Definition 2. A point x^* is said to be a *strict minimum* of the problem in (1) if $f(x^*) < f(x)$, $\forall x \in K(x^*; \epsilon) \cap S$, where $K(x^*; \epsilon)$ is a neighborhood of x^* with the radius $\epsilon > 0$ and S is the feasible region of the problem.

It is well known [17] that if x is a local minimum as well as a regular point of problem (1), there exists a unique vector $y \in R^m$ such that the following Karush-Kuhn-Tucker (KKT) conditions hold:

$$\begin{cases} \nabla f(x) + \nabla g(x)y = 0, \\ y \geq 0, \quad g(x) \leq 0, \quad y^T g(x) = 0 \end{cases}$$

where $\nabla g(x) = (\nabla g_1(x), \dots, \nabla g_m(x))$. The above KKT condition can be equivalently put into the following projection formulation:

$$\begin{cases} \nabla f(x) + \nabla g(x)y = 0, \\ (y + \alpha g(x))^+ = y, \end{cases} \tag{2}$$

where $\alpha > 0$. In the sequel, we denote the KKT point set of (1) or the solution set of (2) by Ω^* .

Define the Lagrangian function associated with problem (1)

$$L(x, y) = f(x) + \sum_{j=1}^m y_j g_j(x). \tag{3}$$

Lemma 1 (Second-order sufficiency conditions [17]). *Suppose that x^* is a feasible and regular point to problem (1). If there exists $y^* \in R^m$, such that (x^*, y^*) is a KKT point pair and the Hessian matrix $\nabla_{xx}^2 L(x^*, y^*)$ is positive definite on the tangent subspace:*

$$M(x^*) = \{d \in R^n \mid d^T \nabla g_j(x^*) = 0, d \neq 0, \forall j \in J(x^*)\},$$

where $J(x^*)$ is defined by $J(x^*) = \{j \in J \mid y_j^* > 0\}$, then x^* is a strict minimum of problem (1).

Now consider the following augmented Lagrangian function associated with problem (1), which differs slightly from the one considered in [16].

$$L_c(x, y) = L(x, y) + \frac{c}{2} \sum_{j=1}^m (y_j g_j(x))^2, \tag{4}$$

where $L(x, y)$ is the regular Lagrangian function defined in (3) and $c > 0$ is a scalar. Let Ω^e denote the solution set of the following equations

$$\begin{cases} \nabla_x L_c(x, y) = 0, \\ (y + \alpha g(x))^+ = y, \end{cases} \tag{5}$$

where $\alpha > 0$. We have the following theorem.

Theorem 1. $\Omega^* = \Omega^e$.

Proof. It is equivalent to prove that under the following condition

$$(y + \alpha g(x))^+ = y \tag{6}$$

the first equation in (2) and the first equation in (5) are identical. The above equation gives $y_j g_j(x) = 0, \forall j \in J$. Then by writing the first equation in (5) as follows

$$\nabla_x L(x, y) + c \sum_{j \in J} y_j^2 g_j(x) \nabla g_j(x) = 0,$$

we can readily see that it is identical to $\nabla_x L(x, y) = 0$. The proof is completed.

3 Neural Network Model and Analysis

Consider a recurrent neural network with its dynamic behavior governed by

$$\frac{d}{dt} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -\alpha(\nabla f(x) + \nabla g(x)y + c\nabla g(x)\Gamma(y^2)g(x)) \\ -y + (y + \alpha g(x))^+ \end{pmatrix}, \tag{7}$$

where $\alpha > 0, c > 0$ are two parameters. Note that the first term on the right-hand-side is the expansion of $\nabla_x L_c(x, y)$. Therefore the equilibrium set of the neural network is actually Ω^e , which is equal to Ω^* as claimed in Theorem 1.

Lemma 2 ([18], p. 68). *Let $P \in R^{n \times n}$ be symmetric and $Q \in R^{n \times n}$ be symmetric and positive semidefinite. Assume that $x^T P x > 0$ for all $x \neq 0$ satisfying $x^T Q x = 0$. Then there exists a scalar c such that $P + cQ > 0$.*

Lemma 3. *Let $u = (x^T, y^T)^T$ and $u^* = ((x^*)^T, (y^*)^T)^T$ be a KKT point satisfying the second-order sufficiency conditions in Lemma 1. There exists $c > 0$ such that $\nabla_{xx}^2 L_c(u) > 0$ and $\nabla F(u) \geq 0$ on $\mathcal{N}(u^*)$, where*

$$F(u) = \begin{pmatrix} \nabla_x L_c(x, y) \\ -g(x) \end{pmatrix}$$

and $\mathcal{N}(u^*) \subseteq R^{n+m}$ is a set containing u^* as one of its interior points.

Proof. A direct reasoning gives

$$\nabla F(u) + \nabla F(u)^T = \begin{pmatrix} 2\nabla_{xx}^2 L_c(u) & 2c\nabla g(x)\Gamma(y)\Gamma(g(x)) \\ 2c\Gamma(g(x))\Gamma(y)\nabla g(x)^T & 0 \end{pmatrix},$$

and

$$\nabla_{xx}^2 L_c(u) = \nabla_{xx}^2 L(u) + c \sum_{j \in J} [y_j^2 \nabla g_j(x) \nabla g_j(x)^T + y_j^2 g_j(x) \nabla^2 g_j(x)].$$

Since u^* is a KKT point, we have

$$\nabla F(u^*) + \nabla F(u^*)^T = \begin{pmatrix} 2\nabla_{xx}^2 L_c(u^*) & 0 \\ 0 & 0 \end{pmatrix},$$

and

$$\nabla_{xx}^2 L_c(u^*) = \nabla_{xx}^2 L(u^*) + c \sum_{j \in J} (y_j^*)^2 \nabla g_j(x^*) \nabla g_j(x^*)^T.$$

Because u^* satisfies the second-order sufficiency conditions, there exists at least one $j \in J$ such that $y_j^* > 0$. By Lemma 2, there exists $c > 0$ such that $\nabla_{xx}^2 L_c(u^*) > 0$. Let $\lambda_j(u), \forall j \in J$ denotes any eigenvalue of the matrix function $\nabla_{xx}^2 L_c(u)$. Then $\lambda_j(u)$ is a continuous function in x and y satisfying $\lambda_j(u^*) > 0$ for any $j \in J$. Therefore, there exists a neighborhood of u^* , denoted by $\mathcal{N}'(u^*)$, on which $\nabla_{xx}^2 L_c(u) > 0$; moreover, u^* is an interior point of this neighborhood. Similarly we can prove that there exists $\mathcal{N}(u^*) \subseteq \mathcal{N}'(u^*)$, on which $\nabla F(u) \geq 0$, where u^* is an interior point of $\mathcal{N}(u^*)$. The proof is completed.

Lemma 4. *For any initial point $(x(t_0)^T, y(t_0)^T)^T \in R^{n+m}$ there exists a unique continuous solution $(x(t)^T, y(t)^T)^T$ for (7). Moreover, $y(t) \geq 0$ if $y(t_0) \geq 0$.*

Proof. Similar to the analysis of Lemma 3 in [12].

Theorem 2. *Let $u^* = ((x^*)^T, (y^*)^T)^T$ be a KKT point of problem in (1) satisfying the second-order sufficiency conditions in Lemma 4. There exists $c > 0$ such that the neural network in (7) is asymptotically stable at u^* , where x^* is a strict local minimum of the problem.*

Proof. Choose $y(t_0) \geq 0$, then from Lemma 4, $y(t) \geq 0, \forall t \geq t_0$. Consider the equivalent form of (7)

$$\frac{du}{dt} = -u + P_\Omega(u - \alpha F(u)),$$

where $F(u)$ is defined in Lemma 3, $\Omega = R^n \times R_+^n$ and $P_\Omega(\cdot)$ is a projection operator defined as $P_\Omega(u) = \arg \min_{v \in \Omega} \|u - v\|$. Define the Lyapunov function

$$V(u) = -F(u)^T r(u) - \frac{1}{2\alpha} \|r(u)\|^2 + \frac{1}{2\alpha} \|u - u^*\|^2$$

where $r(u) = P_\Omega(u - \alpha F(u)) - u$ and u^* is a KKT point satisfying the second-order sufficiency conditions. Lemma 1 indicates that x^* is a strict local minimum of the problem. As y^* is uniquely determined for x^* , there exists $\epsilon > 0$ such that u^* is a unique KKT point in $D(u^*; \epsilon) = \{u \in R^{n+m} \mid \|u - u^*\| < \epsilon\}$. Similar to the proof of [12, Theorem 1], we can obtain

$$V(u) \geq \frac{1}{2\alpha} \|u - u^*\|^2 \quad \forall u \in R^{n+m}$$

and

$$\begin{aligned} dV(u(t))/dt &\leq -F(u)^T(u - u^*) - r(u)^T \nabla F(u)r(u) \\ &\leq -(F(u) - F(u^*))^T(u - u^*) - r(u)^T \nabla F(u)r(u) \quad \forall t \geq t_0. \end{aligned}$$

Select a convex subset $\Omega_c \subseteq \mathcal{N}(u^*)$ which contains u^* as a unique KKT point in it, where $\mathcal{N}(u^*)$ is defined in Lemma 3. By Lemma 3 there exists $c > 0$ such that $\nabla F(u) \geq 0$ on Ω_c , which implies $F(u)$ is monotone on Ω_c . Then $dV(u)/dt \leq 0$ on Ω_c , and the neural network in (7) is stable at u^* .

According to the Lyapunov theorem [19], to prove the asymptotic stability of the neural network, it is only needed to show that $dV(u)/dt = 0$ if and only if $u = u^*$ in Ω_c , or show that $dV(u)/dt = 0$ if and only if $du/dt = 0$ since in Ω_c , $du/dt = 0$ is equivalent to $u = u^*$. Consider a point $u \in \Omega_c$. Clearly, $du/dt = 0$ implies $dV/dt = 0$. Let u be a solution of $dV(u)/dt = 0$. It follows that

$$r(u)^T \nabla F(u) r(u) = \alpha^2 \nabla_x L_c(x, y)^T \nabla_{xx}^2 L_c(x, y) \nabla_x L_c(x, y) = 0,$$

and $dx/dt = -\alpha \nabla_x L_c(x, y) = 0$ since $\nabla_{xx}^2 L_c(x, y) > 0$ on Ω_c . In view that $\nabla F(u) \geq 0$ on Ω_c , we have

$$dV/dt \leq -(F(u^*) - F(u))^T (u^* - u) = - \int_0^1 (x^* - x)^T \nabla_{xx}^2 L_c(x_s, y_s) (x^* - x) ds$$

where $x_s = x + s(x^* - x)$ and $y_s = y + s(y^* - y)$ with $0 \leq s \leq 1$. Because Ω_c is convex, $(x_s^T, y_s^T)^T$ is in Ω_c , and $\nabla_{xx}^2 L_c(x_s, y_s) > 0$ for $0 \leq s \leq 1$. Then $dV(u)/dt = 0$ implies $x = x^*$. With this implication we can deduce

$$dV(u)/dt = 0 \Rightarrow F(u)^T (u - u^*) = 0 \Rightarrow g(x)^T (y - y^*) = 0 \Rightarrow g(x)^T y = 0.$$

By considering $g(x) = g(x^*) \leq 0$ and $y \geq 0$, we have $dy/dt = (y + \alpha g(x))^+ - y = 0$. Thus, u is a solution of $du/dt = 0$. In summary, $dV(u)/dt = 0$ if and only if $du/dt = 0$. Hence, the neural network is asymptotically stable at u^* . The proof is completed.

4 Comparisons with Other Neural Networks

In 1992, Zhang and Constantinides proposed a neural network based on the augmented Lagrangian function for seeking local minima of the following equality constrained optimization problem [15]:

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & h(x) = 0, \end{aligned}$$

where $f : R^n \rightarrow R, h : R^n \rightarrow R^m$ and both f and h are assumed twice differentiable. The dynamic equation of the network is as follows

$$\frac{d}{dt} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -(\nabla f(x) + \nabla h(x)y + c \nabla h(x)h(x)) \\ h(x) \end{pmatrix}, \tag{8}$$

where $c > 0$ is a control parameter. Under the second-order sufficiency conditions, the neural network can be shown convergent to local minima with appropriate choice of c . The disadvantage of the neural network lies in that it handles equality constraints only. Though in theory inequality constraints can be converted to equality constraints by introducing slack variables, the dimension of the neural network will inevitably increase, which is usually not deemed optimal in terms of model complexity. In this sense, the proposed neural network in the present paper can be regarded as an extension of the Lagrange network.

An alternative extension of the neural network in [15] for handling inequality constraints in (II) directly can be found in [16], and its dynamic system is as follows:

$$\frac{d}{dt} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -(\nabla f(x) + \nabla g(x)y^2 + c\nabla g(x)\Gamma(y^2)g(x)) \\ 2\Gamma(y)g(x) \end{pmatrix}. \tag{9}$$

The local convergence of the neural network to its equilibrium set, denoted by $\hat{\Omega}^e$, was proved by using the linearization techniques, and moreover, $\Omega^* \subset \hat{\Omega}^e$. However, it is clear that $\hat{\Omega}^e \neq \Omega^*$. For example, any critical point x of the objective function, which makes $\nabla f(x) = 0$, and $y = 0$ constitute an equilibrium point of (9), but in rare cases such an equilibrium corresponds to a KKT point.

If we set the control parameter $c = 0$ in (7), the neural network becomes

$$\frac{d}{dt} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} -(\nabla f(x) + \nabla g(x)y) \\ -y + (y + g(x))^+ \end{pmatrix}, \tag{10}$$

which is a special case of the neural network proposed in [12] for solving convex version of (II). Clearly, its equilibrium set coincides with Ω^* . However, it will be seen in the next section that this neural network cannot be guaranteed the local convergence to local minima of nonconvex optimization problems. A scheme taking advantages of the neural network in (10) for seeking local minima of nonconvex problems is found in [20]. The idea is to transform the problem in (II) into the following equivalent one, of course under some appropriate assumptions,

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & (g(x) + b)^p \leq b^p, \end{aligned} \tag{11}$$

where $b \geq 0$ is a vector and $p \geq 1$ is a scalar, and then use the neural network in (10) to solve the new problem. Some local convergence capability can be ensured by selecting sufficiently large p . The weak point of this approach is that large p values will introduce great nonlinearity to the system and cause some numerical problems.

5 Illustrative Example

Consider the following optimization problem

$$\begin{aligned} \min \quad & f(x) = 5 - (x_1^2 + x_2^2)/2, \\ \text{s.t.} \quad & g_1(x) = -x_1^2 + x_2 - 1 \leq 0, \\ & g_2(x) = x_1^4 - x_2 \leq 0. \end{aligned}$$

As both $f(x)$ and $g_1(x)$ are concave, the problem is a nonconvex optimization problem. Fig. 1 shows the contour of the objective function and the solutions to $g_1(x) = 0$ and $g_2(x) = 0$ on the $x_1 - x_2$ plane. The feasible region is the nonconvex area enclosed by the bold curves. Simple calculations yield

$$\nabla_{xx}^2 L(x, y) = \begin{pmatrix} -2y_1 + 12x_1^2y_2 - 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

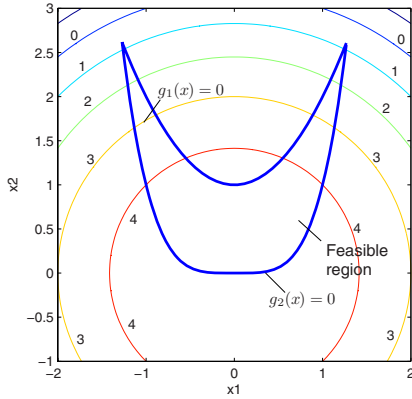
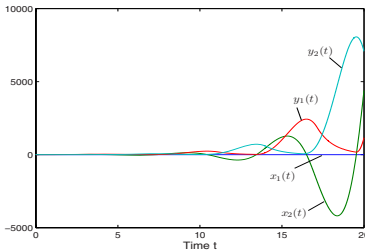
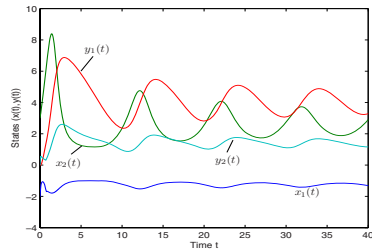


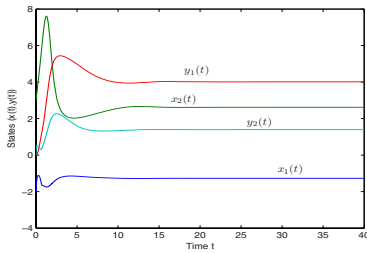
Fig. 1. Contour of the objective function and the feasible region



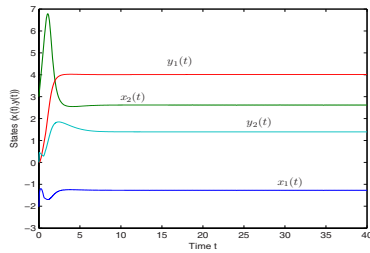
(a) $c = 0$



(b) $c = 0.1$



(c) $c = 0.2$



(d) $c = 0.5$

Fig. 2. Transient behavior the neural network in (7) with different values of c

Evidently, $\nabla_{xx}^2 L(x, y)$ is not positive definite over the entire real space, and the neural network in (10) can not be applied to solve the problem. Now we check if the neural network in (7) can be used to search for the KKT points. There are four KKT points associated with the problem: $u_1^* = (-1.272, 2.618, 4.013, 1.395)^T$, $u_2^* = (1.272, 2.618, 4.013, 1.395)^T$, $u_3^* = (0, 0, 0, 0)^T$, $u_4^* = (0, 1, 1, 0)^T$, but only the first two correspond to local minima. Moreover, it is verified that at either

u_1^* or u_2^* , $J(x^*)$ defined in Lemma 1 is equal to $\{1, 2\}$, and $\nabla g_1(x^*)$, $\nabla g_2(x^*)$ are linearly independent, which indicates $M(x^*) = \emptyset$. So the second-order sufficiency conditions holds trivially at either point. According to Theorem 2, the neural network in (7) can be made asymptotically stable at u_1^* and u_2^* by choosing appropriate $c > 0$. Fig. 2 displays the state trajectories of the neural network with different values of c started from the same initial point $(-2, 3, 0, 0)^T$. When $c = 0$, the neural network reduces to the neural network in (10). It is seen from Fig. 2(a) that some state trajectories is divergent to infinity. When $c = 0.1$, the neural network is not convergent, either, as shown in Fig. 2(b). However, when $c \geq 0.2$, in Figs. 2(c) and 2(d) we observe that the trajectories converge to u_1^* asymptotically.

6 Concluding Remarks

In the paper, a recurrent neural network is proposed for seeking local optima of general nonconvex optimization problems by means of solving Karush-Kuhn-Tucker (KKT) equations. In the proposed scheme, there is no need to introduce slack variables to convert inequality constraints into equality constraints. Moreover, a nice property of the proposed neural network is that its equilibria are in correspondence with the KKT points. Another nice property lies in that by choosing an appropriate control parameter the neural network can be made asymptotically stable at those KKT points associated with local optima under some standard assumptions in the optimization context, although locally. This can be regarded as a meaningful progress in paving the way for designing neural networks for completely solving nonconvex optimization problems, by considering that many existing neural network models are unstable at such KKT points. A numerical example is discussed to illustrate the performance of the proposed neural network.

Acknowledgement

This work was supported by the Hong Kong Research Grants Council under Grant CUHK4165/03E.

References

1. Hopfield, J.J., Tank, D.W.: ‘Neural’ Computation of Decisions in Optimization Problems. *Biol. Cybern.* **52** (1985) 141–152
2. Tank, D.W., Hopfield, J.J.: Simple Neural Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit. *IEEE Trans. Circuits Syst. II* **33** (1986) 533–541
3. Rodríguez-Vázquez, A., Domínguez-Castro, R., Rueda, A., Huertas, J. L., Sánchez-Sinencio, E.: Nonlinear Switched-Capacitor Neural Networks for Optimization Problems. *IEEE Trans. Circuits Syst.* **37** (1990) 384–397

4. Wang, J.: Analysis and Design of a Recurrent Neural Network for Linear Programming. *IEEE Trans. Circuits Syst. I* **40** (1993) 613–618
5. Wang, J.: A Deterministic Annealing Neural Network for Convex Programming. *Neural Networks* **7** (1994) 629–641
6. Bouzerdoum, A., Pattison, T.R.: Neural Network for Quadratic Optimization with Bound Constraints. *IEEE Trans. Neural Networks* **4** (1993) 293–304
7. Liang, X., Si, J.: Global Exponential Stability of Neural Networks with Globally Lipschitz Continuous Activations and Its Application to Linear Variational Inequality Problem. *IEEE Trans. Neural Networks* **12** (2001) 349–359
8. Xia, Y., Wang, J.: A Recurrent Neural Network for Solving Linear Projection Equations. *Neural Networks* **13** (2000) 337–350
9. Xia, Y., Wang, J.: On the Stability of Globally Projected Dynamical Systems. *Journal of Optimization Theory and Applications* **106** (2000) 129–150
10. Xia, Y., Leung, H., Wang, J.: A Projection Neural Network and Its Application to Constrained Optimization Problems. *IEEE Trans. Circuits Syst. I* **49** (2002) 447–458
11. Xia, Y., Feng, G., Wang, J.: A Recurrent Neural Network with Exponential Convergence for Solving Convex Quadratic Program and Linear Piecewise Equations. *Neural Networks* **17** (2004) 1003–1015
12. Xia, Y., Wang, J.: A Recurrent Neural Network for Nonlinear Convex Optimization Subject to Nonlinear Inequality Constraints. *IEEE Trans. Circuits Syst. I* **51** (2004) 1385–1394
13. Xia, Y., Wang, J.: Recurrent Neural Networks for Solving Nonlinear Convex Programs with Linear Constraints. *IEEE Trans. Neural Networks* **16** (2005) 379–386
14. Hu, X., Wang, J.: Solving Pseudomonotone Variational Inequalities and Pseudomonotone Convex Optimization Problems Using the Projection Neural Network. *IEEE Trans. Neural Networks* **17** (2006) 1487–1499
15. Zhang, S., Constantinides, A.G.: Lagrange Programming Neural Networks. *IEEE Trans. Circuits Syst. II* **39** (1992) 441–452
16. Huang, Y.: Lagrange-Type Neural Networks for Nonlinear Programming Problems with Inequality Constraints. In: *Proc. 44th IEEE Conference on Decision and Control and the European Control Conference*. Seville, Spain, Dec. 12–15 (2005) 4129–4133
17. Luenberger, D.G.: *Linear and Nonlinear Programming*. 2nd Edition. Addison-Wesley, Reading, Massachusetts (1984)
18. Bertsekas, D.P.: *Constrained Optimization and Lagrange Multiplier Methods*. Academic Press, New York (1982)
19. Slotine, J.J., Li, W.: *Applied Nonlinear Control*. Prentice Hall, Englewood Cliffs, NJ (1991)
20. Hu, X., Wang, J.: A Recurrent Neural Network for Solving Nonconvex Optimization Problems. In: *Proc. 2006 IEEE International Joint Conference on Neural Networks*. Vancouver, Canada, July 16–21 (2006) 8955–8961

Multi-objective Topology Optimization of Structures Using NN-OC Algorithms*

Xinyu Shao, Zhimin Chen, Mingang Fu, and Liang Gao

Department of Industrial & Manufacturing System Engineering, Huazhong Univ. of Sci. &
Tech. Wuhan, 430074, P.R. China
gaoliang@mail.hust.edu.cn

Abstract. Topology optimization problem, which involves many design variables, is commonly solved by finite element method, a method must recalculate structure-stiffness matrix each time of analysis. OC method is a good way to solve topology optimization problem, nevertheless, it can not solve multiobjective topology optimization problems. This paper introduces an effective solution to Multi-objective topology optimization problems by using Neural Network algorithms to improve the traditional OC method. Specifically, in each iteration, calculate the new neural network link weight vector by using the previous link weight vector in the last iteration and the compliance vector in the last time of optimization, then work out the impact factor of each optimization objective on the overall objective of the optimization in order to determine the optimal direction of each design variable.

1 Introduction

Topology optimization aims at obtaining overall material distribution in a certain solution domain in the initial stage of product manufacturing process. It originates from the idea of minimizing material cost and producing efficient mechanisms. Truss theory (Michell 1904) is considered as the primal thought of topology optimization. Bendsoe and Kikuchi (1988) introduced the numerical approach to topology optimization method for continuum structures, which contributed to the rapid increase of the popularity of topology optimization methods in structural design. In 1989, the so called power-law or Solid Isotropic Material with Penalization model (SIMP) approach was proposed by Bendsoe. An alternative was presented by Zhou and Rozvany (1991). The SIMP approach has now become the most popular way to introduce the notion of topology into structural analysis of topology optimization problem. Later, Bendsoe (1995) gave an overview of the topology optimization method. Currently, with the rapid development and maturation of topology theory, it is being applied to the various industrial problems such as the design of materials and mechanisms.

* This paper is supported by the National Basic Research Program of China (973 Program), No. 2004CB719405 and the National Natural Science Foundation of China, No. 50305008.

SIMP, the most popular and competitive density-stiffness interpolation scheme of density methods, establishes certain nonlinear relation between modulus of elasticity and the relative density of the elements by introducing penalty factor, which penalize the intermediate densities when the design variables are bounded above and below by 1 and 0 in order to make intermediate densities approach 0 or 1 as close as possible. As the penalty factor increase, the intermediate densities are penalized progressively; the intermediate density material is structurally less effective. Thus, continuous variable topology optimization model can approach previous 0-1 discrete variable optimization model to a large extent. As a result, the intermediate density element has a very small corresponding modulus of elasticity, which results in trivial influence to structure-stiffness matrix.

Optimality Criteria (OC) is a kind of optimality design principles based on Tuhn-tucker optimization qualification. By forming Lagrange function, OC method introduces Lagrangian multiplier, transform nonlinear problem with restriction into linear problem without restriction. The development of OC method went through two processes: Continuum-based optimality criteria (COC) and Discretized continuum-based optimality criteria (DCOC).

OC method is a good way to solve topology optimization problem, nevertheless, each iteration of it requires a huge amount of calculation, which leads to the fact that iterative times directly determine computation speed. Due to the intensive computation, the application of topology optimization is confined in narrow fields. Therefore, reducing the iterative times of OC method shows its significance as a part of relative research. It is necessary to find a faster and more computationally efficient method. Here, this paper is primarily intended to focus on how to accelerate OC method of topology optimization.

There are already some methods to solve multi-objective problems such as J. Aguilar Madeira's genetic algorithms with chromosome repairing method and Zhen Luo's compromise programming method. It cost the two methods a long time to get the optimal solution. In this paper, we introduce a new algorithm to reduce the computational time effectively.

In each iteration, the traditional Optimality Criteria method is incapable of dealing with the impact of multi-objectives on the overall objective and each design variable. At the same time, due to the change of the structure in the optimization process, the impact of different objectives on the overall and each design variable changes. Therefore, it is difficult for us to solve multi-objectives problems with traditional OC method. This paper introduces an effective solution to multi-objective topology optimization problems by using Neural Network algorithms to improve the traditional OC method. Specifically, in each iteration, calculate the new neural network link weight vector by using the previous link weight vector in the last iteration and the compliance vector in the last time of optimization, then work out the impact factor of each optimization objective on the overall objective of the optimization in order to determine the optimal direction of each design variable.

2 Topology Optimization Formulation

2.1 Single-Objective Topology Optimizaion Problem and It Solving Model

A single-objective topology optimization problem based on the SIMP approach, whose objective is to minimize compliance can be expressed as:

$$\begin{aligned} \min : \quad & c(x) = U^T K U = \sum_{e=1}^N (x_e)^p u_e^T k_e u_e \quad (1) \\ \text{subject :} \quad & \frac{V(x)}{V_0} = f \\ & : KU = F \\ & : 0 < x_{\min} \leq x \leq 1 \end{aligned}$$

Where $c(x)$ is the compliance of the structure, the global displacement vectors and force vectors are denoted by U and F respectively, K is the global stiffness matrix of the structure, x is the vector of design variables, u_e and k_e are the element displacement vector and stiffness matrix, respectively, N is the total number of discrete elements, x_{\min} is the lower bound constraint of relative densities, p is the penalty factor, $V(x)$ is the material volume, V_0 is the design domain volume, and f is the ratio of design domain volume to material volume.

$$E^p(x_j) = E^{\min} + x_j^p (E^0 - E^{\min}) \quad (2)$$

$$[K] = \sum_{j=1}^n E^p(x_j) [K_j] = \sum_{j=1}^n [E^{\min} + x_j^p (E^0 - E^{\min})] [K_j] \quad (3)$$

Expressed as formula.2 and formula.3, p is the penalty factor in the model. It is introduced to make aimed design approach the 0/1 discrete design gradually when the design variables are bounded above and below by 1 and 0 by increasing p gradually to penalize intermediate design variables progressively. It is required that $p > 2$ for the sake of compressing intermediate material effectively. E^p is the modulus of elasticity after interpolation, E^0 and E^{\min} are the modulus of elasticity of solid part and void part, respectively, $\Delta E = E^0 - E^{\min}$, $E^{\min} = E^0 / 1000$. $x_j (j = 1, 2, \dots, n)$ is the design variable of element j . $[K]$ is the stiffness matrix after interpolation, $[K_j]$ is the stiffness matrix of the element solid material j .

SIMP's corresponding compliance function and sensitivity form can be written as:

$$C(x) = \sum_{j=1}^n [E^{\min} + x_j^p (E^0 - E^{\min})] \{U_j\}^T [K_j] \{U_j\} \quad (4)$$

$$C'(x) = -\sum_{j=1}^n p x_j^{p-1} (E^0 - E^{\min}) \{U_j\}^T [K_j] \{U_j\} \quad (5)$$

Where, $[K_j]$ is the stiffness of element j , $\{U\}$ is the displacement vectors of the structure. x is the design variable, in order to avoid singularity of the overall stiffness matrix, $x_{\min} = 0.001$. n is the number of elements, C is the compliance of structure and C' denotes the sensitivity of design variable with regard to compliance.

According to the site of optimal node, the equation solving SIMP density-stiffness interpolation scheme based on Kuhn-Tucker optimization qualification can be described as:

$$\frac{p(x_j^k)^{p-1} \{U_j\}^T [K_j] \{U_j\} \Delta E}{\Lambda V_j} = \frac{x_j^{k+1}}{x_j^k} \tag{6}$$

$$x_j^{k+1} = x_j^k \frac{p(x_j^k)^{p-1} \{U_j\}^T [K_j] \{U_j\} \Delta E}{\Lambda V_j} \tag{7}$$

Make

$$D_j^k = \frac{p(x_j^k)^{p-1} \{U_j\}^T [K_j] \{U_j\} \Delta E}{\Lambda V_j} \tag{8}$$

$$D_j^k = -\frac{\partial C}{\partial x_j} \times \frac{1}{\Lambda V_j} \tag{9}$$

Thus, the above iteration formula can be described as:

$$x_j^{k+1} = D_j^k x_j^k \tag{10}$$

Λ is the Lagrangian multiplier, it is variable.

2.2 Multi-objective Topology Optimization of Structures

When the structure needed to be topology optimized is influenced by variety of forces from F_1 to F_i , the stress distribution of topology structure is impacted by these forces at the same time. We assume that each force F_i has an impact factor W_i on the topology structure, the structural compliance of each force F_i is C_i , and the structural sensitivity of each force F_i is C_i' .

The overall structural compliance can be expressed as:

$$C = \sum W_i C_i \tag{11}$$

The overall structural sensitivity can be expressed as:

$$C' = \sum W_i C_i' \tag{12}$$

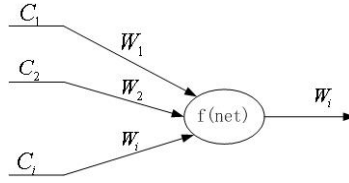


Fig. 1. M-P model neuronal structure

Above approach can be seen as a simple continual M-P model, the impact factor on the overall structure of each force can be seen as neurons’ link weights. Please refer to Fig. 1 for specific neuronal structure.

How to determine the impact factor on the overall structure of each force or neurons’ link weights will directly determine the algorithm’s effect and computing speed.

In this paper, an alternating design method between training neural network and topology optimization is constructed. Its specific method is as follows:

- 1) Calculate each force’s structural compliance C_i^k and sensitivity C_i^k to the overall structure by the formula. 4 and formula. 5, in which k is the number of optimization iterations.
- 2) Adopt Hebb learning rule to amend impact factor vector W_i . Specific formula is as follows:

$$net = W_i^k C_i^k \tag{13}$$

$$f(net) = \frac{2}{1 + e^{-\lambda \cdot net}} - 1 \tag{14}$$

$$W_i^{k+1} = W_i^k + f(net)C_i^k \tag{15}$$

- 3) Calculate the overall structure’s compliance and sensitivity according to the impact factor vector after amendment and each force’s compliance and sensitivity to the overall structure. Specific formula is as follows:

$$C^{k+1} = \sum W_i^{k+1} C_i^k \tag{16}$$

$$C_i^{k+1} = \sum W_i^{k+1} C_i^k \tag{17}$$

- 4) Calculate new design variable according to the overall structure’s compliance and sensitivity. Specific formula is as follows:

$$\frac{p(x_j^k)^{p-1} \{U_j\}^T [K_j] \{U_j\} \Delta E}{\Delta V_j} = \frac{x_j^{k+1}}{x_j^k} \tag{18}$$

$$x_j^{k+1} = x_j^k \frac{p(x_j^k)^{p-1} \{U_j\}^T [K_j] \{U_j\} \Delta E}{\Delta V_j} \tag{19}$$

Make

$$D_j^k = \frac{p(x_j^k)^{p-1} \{U_j\}^T [K_j] \{U_j\} \Delta E}{\Delta V_j} \tag{20}$$

$$D_j^k = -\frac{\partial C}{\partial x_j} \times \frac{1}{\Delta V_j} = -C^{k+1} \times \frac{1}{\Delta V_j} \tag{21}$$

The determine of Lagrangian multiplier Λ refers to Section 2.1.

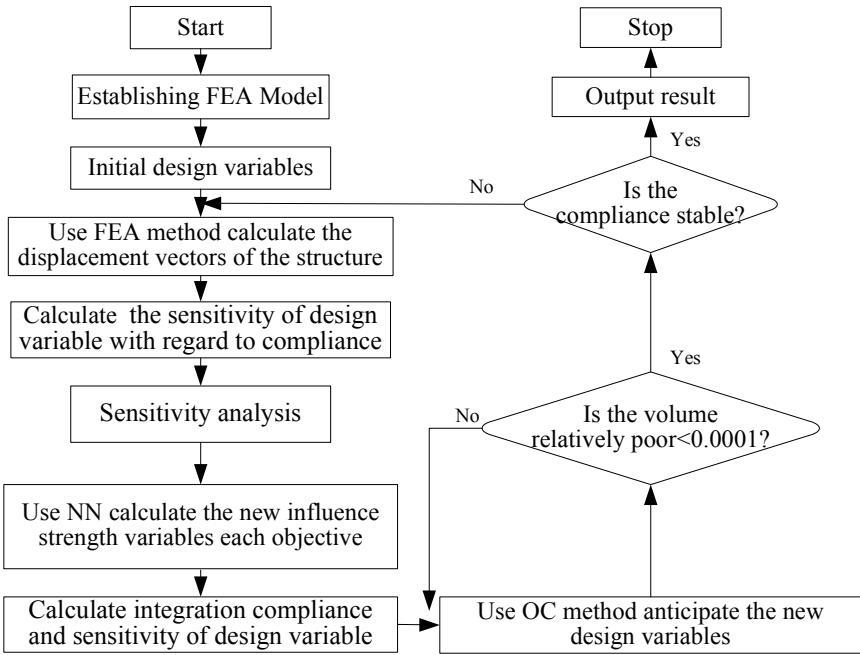


Fig. 2. NN-OC method's Computational model

3 Computational Model of NN-OC

It can be seen from Fig. 2 that there is much difference between the improved OC method and traditional OC method in respect of solving process. The improved OC method re-evaluates each objective's impact on the overall structural compliance and sensitivity, then works out a comprehensive compliance and a comprehensive sensitivity by which variables' changed directions are calculated, that is what the new method different from the old.

4 Case Studies

4.1 The Single Beam Model with Multi-objective Optimization

The design domain of the problem needed to be solved is shown as fig (), where the load case $F1 = F2 = F3 = F4 = 3.5KN$. The modulus of elasticity $E=200GPa$, the Poisson’s rate $\nu=0.3$, the volume ratio $f=0.5$. Use the SIMP model, penalty factor $p=3.0$, the initial number of elements is 1600. The mobile limit $m=0.3$. The force structure is shown as Fig. 3.

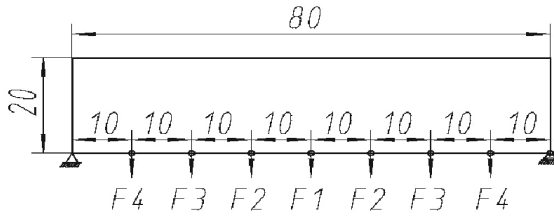


Fig. 3. The design domain of the single beam model with Multi-objective optimization

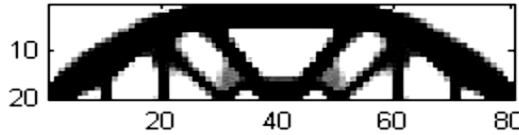


Fig. 4. The optimized result

The optimized result is shown as Fig. 4, for this moment, the comprehensive structural compliance $c=0.4995$.

4.2 Simple Bridge Model with Multi-objective Optimization

The design domain of the problem needed to be solved is shown as Fig. 6, where the load case $F1 = F2 = F3 = F4 = F5 = F6 = F7 = 3.5KN$. The modulus of elasticity $E=200GPa$, the Poisson’s rate $\nu=0.3$, the volume ratio $f=0.3$. Use the SIMP model, penalty factor $p=3.0$, the initial number of elements is 3600. The mobile limit $m=0.3$. the force structure is shown as Fig. 5.

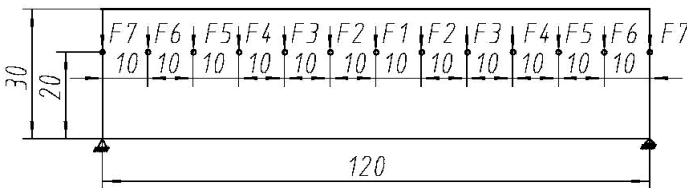


Fig. 5. The design domain simple bridge model with multi-objectives optimized

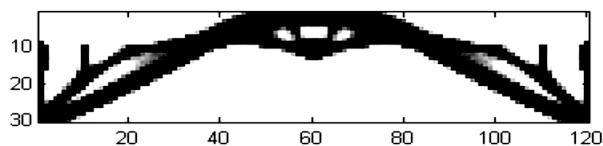


Fig. 6. The optimized result

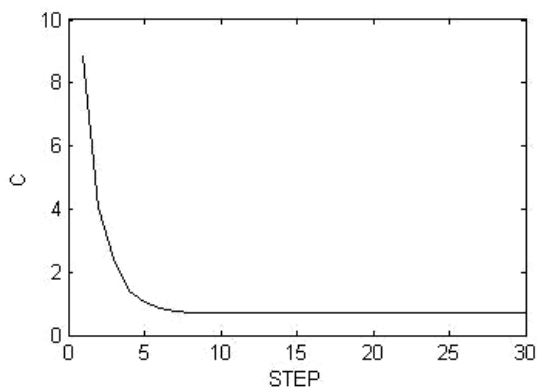


Fig. 7. The movement of compliance's value as steps of computation increases

The optimized result is shown as Fig. 6, for this moment, the comprehensive structural compliance $c=0.6997$.

It can be seen from Fig. 7, NN-OC optimization method can work out the optimal solution of multi-objectives topology optimization problem with a fast speed and a high stability.

5 Conclusions

This paper successfully realizes solving multi-objectives topology optimization problem by improving traditional OC method with neural network. Its design method owns following characteristics:

- 1) In each iteration, calculate the new neural network link weight vector by using the previous link weight vector in the last iteration and the compliance vector in the last time of optimization. That is alternation between neural network training and structural optimization process.

- 2) With the new link weight vector, work out the impact factor of each optimization objective on the overall objective of the optimization in order to determine the comprehensive optimal direction. Although in the started several times this direction may not change towards the optimal direction, as the neural network structural optimization and topology structural optimization process, this direction will become closer and closer to the optimal solution.

It is a completely new improvement to traditional OC method. Experimental results have demonstrated that such new method is very effective, it improves traditional OC method to endow it with the ability to solve multi-objectives and faster computation speed (normally work out the optimal solution before 10 times of iterations). This will be a breakthrough in the application of OC method to solve multi-objectives. But the current speed can still be accelerated by improving neural network algorithm, for this consideration, it is very necessary to continue the study about how to improve OC method by adding method of neural networking. At the same time, the application of this new method to 3D structure's multi-objectives topology optimization is urged.

References

1. Zhou, M., Rozvany, G I N.: The COC algorithm, Part II: topological, geometry and generalized shape optimization. *Computer Methods in Applied Mechanics and Engineering* **89** (1991) 197-224
2. Zhou, M., Rozvany, G I N.: DCOC: an optimality criteria method for large systems, Part I: Theory. *Structural optimization*, Springer-Verlag **5** (1995) 12-25
3. Zhou, M., Rozvany, G I N.: DCOC: an optimality criteria method for large systems, Part II: Algorithm. *Structural optimization*, Springer-Verlag **6** (1995) 250-262
4. Sigmund, O.: A 99 line topology optimization code written in Matlab, *Struct Multidisc Optim*, Springer-Verlag **21** (2001) 120-127
5. Aguilar Madeira, J., Rodrigues, H.C., Pina, H.: Multiobjective topology optimization of structures using genetic algorithms with chromosome repairing. *Struct Multidisc Optim*, Springer-Verlag **32** (2006) 31-39
6. Luo, Z., Chen, L.P., Yang, J.Z., Zhang, Y.Q.: Multiple stiffness topology optimizations of continuum structures. *Int J Adv Manuf Technol*, Springer-Verlag London Limited **30** (2006) 203-214
7. Chen, T.Y., Shieh, C.C.: Fuzzy multiobjective topology optimization. *Computers and Structures*, Elsevier Science Ltd, Exeter, United Kingdom **78** (2000) 459-466
8. Kim, T.S., Kim, Y.Y.: Multiobjective topology optimization of a beam under torsion and distortion. *AIAA Journal* **40** (2002) 376-381
9. Mohandans, S.U., Phelps, T.A., Ragsdell, K.M.: Structural optimization using a fuzzy goal programming approach. *Computers and Structures* **37** (1990) 1-8
10. Min, S., Nishiwaki, S., Kikuchi, N.: Unified topology design of static and vibrating structures using multiobjective optimization. *Computers and Structures* **75** (2000) 93-116
11. Fujii, H., Ito, H., Aihara, K., Ichinose, N., Tsukada, M.: Dynamical cell assembly hypothesis - Theoretical possibility of spatio-temporal coding in the cortex. *Neural Networks*, Pergamon Press Inc, Tarrytown, NY, United States **9** (1996) 1303-1350
12. Brown, T. H., Kairiss, E.W., Keenan, C. L.: Hebbian synapses: Biophysical mechanisms and algorithms. *Annual Review of Neuroscience* **13** (1990) 475-511

A Hybrid Particle Swarm Algorithm for the Structure and Parameters Optimization of Feed-Forward Neural Network

Tang Xian-Lun, Li Yin-Guo, and Zhuang Ling

College of Automation, Chongqing University of Posts and Telecommunications,
Chongqing, 400065, P.R. China
tangxianlun@hotmail.com

Abstract. A novel and efficient method combining chaos particle swarm optimization (CPSO) and discrete particle swarm optimization (DPSO) is proposed to optimize the topology and connection weights of multilayer feed-forward artificial neural network (ANN) simultaneously. In the proposed algorithm, the topology of neural network is optimized by DPSO and connection weights are trained by CPSO to search the global optimal ANN structure and connectivity. The proposed algorithm is successfully applied to fault diagnosis, able to eliminate some bad effects on the diagnosis capacity of network introduced by redundant structure of ANN. Compared with genetic algorithm (GA), the proposed method shows its superiority on convergence property and efficiency in training ANN. It is validated by the good diagnosis results of experiments.

1 Introduction

Artificial neural network (ANN) has been widely used in various fields, such as pattern recognition, intelligent control, fault diagnosis, optimization etc.. However, the optimal structure of neural network is still difficult to determine now. An oversimplified network architecture might hamper the convergence of the network. On the other hand, an oversized network would lead to overfitting, and thus poor generalization performance. So, it's highly desirable to design a methodology capable to find the appropriate network architecture and connectivity simultaneously.

Genetic algorithm (GA) has been applied to train ANN to obtain appropriate topology and connection weights. Nevertheless, complex evolution operators of GA such as crossover and mutation make the training time increasing according to exponential series following the scale and complexity of problem [1].

Particle Swarm Optimization (PSO) is an evolutionary algorithm based on swarm intelligence. Based on a set of potential solutions named population, it searches the optimal problem solution through cooperation and competition among the individuals of population [2], PSO method often provides solutions to many complex optimization tasks such as neural network training, system identification and engineering design [3]-[4]. However, PSO has its disadvantages when facing a complex task with many local optima. In this paper, an improved PSO algorithm (CPSO) is proposed, which combines

PSO and chaotic search for the weight training of ANN. CPSO improves convergence speed and the abilities of searching for the global optima. Most versions of PSO, such as CPSO, have operated in continuous and real-number space, however, many optimization problems are set in a space featuring discrete, so, James Kennedy and R. C. Eberhart proposed an algorithm operates on discrete binary variables (DPSO) [5].

In this article, CPSO is used for the weight training of ANN, and DPSO is applied to find the appropriate network architecture. Network structure and connectivity are searched simultaneously. CPSO and DPSO can jointly search the global optimal ANN architecture and weights. Experiment results demonstrate that the proposed method is a useful tool for training ANN, which converges quickly and can avoid overfitting to some extent.

2 Correlative Algorithms

2.1 Particle Swarm Optimization

PSO is initialized with a swarm including N random particles. Each particle is treated as a point in a D -dimensional space. The i th particle is represented as $x_i = (x_{i1}, x_{i2}, \dots, x_{iD})$, x_{ij} is set in the range $[a_j, b_j]$. The best previous position of the i th particle is represented as $P_i = (p_{i1}, p_{i2}, \dots, p_{iD})$. The best particle among all the particles in the population is represented by $P_g = (p_{g1}, p_{g2}, \dots, p_{gD})$. The velocity of particle i is represented as $V_i = (v_{i1}, v_{i2}, \dots, v_{iD})$. After finding the aforementioned two best values, the particle updates its velocity and position according to the following equations:

$$v_{id} = \omega v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \quad (1)$$

$$x_{id} = x_{id} + v_{id} \quad (2)$$

Where ω is the inertia weight, c_1 and c_2 are two positive constants called learning factors, r_1 and r_2 are random numbers in the range of $[0, 1]$.

2.2 Chaos Particle Swarm Optimization

Chaos is characterized as ergodicity, randomness and regularity. Because chaos queues can experience all the states in a specific area without repeat, chaotic search becomes a novel tool used as an optimizer. Here, Logistic Equation is employed to obtain chaos queues, which is represented as follows:

$$z_{n+1} = \mu z_n (1 - z_n) \quad n = 0, 1, 2, \dots \quad (3)$$

Where μ is the control parameter, suppose that $0 \leq z_0 \leq 1$, when $\mu = 4$, the system of (3) has been proved to be entirely chaotic. Chaos queues z_1, z_2, z_3, \dots are generated by iteration of Logistic Equation.

The basic steps of CPSO algorithm is described as follows:

Step 1: Initialize a swarm including N particles, which locates each particle in a specific range randomly and sets each particle's velocity as a random value in the range $[-1,1]$.

Step 2: Using the global best and individual best of each particle, update each particle's velocity and position of connectivity variable according to (1) and (2).

Step 3: Optimize P_g by chaos search. Firstly, scale $P_{gi} (i = 1, 2, \dots, D)$ into $[0,1]$ according to $z_i = (p_{gi} - a_i) / (b_i - a_i), (i = 1, 2, \dots, n)$, and generate chaos queues

$z_i^{(m)} (m = 1, 2, \dots)$ by iteration of Logistic Equation, then, transfer the chaos queues into the optimization variable $p_g^{(m)} (m = 1, 2, \dots)$ according to following equation:

$$p_{gi}^{(m)} = a_i + (b_i - a_i)z_i^{(m)}, \text{ upon that the solution set is obtained:}$$

$$p_g^{(m)} = (p_{g1}^{(m)}, p_{g2}^{(m)}, \dots, p_{gD}^{(m)}), (m = 1, 2, \dots).$$

Compute the fitness value of each feasible solution $p_g^{(m)} (m = 1, 2, \dots)$ in the problem space during chaotic search, and get the best solution p^* .

Step 4: Replace the position of one particle selected randomly from the swarm with p^* .

Step 5: If one of the stopping criteria is satisfied, then stop. Otherwise, loop to step 2.

2.3 Discrete Particle Swarm Optimization

For a discrete problem expressed in a binary notation, a particle moves in a search space restricted to 0 or 1 on each dimension, where the velocity v_{id} represents the probability of bit x_{id} taking the value 1 or 0. Since it is a probability, must be constrained to the interval $[0.0,1.0]$, logistic transformation $sigmoid(v_{id})$ can be used to accomplish. The velocity and position of each particle is updated according to the following equations:

$$v_{id} = \omega v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}), \tag{4}$$

$$x_{id} = \begin{cases} 1, & \text{if } \rho_{id} < sigmoid(v_{id}), \\ 0, & \text{if } \rho_{id} > sigmoid(v_{id}). \end{cases} \tag{5}$$

Compared with standard PSO, the formula for velocity updating remains unchanged, except that now p_{id} and x_{id} are integers in $\{0,1\}$. In (5), ρ_{id} is a random number selected from a uniform distribution in $[0.0, 1.0]$.

3 ANN Structure and Parameters Optimizing by Proposed Hybrid PSO – A Combination of CPSO and DPSO

3.1 Particle Encoding and Fitness Computing

The connection structure of ANN featuring discrete, which is encoded by a string of binary bits, bit “1” is encoded if the corresponding connection between nodes exists; otherwise “0” is encoded. DPSO is applied to find the appropriate network structure. In CPSO, real-number strings were adopted to encode all the particles, each real-number coded string stands for a set of weights.

All the initial strings in CPSO and DPSO are randomly initialized with an appropriate size of a population. If the weight number of a fully connected ANN is d , then the particle is made up of weights in d dimension and connectivity in d dimension, the size of each particle should be $2 * d$.

Decode the binary string which represents the structure of each particle, a bit “0” in a particle represents the uselessness of the corresponding weight. The fitness is judged of all the samples in the training set, which is taken as the objective function. The performance of each particle is measured according to the fitness function. Here, we define Mean Square Error (MSE) as the fitness function, whose minimization will generate an optimum network configuration.

3.2 ANN Training by Hybrid PSO

The basic steps of ANN training by hybrid PSO are described as follows:

Step 1: Initialize a swarm including N particles, which locates each particle’s position and velocity of weight variables in a specific range randomly and sets each particle’s velocity of connectivity variables, then we can get the initial position according to (4).

Step 2: Evaluate the fitness of each particle.

Step 3: for all the particles of the swarm, take the following steps:

1) Using the global best and individual best of each particle, each particle’s velocity and position of weight variables is updated according to (1) and (2).

2) Update each particle’s velocity and position of connectivity variable according to (4) and (5).

3) Evaluate the fitness of each particle and compare the evaluated fitness value of each particle to its individual best p_i . If current value is better than p_i , then update p_i as current position.

4) If current value is better than the global best p_g , update p_g as current position.

Step 4: According to step 4 in 2.2, optimize the global best position of weight variables by chaos search. Compute the fitness value of each feasible solution $p_g^{(m)}$ ($m=1, 2, \dots$) in the problem space during chaotic search, and get the best solution p^* .

Step 5: Replace the position of one particle selected randomly from the swarm with p^*

Step 6: If one of the stopping criteria is satisfied, then stop. Otherwise, loop to step 3.

4 Experiment and Analysis

Large-scale revolving mechanism is a very important device widely used in electric power, metallurgy etc., it's very significant to constantly supervise its state of operation at real time and diagnose the faults accurately. In this section, we carried out an experiment on fault diagnosis of revolving machine with the method mentioned above based on a neural network. In this network, according to the set of fault symptom, the number of input neurons of the neural networks is 5, and the number of output neurons is 3 in term of fault cause set. Due to experience formula, the number of neurons of hidden layer is set as $n_h = 7$.

Apply the proposed algorithm and genetic algorithm (SW-GA) which also optimize the topology and connection weights simultaneously in ANN training, the best Mean Square Errors following iteration number are shown in figure 1. From the results of comparison, it is obvious that the proposed algorithm has a better result than that of SW-GA, it shows its superiority in convergence speed and efficiency in training ANN.

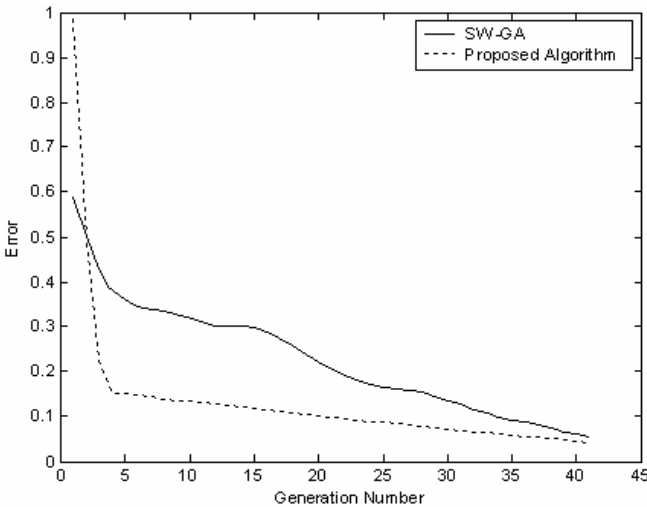


Fig. 1. Best error history of network training

BP and PSO only training the weights of the fully connected ANN, SW-GA and the proposed algorithm are utilized for testing the neural network. The test error, the number of network connections and the runtime of CPU in training process are shown in table 1.

Table 1. Performance comparison between networks trained by four algorithms

Evaluation factor	BP	PSO	SW-GA	Proposed algorithm
Test error	0.6227	0.4332	0.4526	0.3294
Connection number	56	56	38	34
Runtime of CPU (s)	6.1250	3.5330	5.5750	4.1230

From table 1, it can be seen that the proposed algorithm improves the capacity of fault pattern recognition by optimizing the structure of the neural network, this indicates that the network connections are redundant, and validates that redundant structure can introduce bad effect on the performance of ANN. By comparing the results of different methods mentioned above, it is clearly to see that the training efficiency is highly improved by using the proposed algorithm. Though optimizing the structure increases the complexity of the network, the accuracy is improved when applying the neural network whose structure is optimized in practice. The structure optimized neural network especially adapts to dispose large-scale data real time.

5 Conclusion

Redundant connections not only decrease the processing speed, large numbers of redundant connections but also affect the performance of ANN, therefore, finding the appropriate network architecture and its connectivity simultaneously is essential in ANN training. In this article, CPSO is used for the weight training of ANN, and DPSO is applied to find the appropriate network architecture, CPSO and DPSO can jointly search the global optimal ANN architecture and weights. The method can delete the redundant connections when training the weights of ANN to optimize the structure.

References

- [1] Yang, J.M., Kao, C.Y.: A Robust Evolutionary Algorithm for Training Neural Networks. *Neural Computing and Application* **10**(3) (2001) 214-230
- [2] Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization Proc. IEEE Service Center ed. IEEE International Conference on Neural Networks, Perth, Australia, 1995. Piscataway: IEEE Press (1995) 1942-1948
- [3] Voss, M.S., Xin, F.: Arma Model Selection Using Particle Swarm Optimization and AIC Criterial. The 15th Triennial World Congress, Barcelona, Spain (2002)
- [4] Parsopoulos, K., Vrahatis, M.: Recent Approaches to Global Optimization Problems Through Particle Swarm Optimization. *Natural Computing* **1**(2-3) (2002) 235-306
- [5] Kennedy, J., Eberhart, R. C.: A Discrete Binary Version of the Particle Swarm Algorithm. Proc. of the 1997 Conf. on Systems, Man, and Cybernetics. Piscataway: IEEE Press (1997) 4104-4108

Designing Neural Networks Using PSO-Based Memetic Algorithm*

Bo Liu, Ling Wang, Yihui Jin, and Dexian Huang

Department of Automation, Tsinghua University, Beijing 100084, China
Liub01@mails.tsinghua.edu.cn

Abstract. This paper proposes an effective particle swarm optimization (PSO) based memetic algorithm (MA) for designing artificial neural network. In the proposed PSO-based MA (PSOMA), not only the evolutionary searching mechanism of PSO characterized by individual improvement plus population cooperation and competition is applied to perform the global search, but also several adaptive high-performance faster training algorithms are employed to enhance the local search, so that the exploration and exploitation abilities of PSOMA can be well balanced. Moreover, an effective adaptive Meta-Lamarckian learning strategy is employed to decide which local search method to be used so as to prevent the premature convergence and concentrate computing effort on promising neighbor solutions. Simulation results and comparisons demonstrate the effectiveness and efficiency of the proposed PSOMA.

1 Introduction

Evolving artificial neural network is an important issue in neural networks (NN), evolutionary computation (EC) and engineering fields. As we know, essentially it is an optimization problem to design a neural network. So far, many techniques have been proposed to train a network, such as gradient-descent method (i.e., BP, conjugate gradient algorithms), tabu search, simulated annealing, evolutionary computation etc. [1]. It is concluded that gradient-descent methods are very slow, easy to be trapped in local optima, short of generalization and rather sensitive to initial weights. During the past two decades, evolutionary computation (EC) techniques especially gained much attention for neural network design [2]. As a new kind of EC technique, particle swarm optimization (PSO) has been applied to many kinds of problems, including neural network design [3], [4]. In this paper, an effective particle swarm optimization (PSO) based memetic algorithm (MA) is proposed for neural network design and the trading-ration system of water market simulation. In the proposed PSO-based MA (PSOMA), not only the evolutionary searching mechanism of PSO characterized by individual improvement plus population cooperation and competition is applied to perform the global search, but also several adaptive high-performance faster training

* This work is supported by National Natural Science Foundation of China (Grant No. 60204008, 60374060 and 60574072) and National 973 Program (Grant No. 2002CB312200).

algorithms are employed to enhance the local search, so that the exploration and exploitation abilities of PSOMA can be well balanced. Moreover, an effective adaptive Meta-Lamarckian learning strategy is employed to decide which local search method to be used so as to prevent the premature convergence and concentrate computing effort on promising neighbor solutions. Simulation results and comparisons demonstrate the effectiveness and efficiency of the proposed PSOMA.

The remaining content of this paper is organized as follows. In Section 2, the PSOMA is proposed after presenting the basic idea of PSO, local search combining adaptive Meta-Lamarckian learning strategy. Numerical simulations and comparisons are provided in Section 3. Finally, Section 4 provides some concluding remarks.

2 PSOMA

2.1 PSO

Recently, a novel evolutionary technique, namely particle swarm optimization (PSO) has been proposed [5], which was developed based on bird flocking, fish schooling, and swarm theory. Due to the simple concept, easy implementation and quick convergence, nowadays PSO has gained much attention and wide applications in a variety of fields [3], [4], [6], [7], [8].

In PSO, it starts with the random initialization of a population (*swarm*) of individuals (*particles*) in the search space and works on the social behavior of the particles in the swarm. Therefore, it finds the global best solution by simply adjusting the trajectory of each individual towards its own best location and towards the best particle of the swarm at each time step (*generation*). However, the trajectory of each individual in the search space is adjusted by dynamically altering the velocity of each particle, according to its own flying experience and the flying experience of the other particles in the search space.

The position and the velocity of the i -th particle in the d -dimensional search space can be represented as $X_i = [x_{i,1}, x_{i,2}, \dots, x_{i,d}]$ and $V_i = [v_{i,1}, v_{i,2}, \dots, v_{i,d}]$ respectively. Each particle has its own best position (*pbest*) $P_i = (p_{i,1}, p_{i,2}, \dots, p_{i,d})$, corresponding to the personal best objective value obtained so far at time t . The global best particle (*gbest*) is denoted by P_g , which represents the best particle found so far at time t in the entire swarm. The new velocity of each particle is calculated as follows:

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1r_1(p_{i,j} - x_{i,j}(t)) + c_2r_2(p_{g,j} - x_{i,j}(t)), j = 1, 2, \dots, d \quad (1)$$

where c_1 and c_2 are constants called acceleration coefficients, w is called the inertia factor, r_1 and r_2 are two independent random numbers uniformly distributed in the range of $[0, 1]$.

Thus, the position of each particle is updated in each generation according to the following equation:

$$x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1), j = 1, 2, \dots, d \quad (2)$$

Due to the simple concept, easy implementation and quick convergence, nowadays PSO has gained much attention and wide applications in different fields. However, the performance of simple PSO greatly depends on its parameters, and it often suffers the problem of being trapped in local optima for the lost of diversity of swarm.

2.2 Local Search Combining Adaptive Meta-Lamarckian Learning Strategy

Some recent studies on the choice of local search methods have shown that the choice significantly affects the efficiency of problem searches. In this paper, we design a local search with multiple faster training algorithms to enrich the local searching behavior and to avoid premature convergence. Moreover, an effective adaptive strategy for Meta-Lamarckian learning in [9], [10] is employed to decide which training algorithm to be used for local search so as to reward the utilization times of those algorithms resulting in solution improvement.

In our paper, four training algorithms are utilized as the pool of local searches, including Powell-Beale, Scaled Conjugate Gradient, BFGS, and Levenberg-Marquardt algorithms. The adaptive Meta-Lamarckian learning strategy in our PSOMA is illustrated as follows. We divide local search into training phase and non-training phase. During the Meta-Lamarckian training phase, each training algorithm is applied for the same times, i.e., k epochs. Then, the reward η of each training algorithm is determined using the following equation:

$$\eta = |pf - cf| / k \quad (3)$$

where pf is the objective value before local search, and cf is the objective value after the local search. After the reward of each training algorithm is determined, the utilizing probability p_{ut} of each training algorithm is adjusted using the following equation:

$$p_{ut,i} = \eta_i / \sum_{j=1}^K \eta_j \quad (4)$$

where η_i is the reward value of the i -th training algorithm, K is the total number of training algorithm.

At non-training phase, according to the utilizing probability of each neighborhood, a roulette wheel rule is used to decide which training algorithm to be used for local search. If the i -th training algorithm is used, its reward will be updated by $\eta_i = \eta_i + \Delta\eta_i$, where $\Delta\eta_i$ is the reward value of the i -th training algorithm calculated during a non-training phase. Of course, the utilizing probability p_{ut} of each training algorithm should be adjusted again for next generation of PSOMA.

2.3 PSOMA

The PSOMA is proposed in this section, whose procedure is illustrated in Fig. 1.

Step 1: Set $k = 0$. For each particle i in the population:

Step 1.1: initialize X_i randomly.

Step 1.2: initialize V_i randomly.

Step 1.3: evaluate f_i .

Step 1.4: initialize P_g with the index of the particle with the best function value among the population.

Step 1.5: initialize P_i with a copy of X_i , $\forall i \leq N$.

Step 2: Repeat until a stopping criterion is satisfied:

Step 2.1: find P_g such that $f[P_g] \leq f_i, \forall i \leq N$.

Step 2.2: for each particle i , $P_i = X_i$ if $f_i < f_{best}[i], \forall i \leq N$.

Step 2.3: perform PSO operator for each particle i to update V_i and X_i according to equation 1 and 2.

Step 2.4: evaluate f_i for all particles.

Step 3: Implement the local search combining adaptive Meta-Lamarckian learning strategy for the best particle, and update the best particle.

Step 4: If a stopping criterion is satisfied, output the solution found best so far, otherwise, Let $k = k + 1$ and go back to step 2.

Fig. 1. Procedure of PSOMA

3 Experiments

In this section, we apply the PSOMA for designing the weight of multi-layer feed-forward networks. Three different problems with different network structures, including two classification problems [11] (Ionosphere and Cmc) and one approximation problem (Henon), are used for testing. The training objective is to minimize mean squared error (MSE). In particular, each particle in PSO denotes a set of weights of the network. The dimension of each particle is same as the number of weights of a certain network. Training a network using PSO means moving the particle among the weight space to minimize the MSE.

We apply three training algorithms, including PSOMA, standard PSO and Powell-Beale's conjugate gradient algorithm for comparison. In PSO and PSOMA, population size is 10, c_1 and c_2 are set to 2.0, v_{\max} is clamped to be 15% of the search space and use linearly varying inertia weight over the generations, varying from 1.2 at the beginning of the search to 0.2 at the end. The Powell-Beale's method is a standard training algorithm in Matlab neural network toolbox. The total number of function evaluation is set to 20000 in each run. All experiments were conducted 20 runs. In each experiment, each data set was randomly divided into two parts: 2/3 as training set and 1/3 as test set. MSE_T and MSE_G refer to mean square error averaged over 20 runs on the training and test set, respectively. And $Error_T$ and $Error_G$ referred to the error rate of classification and generalization averaged over 20 runs for the training and test sets, respectively. The information of data sets as well as the designing results of the three algorithms is listed in Table 1.

Table 1. Information of data sets and results of the three algorithms

Case/ Architecture	Index	PSOMA	PSO	Powell-Beale
Ionosphere/34-2-2	MSE_T	1.13e-06	1.36e-01	1.82e-03
	$Error_T$	0	26.1538	0
	MSE_G	1.23e-01	9.44e-01	7.24e-01
	$Error_G$	14.9501	48.5897	28.6325
Cmc/9-5-3	MSE_T	1.19e-02	1.97e-01	1.91e-01
	$Error_T$	51.7920	76.4950	77.9600
	MSE_G	1.99e-01	2.09e-01	2.04e-01
	$Error_G$	59.2722	77.5370	80.3171
Henon/2-5-1	MSE_T	2.18 e-05	2.20e-03	4.49e-05
	$Error_T$	0	0.0250	0
	MSE_G	1.51e-05	2.00e-03	6.45e-05
	$Error_G$	0	0	0

From Table 1, it can be seen that MSE_T , MSE_G , $Error_T$ and $Error_G$ of PSOMA outperform those of both PSO and Powell-Beale. So, it is concluded that PSOMA is more effective for neural network design, more robust on initial conditions and can obtain networks with better generalization property. In a word, the proposed PSOMA is a viable approach for neural networks design.

4 Conclusions

In this paper, an effective PSOMA is proposed for designing neural networks, in which PSO, local search combining adaptive Meta-Lamarckian learning strategy are well combined. Simulation results and comparisons based on three typical problems demonstrated the effectiveness of the PSOMA in term of searching quality, robustness on initial conditions and generalization property. The future work is to apply such approach to design other kinds of neural networks.

References

1. Yao, X.: Evolving Artificial Neural Networks. Proceedings of the IEEE. **87** (1999) 1423-1447
2. Wang, L.: Intelligent Optimization Algorithms with Applications. 4th edn. Tsinghua University & Springer Press, Beijing (2001)
3. Liu, B., Wang, L., Jin, Y.H., Huang, D.X.: Advances in Particle Swarm Optimization Algorithm. Control and Instruments in Chemical Industry **32** (2005) 1-6
4. Liu, B., Wang, L., Jin, Y.H., Huang, D.X.: Designing Neural Networks Using Hybrid Particle Swarm Optimization. Lecture Notes in Computer Science. Springer-Verlag, Berlin Heidelberg, New York **3496** (2005) 391-397
5. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. Proc. IEEE Int. Conf. on Neural Networks (1995) 1942-1948
6. Liu, B., Wang, L., Jin, Y.H., Tang, F., Huang, D.X.: Improved Particle Swarm Optimization Combined with Chaos. Chaos, Solitons and Fractals **25** (2005) 1261-1271

7. Liu, B., Wang, L., Jin, Y.H., Tang, F., Huang, D.X.: Directing Orbits of Chaotic Systems by Particle Swarm Optimization. *Chaos, Solitons and Fractals* 29 (2006) 454-461
8. Liu, B., Wang L., Jin, Y.H.: An Effective PSO-based Memetic Algorithm for Flow Shop Scheduling. *IEEE Trans. System, Man and Cybernetics Part B* 37 (2007) 18-27
9. Ong, Y.S., Keane, A.J.: Meta-Lamarckian Learning in Memetic Algorithms. *IEEE Trans. Evol. Comput.* **8** (2004) 99-110
10. Ong, Y.S., Lim, M.H., Zhu, N., Wong, K.W.: Classification of Adaptive Memetic Algorithms: A Comparative Study. *IEEE Trans. System, Man and Cybernetics, Part B* **36** (2006) 141-152
11. Blake, C., Keogh, E., Merz, C.J.: UCI Repository of Machine Learning Databases, <http://www.ics.uci.edu/~mllearn/MLRepository.html>. University of California, Irvine, Dept. of Information and Computer Sciences (1998)

Simultaneous Optimization of ANFIS-Based Fuzzy Model Driven to Data Granulation and Parallel Genetic Algorithms

Jeoung-Nae Choi¹, Sung-Kwun Oh¹, and Ki-Sung Seo²

¹ 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

² Department of Electronics Engineering, Seokyeong University, South Korea

Abstract. The paper concerns the simultaneous optimization for structure and parameters of fuzzy inference systems that is based on Hierarchical Fair Competition-based Parallel Genetic Algorithms (HFCGA) and information data granulation. HFCGA is used to optimize structure and parameters of ANFIS-based fuzzy model simultaneously. The granulation is realized with the aid of the C-means clustering. Through the simultaneous optimization mechanism to be explored, we can find the overall optimal values related to structure as well as parameter identification of ANFIS-based fuzzy model via HFCGA, C-Means clustering and standard least square method. A comparative analysis demonstrates that the proposed algorithm is superior to the conventional methods.

1 Introduction

Recently, a lot of attention has been directed to advanced techniques of system modeling. Specially, many researchers are concerned about ANFIS-based fuzzy modeling and there has been a diversity of approaches to ANFIS-based fuzzy modeling. Some enhancements to the model have been proposed by Oh and Pedrycz [5,11-13]. As one of the enhanced ANFIS-based fuzzy model, fuzzy relation model based on information granulation and genetic algorithms was introduced [5]. Here, binary coded genetic algorithm (GAs) was used to optimize structure and premise parameters of fuzzy model. And sequential optimization method by means of GAs was studied. It includes two optimization procedures such as structural and parametric identification. First the structural optimization is carried out and then using the results of structural optimization, the parametric optimization is executed. Yet the problem of finding “good” initial parameters of the fuzzy sets in the rules remains open.

This study concentrates on optimization of information granulation-oriented ANFIS-based fuzzy model. We propose to use hierarchical fair competition based parallel genetic algorithm (HFCGA) for optimization of ANFIS-based fuzzy model.

Based on the IG and the HFCGA, the simultaneous optimization method is introduced. In the ANFIS-based fuzzy model, there are two classes of variables to be optimized. Structural optimization involves the number of input variables, a collection of specific subset of input variables, the number of membership functions per input

variable, and the polynomial type of the consequence part of fuzzy rules, the parametric optimization concerns apexes of membership functions used in the premise part of the fuzzy. All of two classes of variables are optimize through HFPGA, C-Means clustering and LSM at once.

2 Design of ANFIS-Based Fuzzy Model Based on Information Data Granulation

In the premise part of the rules, we confine ourselves to a triangular type of membership functions whose parameters are subject to some optimization. The C-Means clustering [7] helps us organize the data into cluster so in this way we capture the characteristics of the experimental data. In the regions where some clusters of data have been identified, we end up with some fuzzy sets that help reflect the specificity of the data set

The identification of the premise part is completed in the following manner. Given is a set of data $\mathbf{U}=\{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_l; \mathbf{y}\}$, where $\mathbf{x}_k=[x_{1k}, \dots, x_{mk}]^T$, $\mathbf{y}=[y_1, \dots, y_m]^T$, l is the number of variables and , m is the number of data.

[Step 1] Arrange a set of data \mathbf{U} into data set \mathbf{X}_k composed of respective input data and output data.

$$\mathbf{X}_k=[\mathbf{x}_k; \mathbf{y}] \quad (1)$$

\mathbf{X}_k is data set of k -th input data and output data, where, $\mathbf{x}_k=[x_{1k}, \dots, x_{mk}]^T$, $\mathbf{y}=[y_1, \dots, y_m]^T$, and $k=1, 2, \dots, l$.

[Step 2] Complete the C-Means clustering to determine the centers (prototypes) \mathbf{v}_{kg} with data set \mathbf{X}_k .

[Step 3] Partition the corresponding isolated input space using the prototypes of the clusters \mathbf{v}_{kg} . Associate each clusters with some meaning, say Small, Big, etc.

[Step 4] Set the initial apexes of the membership functions using the prototypes \mathbf{v}_{kg} .

After premise part of identification, we identify the structure considering the initial values of the polynomial functions based on the information granules realized for the consequence and antecedents parts.

[Step 1] Find a set of data included in the fuzzy space of the j -th rule.

[Step 2] Compute the prototypes \mathbf{V}_j of the data set by taking the mean of each rule.

$$\mathbf{V}_j = \{V_{1j}, V_{2j}, \dots, V_{kj}; M_j\} \quad (2)$$

[Step 3] Set the initial values of polynomial functions with the center vectors \mathbf{V}_j .

The identification of the conclusion parts of the rules deals with a selection of their structure (type 1, type 2, type 3 and type 4) that is followed by the determination of the respective parameters of the local functions occurring there.

The conclusion part of the rule that is extended form of a typical fuzzy rule in the TSK (Takagi-Sugeno-Kang) ANFIS-based fuzzy model has the form

$$R^j : \text{If } x_1 \text{ is } A_{1c} \text{ and } \dots \text{ and } x_k \text{ is } A_{kc} \text{ then } y_j - M_j = f_j(x_1, \dots, x_k) \quad (3)$$

In case of Type 3 (Quadratic Inference):

$$f_j = a_{j0} + a_{j1}(x_1 - V_{1j}) + \dots + a_{jk}(x_k - V_{kj}) + a_{j(k+1)}(x_1 - V_{1j})^2 + \dots + a_{j(2k)}(x_k - V_{kj})^2 + a_{j(2k+1)}(x_1 - V_{1j})(x_2 - V_{2j}) + \dots + a_{j((k+2)(k+1)/2)}(x_{k-1} - V_{(k-1)j})(x_k - V_{kj}) \tag{4}$$

The calculations of the numeric output of the model, based on the activation (matching) levels of the rules there, rely on the following expression

$$y^* = \frac{\sum_{j=1}^n w_{ji} y_i}{\sum_{j=1}^n w_{ji}} = \frac{\sum_{j=1}^n w_{ji} (f_j(x_1, \dots, x_k) + M_j)}{\sum_{j=1}^n w_{ji}} = \sum_{j=1}^n \hat{w}_{ji} (f_j(x_1, \dots, x_k) + M_j) \tag{5}$$

where, $\hat{w}_{ji} = \frac{w_{ji}}{\sum_{j=1}^n w_{ji}}$, $\hat{w}_{ji} = \frac{A_{j1}(x_{1i}) \times \dots \times A_{jk}(x_{ki})}{\sum_{j=1}^n A_{j1}(x_{1i}) \times \dots \times A_{jk}(x_{ki})}$

The consequence parameters a_{jk} can be determined by the standard least-squares method that leads to the expression

3 Hierarchical Fair Competition–Based Parallel Genetic Algorithm and Simultaneous Optimization of ANFIS-Based Fuzzy Model

One of the central problems in evolutionary computation is to combat premature convergence and to achieve balanced exploration. Parallel Genetic Algorithm (PGA) is devised to solve this problem, and there are various PGA models such as global, fine-grained, and coarse-grained model [8]. The most popular model is coarse-grained model and Hierarchical Fair Competition model (HFC) is one type of PGA. It has multiple-deme (subpopulation), individuals evolve within each deme independently, and specified individuals migrate to other deme in regular generation interval [9]. Evolutionary process is similar to traditional GAs, but it include migration algorithm.

In HFCGA, migration is executed in regular generation interval. Fig. 1 show the migration topology of HFCGA

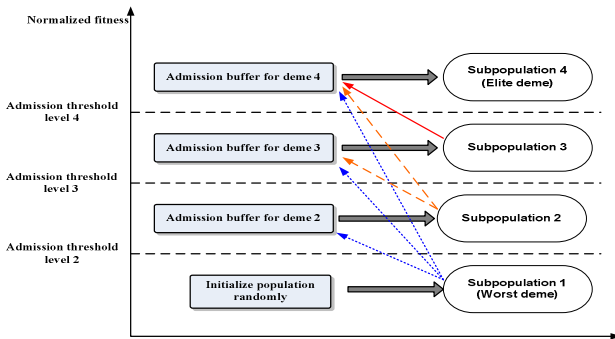


Fig. 1. The migration topology of HFCGA (In case when the number of demes is 4)

The optimization of ANFIS-based Fuzzy model is carried out through proposed the simultaneous optimization method.

In the sequential optimization method, first the structure identification is carried out and then the parameter identification is implemented based on results obtained in the structure identification. For structure identification, we search the structurally optimized model using the fixed membership parameters, and then find the parametrically optimized model via the parameter identification using the results obtained by the structurally optimized model. So, the boundary range of search space for parameter identification to find the optimized ANFIS-based fuzzy model is restricted within a certain area for structure identification.

In order to alleviate the problem, the simultaneous optimization method is proposed. Genes for the structure and parameter identification of the ANFIS-based fuzzy model are arranged only within a chromosome. In the sequel, we can expand the boundary range of search space to find an optimized model. Hence by using the structurally as well as parametric optimization procedure simultaneously based on HFCGA, we can obtain the optimized ANFIS-based fuzzy model.

Fig. 2 depicts the arrangement of chromosomes for the simultaneous optimization.

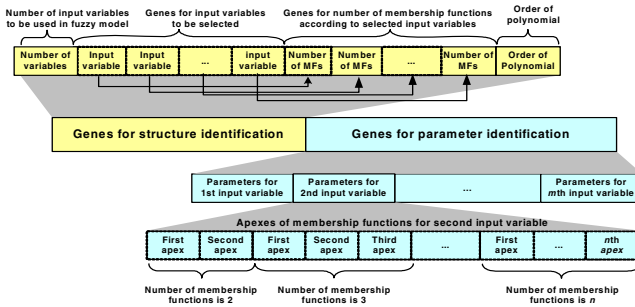


Fig. 2. Arrangement of chromosomes for simultaneous optimization

4 Experimental Studies

In this section, we provide numerical examples to evaluate the advantages and the effectiveness of the proposed approach. We deal with the NOx emission process data of gas turbine power plant. Till now, almost NOx emission processes are based on ‘standard’ mathematical model in order to obtain regulation data from control process. However, such models do not develop the relationships between variables of the NOx emission process and parameters of its model in an effective manner. A NOx emission process of a GE gas turbine power plant located in Virginia, USA, is chosen in this experiment.[11-13]

Using NOx emission process data, the regression equation reads in the form

$$y = -163.77341 - 0.06709x_1 + 0.00322x_2 + 0.00235x_3 + 0.26365x_4 + 0.20893x_5 \quad (6)$$

In the sequential and simultaneous optimization process, the number of input variables to be selected is confined to the range of two to five (2-4), the number of membership bound is two and three (2-3), and the polynomial order of the consequent

part of fuzzy rules is chosen from four types, that is Types 1-4. Fig. 3 depicts traces of performance indexes when running sequential and simultaneous optimization base on HFCGA.

In Fig. 4, the upper parts of the figure depict cluster groups and central values generated through C-Means clustering for each selected input variable, where central values are used to design the IG based fuzzy model. They are also used as initial apexes of the membership functions in the case of structure optimization in sequential method. The lower parts represent the tuned apexes of membership function carried out by simultaneous optimization base on HFCGA. Table 1 summarizes the results of comparative analysis of the proposed model with respect to other constructs.

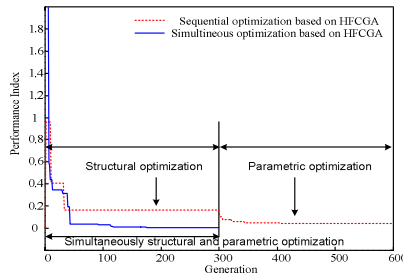


Fig. 3. Performance index for sequential method and simultaneous method

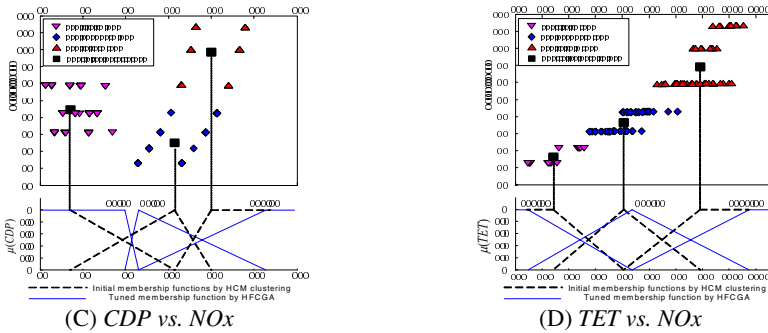


Fig. 4. Results of the C-Means clustering and the tuned apexes of the membership functions by means of simultaneous optimization method base on the HFCGA

Table 1. Summary of performance of various intelligent models

Model			Performance index	
			PI	E_PI
Regression model			17.68	19.23
Hybrid FR-FNNs [11]			2 (Linear)	0.080 0.190
Multi-FNN [12]			2 (Linear)	0.720 2.025
SOFNN [13]			Second layer	0.012 0.094
Our model	Sequential	3 (Quadratic)	0.0117	0.0670
	Simultaneous	2 (Linear)	0.00043	0.01221

5 Conclusions

In this study, we have developed a simultaneous optimization method for structure and parameters of fuzzy inference system that is based on Hierarchical Fair Competition-based Genetic Algorithms (HFCGA) and information data granulation. The HFCGA is used as optimization algorithm for ANFIS-based fuzzy model. Information granulation based on the C-Means clustering helps determine several parameters of ANFIS-based fuzzy model such as prototypes to be used in the consequence part of the fuzzy rules and the range of search space for parameters of premise part being used in HFCGA. Also, the structure and parameter of ANFIS-based fuzzy model are optimized through simultaneous optimization methodology. The experimental studies showed that performance is better than some other previous models. The proposed model is effective for nonlinear complex systems, so we can construct a well-organized model.

Acknowledgement. This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD)(KRF-2006-311-D00194).

References

1. Tong RM.: Synthesis of Fuzzy Models for Industrial Processes. *Int. J Gen Syst.* **4** (1978) 143-162
2. Pedrycz, W.: An Identification Algorithm in Fuzzy Relational System. *Fuzzy Sets Syst.* **13** (1984) 153-167
3. Takagi, T., Sugeno, M.: Fuzzy Identification of Systems and Its Applications to Modeling and Control. *IEEE Trans. Syst, Cybern.* **SMC-15**(1) (1985) 116-132
4. Oh, S.K., Pedrycz, W.: Identification of Fuzzy Systems by Means of an Auto-Tuning Algorithm and Its Application to Nonlinear Systems. *Fuzzy Sets and Syst.* **115**(2) (2000) 205-230
5. Pedrycz, W., Vukovich, G.: Granular Neural Networks. *Neurocomputing.* **36** (2001) 205-224
6. Krishnaiah, P.R., Kanal, L.N., editors.: Classification, Pattern Recognition, and Reduction of Dimensionality, volume 2 of Handbook of Statistics. North-Holland, Amsterdam. (1982)
7. Lin, S.C., Goodman, E., Punch, W.: Coarse-Grain Parallel Genetic Algorithms: Categorization and New Approach. *IEEE Conf. on Parallel and Distrib. Processing.* Nov. (1994)
8. Hu, J.J., Goodman, E.: The Hierarchical Fair Competition (HFC) Model for Parallel Evolutionary Algorithms. *Proceedings of the 2002 Congress on Evolutionary Computation: CEC2002.* IEEE. Honolulu. Hawaii. (2002)
9. Lyu, M.R.: Handbook of Software Reliability Engineering. McGraw-Hill, New York. (1995) 510-514
10. Park, H.S., Oh, S.K.: Fuzzy Relation-Based Fuzzy Neural-Networks using a Hybrid Identification Algorithm. *International Journal of Control, Automation, and Syst.* **1**(3) (2003) 289-300
11. Park, H.S., Oh, S.K.: Multi-FNN Identification Based on HCM Clustering and Evolutionary Fuzzy Granulation. *International Journal of Control, Automation, and Syst.* **1**(2) (2003) 194-3202
12. Oh, S.K., Pedrycz, W.: A New Approach to Self-Organizing Multi-Layer Fuzzy Polynomial Neural Networks Based on Genetic Optimization. *Advanced Engineering Informatics.* **18** (2004) 29-39

A Neural Network Based Optimization Method for a Kind of QPPs and Application

ShiehShing Lin

Department of Electrical Engineering, Saint John's University
499, Sec. 4, Tam King Road, Tamsui, Taipei, Taiwan
sslin@mail.sju.edu.tw

Abstract. In this paper, we present a neural network based optimization method for solving a kind of quadratic programming problems (QPPs) with equality and inequality constraints. The proposed method is appropriate for distributed implementation and can be used as a basic optimization module for managing optimization problems of large distributed systems. We test the proposed method in a real PC-Network for power system state estimation problem. Several cases are considered and obtain some successfully results.

1 Introduction

Nearly all the practical network-type systems are large-scale and formed by interconnected subsystems. For such kind of systems, process organization and state control are usually very complicated. Typically, a central control center is employed to govern the operations of the whole system. Currently, since computer communication technologies have become more grown-up, decentralized organization and control seems to be a tendency. For instance, they are applied on the current deregulated large power systems [1] formed by area-like subsystems interconnected through tier lines. Quite a few documents that are concerned with decentralized control algorithms for maintaining the stability of the system have been published [2], [3]. However, there are few documents related to distributed algorithms for solving optimization problems of large-scale power systems. Presenting a neural network based optimization method to handle a larger rank of optimization problems is the purpose of this paper. In this paper, we assume the considered large-scale system is formed by n interconnected subsystems [7]. Let x_i and x_{i_b} denote the states vector and the boundary states vector of subsystem i , and $x_{i_b}^r$ denotes the boundary states vector of the other subsystems connecting with subsystem i . A function $J(x)$ is said to be block additive if $J(x)$ can be expressed as $\sum_{i=1}^n J_i(x_i, x_{i_b}^r)$.

A kind of QPPs of the following form is the object problem of this paper:

$$\min_x \sum_{i=1}^n \left[\frac{1}{2} x_i^T D_{ii} x_i + r_i^T x_i + r_{i_b}^T x_{i_b}^r \right], \quad (1a)$$

subject to

$$E_i x_i = c_i, \quad i = 1, 2, \dots, n, \tag{1b}$$

$$\underline{x}_i \leq x_i \leq \bar{x}_i, \quad i = 1, 2, \dots, n, \tag{1c}$$

where the matrix D_{ii} is diagonal block sub matrix of D , D denotes the Hessian of the objective function in (1a) and is positive definite, r_i and r_{i_b} are constant vectors, E_i is a full-rank real matrix, and c_i is a constant vector. (1b) denote the equality constraints, and inequality constraints (1c) denote the bounded constraints on the states such that the vectors \bar{x}_i and \underline{x}_i denote the upper and lower limits of each subsystem's states x_i , respectively. We see that the objective function in (1a) is block additive. The paper is organized in the following manner. Section 2 presents the neural network based optimization method for solving the QPPs (1a)-(1c). The application examples and simulation test results are given in Section 3. Finally, we conclude this paper in Section 4.

2 Solution Method

2.1 The Duality Problem Formulation

The dual problem of (1a)-(1c) can be stated as

$$\max_{\lambda} \phi(\lambda), \tag{2}$$

where the dual function is

$$\phi(\lambda) = \min_{x \in \Gamma} \sum_{i=1}^n \left[\frac{1}{2} x_i^T D_{ii} x_i + r_i^T x_i + r_{i_b}^T x_{i_b} + \lambda_i^T (E_i x_i - c_i) \right], \tag{3}$$

$\lambda = (\lambda_1, \lambda_2, \dots, \lambda_n)$, is the vector of Lagrange multiplier, and Γ denotes the set of inequality constrains, that is, $\Gamma = \{x \mid \underline{x}_i \leq x_i \leq \bar{x}_i, i = 1, 2, \dots, n\}$.

2.2 The Neural Network Based Optimization Method

The neural network based optimization method for solving (2) stated in the following:

$$\lambda_i(t+1) = \lambda_i(t) + \alpha(t) d\lambda_i(t), \quad i = 1, 2, \dots, n, \tag{4}$$

where t is the iteration index, $\alpha(t)$ is a weighting, and the increment vector $d\lambda(t) = (d\lambda_1(t), \dots, d\lambda_n(t))$ is the solution of the following quadratic approximate problem of (2) at $\lambda(t)$:

$$\max_{d\lambda} \frac{1}{2} d\lambda^T \Phi d\lambda + \nabla \phi^T(\lambda(t)) d\lambda. \quad (5)$$

The matrix Φ is given by

$$\Phi = \begin{bmatrix} \Phi_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \Phi_n \end{bmatrix}, \quad (6)$$

where the block sub matrix Φ_i is selected as

$$\Phi_i = -E_i D_{ii}^{-1} E_i^T. \quad (7)$$

The $\nabla \phi(\lambda(t))$ in (5) can be computed by the following [4]:

$$\nabla \phi(\lambda(t)) = \begin{bmatrix} E_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & E_n \end{bmatrix} \begin{bmatrix} \hat{x}_1(\lambda(t)) \\ \vdots \\ \hat{x}_n(\lambda(t)) \end{bmatrix} - \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix}, \quad (8)$$

where $\hat{x}(\lambda(t)) = (\hat{x}_1(\lambda(t)), \dots, \hat{x}_n(\lambda(t)))$, and Eq. (8) can be decomposed into n independent sets of linear equations:

$$\nabla_{\lambda_i} \phi(\lambda(t)) = E_i \hat{x}_i(\lambda(t)) - c_i, \quad i = 1, 2, \dots, n. \quad (9)$$

The $d\lambda(t)$ can be obtained by solving the following [4]:

$$\Phi d\lambda(t) = -\nabla \phi(\lambda(t)), \quad (10)$$

which can be decomposed into n independent sets of linear equations

$$\Phi_i d\lambda_i(t) = -\nabla_{\lambda_i} \phi(\lambda(t)), \quad i = 1, 2, \dots, n. \quad (11)$$

Computation of $\nabla_{\lambda_i} \phi(\lambda(t))$. This can be achieved using the following two-phase algorithm [5], [6]:

Phase 1: Solve the following unconstrained minimization problem

$$\min_x \sum_{i=1}^n \left[\frac{1}{2} x_i^T D_{ii} x_i + r_i^T x_i + r_{i_b}^T x_{i_b} + \lambda_i^T(t) (E_i x_i - c_i) \right], \quad (12)$$

using a PBSGDS method [7] to obtain $\tilde{x}(\lambda(t)) = (\tilde{x}_1(\lambda(t)), \dots, \tilde{x}_n(\lambda(t)))$.

Phase 2: Project $\tilde{x}(\lambda(t))$ onto Γ where $\Gamma = \cup_{i=1}^n \Gamma_i$ $\Gamma_i = \{x_i \mid \underline{x}_i \leq x_i \leq \bar{x}_i\}$, and the resulting projection $\hat{x}(\lambda(t)) = (\hat{x}_1(\lambda(t)), \dots, \hat{x}_n(\lambda(t)))$ computed by

$$\hat{x}_i(\lambda(t)) = \begin{cases} \underline{x}_i, & \text{if } \tilde{x}_i(\lambda(t)) < \underline{x}_i, \\ \bar{x}_i, & \text{if } \tilde{x}_i(\lambda(t)) > \bar{x}_i, \\ \tilde{x}_i, & \text{Otherwise.} \end{cases} \tag{13}$$

2.3 Algorithm of the Proposed Method

The minimization (12) can be rewritten as follows:

$$\min_x \sum_{i=1}^n \left[\frac{1}{2} x_i^T D_{ii} x_i + (r_i + E_i^T \lambda_i(t))^T x_i + r_{i_b}^T x_{i_b} - \lambda_i^T(t) c_i \right], \tag{14}$$

which is a block additive unconstrained optimization problem. We define

$$J_i(x_i, x_{i_b}) = \frac{1}{2} x_i^T D_{ii} x_i + (r_i + E_i^T \lambda_i(t))^T x_i + r_{i_b}^T x_{i_b} - \lambda_i^T(t) c_i, \tag{15}$$

and then

$$J(x) = \sum_{i=1}^n J_i(x_i, x_{i_b}), \tag{16}$$

which denotes the objective function of (12).

Weighting determination. The weighting $\alpha(t)$ in (4) can be chosen by the centralized Armijo’s rule [4]. We define

$$\psi_i(\lambda(t)) = J_i(\hat{x}_i(\lambda(t)), \hat{x}_{i_b}(\lambda(t))). \tag{17}$$

From the definition of J_i and ψ_i lead to the following:

$$\phi(\lambda(t)) = \sum_{i=1}^n \psi_i(\lambda(t)). \tag{18}$$

The centralized Armijo’s rule [4] sets the weighting $\alpha(t)$ to be $\beta^{m(t)}$ where $m(t)$ is the smallest nonnegative integer m for which the following condition holds

$$\sum_{i=1}^n \psi_i(\lambda(t) + \beta^m d\lambda(t)) > \sum_{i=1}^n [\psi_i(\lambda(t)) + \rho \beta^m \|d\lambda_i(t)\|_2^2], \tag{19}$$

where scalar $\rho > 0$.

Algorithm. The neural network based optimization algorithm for solving the QPPs.

For every subsystem $i, i=1, \dots, n$

Step 0: Set ρ, β initial value $\lambda_i(0), t=0, m=0$. Calculate Φ_i by (7).

Step 1: Use the PBSGDS method [7] to solve (12) and obtain $\tilde{x}_i(\lambda(t))$.

Step 2: Calculate $\hat{x}_i(\lambda(t))$ from (13).

Step 3: Compute $\nabla_{\lambda_i} \phi(\lambda(t))$ by (9).

Step 4: Compute $d\lambda_i(t)$ by (11) and calculate $\|d\lambda_i(t)\|_2^2$.

Step 5: Calculate $\psi_i(\lambda(t))$ by (17).

Step 6: Send the values of $\psi_i(\lambda(t))$ and $\|d\lambda_i(t)\|_2^2$ to the central computer.

Step 7: Check whether the inequality (19) holds and send the result to each subsystem.

Step 8: If (19) holds, set $\lambda_i(t+1) = \lambda_i(t) + \beta^m d\lambda_i(t)$, go to step 9; otherwise, set

$$m = m + 1, \text{ update } \lambda_i(t) = \lambda_i(t) + \beta^m d\lambda_i(t) \text{ and go to Step 1.}$$

Step 9: If $\|\alpha(t)d\lambda(t)\|_\infty \leq \varepsilon$, x_i is the solution; otherwise, set $t = t + 1$ and go to Step 1.

3 Application of the Neural Network Based Optimization Method

3.1 The Constrained Weighted Least Squares Problem (CWLS) [8]

The constrained state estimation problems are a kind of CWLS problems in its exact formulation and can be formulated as QPPs [9]. The IEEE 118-bus with eight subsystems is used for our test system and we use eight areas PC-Network as our experimental computer network.

3.2 Test Results

The initial values of states are all 1.0 p.u. for voltage magnitudes and 0 radians for phase angles, $\beta = 0.95$, $\rho = 0.90$, $\lambda_i(0) = 0$, $i = 1, 2, \dots, n$ and the termination criteria are set $\varepsilon \leq 0.001$ in all cases. We choose four different cases of QPPs including different numbers of equality constraints (e.c.) and inequality constraints (i.c.); Case (a) 60 e.c., 88 i.c., Case (b) 80 e.c., 98 i.c., Case (c) 100 e.c., 108 i.c., and Case (d) 120 e.c., 118 i.c., and the total numbers of variables are 236 in all cases. To show the efficiency, we compare with the existing state-of-the-art parallel algorithm for QPPs [10]. The paper dealt with a parallel implementation of an interior point (IP) method which uses the preconditioned conjugate gradient (PCG) algorithm as the inner solver. We abbreviate this method as IPPCG algorithm. Based on parallel approach, we used our algorithm and the IPPCG algorithm to solve the same QPPs in the IEEE 118-bus with eight subsystems with the same initial condition and stop

criteria. We find that the speed up ratio of our algorithm exceeds about 22.18, 23.82, 25.91, and 27.65 times over than the IPPCG algorithm in Cases (a)-(d), respectively, in solving the QPPs in the IEEE 118-bus system. Furthermore, we find that, the **computational efficiency** of our method represents better results while the numbers of equality and/or inequality constraints are **increasing**. This addresses that the proposed algorithm is efficient for handling the QPPs with large number of equality and inequality constraints.

4 Conclusion

We presented a neural network based optimization method for solving a kind of QPPs with block additive objective function. The speed-up effect was observed when we implemented the proposed algorithm on a real eight areas PC-Network. Furthermore, the **computational efficiency** of our method represents better results while the numbers of equality and/or inequality constraints are **increasing**.

References

1. Dunn, W., Rossi, M., Avaramovic, B.: Impact of market restructuring on power systems operation. *IEEE Computer Application in Power Engineering* **8** (1) (1995) 42-47
2. Miller, D., Davison, E.: Adaptive control of a family of plants. in *Control of Uncertain Systems*. Berkhauser Press (1990) 197-219
3. Aghdam, A., Davison, E.: Decentralized switch in control. *IFAC LSS* **2** (1998) 1-7
4. Luenberger, D.G.: *Linear and nonlinear programming*. 2nd ed (Addison-Wesley) (1984)
5. Lin, C.H., Lin, S.Y.: A new dual type method used in solving optimal power flow problems. *IEEE Transactions on Power Systems* **12** (1997) 1667-1675
6. Lin, S.Y., Lin, C.H.: A computational efficient method for nonlinear multicommodity network flow problems. *Networks* **29** (1997) 225-244
7. Lin, S.Y., Lin, S.S.: A parallel block scaled gradient method with decentralized step-size for block additive unconstrained optimization problems of large distributed systems. *Asian Journal of Control* **5** (1) (2003) 104-115
8. Debs, A.: *Modern Power systems control and operation*. (Kluwer, Boston) (1989)
9. Lin, S.S.: A Parallel dual-type methods for solving CWLS problem. *Electric Power Systems and Research* **66** (2003) 227-232
10. Durazzi, C., Ruggiero, V.: Numerical Solution of special linear and quadratic programs via a parallel interior-point method. *Parallel Computing* **29** (2003) 485-503

Multilayer Perceptron Networks Training Using Particle Swarm Optimization with Minimum Velocity Constraints*

Xiaorong Pu, Zhongjie Fang, and Yongguo Liu

Computational Intelligence Laboratory, School of Computer Science & Engineering,
University of Electronic Science & Technology of China, Chengdu 610054, P.R. China
puxiaor@uestc.edu.cn
<http://cilab.uestc.edu.cn>

Abstract. Multilayer perceptron networks have been successfully trained by error backpropagation algorithm. We show that Particle Swarm Optimization(PSO) with minimum velocity constraints can efficiently be applied to train multilayer perceptrons to overcome premature convergence and alleviates the influence of dimensionality increasing. The experiments of two multilayer perceptrons trained by PSO with minimum velocity constraints are carried out. The result clearly demonstrate the improvement of the proposed algorithm over the standard PSO in terms of convergence.

1 Introduction

Multilayer perceptron is one of the most widely used neural networks for pattern classification and function approximation, which typically consists of a set of an input layer of source nodes, one or more hidden layers of computation nodes, and an output layer of computation nodes [1]. The input signal propagates through the network in a forward direction layer by layer. The actual response of the network is subtracted from a desired response to produce an error signal which is then propagated backward through the network. The synaptic weights are adjusted to make the actual response of the network move closer to the desired response. Multilayer perceptron is generally trained by backpropagation algorithm [2,3,4], an approximate steepest descent algorithm in which the performance index is mean square error.

However, the basic backpropagation training algorithm is too slow for many practical applications. There are considerable researches on methods to accelerate the convergence of the algorithm [5,6,7]. The storage and computational requirements of these training algorithms are different and suffer from the disadvantage that none of them is suitable for pattern recognition as well as function approximation under all conditions all the time.

* This work was supported by National Science Foundation of China under Grant 60471055, Specialized Research Fund for the Doctoral Program of Higher Education under Grant 20040614017, and the 2006 Youth Science and Technology Key Foundation of UESTC.

A number of research efforts have shown that Particle Swarm Optimization (PSO) is another tool for training neural networks more effective than backpropagation training algorithms [8,9,10]. Gudise [11] proved that the feedforward neural network weights converge faster with the PSO than with the backpropagation algorithm by comparing PSO with backpropagation for training feedforward neural network learning a nonlinear function. However, the performance of PSO deteriorates as the dimensionality of search space increases which is a common situation for complex neural networks. In high dimension problems, swarm particles are attracted to local minimum easily, thus get into premature convergence. This paper proposes a minimum velocity threshold to control the velocity of PSO in order to overcome premature convergence problem effectively.

The rest of this paper is organized as follows. Section 2 describes canonical particle swarm optimization algorithm briefly. Our proposed PSO with minimum velocity constraints is discussed in Section 3. Section 4 presents the experiments and their results. The conclusions are given in section 5.

2 Canonical Particle Swarm Optimization

The particle swarm optimization algorithm was originally developed by Eberhart and Kennedy [12,13] simulating some swarming behaviors observed in flocks of birds, schools of fish, or swarms of bees, and even human social behavior. Each particle represents a candidate solution to the optimization problem and moves with an adaptable velocity within the search space, and retains in its memory the best position it ever encountered.

In a PSO system, a swarm of particles fly through the real-number space. The position of a particle is influenced by the best position visited by itself and the position of the best particle in its neighborhood. Assume in an n -dimensional search space, $S \subset \mathfrak{R}^n$, a particle is an n -dimensional vector, and the size of the swarm is m . The i th particle is denoted as $X_i = (x_{i1}, x_{i2}, \dots, x_{in})^\top \in S$. Its velocity is also an n -dimensional vector $V_i = (v_{i1}, v_{i2}, \dots, v_{in})^\top \in S$. The previous best position of the i th particle so far is a point in S , denoted as $P_i = (p_{i1}, p_{i2}, \dots, p_{in})^\top$. The best position among the swarm so far is stored in a vector \tilde{P} , and its j th dimensional value is \tilde{P}_j . According to Eberhart and Kennedy [12], the velocity and position will be updated after each iteration by equation (1) and (2), respectively.

$$v_{ij}(t+1) = wv_{ij}(t) + c_1r_1(p_{ij}(t) - x_{ij}(t) + c_2r_2(\tilde{p}_j(t) - x_{ij}(t))). \quad (1)$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1). \quad (2)$$

where

w : the inertia factor, manipulating the impact of the previous history of velocities on the current velocity.

c_1, c_2 : positive constants, referred as cognitive and social parameters, respectively.

r_1, r_2 : random numbers, uniformly distributed in $[0,1]$.

j : index of the particle that attained the best previous position among all the individuals of the swarm.

t : the iteration counter.

3 Particle Swarm Optimization with Minimum Velocity Constraints

The particles flying in the search space may be failure to converge to the global minimum, in which the particles may go out of the search space when the velocity of particles increases rapidly; the particles may nearly stop when the velocity of particles decreases rapidly; or the particles cannot go out of the locally optimal solutions [14,15]. In order to guide the particles effectively in the search space, the maximum moving distance during one iteration must be clamped in between the maximum velocity $[-v_{max}, v_{max}]$, which can be described as follows:

$$\text{if } (v_{ij} > v_{max}) \text{ or } (v_{ij} < -v_{max})$$

$$v_{ij} = \text{sign}(v_{ij}) \times v_{max}$$

where variable v_{max} is the maximum velocity threshold.

Some previous studies have shown that the performance of PSO deteriorates as the dimensionality of the search space increases [14,15], in which PSO easily get in trap of premature convergence: when the algorithm stops, it may not converge to the global minimum. To alleviate the impact of dimensionality, we propose a minimum velocity threshold based on the average velocity of all the particles so far. The proposed strategy can drive those struck particles and let them explore better solutions in the search space. The velocity with minimum velocity constraints is adjusted as,

$$\text{if } (v_{ij} < v_{min}) \text{ and } (v_{ij} > -v_{min})$$

$$v_{ij} = \text{sign}(v_{ij}) \bar{v}$$

where variable v_{min} is the minimum velocity threshold, \bar{v} is the average velocity of all the particles so far calculated as $\bar{v} = \frac{1}{M \cdot N} \sum_{i=1}^M \sum_{j=1}^N (|v_{ij}|)$ (M is the size of the swarm, and a particle with N dimensionality).

4 Experiments

Training a multilayer perceptron to find a set of connection weights to minimize output errors is not a trivial problem because it is nonlinear and dynamic, which equals to searching a solution in a high dimensionality PSO space. To show the efficiency of the proposed strategy, two groups of comparative experiments are carried out. In experimental group 1, four continuous benchmark functions, i.e. De Jong function, Griewangk function, Rastrigin function and Sum of different powers function, are used to test the proposed PSO and the traditional PSO. On the other hand, two comparative experiments in experimental group 2 are designed for training two perceptrons with different layers. The one is for training

a 2-layer perceptron to solve the nonlinear XOR classification problem, and the other one is for training a 3-layer perceptron to address a more complex classification problem. The objectives of the two groups of experiment are to compare the performance of the standard PSO with our proposed method.

4.1 Simulations on 4 Continuous Benchmark Functions

The first part of the simulation, as shown in fig 1, shows the setting result of the proposed PSO in comparison with the traditional continuous PSO, applied to four famous benchmark functions: De Jong function, Griewangk function, Rastrigin function and Sum of different powers function [16], in which De Jong's function and Sum of different powers function are unimodal with single minimum, while the other two functions are highly multimodal with multiple local minima. The 4 benchmark functions are defined as follows,

- 1) *De Jong's function 1*

$$f_1(x) = \sum_{i=1}^n x_i^2$$

$$-512 \leq x_i \leq 512$$

Global minimum: $x_i = 0$ $f(x) = 0$
- 2) *Rastrigin's function 2*

$$f_2(x) = 10 \cdot n + \sum_{i=1}^n (x_i^2 - 10 \cdot \cos(2 \cdot \pi \cdot x_i))$$

$$-512 \leq x_i \leq 512$$

Global minimum: $x_i = 0$ $f(x) = 0$
- 3) *Griewangk's function 3*

$$f_3(x) = \sum_{i=1}^n \frac{x_i^2}{4000} - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

$$-600 \leq x_i \leq 600$$

Global minimum: $x_i = 0$ $f(x) = 0$
- 4) *Sum of different powers function 4*

$$f_4(x) = \sum_{i=1}^n |x_i|^{(i+1)}$$

$$-1 \leq x_i \leq 1$$

Global minimum: $x_i = 0$ $f(x) = 0$

Figure 1 illustrates the mean best function values for the 4 functions with two different dimensions, i.e. 30-D and 60-D, using the proposed PSO and the traditional PSO. It is evident that the proposed PSO performance outperforms the traditional PSO in terms of convergence speed and searching ability. Moreover, the proposed PSO overcomes the premature convergence problem and alleviates the influence of dimensionality increasing.

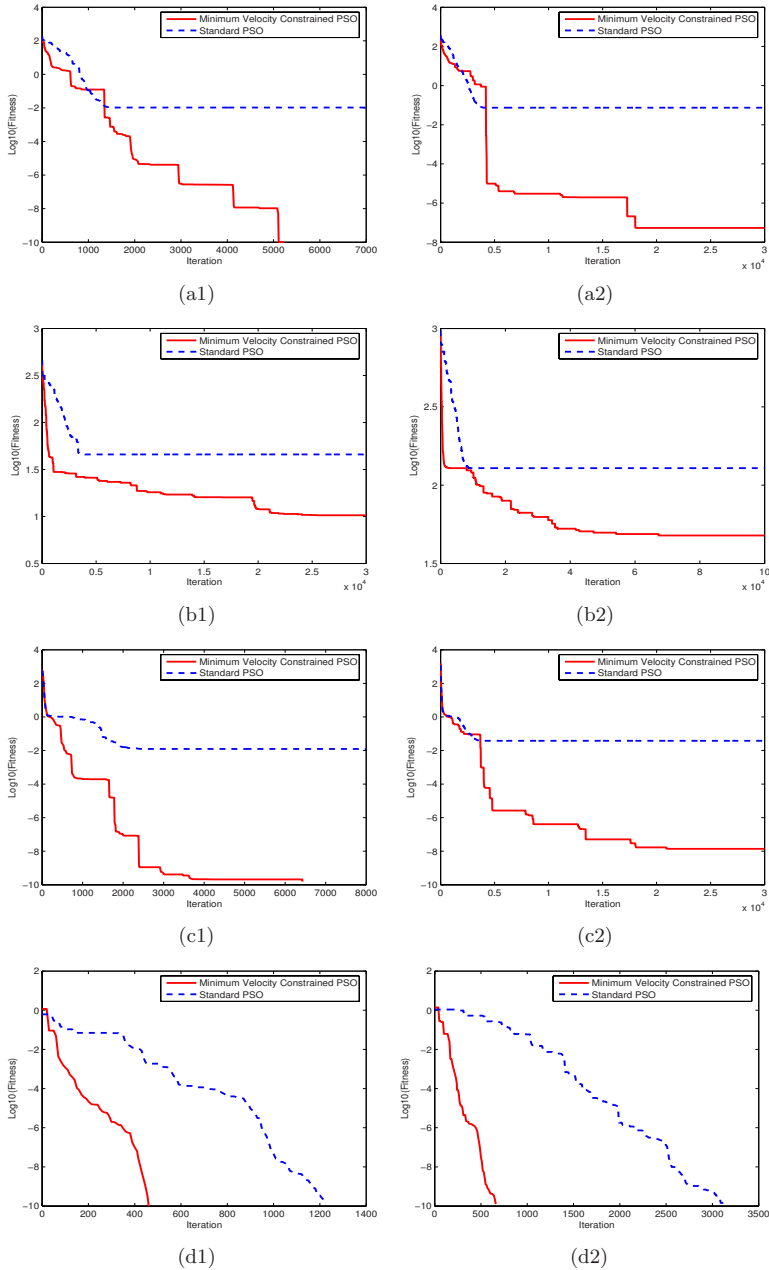


Fig. 1. Performance comparison the proposed PSO with the original one on 4 functions (a1)30-D DeJong function performance, (a2)60-D DeJong function performance,(b1)30-D Rastrigin function performance, (b2)60-D Rastrigin function performance,(c1)30-D Griewangk function performance, (c2)60-D Griewangk function performance,(d1)30-D Sum of different powers function performance, (d2)60-D Sum of different powers function performance

4.2 Training a 2-Layer Perceptron for XOR Problem

The 2-layer perceptron for XOR classification is displayed in Fig. 2. The network has two inputs $p1, p2$ and one output a . The output a returns 1 if the two inputs are the same, i.e. the input vector is $(0, 0)$ or $(1, 1)$, while the output a returns 0 if the two inputs are different $(0, 1)$ or $(1, 0)$. The connection weights and biases notated as $S1, S2, \dots, S9$ will be calculated by training the XOR network. By using PSO approach, all the particles each represented by a

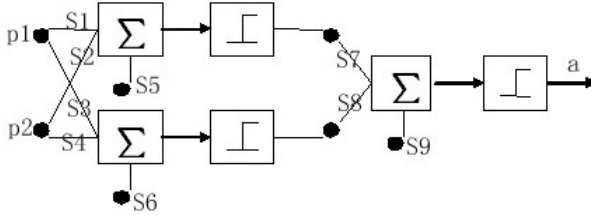


Fig. 2. The 2-layer perceptron for XOR problem

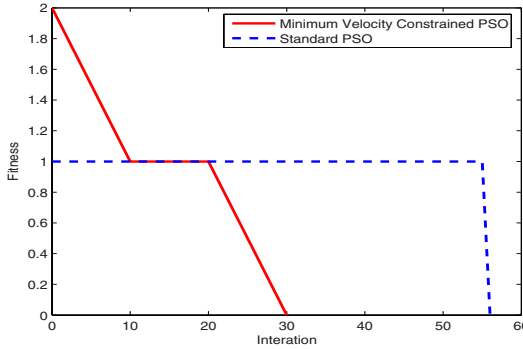


Fig. 3. Performance comparison between standard PSO and proposed PSO

vector with 9 elements fly through the 9-dimensional hyperspace to search for an optimal solution. The comparative experiments use a population of 20 particles and the results are the best answers among 10 tries with each algorithm. The Fig. 3 clearly shows that the great improvement of proposed PSO in terms of convergence which efficiently prevents premature convergence and converges to the global minimum of the output error term successfully and quickly.

4.3 Training a 3-Layer Perceptron for a Complex Pattern Classification

Another experiment is designed to train a 3-layer perceptron to solve a relatively complex classification problem as shown in Fig. 4 in which the light circles

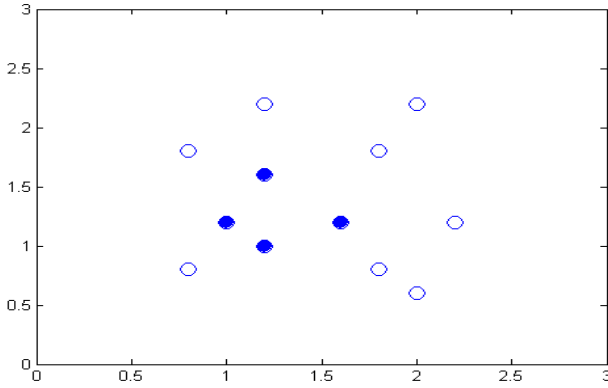


Fig. 4. A complex nonlinear classification problem

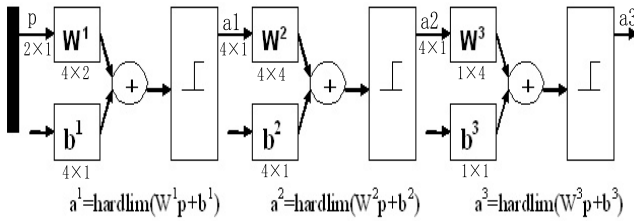


Fig. 5. The 3-layer perceptron for the complex nonlinear classification problem

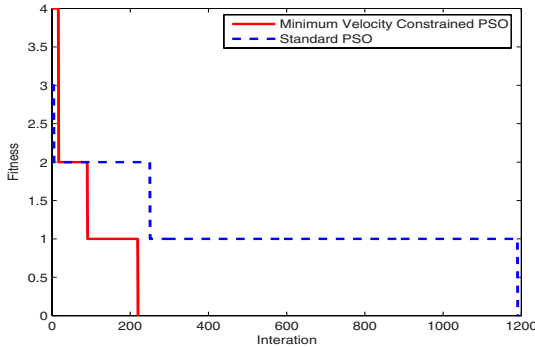


Fig. 6. Performance comparison between standard PSO and proposed PSO

represent one class and the dark circles represent another class. The 3-layer perceptron network is described in Fig. 5. For training this perceptron network using PSO, it is necessary that the particles fly through 37-dimensional hyper-space. As a result, each particle is initialized with position and velocity vectors of 37 elements. The size of the particles in this experiment is also set to 20 and

the comparative experiment between our proposed PSO and the standard PSO is carried out with 10 tries. The experiment results are shown in Fig. 6.

The dimensionality of search space, training this perceptron for solving the complex classification problem, is much higher than that for training the XOR perceptron network. Compared with Fig. 3, the performance of the standard PSO in Fig. 6 degrades remarkably than that in Fig. 3 due to the dimensionality increasing, while the influence of the high dimensionality for PSO with minimum velocity constraints is quite little.

5 Conclusions

A particle swarm optimization with minimum velocity constraints for training multilayer perceptrons is proposed in this paper. Comparative experimental results have clearly shown that the proposed PSO is nonsensitive to the dimensionality increasing in training multilayer perceptrons for nonlinear classification problems and can overcome premature convergence significantly. Further research on PSO with velocity constraints will be carried out in the future.

References

1. Simon Haykin: *Neural Networks-A Comprehensive Foundation*. 2nd edn. Tsinghua University Press, Beijing (2002)
2. Werbos, P.J.: Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE* **78** (1990) 1550-1560
3. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by Back-propagating errors. *Nature* **323** (1986) 533-536
4. Hornik, K.M., Stinchcombe, M., White, M.: Multilayer feedforward networks are universal approximators. *Neural Networks* **2** (1989) 359-366
5. Battiti, R.: First- and second-order methods of learning: between steepest descent and Newton's method. *Neural Computation* **4** (1991) 141-166
6. Rosario, R.: A Conjugate-gradient based algorithm for training of the feed-forward neural networks, Ph.D. Dissertation, Melbourne: Florida Institute of Technology, September 1989
7. Hagan, M.T., Menhaj, M.B.: Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks* **5** (1994) 989-993
8. Salerno, J.: Using the Particle Swarm Optimization Technique to Train a Recurrent Neural Model, *Proceedings of the IEEE International Conference on Tools with Artificial Intelligence*, (1997) 45C49
9. Zhang, C., Shao, H., Li, Y.: Particle swarm optimisation for evolving artificial neural network. *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics* **4** (2000) 2487-2490
10. Settles, M., Rylander, B.: Neural Network Learning using Particle Swarm Optimizers. *Advances in Information Science and Soft Computing* (2002) 224-226
11. Gudise, V.G., Venayagamoorthy, G.K.: Comparison of Particle Swarm optimization and backpropagation as training algorithms for neural networks. *Proceedings of the IEEE Swarm Intelligence Symposium* (2003) Indiana, 110-117

12. Kennedy, J., Eberhart, R.C.: "Particle swarm optimization," *Proc. IEEE Intl. Conf. on Neural Networks*. IV, 1942-1948, IEEE Service Center, Piscataway, NJ, (1995).
13. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995, 39-43. Piscataway, NJ: IEEE Service Center.
14. Yasuda, K., Iwasaki, N.: Adaptive particle swarm optimization using velocity information of swarm. Proceedings of IEEE International Conference on Systems, Man and Cybernetics. (2004) 3475-3481
15. Liu H., Abraham A.: Fuzzy Turbulent Particle Swarm Optimization. Proceeding of the 5th International Conference on Hybrid Intelligent Systems, Brazil, (2005) IEEE CS Press, USA.
16. Pohlheim, H.: Genetic Algorithm Toolbox Test Functions. Department of Automatic Control and systems Engineering University of Sheffield.

Characterization and Optimization of the Contact Formation for High-Performance Silicon Solar Cells[★]

SungJoon Lee¹, A. Pandey², DongSeop Kim², A. Rohatgi²,
Gary S. May², SangJeen Hong¹, and SeungSoo Han¹

¹ Myongji University, Department of Information Engineering,
& Myongji IT Engineering Research Institute (MITERI)
Yongin, Kyunggi 449-728, Korea
shan@mju.ac.kr

² Georgia Institute of Technology, School of Electrical Engineering,
Atlanta, GA 30332, USA

Abstract. In this paper, p-n junction formation using screen-printed metallization and co-firing is used to fabricate high-efficiency solar cells on single-crystalline (SC) silicon substrates. In order to form high-quality contacts, co-firing of a screen-printed Ag grid on the front and Al on the back surface field is implemented. These contacts require low contact resistance, high conductivity, and good adhesion to achieve high efficiency. Before co-firing, a statistically designed experiment is conducted. After the experiment, a neural network (NN) trained by the error back-propagation algorithm is employed to model the crucial relationships between several input factors and solar cell efficiency. The trained NN model is also used to optimize the beltline furnace process through genetic algorithms.

1 Introduction

To obtain high-quality contact formation, both screen-printed metallization and co-firing in an infrared (IR) beltline furnace have played important roles in the high-efficiency photovoltaic industry. P-N junction formation has not only influenced crucial electrical characteristics (such as efficiency, open circuit voltage, and fill factor), but also manufacturing cost and time.

Even though there are several disadvantages (such as high grid shading, junction shunting, low metal conductivity, and high contact resistance), screen-printing technology is used extensively for commercial solar cell manufacturing [1]. Screen-printed metallization makes p-n junction contact formation more rapid, simple, and economical compared to the conventional methods such as photolithography, buried-contact, and silicon ribbon technologies. Today, a matter of concern is how to alleviate high contact resistance and junction shunting [2-3].

[★] This work was supported by the Korea Research Foundation (Grant KRF-2006-013-D00242).

In order to make the co-firing process more stable and controllable, a systematic characterization experiment to determine and clarify the relationship between process parameters and global optimization using variable parameters is necessary.

2 Experiments

To fabricate $n^+ - p - p^+$ SC silicon solar cells, p -type, $1.5 \sim 2.0 \Omega\text{-cm}$, square textured float zone (FZ) wafers were employed. Saw damage removal by potassium hydroxide (KOH) etching was performed for 3 minutes at 85°C followed by a cleaning step. Before the emitter diffusion process with phosphoric acid on the surface, in order to acquire a hydrophilic feature, sulfuric acid (H_2SO_4) was oxidized uniformly. Phosphoric acid was then deposited on the front surface of the wafers by spin coating. The coated wafers were fully annealed in a ceramic roller hearth furnace with no metal contamination and eight heating zones, and this was followed by parasitic junction removal through edge isolation. After emitter formation by diffusion, phosphosilicate glass (PSG) was etched from the surface in a dilute HF solution. The next step was depositing a SiN_x single layer antireflection coating (ARC) by low-frequency (50 KHz) plasma-enhanced chemical vapor deposition (PECVD) to alleviate reflection losses and to enhance surface and bulk passivation. Ag and Al paste were then screen-printed on the front and back surface, respectively. Lastly co-firing was conducted in an IR beltline furnace with three dissimilar temperature zones to accomplish photo generation in a light-absorbing surface and separation of electrons and holes.

2.1 Statistical Experimental Design

The typical experimental method of collecting large quantities of data while holding each factor constant until all possibilities have been tested is an approach that becomes impossible as the number of factors increases. In addition, the role of each step in determining output response is generally not clear. Statistical experimental design techniques can be used to examine all possible combinations of a set of factors within a process at once and observe the data collectively. These techniques are highly efficient, and the structured approach for characterization using a relatively small number of experiments significantly improve process performance and makes the process more robust [4-5].

During the co-firing process, a set of experimental data was acquired from 2^4 full factorial designs with 3 center points, 8 axial points, and 5 random points to characterize process variations with four controllable input factors, consisting of the three different zone temperatures and belt speed (all of which have two levels). Random samples were fabricated for the purpose of verification.

2.2 Neural Networks

NN methodology has recently emerged as a powerful alternative for assisting with developing models with noisy and/or nonlinear data and establishing relationships between highly complicated input and the output parameters [6]. NNs consist of several layers of neurons, each of which are interconnected in such a way that

information is stored in the weights assigned to the connections. The NN learning system is designed to determine appropriate sets of interconnection weights that facilitate the activation of the neurons to achieve a desired state related to a given set of sampled patterns.

By forward propagation, the output of each neuron is calculated by summing the weighted input connections of the previous layer and then filtering this sum through a sigmoidal transfer function. Error back-propagation occurs when the acquired error in between the predicted output of the network and experimental one is used to generate feedback toward the network for learning [7]. This is a gradient descent approach to minimize the error of the output computed by the network. The weights are initially randomized but they are updated to minimize an error function in back-propagation.

Performance evaluation for the trained neural network is performed in terms of the root mean squared error (RMSE), which is given by:

$$RMSE = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \tag{1}$$

where n is the number of trials, y_i is the measured value, and \hat{y}_i is the predicted value.

The performance the network depends on both the number of neurons in each layer and parameters such as learning rate, momentum, and training tolerance. A typical multilayered neural network structure is depicted in Fig. 1 [8].

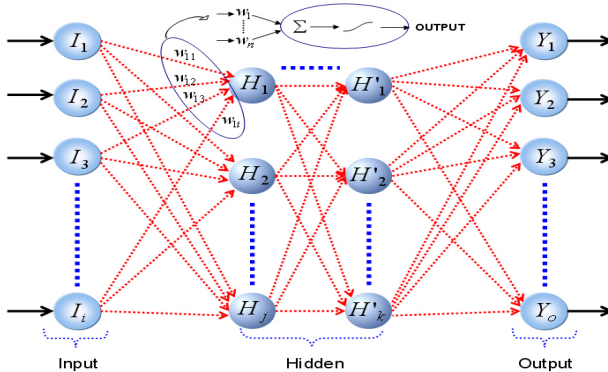


Fig. 1. A multilayered network

Table 1. Parameters in training network

Parameters	Networks Setting
Learning Rate (η)	0.1
Momentum (a)	0
Training Method	Vector
Maximum Iterations	100,000+
# of Neurons in the 1 st / 2 nd Hidden Layer	4 / 4

For the purpose of establishing and optimizing models, the Object-Oriented Neural Network Simulator (ObOrNNS) software tool, which was developed by the Intelligent Semiconductor Manufacturing group at the Georgia Institute of Technology, was employed in this study. Training was performed on 70% of the experimental data, and the remaining 30% were used to test the performance of the trained networks. Table 1 illustrates the parameters used for training.

2.3 Genetic Algorithms

Genetic algorithms (GAs) are widely employed in a variety of fields to determine optimal set points in processes requiring optimization. They are parallel and stochastic search methods that generate optimal recipes for desired target responses [10]. A GA mimics the natural evolutionary process, exhibited by a superior solution evolving within the population while inferior solutions deteriorate with each generational progression.

Randomly generated solutions are encoded to binary strings (chromosomes). Each string is evaluated with a fitness function to determine if the termination criteria for optimality are met. Selection and operations such as crossover and mutation then generate new individuals. The chromosomes are directly decoded and evaluated with the fitness function. More fit solutions are more likely to be selected and have more chance to reproduce.

3 Results and Discussion

3.1 Response Surface Methodology

After NN process models were established, response surfaces were generated to illustrate the relationships between the input parameters and the responses. Any two of the four process parameters were simultaneously varied, while the remaining parameters were fixed at their mid-range values.

For each model, a temperature variation was observed in zone 3 which has a predominant effect on cell efficiency. The plots in Fig. 2 a) and b) indicate that an increase or decrease in zone 3 temperature results in higher cell efficiency. In addition, high efficiency was observed at belt speeds at the approximate middle range. To highlight the effect of zone 3 temperature on cell efficiency, the temperature in zone 3 and belt speed were fixed at 845°C, 870°C at 115 rpm, respectively. The response surface plots at the fixed values are depicted in Fig. 2 c) and d).

3.2 Optimization Using GAs

Using NN models in combination with GAs, the ideal input parameters that lead to high-quality contact formation can be determined. The GA parameters and synthesized recipe for optimizing the co-firing process are listed in Tables 2 and 3, respectively.

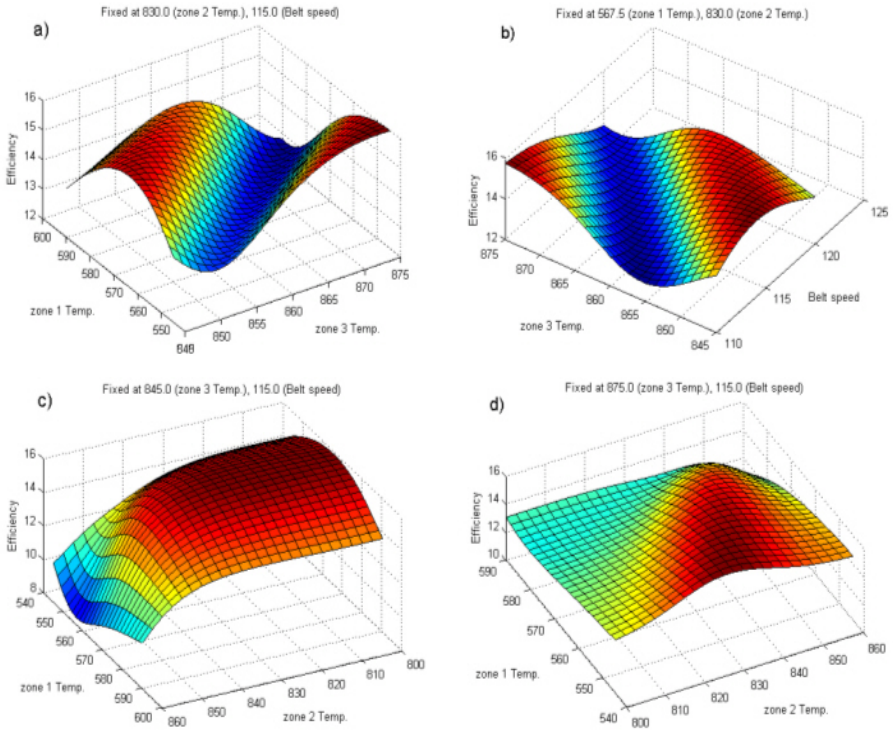


Fig. 2. Response surface plots for efficiency

Table 2. GA parameters

Parameters	Networks Setting
Weights	1.0
Population Size	100
Maximum Generations	100
Minimum Error	0.05
Probability of Crossover	0.65
Probability of Mutation	0.01

Table 3. Recipe results

Zone 1 Temp.	Zone 2 Temp.	Zone 3 Temp.	Belt speed	Efficiency
557.24 (°C)	821.45 (°C)	847.52 (°C)	117.61 (rpm)	15.81 (%)

3.3 Verification

In order to verify the performance of GA in finding the optimal process recipe, three additional verification experiments were implemented using the optimal recipe shown

in Table 3. One wafer was broken by over-firing, but the measured efficiency of the other two wafers was 15.18% and 15.69%. The average efficiency of these two, 15.44%, is 3.9% higher than the average efficiency of the previous 32 DOE experiments, 14.86%.

4 Conclusions

In summary, co-firing experiments were conducted using a central composite experimental design. Four parameters associated with cell efficiency were manipulated, and their effects were investigated using NN models. Mid-range belt speed at fixed middle values of zone 1 and zone 2 temperatures resulted in a high-efficiency cell. Optimization of the co-firing process conditions was successfully achieved using genetic algorithms, and performance was verified by additional experiments. The verification experiments showed an average of 15.44% in efficiency, which is 3.9% improvement compared to the previous DOE experiments.

References

1. Doshi, P., Mejia, J., Tate, K., Rohatgi, A.: Modeling and Characterization of High-Efficiency Silicon Solar Cells Fabricated by Rapid Thermal Processing, Screen Printing, and Plasma-Enhanced Chemical Vapor Deposition. *IEEE Transactions on Electron Devices* **44** (1997) 1417-1424
2. Nijs, J.F., Szlufcik, J., Poortmans, J., Sivoththaman, Mertens, R.P.: Advanced Manufacturing Concepts for Crystalline Silicon Solar Cells. *IEEE Transactions on Electron Devices* **46** (1999) 1948-1969
3. Rohatgi, A., Hilali, M.M., Nakayashiki, K.: High-efficiency screen-printed solar cell on edge-defined film-fed grown ribbon silicon through optimized rapid belt co-firing of contacts and high-sheet-resistance emitter. *Applied Physics Letter* **84** (2004) 3409-3411
4. Box, G., Hunter, W., Hunter, J.: *Statistics for Experimenters*. Wiley, New York (1978)
5. Montgomery, D.C.: *Introduction to Statistical Quality Control*. 5th edn. Wiley & Sons, Inc. (2005)
6. May, G. S.: Manufacturing ICs the Neural Way. *IEEE Spectrum* (1994) 47-51
7. Fausett, L.: *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*, Prentice-Hall, Inc. (1994)
8. Han, S.S., Cai, L., May, G.S., Rohatgi, A.: Modeling the Growth of PECVD Silicon Nitride Films for Solar Cell Application Using Neural Networks. *IEEE Transactions on Semiconductor Manufacturing* **9** (1996) 303-311
9. Han, S.S., May, G.S.: Modeling the Plasma Enhanced Chemical Vapor Deposition Process Using Neural Networks and Genetic Algorithms. *Tools with Artificial Intelligence*, 1994 Proceedings, Sixth International Conference (1994) 760-763
10. Hong, S.J., May, G.S., Park, D.C.: Neural Network Modeling of Reactive Ion Etching Using Optical Emission Spectroscopy Data. *IEEE Transactions on Semiconductor Manufacturing* **16** (2003) 598-608

Optimizing Process Parameters for Ceramic Tile Manufacturing Using an Evolutionary Approach

Thitipong Navalertporn and Nitin V. Afzulpurkar

Asian Institute of Technology,
P.O. Box 4, Klong Luang, Pathumthani 12120, Thailand
{st100613, nitin}@ait.ac.th

Abstract. In this paper, an integrated approach of neural network and bidirectional particle swarm is proposed. The neural network is used for obtaining the relationships between decision variables and the measures of interest while the bidirectional particle swarm is utilized to perform the optimization where multiple objectives and multiple constraints are presented. Finally, the proposed algorithm is applied to solve a process parameter design problem in ceramic tile manufacturing. The results showed that bidirectional particle swarm is an effective method for solving multi-objective optimization problems, and an integrated approach of neural networks and bidirectional particle swarm can be used in solving complex process parameter design problems.

1 Introduction

A proper selection of process parameters has always been a critical issue in ceramic tile manufacturing. Even at the present time, traditional practice still done more or less by human where engineers/operators base their decisions on individual experience, skill, intuition, and attempt to optimize the parameters via trial and error. Nevertheless, this approach nearly always results in delay and causes considerable wastes, and there is no guarantee that the selected parameters are truly optimal for there is no analytical description of process dynamics. In most cases, a number of crucial quality characteristics of tiles, which are influenced by numerous process parameters, must be optimized simultaneously. Consequently, the problem becomes a complex multi-objective optimization dilemma.

Traditionally, the methods for handling multi-objective problems has always involve the use of either the desirability function or the loss function to scale down multi-objective problems into a single objective problem prior to optimization. These approaches, however, constitute some problems. Primarily, the approaches require that a great deal of preference information of the decision maker be extracted before solving the problem. On the other hand, some of this information is difficult to present mathematically and the articulation is very complex to implement in practice. Secondly, as the multi-objective problems are scaled to a single objective problem before optimization, the result produces a single Pareto optimum for each run of the optimization process and the result is highly sensitive to the weight vector used in the scaling process.

Since its introduction in 1995, particle swarm optimization (PSO) has gained its reputation as a viable tool for solving multi-objective problems [1]. Accordingly, PSO is a competent alternative over other stochastic and population-based search algorithms, especially when dealing with multi-objective optimization problems, where the ability to efficiently locate the Pareto front of the problem is critically required.

In this study, an integrated approach of bidirectional particle swarm optimization (BPSO), an extended version of PSO and neural network is proposed as an alternative means for solving optimization problems. Here, neural network is utilized for modeling the relationships between decision variables and the performance measures of interest while the BPSO is used to compute Pareto optimal non-dominated solutions. Finally, the proposed approach is applied to solve a process parameter design problem in ceramic tile manufacturing where an optimization of multi-objectives with multi-constraints is involved.

2 Previous Works

Process parameter optimization has been and continues to be an active research area. Principally, [2] developed a hybrid system integrating fuzzy logic, neural networks and algorithmic optimization to optimize ceramic slip casting process. Afterward, [3] applied multi-objective genetic algorithm approach for optimizing surface grinding operation. Subsequently, [4] presented an application of genetic algorithm to the optimization of metal cutting process. In addition, [5] proposed an integrated optimization approach of fuzzy theory, neural network, exponential desirability function, and genetic algorithm for determining the optimal parameter setting in the thin quad flat back (TQFB) molding process. Further, [6] presented an approach using multi-objective particle swarm optimization to solve economic load dispatch problem in power system. As well as [7] proposed an approach integrating neural network, exponential desirability function, and genetic algorithm to solve process parameter design problems for fiberoptic industry. Finally, [8] developed an integrated system of neural network and swarm intelligence to optimize machining parameters in hard turning processes.

3 Multi-Objective Optimization (MOO)

Many optimization problems in engineering applications have always involve simultaneous optimization of multiple objectives. A general form of multi-objective optimization problems can be defined as:

$$\begin{array}{ll}
 \text{Minimize} & f(x) = (f_1(x), f_2(x) \dots f_k(x)) \\
 \text{Subject to} & g_j(x) \leq 0 \text{ for } j = 1, 2, \dots, p \\
 \text{and} & h_j(x) = 0 \text{ for } j = p+1, \dots, m \\
 & x = (x_1, x_2, \dots, x_n) \in R^n
 \end{array}$$

In this formulation; $f_i(x)$ denotes the objective function, $g_i(x)$ and $h_i(x)$ indicate respectively the inequality and equality constraints and the decision variables are shown with the vector x .

Unlike uni-objective optimization (UOO) where only one objective is optimized, MOO problems often involve the presence of conflicting objectives, where improvement in one objective may cause deterioration in another objective, and the task is, therefore, to find solutions which balance such trade-offs. A balance is achieved when a solution cannot improve any objective without degrading one or more of the other objectives. These solutions are referred to as non-dominated solutions, of which many may exist. Consequently, the ultimate goal when solving MOO is to produce a set of solutions that balance the trade-offs between such conflicting objectives. This set of solution is called a *Pareto optimal set*. The corresponding objective vectors in objective space, on the other hand, are referred to as the *Pareto front*.

Conventionally, there are numerous ways to solve MOO problems. However, the most common approach is to scale a set of objectives into a single objective by multiplying each objective with weight parameters which represents the relative importance among the objective and then solve the problem by standard uni-objective optimization methods, such as gradient-based methods, evolutionary algorithms, and etc. However, such approach is not very practical since the optimization results significantly depend on the selection of weight parameters, which in most cases, is very difficult to determine properly.

4 Particle Swarm Optimization (PSO) and Bidirectional Particle Swarm Optimization (BPSO)

Particle Swarm Optimization was first introduced by Kennedy and Eberhart, which was driven by the social behavior of a bird flock and can be seen as a population based stochastic optimization algorithm. In PSO, the group is a community composed of individuals called particles, and all particles fly around in a multidimensional search space. During flight, each particle adjusts its own “flying” according to its own flying experience as well as the flying experience of neighboring particles.

Let x and v denote a particle position and its velocity in a search space, respectively. Therefore, the i th particle is represented as $X_i = (x_{i1}, x_{i2} \dots x_{iD})$ in the D -dimensional search space. The best previous position of the i th particle is recorded and represented as $P_i = (p_{i1}, p_{i2} \dots p_{iD})$. The index of the best particle in the group, i.e. the particle with the smallest function value, is represented by the $P_g = (p_{g1}, p_{g2} \dots p_{gD})$, while the velocity of the i th particle is represented as $V_i = (v_{i1}, v_{i2} \dots v_{iD})$. The modified velocity and position of each particle can be manipulated according to the following formulas:

$$v_{id} = wv_{id} + c_1r_1(p_{id} - x_{id}) + c_2r_2(p_{gd} - x_{id}) \quad (1)$$

$$x_{id} = x_{id} + \chi v_{id} \quad (2)$$

Where c_1 and c_2 are positive constants and constrain the velocity toward global and local best, χ is a constriction factor which controls and constricts the velocity's magnitude; w is the inertia weight; and r_1 and r_2 are two random numbers within the range [0,1].

Nevertheless, this basic version of PSO is not suitable for solving MOO problems because there is no absolute global minimum (or maximum) in a multi-objective problem, but rather a set of non-dominated solutions. Recently, a number of modifications on PSO have been made to solve MOO problems. Introduced in [9], a method that use weighted aggregation technique with fixed adaptive weights to convert multiple objectives into a single objective. The methods need to run K times to produce K estimated Pareto optimal point. Presented in [10], a dynamic neighborhood method which use one-dimensional optimization to deal with multiple objectives. However, this algorithm is suitable to the two objectives optimization only. Presented in [1], a MOPSO (multi-objective particle swarm optimization) which maintains previously found non dominated solutions and used them to guide particle's flight. Nevertheless, this method is difficult to maintain the diversity of solutions with the increment of constraints and variables.

In this study, bidirectional particle swarm optimization (BPSO) methodology proposed in [11] is adopted for its documented ability to provide solutions with good distribution and its robust constraint-handling mechanism.

Generally, the basic principle of BPSO is to guide each particle to search simultaneously in its neighborhood and the region where particles are distributed sparsely by combining the isolated searching strategy and neighborhood searching strategy.

In addition to the standard population POP, an external archive (regarded as GBEST) is used for storing the global best particles found by the algorithm. Once better particles are found, they are entered into the archive and existing members of the archive are removed if they are dominated by the new particles. The use of such historical archive of previously found non-dominated vectors would encourage convergence towards globally non-dominated solutions.

In this approach, two offspring particles are derived from each particle. For the i -th particle, x_i is its location and v_i is its velocity. Assume *child1* and *child2* are derived from the i -th particle, their location and velocity are denoted as x_l, v_l, x_r, v_r , and

$$x_l = x_r = x_i$$

$$v_l = v_r = v_i$$

In order to maximize the spread of the obtained non-dominated front, here the isolated point searching strategy and neighborhood searching strategy are used for selecting a global optimum for the *child1* and *child2* respectively. Then the velocity and location of *child1* and *child2* are updated with formula (1) and (2). If the new solutions x'_l and x'_r are feasible and not dominated by any solution in GBEST, GBEST is updated by adding x'_l and x'_r to it and eliminate the solutions dominated by x'_l and x'_r .

When the offspring particles finish their flying, the i -th particle also should be updated. If x'_l is feasible, then the i -th particle is updated with *child1*, i.e.

$$x_i = x'_i \quad \text{and} \quad v_i = v'_i$$

Else it is updated with *child2*.

Finally, the best previous position (p_i) of the i -th particle is updated by comparing the Pareto dominance and the constraint violation relation between x_i and p_i .

To verify its capability, BPSO has been applied to solve SRN problem, suggested in [12], which is a two variables problem:

$$\text{Minimize } f_1(x) = 2 + (x_1 - 2)^2 + (x_2 - 1)^2$$

$$\text{Minimize } f_2(x) = 9x_1 - (x_2 - 1)^2$$

Subject to

$$e_1(x) = x_1^2 + x_2^2 \geq 225$$

$$e_2(x) = x_1 - 3x_2 + 10 \leq 0$$

Where

$$-20 \leq x_1 \leq 20$$

$$-20 \leq x_2 \leq 20$$

The Pareto front produced by BPSO is shown in Fig. 1. Here, BPSO is able to provide solutions that are consistently spaced and have a good distribution along the Pareto front, which validates the potential of BPSO as a viable method for solving constrained multi-objective problems.

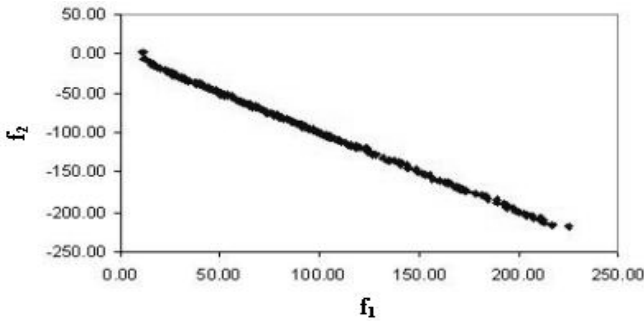


Fig. 1. Pareto Front Produces by BPSO

5 Description of the Proposed Approach

The proposed approach consists of two consecutive stages which include:

1. Articulation of the relationship between decision variables and performance measures of interest.
2. Computation of the Pareto optimal set.

To perform the first stage, Neural Network has been selected as a method of choice for its astounding ability to identify arbitrary nonlinear multi-parametric discriminant functions directly from experimental data which makes it suitable for applications where a mechanistic description of the dependency between dependent and independent variables is either unknown or very complex.

The execution of the second stage, on the other hand, utilizes bidirectional particle swarm as an optimization method for its proven ability to provide solutions with good distribution even for a constrained MOO problem as discussed and illustrated in the previous section.

Consequently, an integrated approach of neural network and bidirectional particle swarm optimization to perform process parameter optimization is proposed. The schematic diagram of the algorithm is shown in Fig. 2.

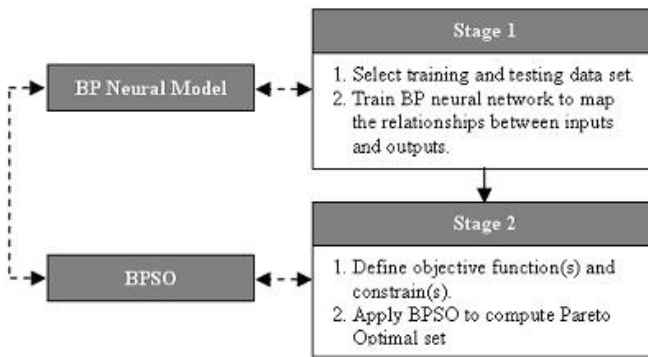


Fig. 2. Algorithm of the Proposed Approach

The algorithm begins when the neural network is trained with typical back propagation (BP) method to articulate the relationships between decision variables (input) and performance measures of interest (output) from the given data. Once the network is trained, its output is then manipulated by the bidirectional particle swarm optimization (BPSO) where the objective functions and constraints are defined. As a result, a Pareto optimal set corresponding to the objective function is produced.

6 Experiments and Results

In this study, the proposed approach is applied to solve a complex process parameter design problem in ceramic tile manufacturing. Due to non-availability of experimental data, designed data, assuming polynomial relationship between inputs and outputs, was used to illustrate the framework of the proposed system. A sample data is given in Table 1.

Table 1. Sample Data

% Residue	Temperature	Pressure	Porosity	Curvature
2.5	1204	55	0.23	0.19
3.5	1200	75	0.43	0.36
2.9	1198	63	0.30	0.25
3.2	1201	55	0.23	0.19
3.4	1204	64	0.31	0.25

The neuron network model for this problem uses %residue, temperature, and pressure as inputs to predict porosity and curvature. A feed forward neural network with single hidden layer is chosen and trained with Levenberg – Marquardt (LM) method for its documented training speed and robustness. Architecture of the network is displayed in Fig. 3.

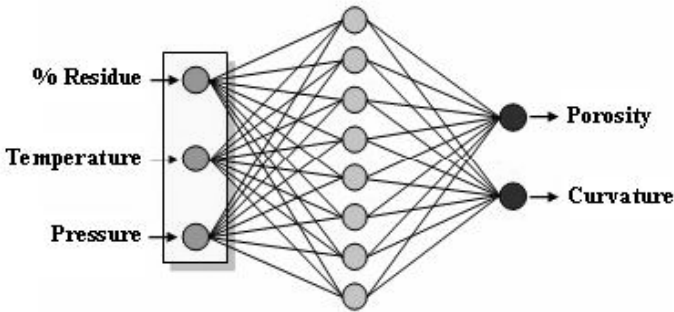


Fig. 3. Neural Network Architecture

The comparison of predicted and actual data of porosity is shown in Fig. 4.

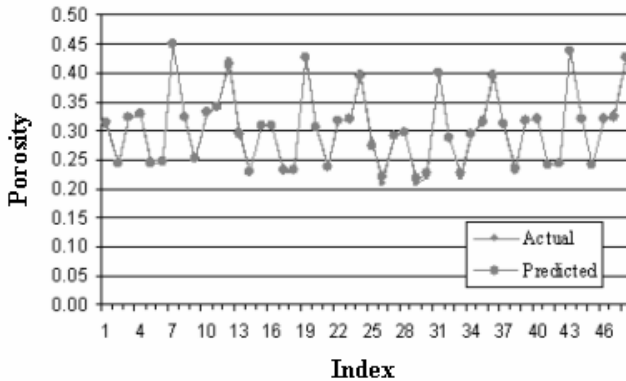


Fig. 4. Porosity Forecasted by Neural Network

Next, the following optimization problem is defined and used with BPSO. Decision variables %residue x_1 , temperature x_2 , and pressure x_3 are constrained within the specified ranges.

$$f_1 = \text{Minimize (Porosity)}$$

$$f_2 = \text{Minimize (Curvature)}$$

Subject to

$$e_1(x) = x_3^2 - 3x_2 \leq 0$$

$$e_2(x) = 3x_1^2 x_3 - x_2 \geq 0$$

Where

$$0 \leq x_1 \leq 4,$$

$$1196 \leq x_2 \leq 1210,$$

$$55 \leq x_3 \leq 81$$

The Pareto front of this problem is shown in Fig. 5. The decision makers may select the most optimal conditions among these non-dominated solutions. Some of the optimal %residue, temperature, pressure and corresponding porosity and curvature are shown in Table 2.

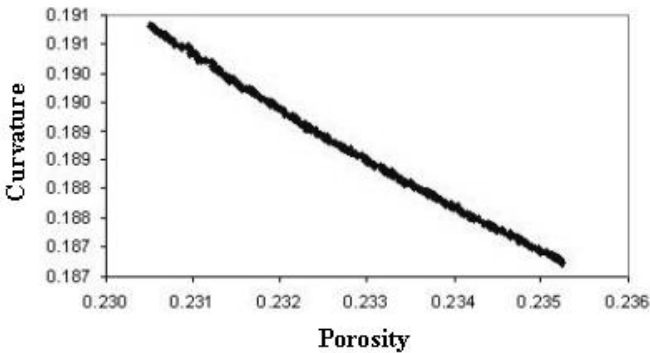


Fig. 5. Pareto Front Produces by BPSO

Table 2. Selected Optimal Solutions

% Residue	Temperature	Pressure	Porosity	Curvature
0.76	1209.88	55.02	0.22	0.20
1.19	1209.83	55.02	0.22	0.20
1.43	1209.89	55.00	0.22	0.20
2.31	1208.83	55.01	0.23	0.19
3.09	1209.11	55.01	0.23	0.19

7 Conclusions

This study presents a system integrating neural network with BPSO to obtain optimal process parameters in ceramic tile manufacturing. The neural network is utilized for modeling the relationships between decision variables and the performance measures of interest, while BPSO, on the other hand, is used to compute Pareto optimal non-dominated solutions. The methodology then is finally applied to solve a complex process parameter design problem in ceramic tile manufacturing and a group of optimal solutions for the problem are obtained. Once properly developed, such system can be used as a tool to bring the quality of product to the desired level. This is the important aspect of the research because it is with the "what if" analysis using the proposed methodology that the quality engineer can consistently and reliably design the process parameters. This eliminates the need for trial and error and ensures an acceptable quality level as well as minimizes product waste.

References

1. Coello, C.A., Lechuga, M.S.: Mopso: A Proposal for Multiple-Objective Particle Swarm Optimization. Proceedings of IEEE, World Congress on Computational Intelligence (2002) 1051-1056
2. Lam, S.Y., Petri, K.L., Smith, A.E.: Prediction and Optimization of a Ceramic Casting Process Using a Hierarchical Hybrid System of Neural Networks and Fuzzy Logic. IEEE Transactions on Design and Manufacturing (1999)
3. Saravanan, R., Asokan, P., Sachidanandam, M.: A Multi-Objective Genetic Algorithm (GA) Approach for Optimization of Surface Grinding Operations. International Journal of Machine Tools and Manufacture **42** (2002) 1327-1334
4. Cus, F., Balic, J.: Optimization of Cutting Process by GA Approach. Robotics and Computer Integrated Manufacturing **19** (2003) 113-121
5. Chiang, T., Su, C.: Optimization of TQFP Molding Process Using Neuro-Fuzzy-GA Approach. European Journal of Operational Research **147** (2003) 156-164
6. Zhao, B., Cao, Y.J.: Multiple Objective Particle Swarm Optimization Technique for Economic Load Dispatch. Journal of Zhejiang University Science (2004) 420-427
7. Hsu, C.M., Su, C.T., Liao, D.: A Novel Approach for Optimizing the Optical Performance of the Broadband Tap Coupler. International Journal of Systems Science **3** (2003) 215-226
8. Karpat, Y., Özel, T.: Hard Turning Optimization Using Neural Network Modeling and Swarm Intelligence. Transactions of North American Manufacturing Research Institute **34** (2005) 179-186
9. Parsopoulos, K.E., Vrahatis, M.N.: Particle Swarm Optimization Method in Multiobjective Problems. Proceedings of the 2002 ACM Symposium on Applied Computing (SAC 2002) (2002) 603-607
10. Hu, X., Eberhart, R.: Multiobjective Optimization Using Dynamic Neighborhood Particle Swarm. Proceedings of the IEEE World Congress on Computational Intelligence (2002) 1677-1681
11. Zhang, Y., Huang, S.: A Novel Multiobjective Particle Swarm Optimization for Buoy-arrangement Design. Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'04) (2004)
12. Srinivas, N., Deb, K.: Multi Objective Function Optimization Using Non-Dominated Sorting Genetic Algorithms. Evolutionary Computation (1995) 221-248

Application of RBF Neural Network to Simplify the Potential Based Optimization^{*}

Kanjian Zhang and Chunbo Feng

Research Institute of Automation, Southeast University,
Nanjing 210096, P.R. China
kjzhang@seu.edu.cn

Abstract. A policy iteration approach to optimal control problems for a class of nonlinear stochastic dynamic system is introduced. Some parameters and nonlinearities of the system are not required to be known a-priori. An optimality equation is developed based on performance potential. The potential can be estimated by a sample path, and then it is approximated by RBF neural network. As a result, an on-line algorithm is proposed by using a sample path of the given control system.

1 Introduction

Optimal control of stochastic dynamic systems is a difficult problem. Because the exact solution of the Bellman equation can only be obtained under rather strict restrictions on the system structure, various approximate synthesis methods have been developed [5], [6]. However, the transition probabilities or the parameters and nonlinearity have to be known in most of these methods. This is not amenable to on-line implementation.

An optimal algorithm for the discrete nonlinear stochastic system is proposed in this paper. The main concept with this approach is the performance potential [2], [3]. With the performance potential, an optimality equation is established. And an iteration procedure is provided to approximate the optimal solution. The potentials can be estimated by a single sample path without knowing any other information of the systems. No knowledge about the entire transition matrix and nonlinearities is needed. To reduce the cost of computing, radial basis function (RBF) neural networks are used for approximate the potentials and transition probabilities. Compared with the optimal algorithms in the literatures, our potential based approach can be implemented on-line without knowing all the system parameters.

2 System Description

Consider a nonlinear stochastic control system of the form

$$x_{k+1} = f(x_k) + h(x_k, u_k) + \xi_k, k = 0, 1, \dots, \quad (1)$$

^{*} This work was supported by National Natural Science Foundation of China under Grant 60404006.

where $x_k \in R^n$ is the state of the system, the disturbance $\xi_k \in R^n$ forms the sequence of independent and identically distributed(i.i.d.) random variables, and $u \in U$ is the admissible control input, U is a specified control constraint set of R^m . The function $f(x)$ is sufficiently smooth and may contain some unknown information, and $h(x_k, u_k)$ is known sufficiently smooth function. Associated with the system (1) is the cost

$$J = \lim_{N \rightarrow \infty} \frac{1}{N} E\left\{ \sum_{k=0}^{N-1} r(x_k, u_k) \right\}, \tag{2}$$

where $r : R^n \times R^m \rightarrow R$ is some suitable given function. The optimal control problem for system (1) with state feedback, is to find a control action $u_k = \alpha(x_k) \in U$ such that the performance J in (2) is minimized.

Our control problem of nonlinear stochastic system (1) will be studied in the framework of Markov process. We begin with some basic definitions of Markov process that will be needed later.

Let $\mathcal{B}(R^n)$ denotes the family of Borel sets in R^n , $b\mathcal{B}(R^n)$ is the set of bounded measurable function with respect to $\mathcal{B}(R^n)$. Let $P(x, B)$ be a transition function and $f(x) \in b\mathcal{B}(R^n)$, define a linear operator from $b\mathcal{B}(R^n)$ to $b\mathcal{B}(R^n) : Pf(x) = \int_{R^n} f(y)P(x, dy)$, such a linear operator is also written as P . For any transition function P , let $e(x) = 1$ for any $x \in R^n$, then $Pe(x) = 1, \forall x \in R^n$, or $Pe = e$.

Suppose that $x_0, x_1, \dots, x_k, \dots$ is a time-homogeneous Markov chain which takes values on the space R^n and let P be the one-step transition function of the Markov chain, i.e., $P(x, B)$ is the probability of $x_{k+1} \in B$ if $x_k = x$, we set $P^1(x, B) = P(x, B)$ and define recursively the k -step transition function $P^k(x, B)$ by $P^{k+1}(x, B) = \int_{R^n} P^k(y, B)P(x, dy)$. Then it is easy to verify that the linear operator of $P^k(x, B)$ is the product of P in power k , or P^k .

Definition 1. *The Markov chain $\{x_k\}$ on R^n is said to be ergodic if there exists an invariant probability $\pi(B) = \pi P(B), \forall B \in \mathcal{B}(R^n)$ such that $\lim_{k \rightarrow \infty} \|P^k(x, \cdot) - \pi(\cdot)\| = 0, \forall x \in R^n$, where $\|\cdot\|$ represents the total variation norm on $\mathcal{B}(R^n)$.*

Then for an ergodic Markov chain, the infinite horizon average-cost J can be given as follows

$$J(x) = \lim_{N \rightarrow \infty} \frac{1}{N} E\left\{ \sum_{k=0}^{N-1} r(x_k) | x_0 = x \right\} = \int_{R^n} r(x)\pi(dx) \tag{3}$$

We now return our attention to the nonlinear system (1). We assume that there exists a initial controller $u_k = \alpha_1(x_k) \in U$ for system (1) such that the performance is bounded. Our objective is to find control sequence $\alpha_i(\cdot), i = 1, 2, \dots$, by policy iteration, such that the corresponding performance defined by (2) improves step by step, or J tends smaller and smaller. While the sequence $\{x_k\}$ obtained from the system (1) with the control law $u_k = \alpha_i(x_k)$ forms a time homogeneous Markov chain. By the discussion in [1], if the closed system formed by the system (1) and $u_k = \alpha_i(x_k)$ is stable, or $E(\|x_k\|)$ is bounded,

then the Markov chain is ergodic with some mild assumption. On the other hand, the assumption that if the performance J of the system is bounded then the closed system is stable is usually needed. Since our control law will make the J improves step by step, boundedness of J can be ensured. Therefore, starting from the controller $u_k = \alpha_1(x_k)$, we assume that the ergodic condition is always satisfied in the iteration procedure.

3 Optimization Based on Potential

For ergodic Markov chain $\{x_k\}$ and its transition function P , define P^* as follows $P^* = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{n=0}^{N-1} P^n$. The performance potential g is defined as

$$(I - P + P^*)g = r. \tag{4}$$

Then by the definition [\(1\)](#) and the dominated convergence theorem we have

Lemma 1. P^* exists and the following equations are satisfied:

1. $PP^* = P^*P = P^*P^* = P^*, P^*e = e;$
2. $Je = P^*r, PJe = Je;$
3. $Je = P^*g,$ and

$$Je + (I - P)g = r. \tag{5}$$

Here J is defined by [\(3\)](#).

Lemma 2. For any $r \in b\mathcal{B}(R^n)$,

$$g = \sum_{k=0}^{\infty} (P^k - P^*)r \tag{6}$$

is a solution of [\(4\)](#).

[\(5\)](#) is called as the Possion equation. Its solution is only up to an additive constant, i.e., if g is a solution to [\(5\)](#), then so is $g + ce$ with any constant c .

Let $u = \alpha(x)$ and $\tilde{u} = \tilde{\alpha}(x)$ be two control laws, P, \tilde{P} and r, \tilde{r} are corresponding transition functions and performance functions respectively. Then, the following results can be established by the above two lemma.

Lemma 3. The following equation is satisfied:

$$(\tilde{J} - J)e = \tilde{P}^*[(\tilde{r} + \tilde{P}g) - (r + Pg)].$$

Here P and \tilde{P} are the two transition functions defined above, g and \tilde{g} are the corresponding solution of [\(4\)](#).

Theorem 1. A control law $u^* \in U$ is optimal if and only if

$$P^{u^*}g^{u^*} + r^{u^*} \leq P^u g^u + r^u, a.e., \forall u \in U. \tag{7}$$

Where $(\cdot)^{u^*}$ and $(\cdot)^u$ mean that the corresponding Markov chains are obtained by the system [\(1\)](#) under the control laws u^* and u respectively.

Theorem 1 shows that the control law u^* is optimal if and only if $P^{u^*} g^{u^*} + r^{u^*} = \min_u \{P^u g^{u^*} + r^u\}$. Thus, we can construct the controller by the optimality equation in Theorem 1 as in 2. Roughly speaking, at the k th step with control $\tilde{u} = \alpha_k(x)$, we can set the control law for the next step as

$$\alpha_{k+1}(x) = \arg \min_u \{P^u g^{\tilde{u}}(x) + r^u(x)\}, \tag{8}$$

with $g^{\tilde{u}}$ being any solution to the Poisson equation for transition function $P^{\tilde{u}}$. This results in an iteration procedure. From Lemma 3, the performance improves in such a procedure. Theorem 1 shows that the minimum is reached when no performance improvement can be further achieved. The implementation details of the above results will be discussed in the paper.

4 On-Line Algorithm

Theorem 1 shows that the optimal control law of the system (1) with performance (2) can be approximated by (8) step by step. However, exact solution of the performance potential g can not be obtained in general. We shall present a numerical approach to estimate the performance potential g by sample path of the given system.

For simple discussion, we consider one dimensional system as a illustration. We discretize the state space with a small spatial step Δ which is to be given in advance. That is, the state space is divided into $[i\Delta, (i + 1)\Delta)$ with $i = \dots, -1, 0, 1, \dots$.

From (6), we have $g(x) = E\{\sum_{k=0}^{\infty} (r(x_k) - J) | x_0 = x\}$. From deduction of Lemma 2, the right side of the above equation converges. Thus, we have

$$\frac{\int_B g(x)\pi(dx)}{\int_B \pi(dx)} = E\left\{\sum_{k=0}^{\infty} (r(x_k) - J) | x_0 \in B\right\} \approx E\left\{\sum_{k=0}^M (r(x_k) - J) | x_0 \in B\right\} \tag{9}$$

with a sufficiently large constant M . Therefore, from (9) if $g(x)$ is continuous, then $g(i\Delta)$ can be approximated by $g(i\Delta) \approx E\{\sum_{k=0}^M r(x_k) | x_0 \in B_i\} - MJ$, where the set B_i is defined as $[i\Delta, (i + 1)\Delta)$. Since g is still called performance potential if a constant MJ is added to it, we can ignore the constant MJ . Thus, ignoring the constant MJ , the solution of the Poisson equation (5) can be approximated by $g(i\Delta) \approx E\{\sum_{k=0}^M r(x_k) | x_0 \in B_i\}$. Since the Markov chain of the closed system is ergodic, as in 2, this leads to

$$g(i\Delta) \approx \lim_{K \rightarrow \infty} \left\{ \frac{\sum_{k=0}^{K-M-1} \sum_{j=0}^{M-1} r(x_{k+j})}{\sum_{k=0}^{K-L-1} \epsilon_i(x_k)} \right\}, \epsilon_i(x) = \begin{cases} 1 & x \in B_i \\ 0 & otherwise \end{cases} \tag{10}$$

Because the number of $x_k < x$ and $x_{k-1} \in B_i$ can be obtained from sample path, the probability distribution $P(B_i, \cdot)$ can be estimated by the general statics approach.

Although the above process of estimating the function P and g is simple, the cost of computing is huge. Estimating the potential $g(x)$ with a great lots of discrete states may lead to curse of dimension, especially for high dimension system. Even if the probability distribution P and the potential g are obtained, integral on the function P and g must be calculated to search the optimal control by (8). The computational cost is also large.

To lower the cost of computing, we construct RBF neural networks [4] from the above analysis to approximate the potential $g_i(x)$ by (10) and probability density function $p_i(x, y)$ from the sample path. That is

$$g_i(x) = \psi_i(x)^T \theta_i, p_i(x, y) = \phi_i(x, y)^T \eta_i. \tag{11}$$

Where θ and η are weights, $\psi_i(x) = [\psi_{i1}(x) \psi_{i2}(x) \cdots \psi_{it}(x)]$ and $\phi_i(x, y) = [\phi_{i1}(x, y) \phi_{i2}(x, y) \cdots \phi_{is}(x, y)]$ are Gaussian activation functions with

$$\psi_{ij}(x) = e^{-\frac{\|x - \mu_{ij}\|^2}{\sigma_{ij}^2}}, \phi_{ij}(x, y) = e^{-\frac{\|x - \lambda_{ij}\|^2}{\tau_{ij}^2}}, \bar{x} = \begin{bmatrix} x \\ y \end{bmatrix},$$

t and s are the number of hidden units. The number i index the iterative step.

Then, with control $u = \alpha_i(x)$ at the i th step, we obtain the potential $g_i(x)$ and probability density function $p_i(x, y)$ as in (11). Since $\xi_k \in R^n, k \in \mathcal{N}$, form i.i.d. random variables, it follows that $P_i(x, B) = P_\xi\{y - f(x) - h(x, \alpha_i(x)) | y \in B\}$, where P_ξ represents the distribution of the random variable ξ . Since the function $h(x_k, u_k)$ is known, from (11) P^u can be calculated by $P^u(x, B) = P_i(x, \bar{B}), \bar{B} = \{y - h(x, u) + h(x, \alpha_i(x)) | y \in B\}$. This leads to $p^u(x, y) = p_i(y - h(x, u) + h(x, \alpha_i(x)))$. Thus, for the right side of (8), we have

$$\begin{aligned} P^u g_i(x) + r^u(x) &= \int_{R^n} g_i(y) p_i(x, y - h(x, u) + h(x, \alpha_i(x))) \\ &= \int_{R^n} \theta_i^T \psi_i(y) \phi_i(x, y - h(x, u) + h(x, \alpha_i(x)))^T \eta_i dy \\ &= \theta_i^T \left[\int_{R^n} \psi_{ij}(y) \phi_{il}(x, y - h(x, u) + h(x, \alpha_i(x)))^T dy \right]_{j1} \eta_i \end{aligned} \tag{12}$$

Since $\frac{\|y - \mu_{ij}\|^2}{\sigma_{ij}^2} + \frac{\|z - \lambda_{il}\|^2}{\tau_{il}^2}$ can be expressed by $\frac{\sigma_{ij}^2 + \tau_{ij}^2}{\sigma_{ij}^2 \tau_{ij}^2} \|y - h_1(x, u)\|^2 + h_2(x, u)$ with suitable function $h_1(x, u)$ and $h_2(x, u)$, where $z = \begin{bmatrix} x \\ y - h(x, u) + h(x, \alpha_i(x)) \end{bmatrix}$.

It can be obtained that

$$\int_{R^n} e^{-\frac{\sigma_{ij}^2 + \tau_{ij}^2}{\sigma_{ij}^2 \tau_{ij}^2} \|y - h_1(x, u)\|^2} = 2^{\frac{n-1}{2}} \pi^{\frac{n}{2}} \frac{\sigma_{ij} \tau_{ij}}{\sqrt{\sigma_{ij}^2 + \tau_{ij}^2}},$$

since u is a feedback control which only depends on x . Thus, we can obtain from (12) that there exists a known value function $\bar{h}_i(x, u)$ such that $p^u(x) g_i(x) + r^u(x) = \bar{h}_i(x, u)$. Furthermore, the better control can be taken by (8) as follows:

$$\alpha_{i+1}(x) = \arg \min_u \bar{h}_i(x, u). \tag{13}$$

From the above discussion, the following algorithm can be introduced with a given positive constant ε :

1. Select a control law $\alpha_1(x)$ such that the closed system is bounded and set $i = 1$;
2. Obtain the sample path from the closed system, approximate the potential $g_i(x)$ by RBF neural network as in (11);
3. Approximate the probability density $p_i(x, y)$ from the sample path as in (11);
4. Obtain the value function $\hat{h}_i(x, u)$ and solve the equation (13);
5. If $\|\alpha_i(x) - \alpha_{i-1}(x)\| > \varepsilon$ then set $i = i + 1$ and return to step 2.

It is worth to point out that selecting the Gaussian functions as basis is rational, since the integral Pg can be omitted on the space that $\|x\|$ large enough by convergence property of Pg . Since the function $\hat{h}_i(x, u)$ is a value in R , there exist many approaches to solve (13) in literatures. We do not discuss it furthermore. Since the function P and g are estimated by the sample path, the above algorithm is valid even if the function $f(x)$ in the system (1) does not be known a-priori.

5 Conclusions

In this paper, we presented an algorithm for solving the optimal control problems of nonlinear stochastic dynamic systems. The algorithm can be implemented on-line. Compared with some existing approaches, our approach offers some flexibility and may save computation and does not require the information of the nonlinear function $f(x)$ in the system (1). RBF neural networks are introduced to approximate the probability density and performance potentials to lower the cost of computing. This simplify the potential based optimality equation.

References

1. Brockwell, A., Borovkov, K., Evans, R.: Stability of an adaptive regulator for partially known nonlinear stochastic systems. *Siam J. Contr. Optim.* **37** (1999) 1553-1567
2. Cao, X.R.: A unified approach to Markov decision problems and performance sensitivity analysis. *Automatica* **36** (2000) 771-774
3. Cao, X.R.: From perturbation analysis to Markov decision processes and reinforcement learning. *Discrete Event Dynamic Systems: Theory and Applications* **13** (2003)9-39
4. Haykin,S.: *Neural Networks, A Comprehensive Foundation*. Macmillan, New York: NY, 1994
5. Kolosov, G.E.: *Optimal Design of Control Systems: Stochastic and Deterministic Problem*. New York : M. Dekker, 1999
6. Kushner, H.J., Paul, G.: *Numerical methods for stochastic control problems in continuous time*. New York: Springer-Verlag, 1992

Satisficing Approximation Response Model Based on Neural Network in Multidisciplinary Collaborative Optimization

Ye Tao^{1,2}, Hong-Zhong Huang³, and Bao-Gui Wu¹

¹ Key Laboratory for Dalian University of Technology Precision & Non-traditional
Machining of Ministry of Education, Dalian, Liaoning 116023, China

² School of Information Eng., Dalian Fisheries University, Dalian, Liaoning, 116023, China
taoye18@163.com

³ School of Mechatronics Eng., University of Electronic Science and Technology of China,
Chengdu, Sichuan 610054, China
hzhuang@uestc.edu.cn

Abstract. Collaborative optimization (CO), one of the multidisciplinary design optimization (MDO) approaches, is a two-level optimization method for large-scale and distributed-analysis engineering design problem. In practical application, CO exists some known weaknesses, such as slow convergence, complex numerical computation, which result in further difficulties when modeling the satisfaction degree in CO. This paper proposes the use of approximation response model in place of discipline-level optimization in order to relieve the aforementioned difficulties. In addition, a satisficing back propagation neural network based on multiple-quality and multiple-satisfaction mapping criterion is applied to the design of the satisfaction degree approximation for disciplinary objective. An example of electronic packaging problem is provided to demonstrate the feasibility of the proposed method.

1 Introduction

Large-scale and complex systems are usually concerned with different special disciplines and lots of coupling factors. However, when applying conventional optimization methods, mathematical models are considerably intricate. Further, since optimization strategies are often in serial, they lead to lower efficiency and hardly acquire the optimal solutions. Multidisciplinary design optimization is an approach to decompose single system model into smaller units, which can be assigned to different groups of engineers and experts in related areas. Collaborative optimization was therefore developed to follow the multidisciplinary characteristics of engineering design. CO decomposes a single system into two levels, i.e., system-level and disciplinary-level in parallel. In the application of CO, Braun and Powell [1] employed it for launch-vehicle design, Sobieski and Kroo [2] for aircraft configuration. In addition, Tappeta and Renaud [3] presented MOCO.

Recently, CO has also been widely used in decision-making and conceptual design. However, the decomposition of CO results in some problems, such as the increase of

computational time. Different from single level optimization, the bi-level architecture of CO requires each disciplinary optimization to be performed once so as to evaluate the compatibility constraints of the system level optimization. At the same time, to calculate the coupling output variables, the disciplinary level also needs complex disciplinary analyses, which consumes more time.

As an alternative to eliminate the above problems, the use of approximation model has been proposed in place of the disciplinary design. This model was initially presented by Sobiebski *et al.* through directly modeling of the discrepancy function and optimal interdisciplinary design variables as a function of the target variables [4]. After that, Alexandrov *et al.* [5] employed so-called trust region approach to design the response surface. The advantage of using approximation is to reduce the dimensionality in analysis system. On the other hand, some global optimization methods that involve time-consuming stochastic computation, such as genetic algorithm (GA) and simulated annealing (SA) method, can efficiently be applied in CO to solve the convergence problem [6].

This paper proposes the use of approximation response model as part of discipline-level's conventional model in order to construct satisfaction degree response model for disciplinary objectives. Typical approximation response architecture of satisficing CO based on asymmetric fuzzy model is developed, and artificial neural network using multiple-satisfaction criterion is presented to design the satisfaction degree functions. The strategy provides approximation response models of MOCO with the ability to handle satisficing approach. In algorithm, this strategy quotes the approach suggested by Balling and Wilkinson [7] and formulation II method by Tappeta and Renaud [3].

2 Satisficing Collaborative Optimization Using Approximation Response Model

Since CO usually contains multiple objectives, the overall satisfaction degree can be used as the system-level objective, and at the same time, the discipline-level objective adopts the inequalities of subspace sufficiency degree as its local constraints. This typical model is attributed to satisficing problem based on asymmetric fuzzy models, as shown in Fig.1. Note that there are approximation models between system optimizers and subspace optimizers. In fact, the approximation models are the mapping functions for system-level design variables, i.e., $d_i^* = f_i(Z^*)$. When system-level optimizer delivers the target valuables to approximation response, the response surface is able to compute the discrepancy function value and return it to the system. The approximation response model can efficiently avoid the fact that system optimizer has to use the disciplinary optimizer at its each optimization iteration to acquire the discrepancy function values and keep the interdisciplinary constraints consistent.

In the response model for satisficing CO, there are two kinds of approximations, namely, conventional and satisficing approximation, respectively. The conventional approximation has the responsibility to estimate the shared and auxiliary variables, which can be solved efficiently by the method presented by Jang *et al.* [6] and Sobiebski *et al.* [4]. In view of satisficing theory, this paper adopts a novel model, i.e., satisficing approximation and uses it as satisfaction degree to map disciplinary

objectives. That is to say, d_i^* comprises two kinds of output values: (1) interdisciplinary design variables $(x_{sh})_i, (x_{aux})_i$ and (2) overall satisfaction degree $s_{\bar{F}_i}$ of disciplinary objective.

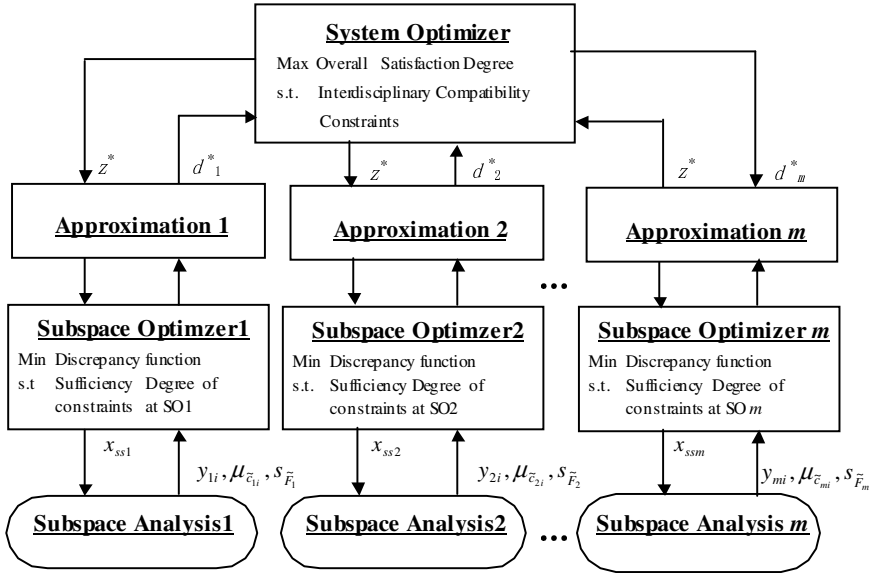


Fig. 1. The architecture of approximation for satisficing CO

2.1 Fussy Satisfaction Degree for Objectives

In 1947, Simon, awarded Nobel Prize, first introduced satisficing criterion and proposed the satisfactory solutions in place of the traditional optimal ones in some situations, which provides a new approach to solve the optimization problem [8]. Since then, satisficing theory has been extensively studied and applied. Takatsu presented the basic mathematical theory and characteristics [9]. In engineering application, Goodrich studied the theory of satisficing control, and applied it to some classic control problems [10].

Definition 1 [11]. Given a satisfactory feasible solution set $Z, Z \subset \mathbf{R}^n$, a multivariate function can be defined as follows:

$$\left. \begin{aligned} f : Z &\rightarrow Q \\ q = f(z) &\in Q, z \in Z \end{aligned} \right\}. \tag{1}$$

The function f is called quality criterion function. Q is the quality set, and q is used to describe the quality of solution $z, z \in Z$.

Definition 2. Given a feasible solution set $Z \subset \mathbf{R}^n$, quality set Q , a mapping function $h(\cdot)$ can be defined as follows:

$$\left. \begin{aligned} h : Q &\rightarrow [0,1] \\ s_{\tilde{F}}(x) &= h(q) = h(f(z)) \\ \forall z \in Z, q \in Q, s_{\tilde{F}}(z) &\in [0,1] \end{aligned} \right\}, \tag{2}$$

where \tilde{F} means satisfactory solution set of Z for f . $s_{\tilde{F}}(\cdot)$ is the satisfactory function of \tilde{F} , and $s_{\tilde{F}}(z)$ is called the satisfaction degree of z , $z \in Z$. For simplicity, the satisfactory solution set \tilde{F} may be denoted as $\{Z, f, h\}$. f , h represent quality criterion mapping and satisfactory mapping, respectively. In application, Z , f , and h are often called three fundamental elements in satisfactory solution set. If there exists multiple-quality criterion mappings or multi-satisfactory mappings, \tilde{F} may be denoted as $\{Z, F, h\}$ or $\{Z, F, H\}$.

For the feasible solution set Z , the function $f : Z \rightarrow Q$ can map solution set Z to quality set Q . In order to evaluate the satisfaction degree of quality set Q , different mapping functions $H : Q \rightarrow [0,1]$, $H = \{h_1, h_2, h_3, \dots, h_{k-1}, h_k\}$ are used to describe different satisfaction degree of objective, which is called multiple-satisfaction criterion problem. If quality mapping function f includes multiple criteria, f is represented as F , i.e., $F = \{f_1, f_2, f_3, \dots, f_{m-1}, f_m\}$. The satisficing model belongs to so-called multiple-quality and multiple-satisfaction mapping problem. In CO, f is called disciplinary objective, and m, k represent the total number of discipline-level objectives and satisfaction criterion, respectively.

2.2 The Satisficing Approximation Response Model Based on ANN

2.2.1 Topological Architecture of Satisficing Mapping Using ANN

Multi-layer back propagation artificial neural networks (BPANN) are a layered parallel processing system consisting of input, output, and hidden layers. It has been theoretically proved that a three-layer BPANN with sigmoid activation function in the hidden layer and linear functions in the output layer is a universal approximator for arbitrary continuous functions, if given sufficient neurons in the hidden layer [12]. Therefore, this paper adopts the three-layer BPANN as the satisficing response models in CO. The topology of three-layer BPANN for multi-quality criterion approximation at the i th discipline is shown in Fig. 2.

The symbols in Fig.2 are defined as

$$\left. \begin{aligned} Z_i &= \{z_{i1}, z_{i2}, \dots, z_{i,n-1}, z_{i,n}\} \\ Q_i = F(Z_i) &= \{q_{i1}, q_{i2}, \dots, q_{i,m-1}, q_{i,m}\} = \{f_1(Z_i), f_2(Z_i), \dots, f_{m-1}(Z_i), f_m(Z_i)\} \end{aligned} \right\}. \tag{3}$$

Considering the situation of multi-satisfaction criterion, we define $s_h^i = f_{\text{ANN}}(Q_i)$. The $f_{\text{ANN}}(\cdot)$ is called the approximation response function of satisfaction degree in respect to satisfaction criterion H . The assumption that f_{ANN} is continuous fuzzy

function mapping will serve as the condition, so the mapping function can be arbitrarily closely approximated. The paper also uses BPANN to design the approximation response of satisfaction degree for disciplinary objectives, and at the i th discipline of CO, the satisfaction degree model based on BPANN is shown in Fig. 3.

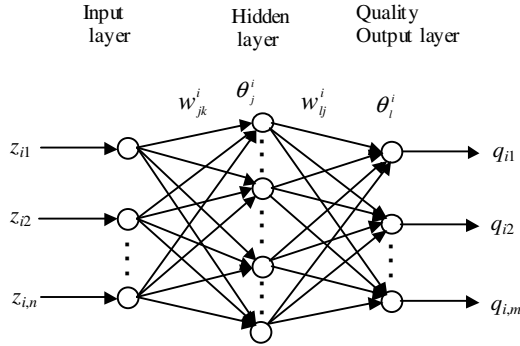


Fig. 2. The topology of BPANN for multi-quality criterion approximation at i th discipline

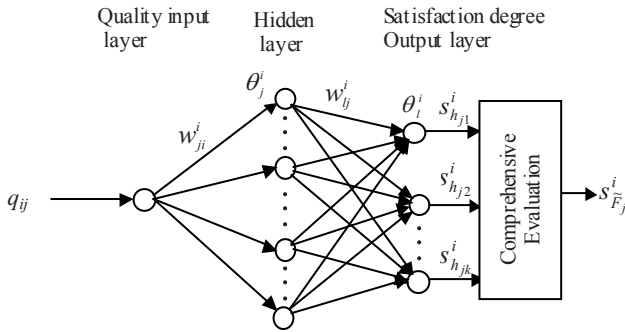


Fig. 3. The topology of BPANN for multi-satisfaction criterion approximation at the i th discipline

Note that the BPANN model approximates the satisfaction degree $s_{h_{jl}}^i$ in respect to function h_{jl} ($l=1, 2, \dots, k$). The neural network to be trained has one input nodes q_{ij} ($j=1, 2, \dots, m$) as the quality function and k output nodes for satisfaction degree at the i th discipline.

In order to consider the comprehensive influence of each satisfaction mapping function h_{jl} , this paper designs the comprehensive evaluation of satisfaction degree, and its main duty is the aggregation of different satisfaction criterion. The achievement of the discipline-level overall satisfaction degree may use the Min operator or linear weighted method.

2.2.2 Structure of Two Approximation Models

Input Layer. Given Z_i^p, Q_i^p , which represent input vector for the models of multi-quality and multi-satisfaction criterion approximation at the i th discipline, respectively, and the input and output of the j th neuron in the input layer are defined as

$$O_{ij}^p = I_{ij}^p = z_{ij}^p, O_{ij}^p = I_{ij}^p = q_{ij}^p. \quad (4)$$

Hidden Layer. Input signals of the input layer can be evaluated through the hidden layer. The input and output of the j th neuron of the hidden layer are defined, respectively, as

$$I_{ij}^p = \sum_{k=1}^n w_{jk}^i \cdot z_{ik}^p + \theta_j^i, I_{ij}^p = w_{ji}^i \cdot q_{ij}^p + \theta_j^i, \quad (5)$$

$$O_{ij}^p = f(I_{ij}^p), \quad (6)$$

where I_{ij}^p is the input of the j th neuron of the hidden layer at the i th discipline. w_{jk}^i represents a weight of a connection between the k th neuron of the input layer and the j th neuron of the hidden layer at the i th discipline. θ_j^i is the threshold of the j th neuron of the hidden layer. O_{ij}^p is the output of the j th neuron of the hidden layer.

Output Layer. Similar to the hidden layer, the input and output of the l th neuron of the output layer are defined, respectively, as

$$I_{il}^p = \sum_{j=1}^{n_H} w_{lj}^i \cdot O_{ij}^p + \theta_l^i, \quad (7)$$

$$O_{il}^p = q_{il}^p = s_{hl}^i = f(I_{il}^p), \quad (8)$$

where I_{il}^p is the input of the l th neuron of the output layer at the i th discipline. w_{lj}^i represents a weight of a connection between the j th neuron of the hidden layer and the l th neuron of the output layer at i th discipline. θ_l^i is the threshold of the l th neuron of the output layer, and O_{il}^p is the output of the l th neuron of the output layer at the i th discipline.

Activation functions. Considering $Q_i \subseteq \mathbf{R}^m$, $s_h^i \in [0,1]$, the activation function of quality output layer for multi-quality criterion approximation may adopt the following linear function

$$y = f(x) = x, \quad (9)$$

and the activation function of hidden layer and output layer of multi-satisfaction criterion approximation may use the sigmoid nonlinear function, depicted as

$$y = \frac{1}{1 + e^{-x}}. \quad (10)$$

2.2.3 The Procedure Using Approximation Response of Satisfaction Degree

1. Design topology architecture of BPANN and initial sample data, $Z_i^0, Q_i^0, s_{h_{ij}}^0$, ($i=1, 2, \dots, m, j=1, 2, \dots, k$) in the sample database at the i th discipline.
2. Train the network to build the quality approximation response of samples $Q_i = (q_{i1}, q_{i2}, q_{i3}, \dots, q_{i,m-1}, q_{im})$.
3. Train the network and establish the representation of satisfaction degree about different satisfaction mapping function h_{ij} ($i=1, 2, \dots, m; j=1, 2, \dots, k$).
4. Receive system-level's target vector Z^* , use two kinds of approximation model to obtain the satisfaction degree $s_{h_{jk}}^i$ of the j th objective in respect to h_k at the i th discipline, and save Z^* in the discipline-level local database.
5. Design the comprehensive evaluation of satisfaction degree, and compute the overall satisfaction degree $s_{\bar{F}_j}^i$ ($j=1, 2, \dots, m$), and return it to system-level.
6. After several iterations, if system-level satisfies convergence conditions then end; else according to subspace analysis, each discipline reinitializes the sample data in local database to set up the more accurate approximation models, and go to step 2.

3 Electronic Packaging Problem (EPP) Example

The electronic packaging problem (Renaud, 1993 [13]) is a benchmark multidisciplinary problem comprising the coupling between electronic and thermal subsystems. Component resistances (in electronic subsystem) are affected by operating temperatures (in thermal subsystem), while the temperature depends on the resistances. The design objective is to minimize negative watt density. According to CO method, the EPP is decomposed into two disciplines: electronic and thermal discipline. The design vectors for each discipline are as shown in Table 1. Note that the electronic subspace analysis undertakes the responsibility to gain disciplinary overall satisfaction degree. In this paper, the assumption that the decision maker uses two kinds of satisfaction criterions served as the condition to establish the satisfaction degree of disciplinary objective.

For simplicity, this paper focuses on the approximation response of electronic discipline, and the topological architecture of discipline 1 is described as follows:

1. The structure of quality approximation includes 5 neurons ($y_6, y_7, y_{11}, y_{12}, y_{13}$) in input layer, 10 neurons in hidden layer, and 1 neuron f_{11} in output layer.
2. The structure of satisfaction approximation includes 1 neuron f_{11} in input layer, 8 neurons in hidden layer and 2 neurons in output layer.

Table 1. The design vectors of each discipline in electronic packaging problem

	Electronic discipline 1	Thermal discipline 2
System targets to be matched	$\{z_1^*, z_2^*, z_3^*, z_4^*, z_5^*, s_{f_1}^{1*}\}$	$\{z_1^*, z_2^*, z_3^*, z_4^*, z_5^*\}$
Shared design vector	Empty	Empty
Auxiliary design vector	$\{y_{11}, y_{12}, y_{13}\}$	$\{y_6, y_7\}$
Local design vector	$\{x_5, x_6, x_7, x_8\}$	$\{x_1, x_2, x_3, x_4\}$
Discipline-level satisfaction criterion for objectives	$s_{h_{11}}^1, s_{h_{12}}^1$ (objective f_{11})	Empty
Subspace analysis	$[y_6, y_7, s_{f_1}^1] = SA1(x_5, x_6, x_7, x_8, y_{11}, y_{12}, y_{13})$	$[y_{11}, y_{12}, y_{13}] = SA2(x_1, x_2, x_3, x_4, y_6, y_7)$

To test the response model, we select 200 samples to learn and set up initial approximation response, and rebuild and update it every fifty iterations. After 300 iterations in CO, the comparison between exact and approximated value (f_{11}^* , f_{11}) of electronic discipline objective are shown in Table 2.

Table 2. The comparison between exact and approximated value of electronic objective

y_6	y_7	y_{11}	y_{12}	y_{13} ($\times 10^{-5}$)	f_{11}^*	f_{11}	f_i / f_{11}^*
7.9664	7.9521	83.8166	84.3835	2.500	636740.271	613180.9	0.9630
0.8414	0.8411	40.0854	40.1787	6.840	24597.1176	24105.18	0.9800
0.4417	0.4416	37.6988	37.7570	13.72	6437.6672	5993.468	0.9310
0.2980	0.2980	36.9008	36.9039	23.68	2517.1232	2446.644	0.9720
0.2242	0.2242	36.4617	36.4646	37.26	1203.2358	1191.203	0.9900
0.1792	0.1792	36.1641	36.1669	55.00	651.6042	641.8301	0.9850
0.1489	0.1489	35.9762	35.9788	77.44	384.5780	382.2705	0.9400
0.1271	0.1271	35.8513	35.8515	105.12	241.8821	236.0769	0.9760
0.0979	0.0979	35.6669	35.6672	178.36	109.8279	106.3134	0.9680

In view of ratio of two objectives, it is demonstrated that with a steady increase of iteration, the quality model has enough precision to approximate original data in the course of optimization.

Fig. 4 depicts two kinds of satisfaction degrees for sample points and approximation profiles about h_{11} , h_{12} using neural network. Note also that the satisfaction degree approximation models are fitted to different preference of sample points, which also proves the approximation model using neural network to be efficient.

In order to combine two satisfactory criteria, this paper adopts linear weighted method ($\omega_1 = 0.4$, $\omega_2 = 0.6$) to establish the overall satisfaction degree $s_{f_1}^1$ at electronic discipline. For the requirement of simplification and demonstration, this paper does not employ the optimal cuts level, but designs a series of cut level, and

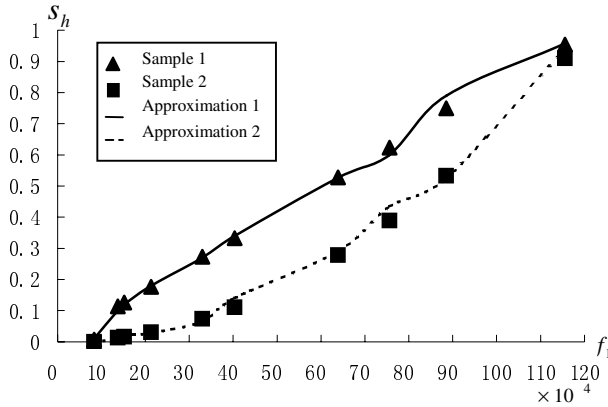


Fig. 4. Two kinds of satisfaction degrees for sample points and approximation profiles

Table 3. The result using approximation response under series of cut levels for CO

λ_1	λ_2	x_1 (m)	x_2 (m)	x_3 (m)	x_4 (m)	x_5 (Ω)	x_6 ($^{\circ}\text{K}^{-1}$)	x_7 (Ω)	x_8 ($^{\circ}\text{K}^{-1}$)	$s_{\tilde{f}_1}^1$
0.6	0.6	0.046	0.046	0.0096	0.052	9.6	0.0039	9.6	0.0039	0.548
1.0	1.0	0.050	0.050	0.0100	0.050	10	0.0040	10	0.0040	0.378
0.2	0.8	0.048	0.048	0.0098	0.051	9.2	0.0047	9.2	0.0047	0.468
0.5	0.9	0.049	0.049	0.0099	0.051	9.5	0.0041	9.5	0.0041	0.433

implements in a comparative study. CO using aforementioned approximation response method computes the optimal solutions at different cut level, respectively. The results are shown in Table 3.

If assume the data of the 3rd row is an optimal design, the watt density is $y_1 = 639779$ and the temperature at each resistor is $y_{11} = 82.37$ and $y_{12} = 82.85$. Therefore, the objective function value for this design is close to that reported by Renaud [13].

4 Conclusions

This paper focuses on the establishment of satisficing approximation response models for collaborative optimization. A satisficing model of multiple-quality and multiple-satisfaction using neural network is presented, which makes MOCO based on asymmetric fuzzy model decrease the number of the disciplinary optimization, and easier to get the satisfaction degree without time-consuming analysis. Moreover, in some complex system, especially multiple-satisfaction criterion situation, the proposed method can avoid more difficulties to provide exact arithmetic expression of disciplinary satisfaction degree. EPP example demonstrates the developed approach can efficiently deal with CO problems in fuzzy and distributed environments.

Acknowledgement

This research was partially supported by the Specialized Research Fund for the Doctoral Program of Higher Education of China under the contract number 20060614016.

References

1. Braun, R.D., Powell, R.W., Lepsch, R.A.: Comparison of Two Multidisciplinary Optimization Strategies for Launch-Vehicle Design. *Journal of Spacecraft and Rockets* **32** (1995) 404-410
2. Sobieski, J., Kroo, I.: Aircraft Design Using Collaborative Optimization. 34th Aerospace Sciences Meeting and Exhibit. Reno, Nevada, AIAA-96-0715 (1995) 15-18
3. Tappeta, R.V., Renaud, J.E.: Multi-objective Collaborative Optimization. *Journal of Mechanical Design* **119** (1997) 403-411
4. Sobieski, I.P., Manning, V.M., Kroo, I.M.: Response Surface Estimation and Refinement in Collaborative Optimization. *Proceedings of 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization* AIAA 98-4753 (1998) 359-370
5. Alexandrov, N.M., Dennis, Jr J.E., Lewis, R.M.: A Trust Region Framework for Managing the Use of Approximation Models in Optimization. *Struct Optim.* **15** (1998) 16-23
6. Jang, B.S., Yang, Y.S., Jung, H.S.: Managing Approximation Models in Collaborative Optimization. *Structural and Multidisciplinary Optimization* **30** (2005) 11-26
7. Balling, R.J., Wilkinson, C.A.: Execution of Multidisciplinary Design Optimization Approaches on Common Test Problems. *AIAA Journal* **35** (1997) 178-186
8. Simon, H A.: *The New Science of Management Decision*. China Social Sciences Press, Beijing (1998)
9. Takatsu, S.: Latent Satisficing Decision Criterion. *Information Science* **25** (1981) 145-152
10. Goodrich, M.A.: A Theory of Satisficing Decisions and Control. *IEEE Transaction Systems, Man and Cybernetics-Part A*. **28** (1990) 763-779
11. Chen, C. J., Lou, G.: Satisfactory Optimization Theory of Design in Control Systems. *Journal of UEST of China* **2** (2000) 186-189
12. Hornik, K., Stinchcombe, M., White, H.: Multilayer Feedforward Networks Are Universal Approximators. *Neural Networks* **2** (1989) 359-366
13. Renaud, J., Gabriele, G.: Approximation in Non-hierarchic System Optimization. *AIAA Journal* **32** (1994) 198-205

A New BP Network Based on Improved PSO Algorithm and Its Application on Fault Diagnosis of Gas Turbine*

Wei Hu^{1,2} and Jingtao Hu¹

¹Department of Industry Control System, Shenyang Institute of Automation Chinese Academy of Science, Shenyang 110016, China

²Graduate School of the Chinese Academy of Science, Beijing 100039, China
{huwei, hujingtao}@sia.cn

Abstract. Aiming at improving the convergence performance of conventional BP neural network, this paper presents an improved PSO algorithm instead of gradient descent method to optimize the weights and thresholds of BP network. The strategy of the algorithm is that in each iteration loop, on every dimension d of particle swarm containing n particles, choose the particle whose velocity decreases most quickly to mutate its velocity according to some probability. Simulation results show that the new algorithm is very effective. It is successful to apply the algorithm to gas turbine fault diagnosis.

1 Introduction

Gas turbines have been widely applied in many fields such as national defense, aviation, and electric power. Accurate fault detection and diagnosis in gas turbines can significantly reduce maintenance costs associated with unanticipated disaster due to engine failures, and incipient fault detection in gas turbines is vitally important to improving the safety and reliability of gas turbine.

Artificial neural networks (ANN) have been widely used in fault diagnosis systems due to their good inner adaptability. Among the various ANN, the back-propagation (BP) algorithm is one of the most important and widely used algorithms and has been successfully applied in many fields [1]. However, the conventional BP algorithm suffers from some shortcomings, such as slow convergence rate and easily sticking to a local minimum. Hence, the researching on improving the BP algorithm is always hot.

This paper presents a new BP network which is based on the improved Particle Swarm Optimization (IPSO) algorithm. The new method can improve the convergence performance of BP network obviously.

2 Improved PSO Algorithm

Particle Swarm Optimization (PSO) was first proposed by Kenney and Eberhart [2], [3] based on the metaphor of social behavior of birds flocking and fish schooling in

* This project was supported by National 863 High-Tech, R&D Program for CIMS, China (Grant No. 2003AA414210) and Shenyang Science and Technology Program (Grant No. 1053084-2-02).

search for food. PSO is a relatively novel approach for global stochastic optimization. PSO is conceptually very simple, and is strong robust and excellent ability of global exploration. Using PSO to optimize the parameters of the BP neural network can improve the convergence ability of the BP neural network, and overcome shortcomings of the BP neural network mentioned above.

Aiming at the case that conventional PSO (CPSO) algorithm could easily stick to local minimum and converge too early, many researchers have proposed different IPSO algorithms to enhance the global optimization ability of the PSO algorithm. Literature [4] added an inertial weight ω in CPSO velocity iteration formula. The researching on the ω finds that the bigger ω is the easier for particle escaping from local minimum and the smaller ω is the better for algorithm converging, so the author proposed a self-adaptive adjusting ω method to improve the performance of PSO. Literature [5] proposed a new PSO algorithm, which is based on the velocity mutation. The mutation strategy is that in each iteration loop, on every dimension d of particle swam containing n particles, find the smallest velocity $v_{T,d}$ of the absolute value of speed, then mutate it according to some probability, and make $v_{T,d}$ distribute on $[-v_{\max}, v_{\max}]$ stochastically, evenly.

This paper presents an improved velocity mutation PSO algorithm, which is based on the acceleration of particle. The mutation strategy based on the acceleration of particle is that in each iteration loop, on every dimension d of particle swam containing n particles, the velocity of the particle whose velocity decreases most quickly is mutated according to some probability, and make it distribute on $[-v_{\max}, v_{\max}]$ stochastically, evenly.

The strategy could be described as:

Find out the acceleration on dimension d of each particle.

$$a_{i,d} = -\frac{v_{i,d}^k - v_{i,d}^{k-\Delta k}}{\Delta k}, \tag{1}$$

where $i = 1, 2, \dots, n$, $d = 1, 2, \dots, D$, Δk is the number of variational iteration, $v_{i,d}^k$ is the velocity in dimension d of particle i at iteration k , $v_{i,d}^{k-\Delta k}$ is the velocity in dimension d of particle i at iteration $k - \Delta k$.

Define $a_{I,d} = \max\{a_{1,d}, a_{2,d}, \dots, a_{n,d}\}$, $I \in [1, n]$. When starting the mutation a mutation rate *Rate* and a stochastic number *Rand* should be set first. Neither bigger *Rate* nor smaller *Rate* does good to the algorithm, this paper uses the *Rate* mentioned in literature [5]. If *Rand* > *Rate*, mutate the velocity $v_{I,d}$ (particle I , dimension d), and distribute the $v_{I,d}$ on $[-v_{\max}, v_{\max}]$ stochastically.

When particle converges nearby the local minimum or global minimum, the velocity of particle would decrease even to zero, there would be a great change of

velocity in that process. The acceleration of particle reflects the change trend of particle velocity, it could foretell the increase or decrease in particle velocity also could foretell the degree of increase or decrease in particle velocity. The particle whose velocity decreases most quickly indicates that the particle is flying to the local minimum and it may be converge too early, so choosing the particle to mutate its velocity could make it explore in more extended space and avoid it sticking to local minimum. The convergence performance can be improved obviously.

This paper used two classical test functions, Rosenbrock and Rastrigin, to compare the convergence performance of the IPSO algorithm proposed by this paper with the convergence performance of the CPSO algorithm.

Rosenbrock is a function of one peak value, when $x = (1, 1, \dots, 1)$ it reaches its global minimum: $f(x) = 0$.

$$f(x) = \sum_{i=1}^n (100(x_{i+1} - x_i)^2 + (x_i - 1)^2), \quad -10 < x_i < 10. \tag{2}$$

Rastrigin is a function of multiple peak values, when $x = (0, 0, \dots, 0)$ it reaches its global minimum: $f(x) = 0$.

$$f(x) = \sum_{i=1}^n (x_i^2 - 10 \cos(2\pi x_i) + 10), \quad -5.12 < x_i < 5.12. \tag{3}$$

In the test, swarm size was set to 30; dimension was set to 10; v_{\max} was set to the initial upper limits according to the two test functions; c_1 and c_2 were both set to 2. Additionally, in IPSO algorithm, *Rate* was set to 0.001, Δk was set to 50.

Defined $|F_{best} - \bar{F}| = E$, F_{best} was the fitness value of the algorithm, \bar{F} was the global minimum. E was the error. The precision was set to 0.1. The two test functions were used to test every algorithm for 50 times, and the average iteration times according to the two test functions respectively could be expressed as Table.1, also the convergence performance could be illustrated by Fig.1.

Table 1. Comparison results between the IPSO algorithm and the CPSO algorithm

Function	IPSO	CPSO
Rosenbrock	375	400
Rastrigin	719	1083

Table.1 and Fig.1 show that, for the function Rosenbrock, the two algorithms have the equivalent convergence performance, so both the two methods have the good ability of local exploring, however, for the function Rastrigin, the IPSO has a better

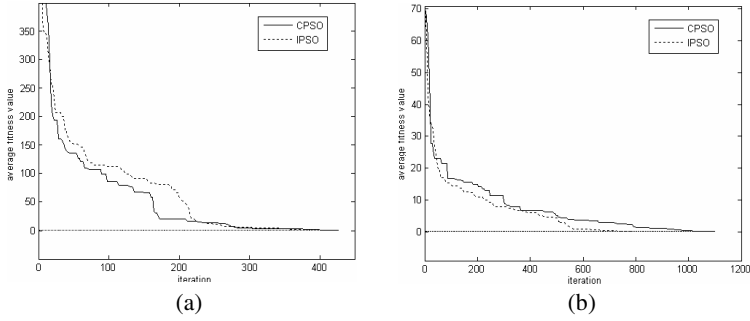


Fig. 1. Convergence performance comparison between the CPSO and the IPSO. (a) The curves of optimizing function Rosenbrock. (b) The curves of optimizing function Rastrigin.

convergence performance than the CPSO obviously, namely, the IPSO has a better global exploring performance than the CPSO. The test indicates that the IPSO has a perfect performance in both global exploring and local exploring, and the IPSO can avoid the particle from sticking to local minimum effectively.

3 BP Neural Network Based on IPSO Algorithm

BP neural network has the excellent abilities of self-adaptive and self-learning, and the researchers’ attentions are always attracted by it in fault diagnosis field. However BP neural network has its inner shortcomings, such as, easily sticking to local minimum, slow convergence rate, sensitivity to original weights and thresholds in algorithm training. This paper presents a new BP learning method based on IPSO algorithm to overcome the shortcomings above. The algorithm uses IPSO instead of gradient descent to optimize the parameters of neural network, and it has the better performance of learning rate and convergence than conventional BP algorithm.

This paper utilizes IPSO algorithm to optimize the weights and thresholds of BP neural network, and the process can be indicated as follows:

Step 1: Initialization. n_i is the number of input nerve cells, n_h is the number of hidden nerve cells, n_o is the number of output nerve cells.

$$D = n_h + n_o + n_i \times n_h + n_h \times n_o, \tag{4}$$

where D is the number of dimensions in the swarm.

Step 2: Set fitness function of the particle, this paper chooses mean square error as fitness function in BP network,

$$E = \frac{1}{M} \sum_k^M \sum_{j=1}^{n_o} (y_{k,j} - \bar{y}_{k,j})^2, \tag{5}$$

Where $y_{k,j}$ is the theoretic output j of sample k in the neural network, $\bar{y}_{k,j}$ is the real output j of sample k of neural network, M is the sample number of the network.

Step 3: Optimize the weights and thresholds of BP network using the IPSO algorithm.

Step 4: Obtain the optimized weights and thresholds according to formula 6.

$$g_{best} = [h_1, h_2, \dots, h_{n_h}, o_1, o_2, \dots, o_{n_o}, ih_1, ih_2, \dots, ih_{n_i \times n_h}, ho_1, ho_2, \dots, ho_{n_h \times n_o}], \quad (6)$$

Where $h_i (i = 1, 2, \dots, n_h)$ is the threshold i of the hidden layer, $o_i (i = 1, 2, \dots, n_o)$ is the threshold i of the output layer, $ih_i (i = 1, 2, \dots, n_i \times n_h)$ is the weight i between input layer and hidden layer, $ho_i (i = 1, 2, \dots, n_h \times n_o)$ is the weight i between hidden layer and output layer.

According to the algorithm mentioned above, this paper trained the BP network using a set of simulated fault data from the gas turbine, and compared the convergence performance between the BP network based on IPSO and conventional BP network which used the same simulated fault data. Both the two neural networks included 4 neural cells in input layer, 10 neural cells in hidden layer and 8 neural cells in output layer. The error was 0.01. Additionally, in network based on IPSO algorithm, swarm size was set to 20, dimension was set to 138, v_{max} was set to 5, c_1 and c_2 were both set to 2, *Rate* was set to 0.001, Δk was set to 50.

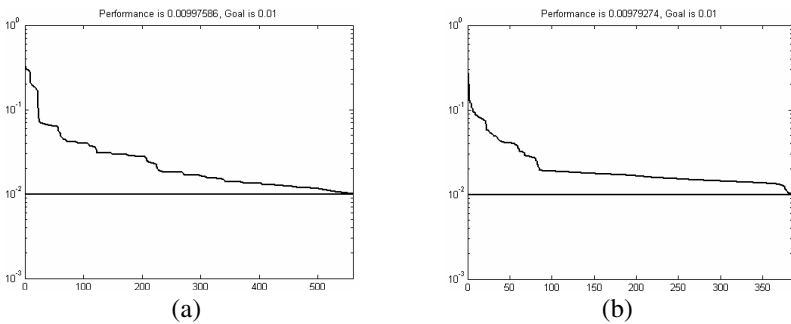


Fig. 2. The convergence curves of the two algorithms. (a) The convergence curve of conventional BP network. (b) The convergence curve of BP network based on IPSO algorithm.

Fig.2 includes two convergence curves of the two methods respectively. (a) shows that conventional BP algorithm converged after 560 iterations with the error 0.01. (b) shows that BP neural network based on IPSO algorithm converged after 385 iterations with the error 0.01. These figures indicate that BP neural network based IPSO has a faster convergence velocity than the conventional BP neural network, and the BP neural network based IPSO has a perfect convergence performance.

4 Fault Diagnosis of Gas Turbine Based on the New BP Network

The working environment of gas turbine components is high temperature and high pressure, and it is possible that some faults occur to these components due to the influence of high temperature and high pressure. It can either bring the huge loss in economy or threaten the safety of people. So the fault diagnosis of gas turbine is very significant. The faults of gas turbine can be classified as: Compress Air Machine Begriming (CAMB); Compress Air Machine Abrasion (CAMA); Compress Air Machine Laminae Mar (CAMLML); Turbine Nozzle Eroding (TNE); Turbine Laminae Abrasion (TLA); Turbine Laminae Begriming (TLB); Turbine Laminae Mar (TLM); Firebox Distortion (FD).

Literature [6], [7] summarized the fault criterion of gas turbine which is the most important thing in fault diagnosis system. Based on the fault criterion of gas turbine, literature [8] concluded the mapping relation between the fault characters ($\Delta\pi_c$: Compress air machine pressure ratio; ΔT_2 : Exhausting air temperature of compress air machine; ΔT_4 : Exhausting air temperature of turbine; ΔW_l : Supply oil quantity of gas turbine) and the 8 faults.

According to the mapping relation between fault characters and fault outputs, this paper applied the BP neural network based on IPSO algorithm and the conventional BP neural network respectively to fault diagnosis of the gas turbine. The configurations of the two BP networks were the same with what mentioned in chapter 3. In each network, chose the 4 fault characters as inputs, and chose the 8 faults as outputs. The applying chose 30 sets of simulated fault samples for every fault type individually, and the sample sum for all the 8 types was 240. 160 of the all samples were used to train every BP neural network, and the others were used to test the outputs of the two trained BP neural networks.

10 samples of each fault type were used to test each method. Table 2 is the average diagnosis accuracy of the BP network based on IPSO algorithm, and Table 3 is the average diagnosis accuracy of the conventional BP network. The results of the two tables indicate that the BP network based on IPSO algorithm has a better diagnosis performance than the conventional BP network, and the diagnosis result of the BP network based on IPSO algorithm is satisfied with the diagnosis average accuracy being 98.75%.

Table 2. Average diagnosis accuracy of the BP network based on IPSO algorithm

	CAMB	CAMA	CAMLML	TNE	TLA	TLB	TLM	FD	SUM
Sample number	10	10	10	10	10	10	10	10	80
Correct number	10	10	10	10	9	10	10	10	79
Wrong number	0	0	0	0	1	0	0	0	1
Accuracy (%)	100	100	100	100	90	100	100	100	98.75

Table 3. Average diagnosis accuracy of the conventional BP network

	CAMB	CAMA	CAMLM	TNE	TLA	TLB	TLM	FD	SUM
Sample number	10	10	10	10	10	10	10	10	80
Correct number	10	10	9	10	9	9	10	10	77
Wrong number	0	0	1	0	1	1	0	0	3
Accuracy (%)	100	100	90	100	90	90	100	100	96.25

5 Conclusions

This paper introduces a new BP network which based on IPSO algorithm and applies it to gas turbine fault diagnosis. In the method a CPSO algorithm is improved, and this paper proposes the velocity mutation algorithm based on particle acceleration. The method can effectively avoid the particle sticking to the local minimum, also it can enhance the global exploring ability for the particle. Applying the method to optimizing weights and thresholds of BP neural network can enhance the convergence performance of BP network, and utilizing the optimized BP network to diagnose the gas turbine fault can obtain a satisfied diagnosis result.

References

1. Filippetti, F., Franceschini, G., Tassoni, C., Vas, P.: Recent Developments of Induction Motor Drives Fault Diagnosis using AI Techniques. *IEEE Trans. Industrial Electronics* 47 (2000) 994-1003
2. Eberhart, R.C., Kennedy, J.: A New Optimizer using Particle Swarm Theory. *Proc. 6th Int. Symp on Micro Machine Human Science* (1995) 39-43
3. Kennedy, J., Eberhart, R.C.: PSO optimization. *IEEE Int. Conf. Neural Networks*. Perth, Australia 4 (1995) 1941-1948
4. Shi, Y., Eberhart, R.C.: A Modified Particle Swarm Optimizer. *IEEE Int. Conf. Evolutionary Computation*, Anchorage, Alaska 5 (1998) 69-73
6. Fu, G. J., Wang, S. M., Liu, S.Y., Li, N.: An Improved Velocity Mutation Particle Swarm Optimizer. *Computer Engineering and Application* 13 (2006) 48-50
7. Mueled, J.D, Taylor, V., Laflamme, J.C.G.: Implanted Component Faults and Their Effects on Gas Turbine Engine Performance. *Engineering for Gas Turbine and Power* 114 (1992) 174-179
8. Diakunchak: Performance Deterioration in Industrial Gas Turbines. *Engineering for Gas Turbine and Power* 114 (1992) 161-168
9. Weng, S. L., Wang, Y.H.: Intelligent Fault Diagnosis of Gas Turbine Based on Thermal Parameters. *Journal of Shanghai Jiaotong University* 36 (2002) 165-168

The Evaluation of BP-ISP Strategy Alignment Degree with PSO-Based ANN*

Lei Wen

Department of Economy Management, North China Electric Power University
No. 204, qingnian road, Baoding, Hebei, China071003
wenlei0312@sohu.com

Abstract. With the development of BP-ISP alignment research, how to evaluate the strategy alignment of BP-ISP become the key problem of this field. In this paper, a set of index system of evaluating the alignment of BP-ISP is established. Based on the index system, an integration algorithm with PSO and neural network is established to evaluate the alignment of BP-ISP. In order to verify the effectiveness of the method, a real case is given and BP neural network is also used to assess the same data. The experimental results show that integration algorithm with PSO and neural network is effective in the alignment evaluation of BP-ISP and achieves better performance than BP neural network.

1 Introduction

With the development and popularization of the information technology (IT), the strategy of information system planning become an important part of the business planning strategy research. The IT resource is viewed as a strategically valuable organization asset. The application of information technology (IT) can create competitive advantage based on the generic strategy of cost leadership, product differentiation. Increased attention has been given to the alignment of IS and business strategy to create the advantage. But the achievement of the IT investment is not so optimism as people's prediction. There are many reasons for it. The main reason is the lack of alignment between Business Planning (BP) and Information Systems Planning (ISP). So how to evaluate the strategy alignment degree of BP-ISP and give some advise-ment to improve the value of IT investment is the key problem of top manager of enterprise [1].

In recent years, The multi-layer feed-forward neural networks of supervised training type, trained with a back-propagation (BP) learning algorithm, are the most popular networks applied in different fields. The objective of the BP algorithm is to minimize an average sum squared error term by doing a gradient descent in the error space. BP neural network as a gradient search algorithm has some limitations associated with overfitting, local optimum problems and sensitivity to the initial values of weights[2].

* This paper was supported by Doctor Foundation of North China Electric Power University. number:20041205 And the nature science planning project of Hebei province education bureau, number: z2005117.

The particle swarm optimization (PSO), firstly proposed by Eberhart and Kennedy [3], is a computational intelligence technique. Some researchers have used PSO to train neural networks and found that PSO-based ANN has a better training performance, faster convergence rate, as well as a better predicting ability than BP neural network[4-6].

In this paper we research the evaluation index system of BP-ISP strategy alignment degree and give a evaluating model of BP-ISP strategy alignment degree with PSO-based ANN. The experiment show that the PSO-based ANN algorithms has a less iteration and a higher accuracy than BP neural network.

2 Evaluating Index System of BP-ISP Strategic Alignment Degree

In this section we introduced the strategy alignment of BP-ISP and give the index system of evaluating the alignment degree of BP-ISP.

With the development and popularization of the information technology (IT), the strategy of information system planning become an important part of the business planning strategy research. The integration of information system planning and business planning strategy can help top Manager gaining information, knowing the opportunity and challenge from out environment. Other wise, the IT investment not only can not support the business process efficiently, even become the burden of enterprise. It is commonly suggested that close integration of the organizational business plan and the IT plan is desirable and is a goal that most organizations should work towards.

The BP-ISP alignment is the linkage of the firm's IS plan and business plan. All the part of enterprise's strategy plan should be mapped the IS plan in some manner to optimize the investment, return and organization. By aligning the IS plan and the business plan, information resource support business objectives and take advantage of opportunities for the strategic use of IS. Alignment need business and IS executives to assume joint responsibility for delivering benefit from the IS investment. This alignment can create significant IS-based competitive advantage. At the same time the alignment is the reflection of IS opportunities in the business plan.

Two sub alignment of BP-ISP alignment exist. The first is the alignment of executive capability between IS department and Business department, The standardization of business process and the higher executive capability can help exerting the function of IS, at the same time, the efficient IS can help business manager making decision more quickly and correctly. The second is the alignment of strategy target between enterprise business plan strategy and the IS plan strategy. The enterprise strategy set consisting of mission, target and plan will guide the design of the IS plan and ensure alignment with the direction of the company. On the other hand, the IS plan can give the manager an efficient tool to adjust the goal of the strategy in both return and the organization.

Based on the analysis of the alignment of BP-ISP, we introduce the index system of evaluating BP-ISP strategy alignment degree. The detail context is showed in Table 1.

Table 1. Index system of evaluating the alignment degree of BP-ISP

Index number	Index name
1	The strategic value of IT
2	The communication between business department and IT department
3	The professional service capability of IT department
4	The degree of top manager trusting IT department
5	The capability of IT department understanding the business profession
6	The IT level of employee in IT department
7	The capability of IT department corresponding the demand of business department
8	The degree of top manager taking part in IS plan
9	The degree of IT department manager take part in the business plan strategy
10	IT government structure
11	The degree of business department making decision based on IS
12	The degree of IT department understanding the business practice of enterprise

3 Algorithms PSO-Based ANN

3.1 Artificial Neural Network

A multiple-layer feed-forward perceptron represents a non-linear mapping between input vector and output vector through a system of simple interconnected neurons to every node in the next and previous layer. The output of a neuron is scaled by the connecting weight and fed forward to become an input through a non-linear activation function to the neurons in the next layer of network. In the course of training, the perceptron is repeatedly presented with the training data. The weights in the network are then adjusted until the errors between the target and the predicted outputs are small enough, or a predetermined number of iteration is passed. The perceptron is then validated by an input vector not belonging to the training pairs. The training processes of ANN are usually complex and high dimensional problems.

Among neural networks, BP neural networks are the most popular type. As the BP algorithm optimizes a target function by using the gradient descent method, the calculation may overflow or fluctuate between the optima. Another problem is that the convergence of the BP is sensitive to the initial selection of the weights. If the initial sets of weights are not selected properly, the optimization solution could be trapped in a local optimum. So how to find another training methods to optimum the weight of neural network is an important task for spreading the use of neural network.

3.2 Principle of Particle Swarm Optimization

PSO simulates a popular behavior such as bird searching food in an area. Like evolutionary algorithm, PSO conducts search using a population, which is called swarm. Candidate solutions to the problem are termed particles or individuals.

Each particle adjusts its flying based on the flying experiences of both itself and its companions. During the process, it keeps track of its coordinates in hyperspace which are associated with its previous best fitness solution, and also of its counterpart corresponding to the overall best value acquired thus far by any other particle in the population.

PSO is initialized with a population of M random particles and then searches for best position (solution or optimum) by updating generations until getting a relatively steady position or exceeding the limit of iteration number. In every iteration or generation, the local bests and global bests are determined through evaluating the performances, i.e., fitness values or objectives, of the current population of particles. Each particle is treated as a point in a K -dimensional space. Two factors characterize a particle status on the search space: its position and velocity. The N -dimensional position for the i th particle in the t th generation can be denoted as

$$X_i(t) = \{x_{i1}(t), x_{i2}(t), \dots, x_{iK}(t)\}. \quad (1)$$

Similarly, the velocity also a K -dimensional vector, for the i th particle in the t th generation can be described as

$$V_i(t) = \{v_{i1}(t), v_{i2}(t), \dots, v_{iK}(t)\}. \quad (2)$$

In each iteration, every particle calculates its velocity according to the following formula:

$$V_i(t) = w(t) V_i(t-1) + c_1 r_1 (X_i^L - X_i(t-1)) + c_2 r_2 (X_i^G - X_i(t-1)) \quad (3)$$

$$X_i(t) = V_i(t) + X_i(t-1) \quad (4)$$

where $i = 1, 2, \dots, M$ and $t = 1, 2, \dots, T$; the local best position of the i th particle associated with the best fitness. encountered after $t - 1$ iterations is represented as

$$X_i^L = \{X_{i1}^L, X_{i2}^L; \dots, X_{iN}^L\} \quad (5)$$

the global best among all the population of particles achieved is represented as

$$X^G = \{X_1^G, X_2^G, \dots, X_N^G\} \quad (6)$$

c_1 and c_2 are positive constants (namely learning factors) and r_1 and r_2 are random number between 0 and 1; $w(t)$ is the inertia weight used to control the impact of the previous velocities on the current velocity, influencing the tradeoff between the global and local exploration abilities during search. Eq. (3) is used to calculate the particle's new velocity according to its previous velocity and the distances of its current position from its own best experience or position and the group's best experience or position. Then the particle flies toward a new position according to Eq. (4).

3.3 ANN Trained by PSO (PSO-Based ANN)

In this paper, we introduced a novel PSO-based ANN, PSO is used for weight training of multiple-layer feedforward neural network. In PSO, real number strings were adopted to code all the particles. Each real number coded string stands for a set of weights, which makes up of one ANN together with all its nodes. Actually, each bit of a particle is a real number that stands for a linking weight of the given ANN. The ANN trained by PSO is described as follows:

Step 1. Randomly initialize all the particle in PSO with an appropriate size of population. Each particle stands for a set of weights of the ANN.

Step 2. Calculate the fitness function of the ANN corresponding to each weight of the particle. If the best object function of the generation fulfills the end condition, the training is stopped with the results output, otherwise, go to the next step.

Step 3. Update the particle swarm according to fitness function by applying the PSO. The all the particles in the swarm will travel to a better location near the optimal value.

Step 4. Go back to the second step and then calculate the fitness of ANN with the connect weight from the renewed population.

Continue to repeat the step above until the fitness of the ANN trained by PSO satisfied the error standard.

The flow chart of PSO-based ANN is showed in Figure1.

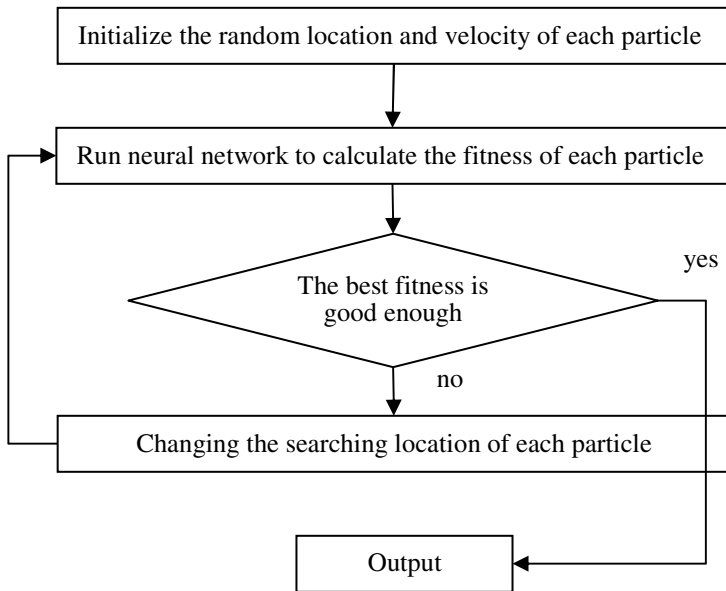


Fig. 1. The flow chart of PSO-based ANN

4 Application of PSO-Based ANN to Evaluate BP-ISP Strategic Alignment Degree

4.1 The Designment of PSO-Based ANN

In this section, we constructed the ANN model of evaluation of BP-ISP strategy alignment degree. A three-layer neural network is chosen in this application: 12 input neurons, four hidden neurons and one output neuron. There are consequently 52 weights to be optimized. This implies that the swarm of particles flies in a 52-dimensional space to search for the weights. A population of 30 individuals is used.

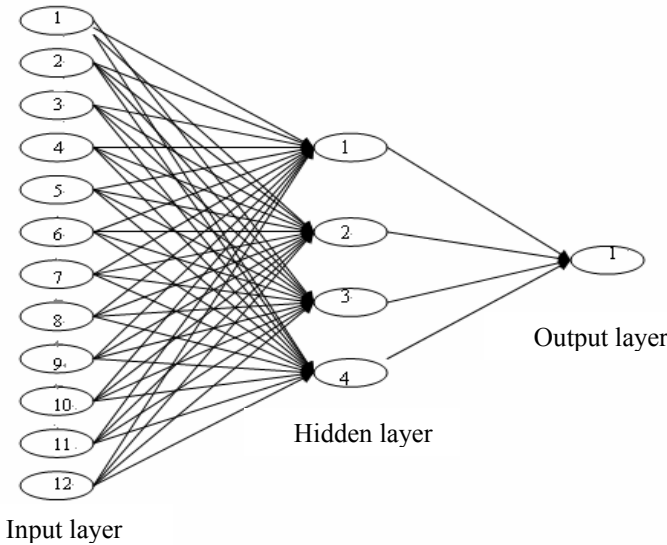


Fig. 2. The construction of ANN model of evaluation of BP-ISP strategy alignment degree

When a PSO is used to train this three-layer neural network, the i th particle of t th iteration is denoted by

$$X_i(t) = \{x_{i1}(t), x_{i2}(t), \dots, x_{iK}(t)\}. \quad K=1, \dots, 52 \quad (7)$$

Where $x_{i1}(t), x_{i2}(t), \dots, x_{i48}(t)$ represent the connection weight matrix between the input layer and the hidden layer, and $x_{i49}(t), \dots, x_{i52}(t)$ represent the connection weight matrix between the hidden layer and the output layer. each $x_{iK}(t)$ is initiated in the range[-10,10].

The position representing the previous best fitness value of any particle is recorded and denoted by

$$X_i^L = \{ X_{i1}^L, X_{i2}^L; \dots, X_{iK}^L \} \quad K=1, \dots, 52 \quad (8)$$

If, among all the particles in the current population, the index of the best particle is represented by the symbol G , then the best matrix is denoted by

$$X^G = \{ X_1^G, X_2^G, \dots, X_K^G \} \quad K=1, \dots, 52 \tag{9}$$

By using eq. (3), (4) the particle's new position is determined according to the new velocity. In this application, the value $c1, c2$ is denoted as 2.0 and $r1, r2$ be two random numbers in the range of $[0, 1]$, respectively. The maximum velocity assumed as 10 and the minimum velocity as -10. For each particle of t th generation, the fitness is counted to determine the local best position and global best position. The fitness of the i th particle is expressed in term of an output mean squared error of the neural networks as follows

$$f(X_i) = \frac{1}{S} \sum_{k=1}^S (t_{kl} - P_{kl}(X_i))^2$$

where f is the fitness value, t_{kl} is the target output; p_{kl} is the predicted output based on X_i , S is the number of training set samples.

4.2 The Experiment Analysis

The part of training data is showed in table 2. all the data is investigated from 30 different enterprise. Each index and the evaluating result is score in the range $[0, 1]$ by the expert of information management fields. We arrange a training dataset with 25 record, after training, and select other 5 record as test set to calculate the accuracy of the model.

The figure 2 showed the iteration number of different algorithms training neural network, we can see that, with different system training accuracy, the PSO-based ANN has a less iteration to finish the training process.

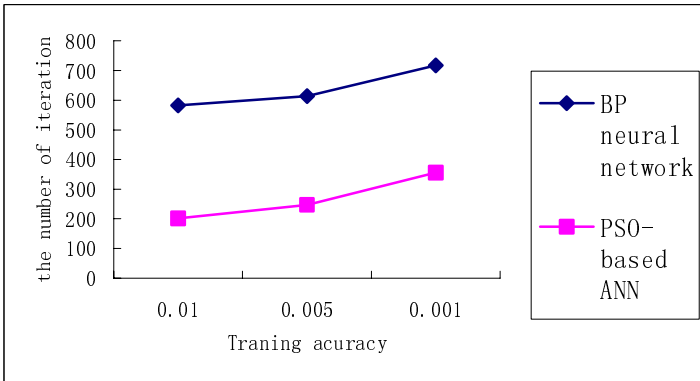


Fig. 3. The iteration number of different algorithms

The fact result of test set and the simulated test result with different algorithms is showed in Table 2. From Table 2, we can discover the PSO-based ANN has a better accuracy than BP neural network.

Table 2. The simulated test result

	1	2	3	4	5	error
Fact result	0.736	0.601	0.519	0.781	0.672	
The result by BP neural net work	0.726	0.607	0.507	0.781	0.678	0.007
The result by PSO-based ANN	0.731	0.605	0.511	0.779	1.673	0.004

5 Conclusions

In this paper, a set of index system of evaluating the alignment of BP-ISP is established. Based on the index system, an integration algorithm with PSO and neural network is established to evaluate the alignment of BP-ISP. The experimental results show that the method PSO- based ANN is over come the shortcoming of BP neural network, and show a less iteration and higher accuracy. In the future, we will try to use the PSO algorithm to optimize the number of hiding layer of ANN to constract a ore efficient model of BP-ISP strategy alignment .

References

- [1] Kearns, G.S., Lederer, A.L.: The Effect of Strategy Alignment on the Use of IS –based Resources for Competitive Advantegy. *Journal of Strategy Information System* 9 (2000) 265-293
- [2] Eberhart, R.C., Kennedy, J.: A New Optimizer Using Particle Swarm Theory. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan* (1995) 39– 43
- [3] Kennedy, J., Eberhart, R.: Particle Swarm Optimization. *Proceedings of the 1995 IEEE International Conference on Neural Networks, Perth* (1995) 1942–1948
- [4] Eberhart, R.C., Shi, Y.: Evolving Artificial Neural Networks. *Proceedings of the International Conference on Neural Networks and Brain, Beijing, PR China* (1998) 5–13
- [5] Zhou, J., Duan, Z., Li, Y., Deng, J., Yu, D.: PSO-based Neural Network Optimization and Its Utilization in A Boring Machine. *Journal of Materials Processing Technology* 178 (2006) 19–23
- [6] Chau, K.W.: Particle Swarm Optimization Training Algorithm for ANNs in Stage Prediction of Shing Mun River. *Journal of Hydrology* 329 (2006) 363– 367

Improved Results on Solving Quadratic Programming Problems with Delayed Neural Network^{*}

Minghui Jiang¹, Shengle Fang¹, Yi Shen², and Xiaoxin Liao²

¹ Institute of Nonlinear Complex Systems, China Three Gorges University, Yichang, Hubei, 443000, China
jmh1239@hotmail.com(M. Jiang)

² Department of Control Science and Engineering, Huazhong University of Science and Technology, Wuhan, Hubei, 430074, China

Abstract. In this paper, in terms of a linear matrix inequality (LMI), using a delayed Lagrangian network to solve quadratic programming-problems, sufficient conditions on delay-dependent and delay-independent are given to guarantee the globally exponential stability of the delayed neural network at the optimal solution. In addition, exponential convergence rate is estimated by the equation in the paper. Furthermore, the results in this paper improved the ones reported in the existing literatures and the proposed sufficient condition can be checked easily by solving LMI. Two simulation examples are provided to show the effectiveness of the approach and applicability of the proposed criteria.

1 Introduction

Optimization problems arise in various of scientific and engineering applications including function approximation, signal processing, regression analysis, and so on. In many scientific and engineering applications, the real-time solution of optimization problems is widely required. However, traditional algorithms for computers may not be efficient since the computing time required for a solution is greatly dependent on the structure and dimension of the problems. The neural network approach can solve optimization problems in running time at the orders of magnitude much faster than those of the most popular optimization algorithms executed on computers. The neural network for solving linear programming problems was first proposed by Tank and Hopfield [1] in 1986. In 1987, Kennedy and Chua [2] proposed an improved model that always guaranteed convergence. However, their new model converges to only an approximation of the optimal solution. In 1990, Rodriguez-Vazquez et al. [3] presented a class of neural networks for solving optimization problems. Based on dual and projection methods [4] Xia et al. [5-9] presented several neural networks for solving linear and

* The work was Supported by Natural Science Foundation of China Three Gorges University(No.604114), Natural Science Foundation of Hubei (Nos.2004ABA055, D200613002) and National Natural Science Foundation of China (No.60574025).

quadratic programming problems. Zhang [10] proposed the Lagrangian network and Wang et al. [11] studied the global convergence of the network. However, these neural networks are based on the assumption that neurons communicate and respond instantaneously without any time delay. In fact, the switching delay exists in some hardware implementation, and the time delay is hard to predict to guarantee the stability of the neural network theoretically. Chen and Fang [12] proposed a delayed neural network for solving convex quadratic programming problems by using the penalty function approach. However, this network can not converge to an exact optimal solution and has an implementation problem when the penalty parameter is very large. To avoid using finite penalty parameters, in [13], the delayed neural network without penalty parameters was proposed by Liu, Wang and Cao. However, the conditions in [13] are hard to verify and our results are expressed in terms of linear matrix inequalities (LMIs), which can be checked numerically using the effective LMI toolbox in MATLAB. Moreover, compared with results on delayed neural networks in [13], the presented conditions in this paper can give greater delay bound for stability, leading to less conservative conditions.

2 Preliminaries

Consider the following quadratic programming problem with constraints:

$$\begin{cases} \min \frac{1}{2}x^T Q_0 x + c^T x, \\ \text{subject to } Ax = b, \end{cases} \tag{1}$$

where $Q_0 \in R^{n \times n}$ is positive semi-definite, $c \in R^n$, $A \in R^{m \times n}$, and $b \in R^m$. In this paper, we always assume that feasible domain $\Omega = \{x \in R^{n \times n} | Ax - b = 0\}$ is not empty.

By Karush-Kuhn-Tucker conditions [14], x^* is a solution to (1) if and only if there exists y^* such that (x^*, y^*) satisfies the following Lagrange condition:

$$\begin{cases} \nabla L_x(x^*, y^*) = Q_0 x^* + c - A^T y^* = 0, \\ \nabla L_y(x^*, y^*) = Ax^* - b = 0, \end{cases} \tag{2}$$

where ∇L is the gradient of the Lagrangian function L .

In [13], the following delayed Lagrangian network for solving problem (1) was proposed:

$$\frac{du}{dt} = -(D + W)u(t) + Du(t - \tau) - J, \tag{3}$$

where $\tau > 0$ denotes the transmission delay, $D \in R^{(n+m) \times (n+m)}$, $W = \begin{pmatrix} Q_0 & -A^T \\ A & 0 \end{pmatrix}$, $J = \begin{pmatrix} c \\ -b \end{pmatrix}$, $y \in R^m$ is the Lagrange multiplier, $u(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}$ with the initial value function $u(s) = \varphi(s)$, $s \in [-\tau, 0]$.

For convenience of discussion, we use the transformation $v(t) = u(t) - u^* = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} - \begin{pmatrix} x^* \\ y^* \end{pmatrix}$ to shift the solution u^* of the delayed neural network (3) to the origin, and get the following form of the delayed neural network (4):

$$\frac{dv}{dt} = A_0v(t) + A_1v(t - \tau), \tag{4}$$

where $A_0 = -(D + W)$, $A_1 = D$.

3 Criteria for Quadratic Programming Problems

3.1 Delay-Dependent Condition

Lemma 1. [15] *Given any $\alpha \in R^{n_a}$, $\beta \in R^{n_b}$, $Y, N \in R^{n_a \times n_b}$, $X \in R^{n_a \times n_a}$, $Z \in R^{n_b \times n_b}$, if the following inequality is satisfied*

$$\begin{pmatrix} X & Y \\ Y^T & Z \end{pmatrix} \geq 0, \tag{5}$$

then

$$-2\alpha^T N \beta \leq (\alpha^T \ \beta^T) \begin{pmatrix} X & Y - N \\ Y^T - N^T & Z \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix}. \tag{6}$$

Theorem 1. *The equilibrium point of the dynamic system (4) is exponentially stable if there are matrices $X = X^T$, Y and positive matrices P, Q, S, Z such that the dynamic system satisfies the following LMIs*

$$\begin{pmatrix} X & Y \\ Y^T & Z \end{pmatrix} \geq 0, \tag{7}$$

and

$$G = \begin{pmatrix} g_{11} & g_{12} & 0 \\ g_{12}^T & g_{22} & 0 \\ 0 & 0 & -S \end{pmatrix} < 0, \tag{8}$$

where $g_{11} = P(A_0 + A_1) + (A_0 + A_1)^T P + Q + \tau X + Y - PA_1 + (Y - PA_1)^T + A_0^T(S + \tau Z)A_0$, $g_{12} = -Q - Y + PA_1 + A_0^T(S + \tau Z)A_1$, $g_{22} = -Q + A_1^T(S + \tau Z)A_1$.

Proof. Choose the candidate Lyapunov functional to be

$$V(x_t) = V_1 + V_2, \tag{9}$$

where

$$V_1 = x^T(t)Px(t),$$

$$V_2 = \int_{t-\tau}^t x^T(s)Qx(s)ds + \int_{t-\tau}^t \dot{x}^T(s)S\dot{x}(s)ds + \int_{-\tau}^0 \int_{t+\theta}^t \dot{x}^T(s)Z\dot{x}(s)dsd\theta.$$

The time derivative of V_1 along the trajectories of (4) and $x(t - \tau) = x(t) - \int_{t-\tau}^t \dot{x}(s)ds$ turn out to be

$$\begin{aligned} \dot{V}_1 &= 2x^T(t)P\dot{x}(t) = 2x^T(t)P[A_0x(t) + A_1x(t - \tau)], \\ &= 2x^T(t)P[(A_0 + A_1)x(t) - A_1 \int_{t-\tau}^t \dot{x}(s)ds], \\ &= 2x^T(t)P(A_0 + A_1)x(t) - \int_{t-\tau}^t 2x^T(t)PA_1\dot{x}(s)ds. \end{aligned} \tag{10}$$

By Lemma 1, we have

$$\begin{aligned} - \int_{t-\tau}^t 2x^T(t)PA_1\dot{x}(s)ds &= \int_{t-\tau}^t (x(t) \dot{x}(s)) \\ &\quad \times \begin{pmatrix} X & Y - PA_1 \\ (Y - PA_1)^T & Z \end{pmatrix} \begin{pmatrix} x(t) \\ \dot{x}(s) \end{pmatrix} ds. \end{aligned} \tag{11}$$

Substituting (11) into (10) yields

$$\begin{aligned} \dot{V}_1 &\leq 2x^T(t)P(A_0 + A_1)x(t) + \int_{t-\tau}^t (x(t) \dot{x}(s)) \\ &\quad \times \begin{pmatrix} X & Y - PA_1 \\ (Y - PA_1)^T & Z \end{pmatrix} \begin{pmatrix} x(t) \\ \dot{x}(s) \end{pmatrix} ds \\ &= 2x^T(t)P(A_0 + A_1)x(t) + \tau x^T(t)Xx(t) + 2x^T(t)(Y - PA_1)^T \int_{t-\tau}^t \dot{x}(s)ds \\ &\quad + \int_{t-\tau}^t \dot{x}^T(s)Z\dot{x}(s)ds. \end{aligned} \tag{12}$$

The time derivative of V_2 along the trajectories of (4) is given by

$$\begin{aligned} \dot{V}_2 &= x^T(t)Qx(t) - x^T(t - \tau)Qx(t - \tau) + \dot{x}^T(t)S\dot{x}(t) \\ &\quad - \dot{x}^T(t - \tau)S\dot{x}(t - \tau) + \tau \dot{x}^T(t)Z\dot{x}(t) - \int_{t-\tau}^t \dot{x}^T(s)Z\dot{x}(s)ds. \end{aligned} \tag{13}$$

Therefore, by the inequalities (12) and (13), we obtain

$$\begin{aligned} \dot{V} &\leq 2x^T(t)P(A_0 + A_1)x(t) + \tau x^T(t)Xx(t) + 2x^T(t)(Y - PA_1)^T(x(t) - x(t - \tau)) \\ &\quad + x^T(t)Qx(t) - x^T(t - \tau)Qx(t - \tau) + \dot{x}^T(t)(S + \tau Z)\dot{x}(t) \\ &\quad - \dot{x}^T(t - \tau)S\dot{x}(t - \tau) \\ &= 2x^T(t)P(A_0 + A_1)x(t) + \tau x^T(t)Xx(t) + 2x^T(t)(Y - PA_1)^T[x(t) - x(t - \tau)] \\ &\quad + x^T(t)Qx(t) - x^T(t - \tau)Qx(t - \tau) + [A_0x(t) + A_1x(t - \tau)]^T(S + \tau Z) \\ &\quad \times [A_0x(t) + A_1x(t - \tau)] - \dot{x}^T(t - \tau)S\dot{x}(t - \tau) \\ &= \xi^T(t)G\xi(t) < \lambda_{max}(G)\|x(t)\|^2, \end{aligned} \tag{14}$$

where $\xi(t) = (x(t) \ x(t - \tau) \ \dot{x}(t - \tau))^T$.

From the definition of V , it follows

$$\begin{aligned} \lambda_{min}(P)\|x(t)\|^2 \leq V(t) \leq & \lambda_{max}(P)\|x(t)\|^2 + \lambda_{max}(Q) \int_{t-\tau}^t \|x(s)\|^2 ds \\ & + \int_{t-\tau}^t \dot{x}^T(s)(S + \tau Z)\dot{x}(s)ds. \end{aligned} \tag{15}$$

By the inequality $2x^T A^T B y \leq x^T A^T A x + y^T B^T B y, \forall x, y, A, B$ with suitable dimensional matrices, we have

$$\begin{aligned} & \int_{t-\tau}^t \dot{x}^T(s)(S + \tau Z)\dot{x}(s)ds = \int_{t-\tau}^t [A_0 x(s) + A_1 x(s - \tau)]^T (S + \tau Z) \\ & \quad \times [A_0 x(s) + A_1 x(s - \tau)] ds \\ \leq & \int_{t-\tau}^t [x^T(s)A_0^T(S + \tau Z)A_0 x(s) + 2x^T(s)A_0^T(S + \tau Z)A_1 x(s - \tau) \\ & \quad + x^T(s - \tau)A_1^T(S + \tau Z)A_1 x(s - \tau)] ds \\ \leq & \lambda_{max}(A_0^T(S + \tau Z + I)A_0) \int_{t-\tau}^t \|x(s)\|^2 ds + \lambda_{max}(A_1^T(S + \tau Z + I)^T \\ & \quad \times (S + \tau Z)A_1) \int_{t-\tau}^t \|x(s - \tau)\|^2 ds, \end{aligned} \tag{16}$$

where I is an identity matrix.

Set $\lambda_1 = \lambda_{max}(A_0^T(S + \tau Z + I)A_0), \lambda_2 = \lambda_{max}(A_1^T(S + \tau Z + I)^T(S + \tau Z)A_1)$, then, by (15) and (16), we have

$$\begin{aligned} V(t) \leq & \lambda_{max}(P)\|x(t)\|^2 + (\lambda_{max}(Q) + \lambda_1) \int_{t-\tau}^t \|x(s)\|^2 ds \\ & + \lambda_2 \int_{t-\tau}^t \|x(s - \tau)\|^2 ds. \end{aligned} \tag{17}$$

There exists at least a positive constant α_0 such that

$$\alpha_0[\lambda_{max}(P) + (\lambda_{max}(Q) + \lambda_1)\tau e^{\alpha_0 \tau} + \lambda_2 \tau e^{2\alpha_0 \tau}] \leq -\lambda_{max}(G). \tag{18}$$

Using (14), (17) and (18), we get

$$\begin{aligned} e^{\alpha_0 t} V(t) - e^{\alpha_0 t_0} V(t_0) &= \int_{t_0}^t \frac{d(e^{\alpha_0 t} V(s))}{ds} ds \\ &\leq \int_{t_0}^t e^{\alpha_0 s} [\alpha_0 (\lambda_{max}(P)\|x(s)\|^2 + (\lambda_{max}(Q) + \lambda_1) \int_{s-\tau}^s \|x(u)\|^2 du \\ & \quad + \lambda_2 \int_{s-\tau}^s \|x(u - \tau)\|^2 du) + \lambda_{max}(G)\|x(s)\|^2] ds. \end{aligned} \tag{19}$$

Exchanging the order of integral, we obtain

$$\begin{aligned} \int_{t_0}^t e^{\alpha_0 s} ds \int_{s-\tau}^s \|x(u)\|^2 du &\leq \int_{t_0-\tau}^t du \int_{u+\tau}^u e^{\alpha_0 s} \|x(u)\|^2 ds \\ &\leq \tau e^{\alpha_0 \tau} \int_{t_0-\tau}^t e^{\alpha_0 u} \|x(u)\|^2 du, \end{aligned} \tag{20}$$

and

$$\begin{aligned} \int_{t_0}^t e^{\alpha_0 s} ds \int_{s-\tau}^s \|x(u-\tau)\|^2 du &\leq \int_{t_0}^t e^{\alpha_0 s} ds \int_{s-2\tau}^s \|x(u-2\tau)\|^2 du \\ &\leq 2\tau e^{2\alpha_0 \tau} \int_{t_0-2\tau}^t e^{\alpha_0 u} \|x(u)\|^2 du, \end{aligned} \tag{21}$$

Applying (20) and (21) to (19) yield

$$\begin{aligned} e^{\alpha_0 t} V(t) - e^{\alpha_0 t_0} V(t_0) &\leq \int_{t_0}^t e^{\alpha_0 s} [\alpha_0 [\lambda_{max}(P) + (\lambda_{max}(Q) + \lambda_1)\tau e^{\alpha_0 \tau} \\ &\quad + \lambda_2 \tau e^{2\alpha_0 \tau} + \lambda_{max}(G) \|x(s)\|^2] ds + q_0(t_0) \\ &\leq q_0(t_0), \end{aligned} \tag{22}$$

where $q_0(t_0) = \alpha_0 \tau e^{\alpha_0 \tau} [(\lambda_{max}(Q) + \lambda_1) \int_{t_0-\tau}^{t_0} e^{\alpha_0 s} \|x(s)\|^2 ds + 2\lambda_2 e^{\alpha_0 \tau} \int_{t_0-2\tau}^{t_0} e^{\alpha_0 s} \|x(s)\|^2 ds]$.

By (15) and (22), we have

$$e^{\alpha_0 t} \lambda_{min}(P) \|x(t)\|^2 \leq e^{\varepsilon t} V(t) \leq \Theta, t \geq t_0. \tag{23}$$

where $\Theta = e^{\alpha_0 t_0} V(t_0) + q_0(t_0)$.

Therefore,

$$\|x(t)\|^2 \leq \frac{\Theta}{\lambda_{min}(P)} e^{-\alpha_0 t}, t \geq t_0. \tag{24}$$

This implies the origin of the delay neural network (4) is global exponentially stable and its exponential convergence rate α_0 is estimated by (18). This completes the proof.

Remark 1. From Theorem 1, it follows that the stability of the delay neural network (4) is related to the size of the delay τ . Furthermore, the condition is easy to check by solving LMIs in Theorem 1 and the maximum delay can be computed by the seeking algorithm.

3.2 Delay-Independent Condition

By applying directly the second Lyapunov method, we can obtain the delay-independent condition on globally exponential stability of delay dynamic system in the following theorem.

Theorem 2. *The equilibrium point of the dynamic system (4) is exponentially stable if there are positive matrices P, Q such that the dynamic system satisfies the following LMI*

$$F = \begin{pmatrix} PA_0 + A_0^T P + Q & PA_1 \\ A_1^T P & -Q \end{pmatrix} < 0. \tag{25}$$

Proof. Take the candidate Lyapunov functional to be

$$V(x_t) = x^T(t)Px(t) + \int_{t-\tau}^t x^T(s)Qx(s)ds. \tag{26}$$

The time derivative of V along the trajectories of (1) is given by

$$\begin{aligned} \dot{V} &= 2x^T(t)P[A_0x(t) + A_1x(t-\tau)] + x^T(t)Qx(t) - x^T(t-\tau)Qx(t-\tau), \\ &= \xi^T(t)F\xi(t) < \lambda_{max}(F)\|x(t)\|^2, \end{aligned} \tag{27}$$

where $\xi(t) = (x(t) \ x(t-\tau))^T$.

From the definition of V , it follows

$$\lambda_{min}(P)\|x(t)\|^2 \leq V(t) \leq \lambda_{max}(P)\|x(t)\|^2 + \lambda_{max}(Q) \int_{t-\tau}^t \|x(s)\|^2 ds. \tag{28}$$

Set $h(\varepsilon) = \lambda_{max}(F) + \lambda_{max}(P)\varepsilon + \lambda_{max}(Q)e^{\varepsilon\tau}\tau\varepsilon$.

Owing to $h'(\varepsilon) > 0, h(0) = \lambda_{max}(F) < 0, h(+\infty) = +\infty$, there exists a unique ε such that $h(\varepsilon) = 0$; i.e., there is $\varepsilon > 0$ satisfies

$$\lambda_{max}(F) + \lambda_{max}(P)\varepsilon + \lambda_{max}(Q)e^{\varepsilon\tau}\tau\varepsilon = 0. \tag{29}$$

From the definition of V , it follows

$$V(t) \leq \lambda_{max}(P)|x(t)|^2 + \lambda_{max}(Q) \int_{t-\tau}^t |x(s)|^2 ds. \tag{30}$$

For $\varepsilon > 0$ satisfying (29), by (28) and (30), we get

$$\begin{aligned} (e^{\varepsilon t}V(x(t)))' &= \varepsilon e^{\varepsilon t}V(x(t)) + e^{\varepsilon t}\dot{V}(x(t)) \\ &\leq e^{\varepsilon t}(\lambda_{max}(F) + \lambda_{max}(P)\varepsilon)|x(t)|^2 \\ &\quad + \varepsilon e^{\varepsilon t}\lambda_{max}(Q) \int_{t-\tau}^t |x(s)|^2 ds. \end{aligned} \tag{31}$$

Integrating the both sides of (31) from 0 to an arbitrary $t \geq 0$, we can obtain

$$\begin{aligned} &e^{\varepsilon t}V(x(t)) - V(x(0)) \\ &\leq \int_0^t e^{\varepsilon s}(\lambda_{max}(F) + \lambda_{max}(P)\varepsilon)|x(s)|^2 ds \\ &\quad + \int_0^t \varepsilon e^{\varepsilon s}\lambda_{max}(Q)ds \int_{s-\tau}^s |x(r)|^2 dr. \end{aligned} \tag{32}$$

And

$$\begin{aligned} & \int_0^t \varepsilon e^{\varepsilon s} \lambda_{max}(Q) ds \int_{s-\tau}^s |u(r)|^2 dr \\ & \leq \int_{-\tau}^t \varepsilon \lambda_{max}(Q) \tau e^{\varepsilon(r+\tau)} |u(r)|^2 dr. \end{aligned} \tag{33}$$

Substituting (33) into (32) and applying (29), we obtain

$$e^{\varepsilon t} V(x(t)) \leq \Theta, t \geq 0. \tag{34}$$

where $\Theta = V(x(0)) + \int_{-\tau}^0 \varepsilon \lambda_{max}(Q) \tau e^{\varepsilon \tau} |u(r)|^2 dr$.

By the above inequality and (28), we get

$$|x(t)|^2 \leq \frac{\Theta}{\lambda_{min}(P)} e^{-\varepsilon t}, t \geq 0. \tag{35}$$

This implies the origin of the delay neural network (4) is globally exponentially stable and its exponential convergence rate α_0 is estimated by (29). This completes the proof.

Applying the above Theorem, we can easily get the following Corollary 1:

Corollary 1. *If there are matrix C and positive matrices P, Q such that the dynamic system (3) satisfies the following LMI*

$$\begin{pmatrix} -C - C^T - PW - W^T P + Q & C \\ C^T & -Q \end{pmatrix} < 0, \tag{36}$$

then the dynamic system (3) with $D = P^{-1}C$ converge exponentially to x^* which is an optimal solution of the quadratic programming problem with constraints (1).

Remark 2. Corollary 1 provides a method on designing the delay-independent neural network (3) to compute the optimal solution of the problem (1).

4 Examples

In this section, we consider the following quadratic programming problem[13]:

$$\begin{cases} \min \frac{1}{2} x^T Q x + c^T x, \\ \text{subject to } Ax = b, \end{cases} \tag{37}$$

where $Q = \begin{pmatrix} 0.1 & 0.1 \\ 0.1 & 0.1 \end{pmatrix}$, $c = (-1 \ 1)$, $A = (0.5, 0.5)$, $b = -0.5$. This problem has a unique solution $x^* = (-0.5, 0.5)^T$.

Example 1. In [13], the following delay neural network was illustrated as follows

$$\frac{du}{dt} = -(D + W)u(t) + Du(t - \tau) - J, \tag{38}$$

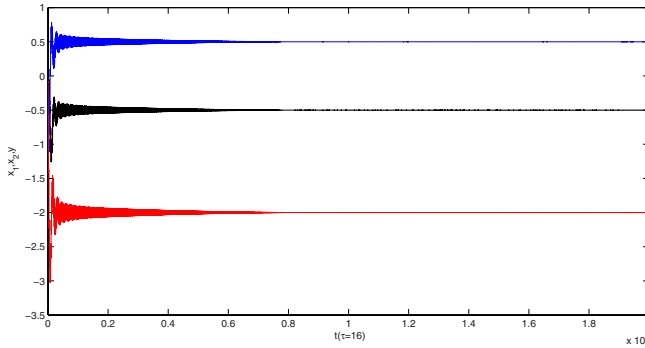


Fig. 1. Behaviors of the delay network with 4 groups of random initial conditions

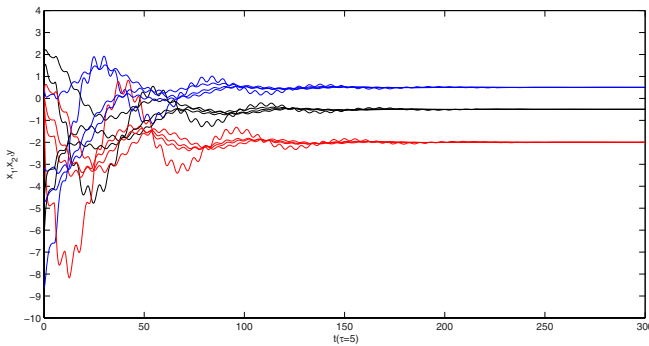


Fig. 2. dynamical behavior of the delay neural network with 4 random initial conditions

where $\tau \geq 0$ denotes the transmission delay, $W = \begin{pmatrix} 0.1 & 0.1 & -0.5 \\ 0.1 & 0.1 & 0.5 \\ 0.5 & -0.5 & 0 \end{pmatrix}$, $J = \begin{pmatrix} -1 \\ 1 \\ 0.5 \end{pmatrix}$,

$u = \begin{pmatrix} x_1 \\ x_2 \\ y \end{pmatrix}$. D is taken as identity matrix, and by Theorem 3 in [13], the equilibrium point of system (3) is asymptotically stable if $0 < \tau < 3.68$. However, applying Theorem 1 in this paper to Example 1, we know that the equilibrium point of system (3) is exponentially stable if $0 < \tau < 21.5$. Figure 1 shows the delay neural network with $\tau = 16$ converge exponentially to the unique optimal solution.

Example 2. By computing LMI (36) in Corollary 1, we can design the following delay neural network: $D = \begin{pmatrix} 0.7964 & -0.2151 & 0.0064 \\ -0.1876 & 0.9301 & -0.0055 \\ 0.0066 & -0.0053 & 0.8951 \end{pmatrix}$. By Theorem 2 in this paper, the stability of the neural network above is delay-independent. Figure 2 demonstrates the neural network designed converges exponentially to $(x_1^*, x_2^*, y^*) = (-0.5, 0.5, -2)$.

5 Conclusions

In this paper, we present theoretical results on the delay neural network for optimization computation. These conditions obtained here are easy to be checked in practice, and are of prime importance and great interest in the design of delay neural network. The criterions are little conservative than the respective criteria reported in existing references. Finally, illustrative examples are also given to verify the effectiveness of the results.

References

1. Tank, D.W.: Tank, D.W., Hopfield, J.J.: Simple Neural Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit. *IEEE Trans. Circuits and Systems* **33**(1986) 533-541
2. Kennedy, M.P., Chua, L.O.: Neural Networks for Nonlinear Programming. *IEEE Trans. Circuits and Systems* **35** (1986) 554-562
3. Rodriguez-Vazquez, A., Dominguez-Castro, R., Rueda, A., Huertas, J.L., Sanchez-Sinencio, E.: Nonlinear Switched-capacitor Neural Networks for Optimization problems. *IEEE Trans. Circuits Syst.* **37** (1990) 384-397
4. Gafini, E.M., Bertsekas, D.P.: Two Metric Projection Methods for Constraints Optimization. *SIAM J. Contr. Optim.* **22** (1984) 936-964
5. Xia, Y., wang, J.: Neural Network for Solving Linear Programming Problems with Bounded Variables. *IEEE Trans. Neural Networks* **6** (1995) 515-519
6. Xia, Y.: A New Neural Network for Solving Linear Programming Problems and Its Applications. *IEEE Trans. Neural Networks* **7** (1996) 525-529
7. Xia, Y., Wang, J.: A Recurrent Neural Network for Solving Linear Projection Equations. *Neural Network A* **13** (2000) 337-350
8. Xia, Y., Leng, H., Wang, J.: A Projection Neural Network and Its Application to Constrained Optimization Problems. *IEEE Trans. Circuits Syst.* **49** (2002) 447-458
9. Xia, Y., Wang, J.: A General Projection Neural Network for Solving Monotone Variational Inequalities and Related Optimization Problems. *IEEE Trans. Neural Networks* **15** (2004) 318-328
10. Zhang, S., Constantinides, A.G.: Lagrange Programming Neural Networks. *IEEE Trans. Circuits and Systems II* **39** (1992) 441-452
11. Wang, J., Hu, Q., Jiang, D.: A Lagrangian Network for Kinematic Control of Redundant Robot Manipulators. *IEEE Trans. Neural Networks* **10** (1999) 1123-1132
12. Chen, Y.H., Fang, S.C.: Neurocomputing with Time Delay Analysis for Solving Convex Quadratic Programming Problems. *IEEE Trans. Neural Networks* **11** (2000) 230-240
13. Liu Q.S., Wang J., and Cao J.D.: A Delayed Lagrangian Network for Solving Quadratic Programming Problems with Equality Constraints. *Lecture Notes in Computer Science* **3971**(2006)369-375
14. Luenberger D.G.: *Linear and Nonlinear Programming*. Addison-Wesley, Reading, MA. (1973)
15. Moon Y.S., Park P., Kwon W.H., and Lee Y. S.: Delay-dependent Robust Stabilization of Uncertain State-delayed Systems. *International Journal of Control* **74** (2001)1447-1455

A Modified Hopfield Network for Nonlinear Programming Problem Solving*

Changrui Yu^{1,2} and Yan Luo¹

¹ Institute of System Engineering, Shanghai Jiao Tong University, Shanghai 200052, China

² School of Information Management and Engineering, Shanghai University of Finance and Economics, Shanghai 200433, China
{yucr, yanluo}@sjtu.edu.cn

Abstract. This paper presents an efficient approach based on Hopfield network for solving nonlinear optimization problems, with polynomial objective function, polynomial equality constraints and polynomial inequality constraints. A modified Hopfield network is developed and its stability and convergence is analyzed in the paper. Then a mapping of nonlinear optimization problems is formulated using the modified Hopfield network. Simulation results are provided to demonstrate the performance of the proposed neural network.

1 Introduction

Optimization problems arise in a wide variety of scientific and engineering applications including decision-making, system identification, function approximation, regression analysis, and so on [1]. In some applications, the real-time solution of optimization problems is widely required. However, traditional numerical algorithms may not be efficient since the computing time required for a solution is greatly dependent on the dimension and structure of the problems. One possible and very promising approach to real-time optimization is to apply artificial neural networks. Because of the inherent massive parallelism, the neural network approach can solve optimization problems in running time at the orders of magnitude much faster than those of the most popular optimization algorithms.

In the neural networks literature, there exist several approaches used for solving nonlinear optimization problems. The first neural approach applied in optimization problems was proposed by Tank and Hopfield in [2] and [3] where the network was used for solving linear programming problems. Although the equilibrium point of the Tank and Hopfield network may not be a solution of the original problem, this seminal work has inspired many researchers to investigate other neural networks for solving linear and nonlinear programming problems. Kennedy and Chua in [4] extended the Tank and Hopfield network by developing a neural network for solving nonlinear programming problems. However, their model converges to only an approximation of the optimal solution. Rodriguez-Vazquez et al. [5] proposed a class of neural

* This research work is supported by the Natural Science Fund of China (# 70501022).

networks for solving optimization problems. Wu et al. [6] introduced a new model that solves both the primal and dual problems of linear and quadratic programming problems. Their new model always globally converges to the solutions of the primal and dual problems. Xia et al. [7] introduced a recurrent neural network for solving the nonlinear projection formulation. Effati et al. [8] presented a new nonlinear neural network that has a much faster convergence. Their new model is based on a nonlinear dynamical system.

Based on the previous research works, we have developed a modified Hopfield network not depending on penalty or weighting parameters, which overcomes shortcomings of the previous approaches. This approach can be applied to solve some types of nonlinear optimization problems, with polynomial objective function, polynomial equality constraints and polynomial inequality constraints. As many nonlinear optimization problems bear this form in many engineering and scientific applications, the neural network proposed in this paper has its practical significance.

The organization of the present paper is as follows. In Section 2, the modified Hopfield network is presented. In Section 3, a mapping of nonlinear optimization problems is formulated using the modified Hopfield network. Simulation results are presented to demonstrate the advanced performance of the proposed approach in Section 4. Finally, the main results of the paper are summarized and some concluding remarks are presented.

2 The Modified Hopfield Network

Based on the continuous model of Hopfield network, we can construct the dynamic function as follows:

$$\begin{aligned}
 C_i \frac{dU_i}{dt} = & -\sum_{l_1=1}^{m_g} \varphi(Z_{l_1}) P_{l_1,i} - \sum_{l_2=1}^{m_h} \psi(Y_{l_2}) S_{l_2,i} + I_i \\
 & + \sum_{j_1=1}^{n_1} T_{ij_1} V_{j_1} + \sum_{j_1=1}^{n_1} \sum_{j_2=1}^{n_1} T_{ij_1 j_2} V_{j_1} V_{j_2} + \dots \\
 & + \sum_{j_1=1}^{n_1} \sum_{j_2=1}^{n_1} \dots \sum_{j_{n_2}=1}^{n_1} T_{ij_1 j_2 \dots j_{n_2}} V_{j_1} V_{j_2} \dots V_{j_{n_2}} - \frac{U_i}{R_i}
 \end{aligned} \tag{1}$$

where

$$V_i = f_i(U_i) \quad (i = 1, 2, \dots, n_1) \tag{2}$$

$$\begin{aligned}
 Z_{l_1} = & \frac{1}{m_{l_1} + 1} \sum_{i=1}^{n_1} \sum_{k_1=1}^{n_1} \sum_{k_2=1}^{n_1} \dots \sum_{k_{m_{l_1}}=1}^{n_1} W_{ik_1 k_2 \dots k_{m_{l_1}}}^{(l_1)} V_i V_{k_1} V_{k_2} \dots V_{k_{m_{l_1}}} + \dots + \frac{1}{2} \sum_{i=1}^{n_1} \sum_{k_1=1}^{n_1} W_{ik_1}^{(l_1)} V_i V_{k_1} \\
 & + \sum_{i=1}^{n_1} W_i^{(l_1)} V_i - b_{l_1} \quad (l_1 = 1, 2, \dots, m_g)
 \end{aligned} \tag{3}$$

$$\begin{aligned}
 Y_{l_2} = & \frac{1}{m_{l_2} + 1} \sum_{i=1}^{n_1} \sum_{k_1=1}^{n_1} \sum_{k_2=1}^{n_1} \cdots \sum_{k_{m_{l_2}}=1}^{n_1} Q_{ik_1k_2 \cdots k_{m_{l_2}}}^{(l_2)} V_i V_{k_1} V_{k_2} \cdots V_{k_{m_{l_2}}} + \cdots + \frac{1}{2} \sum_{i=1}^{n_1} \sum_{k_1=1}^{n_1} W_{ik_1}^{(l_2)} V_i V_{k_1} \\
 & + \sum_{i=1}^{n_1} W_i^{(l_2)} V_i - d_{l_2} \quad (l_2 = 1, 2, \dots, m_h)
 \end{aligned} \tag{4}$$

$$\begin{aligned}
 P_{l_1, i} = & \sum_{k_1=1}^{n_1} \sum_{k_2=1}^{n_1} \cdots \sum_{k_{m_{l_1}}=1}^{n_1} W_{ik_1k_2 \cdots k_{m_{l_1}}}^{(l_1)} V_{k_1} V_{k_2} \cdots V_{k_{m_{l_1}}} + \cdots + \sum_{k_1=1}^{n_1} W_{ik_1}^{(l_1)} V_{k_1} + W_i^{(l_1)} \\
 & (l_1 = 1, 2, \dots, m_g; i = 1, 2, \dots, n_1)
 \end{aligned} \tag{5}$$

$$\begin{aligned}
 S_{l_2, i} = & \sum_{k_1=1}^{n_1} \sum_{k_2=1}^{n_1} \cdots \sum_{k_{m_{l_2}}=1}^{n_1} Q_{ik_1k_2 \cdots k_{m_{l_2}}}^{(l_1)} V_{k_1} V_{k_2} \cdots V_{k_{m_{l_2}}} + \cdots + \sum_{k_1=1}^{n_1} Q_{ik_1}^{(l_2)} V_{k_1} + Q_i^{(l_1)} \\
 & (l_2 = 1, 2, \dots, m_h; i = 1, 2, \dots, n_1)
 \end{aligned} \tag{6}$$

$$\varphi(x) = x$$

$$\psi(x) = \begin{cases} 0 & \text{if } x \geq 0 \\ x & \text{otherwise} \end{cases}$$

Moreover, in Eq. (1), n_1 is the number of the neuron, i.e., variable number; T is coefficient or weight of the connection between neurons; I_i is biased input of neuron i ; $f_i(\bullet)$ is function of neuron i ; C_i and R_i are constants; U_i is input of neuron i (which is also called the potential of a neuron); V_i is output of neuron i (which is also called the firing rate of a neuron); W and Q are constant coefficients; b_{l_1} and d_{l_2} are constants; $n_2, m_{l_1}, m_{l_2}, m_g$, and m_h are positive integers.

Compared with normal continuous Hopfield network model, Eq. (1) is added the first term and the second term on the right side as well as the terms where $n_2 > 1$. Therefore, Eq. (1) is an extension to the continuous Hopfield network model.

We construct the modified energy function as follows:

$$\begin{aligned}
 E = & \sum_{l_1=1}^{m_g} F(Z_{l_1}) + \sum_{l_2=1}^{m_h} G(Y_{l_2}) - \sum_{i=1}^{n_1} V_i I_i - \frac{1}{2} \sum_{i=1}^{n_1} \sum_{j_1=1}^{n_1} T_{ij_1} V_i V_{j_1} - \cdots \\
 & - \frac{1}{n_2 + 1} \sum_{i=1}^{n_1} \sum_{j_1=1}^{n_1} \cdots \sum_{j_{n_2}=1}^{n_1} T_{ij_1 \cdots j_{n_2}} V_i V_{j_1} \cdots V_{j_{n_2}} + \sum_{i=1}^{n_1} \frac{1}{R_i} \int_0^{V_i} f^{-1}(t) d(t)
 \end{aligned} \tag{7}$$

where $F(\bullet)$ and $G(\bullet)$ are functions. Their forms are determined by the following proposition.

Proposition. When the following conditions are satisfied, i.e.,

- (1) $f(\bullet)$ is a monotonically increasing continuous function;
- (2) $C_i > 0$, and $R_i > 0$;

(3) $T_{ij_1j_2 \dots j_{n_x}} = T_{m_1m_2 \dots m_{n_x+1}}$, where $n_x = 1, 2, \dots, n_2$, $m_1, m_2, \dots, m_{n_x+1}$ is a random permutation of $i, j_1, j_2, \dots, j_{n_x}$. That is, the bottom-up weights of all orders the connection between neurons are symmetric. For example, when $n_2 = 2$, this condition is $T_{ij_1j_2} = T_{ij_2j_1} = T_{j_1j_2i} = T_{j_1j_2i} = T_{j_2j_1i}, T_{ij_1} = T_{ji_1}$;

(4) $W_{ik_1k_2 \dots k_{n_x}}^{(l_1)} = W_{m_1m_2 \dots m_{n_x+1}}^{(l_1)}$, where $n_x = 1, 2, \dots, m_1$, $m_1, m_2, \dots, m_{n_x+1}$ is a random permutation of $i, k_1, k_2, \dots, k_{n_x}$. That is, $W^{(l_1)}$ is symmetric. For example, when $m_1 = 2$, the condition is $W_{ik_1k_2}^{(l_1)} = W_{k_1k_2i}^{(l_1)} = W_{k_1k_2i}^{(l_1)} = W_{k_2k_1i}^{(l_1)} = W_{k_2k_1i}^{(l_1)}, W_{ik_1}^{(l_1)} = W_{k_1i}^{(l_1)}$;

(5) $Q_{ik_1k_2 \dots k_{n_x}}^{(l_2)} = Q_{m_1m_2 \dots m_{n_x+1}}^{(l_2)}$, where $n_x = 1, 2, \dots, m_2$, $m_1, m_2, \dots, m_{n_x+1}$ is a random permutation of $i, k_1, k_2, \dots, k_{n_x}$. That is, $Q^{(l_2)}$ is symmetric. For example, when $m_2 = 2$, this condition is $Q_{ik_1k_2}^{(l_2)} = Q_{k_1k_2i}^{(l_2)} = Q_{k_1k_2i}^{(l_2)} = Q_{ik_2k_1}^{(l_2)} = Q_{k_2k_1i}^{(l_2)} = Q_{ik_2k_1}^{(l_2)}, Q_{ik_1}^{(l_2)} = Q_{k_1i}^{(l_2)}$;

(6) $\phi(x) = dF(x) / dx$;

(7) $\psi(x) = dG(x) / dx$,

We have the following conclusions:

(1) $dE / dt \leq 0$.

(2) $dE / dt \leq 0 \Leftrightarrow dV_i / dt = 0$, or $dU_i / dt = 0, i = 1, 2, \dots, n_1$.

The proposition shows that when the seven conditions are satisfied, systematic energy function E described by Eq. (1) is decreasing, i.e., this system is globally stable. According to the proposition, we have

$$F(x) = x^2 / 2$$

$$G(x) = \begin{cases} 0 & x \geq 0 \\ x^2 / 2 & x < 0 \end{cases}$$

According to condition (1) of the proposition, we select the following node function:

$$f_i(U_i) = 0.5(V_i^M - V_i^m)[1 + \tanh(U_i / q_i)] + V_i^m \tag{8}$$

where V_i^M and V_i^m are respectively upper limit and lower limit of the output of neuron i , and q_i is the reciprocal of the gain of neuron i . It is thus apparent that $f_i(U_i)$ is a monotonically increasing continuous function, and its upper limit and lower limit are respectively V_i^M and V_i^m . Therefore, the upper limit and the lower limit of the neuron output can be directly disposed by Eq. (8).

3 Formulation of Nonlinear Optimization Problems Using the Modified Hopfield Network Model

Let us consider the following nonlinear programming problem:

Minimize $S(V)$

subject to

$$g_{l_1}(V) = 0, \quad (l_1 = 1, 2, \dots, m_g)$$

$$h_{l_2}(V) \geq 0, \quad (l_2 = 1, 2, \dots, m_h)$$

where $V = (v_1, v_2, \dots, v_{n_1})^T$ is a vector, and $v_i (i = 1, 2, \dots, n_1)$ are variables.

When $S(V)$, $g_{l_1}(V)$ and $h_{l_2}(V)$ are polynomial functions, they can always be denoted in the following forms:

$$S(V) = -\sum_{i=1}^{n_1} V_i I_i - \frac{1}{2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_1} T_{ij_1} V_i V_{j_1} - \dots - \frac{1}{n_2 + 1} \sum_{i=1}^{n_1} \sum_{j_1=1}^{n_1} \dots \sum_{j_{n_2}=1}^{n_1} T_{ij_1 \dots j_{n_2}} V_i V_{j_1} \dots V_{j_{n_2}}$$

$$g_{l_1}(V) = \frac{1}{m_{l_1} + 1} \sum_{i=1}^{n_1} \sum_{k_1=1}^{n_1} \sum_{k_2=1}^{n_1} \dots \sum_{k_{m_{l_1}}=1}^{n_1} W_{ik_1 k_2 \dots k_{m_{l_1}}}^{(l_1)} V_i V_{k_1} V_{k_2} \dots V_{k_{m_{l_1}}} + \dots + \frac{1}{2} \sum_{i=1}^{n_1} \sum_{k_1=1}^{n_1} W_{ik_1}^{(l_1)} V_i V_{k_1} + \sum_{i=1}^{n_1} W_i^{(l_1)} V_i - b_{l_1} \quad (l_1 = 1, 2, \dots, m_g)$$

$$h_{l_2}(V) = \frac{1}{m_{l_2} + 1} \sum_{i=1}^{n_1} \sum_{k_1=1}^{n_1} \sum_{k_2=1}^{n_1} \dots \sum_{k_{m_{l_2}}=1}^{n_1} Q_{ik_1 k_2 \dots k_{m_{l_2}}}^{(l_2)} V_i V_{k_1} V_{k_2} \dots V_{k_{m_{l_2}}} + \dots + \frac{1}{2} \sum_{i=1}^{n_1} \sum_{k_1=1}^{n_1} W_{ik_1}^{(l_2)} V_i V_{k_1} + \sum_{i=1}^{n_1} W_i^{(l_2)} V_i - d_{l_2} \quad (l_2 = 1, 2, \dots, m_h)$$

where T , W and Q respectively satisfies conditions (3), (4) and (5) of the proposition.

Thus, m_g equality constraints can be satisfied by means of minimizing

$\sum_{l_1=1}^{m_g} F[g_{l_1}(V)]$, and m_h inequality constraints can be satisfied by means of minimizing

$\sum_{l_2=1}^{m_h} G[h_{l_2}(V)]$. We construct the following function:

$$E' = A \bullet S(V) + B \sum_{l_1=1}^{m_g} F[g_{l_1}(V)] + C \sum_{l_2=1}^{m_h} G[h_{l_2}(V)] \tag{9}$$

where A , B and C are positive constraints, and $F(\bullet)$ and $G(\bullet)$ are functions. Their specific forms are shown in the former section. It is apparent that when B and C are large enough (which is the same as solving optimization problems using Hopfield network model [2]), the approximate solution to this programming problem is obtained by minimizing this function. Appropriately fixing the comparative values of A , B and C is a problem that deserves further research. The larger the values of B and C compared with A are, the easier the feasible solutions are obtained.

When $1/R_i$ converges to zero, Eq. (9) and Eq. (7) bear the same form. Thus, the approximate solution to this programming problem can be obtained by solving the dynamic function that corresponds to Eq. (9) and is similar to Eq. (1) until it reaches a stable state. As the terms on the right side of Eq. (1) and Eq. (7) are sequentially corresponding to each other, as long as a programming problem can be defined in the forms of some terms of Eq. (7), the solution to the problem can be obtained by solving Eq. (1) with the terms corresponding to those of Eq. (7).

Compare Eq. (1) with Eq. (7), and it is known that C_i and E are unrelated to each other. Therefore, the value of C_i does not affect the solution to an optimization problem. However, in hardware implementation the value of C_i is connected with the time constant of circuit and affects the time a solution takes to reach a stable state. In software solving, the value of C_i influences the integration process, which is difficult for theoretical analysis. Currently, the value of C_i is 1. The value of R_i is related to E , connected with the time constant of circuit in hardware implementation, and usually influences the quality of solutions. Nevertheless, when the gain is high (i.e., R_i is very large), the terms related to R_i in Eq. (1) and Eq. (7) can be ignored. Therefore, in software solution R_i usually satisfies $1/R_i = 0$.

When we solve a programming problem using the above method, so long as the energy function comprises the equality constraint terms of the real variables of the original problem, we can only obtain an approximate optimal solution. The reason is that in fact this method changes the objective function. The simulated annealing method does not help on this point. If the objective function of the original problem and its constraints are coordinated well, then we can obtain a good solution (although it may not be the optimal solution). For pure integer programming problems, we may obtain the optimal solutions.

The solving method given above is a general one which is suitable for not only programming problems with continuous variables but also for those with discrete variables.

4 Simulation Results

In this section, the modified Hopfield network proposed in previous sections has been used to solve nonlinear optimization problems. We provide two examples to illustrate the effectiveness of the proposed architecture.

Example 1. Consider the following nonlinear optimization problem, which is composed of equality constraints and inequality constraints:

$$\text{Minimize } S(X) = x_1^2 + x_2,$$

subject to

$$g_1(X) = x_1^2 + x_2^2 - 9 = 0$$

$$h_1(X) = -(x_1 + x_2^2) + 1 \geq 0$$

$$h_2(X) = -(x_1 + x_2) + 1 \geq 0$$

We construct the following function:

$$E = AS(X) + BF[g_1(X)] + C\{G[h_1(X)] + G[h_2(X)]\}$$

In the problem solving process, it is pre-determined that $A = 1$, $B = C = 12$ and $q_1 = q_2 = 1$, and the initial values are $x_1 = x_2 = 0$. The computation result is $x_1 = -2.3704$ and $x_2 = -1.8376$. The optimal solution is $x_1^* \approx -2.37$, and $x_2^* \approx -1.84$. Thus, the computation result and the optimal solution are almost the same.

Example 2. Consider the following mixed-integer programming problem, which is composed by inequality constraints and bounded variables:

Minimize

$$S(X) = 0.5x_1^2x_2 + 3.89x_1x_2 + 0.406x_2 + 0.5x_3^2x_4 + 3.51x_3x_4 + 0.444x_4 + x_5^2x_6 + 4.01x_5x_6 + 0.505x_6$$

subject to

$$\begin{aligned} g_1(X) &= x_1x_2 + x_3x_4 + x_5x_6 - 3 = 0 \\ 0.3 \leq x_1 \leq 2.4; & 0.3 \leq x_3 \leq 2.4; 0.2 \leq x_5 \leq 1.5 \\ x_2 = 0, & 1; x_4 = 0, 1; x_6 = 0, 1 \end{aligned}$$

We construct the following function:

$$E = AS(X) + B \sum_{i=1}^3 x_{2 \times i} (1 - x_{2 \times i}) + CF[g_1(X)]$$

The upper limit constraints and the lower limit constraints of the variables are disposed by the node function (see Eq. (8)). Thus, the upper and lower limit constraint terms of x_1 , x_3 and x_5 are not included in the above function. The upper limit constraints of x_2 , x_4 and x_6 are 1, and their lower limit constraints are 0. Then, the minimum value of the second term on the right side of the above function is 0. At this time, $x_{2 \times i} (i = 1, 2, 3)$ is 1 or 0, which satisfies the requirement of the problem.

In the problem solving process, it is pre-determined that $A = 5$, $B = 1$, $C = 50$, $q_{2 \times i-1} = 1 (i = 1, 2, 3)$, and $q_{2 \times i} = 0.01 (i = 1, 2, 3)$. The initial values are $x_{2 \times i-1} = 1 (i = 1, 2, 3)$ and $x_{2 \times i} = 0.5 (i = 1, 2, 3)$. The computation result is

$$\begin{aligned} x_1 &= 1.3056, & x_2 &= 1, \\ x_3 &= 1.6840, & x_4 &= 1, \\ x_5 &= 0.2101, & x_6 &= 0. \end{aligned}$$

This example is a mixed-integer programming problem having four minimum solutions as shown in Table 1.

Table 1. The Four Minimum Solutions of Example 2

Solution No.	$S(X)$	x_1	x_2	x_3	x_4	x_5	x_6
1	14.164	1.310	1	1.690	1	Arbitrary values	0
2	14.386	1.072	1	1.452	1	0.476	1
3	14.937	Arbitrary values	0	2.167	1	0.833	1
4	15.698	2.040	1	Arbitrary values	0	0.960	1

It is apparent that the computation results are close to the optimal solutions. We discover that, when A , B and C are given different values, the function may converge to local minimum points. Usually, the function does not converge to the third and fourth solutions unless the initial values of x_{2xi} ($i = 1, 2, 3$) are very close to the two solutions, which indicates that we can obtain excellent solutions but probably not global optimal solutions using the proposed method to solve combinatorial optimization problems.

5 Conclusions

In this paper, we have developed a modified Hopfield network for solving nonlinear optimization problems, with polynomial objective function, polynomial equality constraints and polynomial inequality constraints. And a mapping of nonlinear optimization problems is formulated using the modified Hopfield network. The simulation results demonstrate that the proposed network is an efficient method to solve nonlinear optimization problems. From the simulation results we can conclude that the modified Hopfield network proposed in this paper has the advantages of global convergence and high accuracy of solutions. In order to guarantee this global convergent behavior, we also analyzed, for each example presented in the paper, the network convergence from several other points randomly generated. For all simulations, the network always converged to the same optimal solutions.

References

1. Bazaraa, M.S., Sherali, H.D., Shetty, C.M.: Nonlinear Programming, Wiley, New York (1993).
2. Hopfield, J.J., Tank, D.W.: Neural computation of decisions in optimization problems, Biol. Cybern. **52** (1985) 141-152.
3. Tank, D.W., Hopfield, J.J.: Simple neural optimization networks: An A/D converter, signal decision circuit and a linear programming circuit, IEEE Trans. Circ. Syst. **33** (1986) 533.
4. Kennedy, M.P., Chua, L.O.: Neural networks for nonlinear programming, IEEE Trans. Circ. Syst. **35** (1988) 554-562.

5. Rodriguez-Vazquez, A., Dominguez-Castro, R., Rueda, A., Huertas, J.L., Sanchez-Sinencio, E.: Nonlinear switched-capacitor neural networks for optimization problems, *IEEE Trans. Circ. Syst.* **37** (1990) 384-397.
6. Wu, Y., Xia, Y., Li, J., Chen, W.: A high-performance neural network for solving linear quadratic programming problems, *IEEE Trans. Neural Networks* **7** (1996) 643-651.
7. Xia, Y., Wang, J.: A projection neural network its application to constrained optimization problems, *IEEE Trans. Circ. Syst.* **49** (2000) 447-457.
8. Effati, S., Baymani, M.: A new nonlinear neural network for solving convex nonlinear programming problems, *Appl. Math. Comput.* (2004) 1-3.

A Novel Artificial Neural Network Based on Hybrid PSO-BP Algorithm in the Application of Adaptive PMD Compensation System

Ying Chen¹, Qiguang Zhu², and Zhiqian Li¹

¹Institute of Electrical Engineering, Yanshan University, Qinhuangdao 066004, China
chenying@ysu.edu.cn

²Institute of Information Science and Engineering, Yanshan University,
Qinhuangdao 066004, China

Abstract. An artificial neural network (ANN) based on hybrid algorithm combining particle swarm optimization (PSO) algorithm with back-propagation (BP) algorithm has been introduced to compensate the polarization mode dispersion (PMD) in the ultra-high speed optical communication system. The hybrid algorithm, also referred to as PSO-BP algorithm, has been adopted to train the weights of ANN, and it can make use of not only strong global searching ability of the PSO algorithm, but also strong local searching ability of the BP algorithm. In the proposed algorithm, a heuristic way was adopted to give a transition from particle swarm search to gradient descending search. The experimental results show that the hybrid algorithm is better than the Adaptive PSO algorithm and BP algorithm in convergent speed and convergent accuracy. And in the PMD compensation system, the ANN is used to optimize the degree of polarization (DOP) signal, which can achieve the random stochastic PMD compensation adaptively. Simulation results show the opening of eye diagram can be improved obviously.

1 Introduction

Recent terabit optical communication transmission experiments use massive WDM technology. Nevertheless, it is still attractive to increase the bit rate per channel instead of the number of wavelength channels, since requirements on wavelength stability, filter quality, and network management are relaxed if the number of wavelength channels is reduced. However, the higher the bit rate per channel the higher is the impairment due to polarization mode dispersion (PMD).

Even in installed fiber links with a mean differential group delay (DGD) as low as 10% of the bit period, system outages may occur, since instantaneous DGD values can still exceed the tolerable margin of about 30% of the bit period. PMD becomes a serious impairment at channel rates of 40 Gb/s and particularly at bit rates as high as 160 Gb/s, where the bit period is only 6.25 ps. Since PMD depends on environmental conditions and varies on different time scales, an adaptive and dynamic approach for PMD compensation is necessary.

Computation techniques play an important role in most electromagnetic engineering problems, in which optimizing processes have to be solved. Usually the complex nature

of many engineering problems involves an effective use of ANNs to solve them. An artificial neural network trained by some algorithms can be used to adaptive feedback control loop to compensate the PMD [1], [2]. One of the most critical phases in managing an ANN is the training, when weights of the neural connections have to be set. The parameters of the network have to be optimized in order to reach a good and accurate output. Therefore the learning process should result in finding the weights configuration associated to the minimum output error.

Usually problems are associated to an objective function to be optimized. The function, also called fitness function, provides the interface between the physical problems and the optimization algorithms. There are often lots of problems related to these algorithms. The huge number of variables is the first difficulty when dealing with one of these optimizing problems. And in addition, there are lots of configurations with values of the objective function that are very similar with each other and very close to the global optimum case, even if these configurations are sub-optimal. In general, finding a solution in an optimization process means to reach a balance among different and often conflicting goals, as a consequence this search could be very difficult.

ANNs are good tools to understand the complex and nonlinear relationships among these data. In this paper, the proposed hybrid PSO-BP algorithm is proposed to train the ANN [3], [4]. This hybrid uses the Adaptive PSO algorithm to do global search in the beginning of stage, and then uses the BP algorithm to do local search around the global optimum P_g . In particular, this hybrid algorithm will be used to train the ANN weights for function approximation and classification problems, respectively compared with the Adaptive PSO algorithm and the BP algorithm in convergent speed and generalization performance.

2 Theoretical Backgrounds

2.1 Adaptive PMD Compensation

It is widely believed that the one-stage compensators are able to compensate PMD to first order. They have three or four parameters to be controlled depending on whether the differential group delay line is fixed or varied. The two-stage compensators can compensate the PMD up to the second order. They have six or seven parameters to be controlled depending on whether the delay line is fixed or varied. The adaptive PMD compensation is a process for a control algorithm to find optimal combinations of control parameters, in order for the feedback signal to reach a global optimum, in an intelligent, fast, and reliable manner.

In the experiment shown in Fig.1, the degree of polarization (DOP), obtained by an in-line polarimeter, was used as feedback signal. The optical pulses at the receiving end have a DOP of one when there is no PMD in the fiber link, and the DOP value decreases as PMD increases. The polarization controller (PC) used in the compensation unit is the electrically controlled one which has four fiber-squeezer cells to be adjusted with voltage of 0~10 V, out of which three cells were used in the experiment. Thus, the problem of adaptive PMD compensation can be described mathematically as a problem of maximization of DOP, that is $MAX(function) \backslash parameters$, where the function represents the DOP value in the experiment. The parameters here are the voltages for

controlling the PCs and the variable delay line. There is no simple method to predict the function in an adaptive PMD compensation system. A good searching algorithm is required to search the global maximum in a D-dimensional hyperspace.

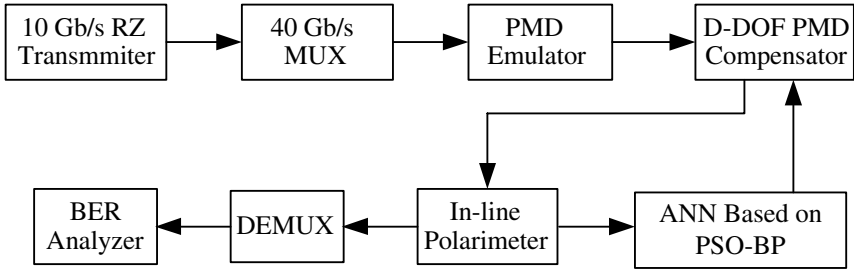


Fig. 1. Experimental setups of adaptive PMD compensation for 40-Gb/s optical time-division-multiplexing transmission systems with ANN based on PSO-BP algorithm

2.2 ANN Architecture

A multi-layer ANN consists of a system of simple interconnected neurons, or nodes. It is a model representing a non-linear mapping between input and output vectors. The architecture of a multi-layer ANN is variable, but, in general, consists of several layers of neurons. The neurons are connected to each other by weighted links over which signals can pass. Each neuron receives multiple inputs from other neurons in proportion to their connection weights and generates a single output, which may be propagated to several other neurons. The input layer plays no computational role but is merely used to pass the input vector to the network. The ANN has the ability to learn through training [5], [6]. The training requires a set of training data, i.e., a series of input and output vectors. During the training, the ANN is repeatedly presented with the training data, and weights in the network are adjusted from time to time till the desired input-output mapping occurs. After the training, input vectors that are not belonging to the training pairs will be used simulate the system and produce the corresponding output vector. By selecting a suitable set of weight and transfer function, it was known that the artificial neural network can approximate any smooth, measurable function between the input and output vectors. The neural networks are widely applied in many areas such as prediction, system modeling and control.

By using the DOP as the object function, we can account for the accuracy of the DOP as measured by in-line polarimeter, which can accurately track the fluctuating PMD conditions. The in-line polarimeter measures the state of polarization of the light. Describing the state of polarization in term of the stokes' vector \vec{S} [7],

$$\vec{S} = \begin{pmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \end{pmatrix} \tag{1}$$

Then, the DOP is calculated as the quotient of the polarized light power and the total power

$$DOP = \frac{\sqrt{S_1^2 + S_2^2 + S_3^2}}{S_0} \quad (2)$$

Polarization can be measured with polarimeters based on rotating waveplates or polarizers. Those polarimeters depend on stable input polarization during plate rotation. Stable input polarization, however, is usually not found in optical transmission systems, and rapid fluctuations may result from fiber touching or other environmental influences. Therefore, real-time polarization measurement is required in the practical communication links.

Thus, in the ANN structure mentioned in this paper, S_1, S_2, S_3 has been regarded as the three inputs of the input layer, and the DOP is used as the output function of the output layer [8]. Compared with the best DOP, waveplates will be modulated and will generate three new inputs of the input layer. And by training the ANN in the certain iteration, a good DOP will be obtained and the PMD condition will be improved obviously.

2.3 Adaptive Particle Swarm Optimization

Adaptive PSO algorithm is an evolutionary algorithm based on a model of social interaction between independent agents (particles) that uses social knowledge in order to find the global maximum or minimum of a function [9], [10]. This computational technique, adopting a pseudo-biological approach, takes its origin from the simulation of social behaviors such as those related to synchronous bird flocking and fish schooling. It is similar to genetic algorithms and simulated annealing, but it operates emulating social interaction between independent agents and utilizes swarm intelligence to achieve the goal of optimizing a specific fitness function in a way easy to understand and implement. Any set of coordinates in the N-dimensional space is a particular position of an agent and represents a solution; it corresponds to a particular value of the fitness function. Each particle also has an associated velocity, and that takes into account the best position reached by all ones and the best position, resulting in a migration of the swarm towards the global optimum [11], [12].

At the starting point, Adaptive PSO algorithm randomly initializes the position and velocity of a random population of particles. In the PSO technique each particle i is defined by its position vectors X_i in the space of the parameters to be optimized, and in addition, such a particle also has a random velocity V_i in the parameter space. At each epoch, the particle moves according to its velocity and the cost function to be optimized $f(X)$ is evaluated for each particle in its current position. The value of the cost function is compared with the best value attained during the previous time steps. The best value ever obtained for each particle is stored, and the position, at which it was attained, P_i is stored too. Velocity of the particle, the main PSO operator, is then stochastically updated by letting the particle be attracted by the position P_i of its individual optimum and the position P_g which is the global optimum. The APSO can be described as follows:

$$V_{i+1} = \omega V_i + c_1 \text{rand}()_1 (P_i - X_i) + c_2 \text{rand}()_2 (P_g - X_i) \quad (3)$$

$$X_{i+1} = X_i + V_i \quad (4)$$

Here $w < 1$ is known as the “inertial weight”. This algorithm by adjusting the parameter w can make w reduce gradually as the generation increases. In the searching process of the Adaptive PSO algorithm, the searching space will reduce gradually as the generation increases. So the Adaptive PSO algorithm is more effective, because the searching space reduces step by step nonlinearly, so the searching step length for the parameter w here also reduces correspondingly. After each generation, the best particle of particles in last generation will replace the worst particle of particles in current generation, thus better result can be achieved. The inertial weight is chosen between 0 and 1 in order to determine to what extent the particle remains along its original course unaffected by the pull of the other two terms, larger values for w result in smoother, more gradual changes in direction through search space. These ones are balanced by the scaling factors c_1, c_2 and two random positive numbers $rand()_1$ and $rand()_2$ with a uniform distribution and a value from 0 to 1. c_1 and c_2 also control how far a particle will move in a single iteration. Typically these are both set to a value of 2.

Generally, in the beginning stages of algorithm, the inertial weight w should be reduced rapidly, when around optimum, the inertial weight w should be reduced slowly. So here, the following selection strategy has been adopted:

$$w = \begin{cases} w_0 - (w_1 / \max 1) * \text{generation}, & 1 \leq \text{generation} \leq \max 1 \\ (w_0 - w_1) * e^{(\max 1 - \text{generation})/k}, & \max 1 \leq \text{generation} \leq \max 2 \end{cases} \quad (5)$$

where w_0 is the initial inertial weight, w_1 is the inertial weight of linear section ending, $\max 2$ is the total searching generations, $\max 1$ is the used generations that inertial weight is reduced linearly, generation , is a variable whose range is $[1, \max 2]$. Through adjusting k , we can achieve different ending values of inertial weight. In particular, the value of $\max 2$ is selected according to empirical knowledge.

2.4 Hybrid PSO-BP Algorithm and ANN Training

The hybrid PSO–BP algorithm is an optimization method combining the PSO with the BP. The PSO algorithm is a global algorithm, which has a strong ability to find global optimistic result, this PSO algorithm, however, has a disadvantage that the search around global optimum is very slow. The BP algorithm, on the contrary, has a strong ability to find local optimistic result, but its ability to find the global optimistic result is weak. The fundamental idea for this hybrid PSO–BP algorithm is that at the beginning stage of searching for the optimum, the PSO is employed to accelerate the training speed. When the fitness function value has not changed for some generations, or value changed is smaller than a predefined number, the searching process is switched to gradient descending searching according to this heuristic knowledge.

Similar to the Adaptive PSO algorithm, the searching process of the PSO–BP algorithm is also started from initializing a group of random particles. First, all the particles are updated according to the Eqs. (3) and (4), until a new generation set of particles are generated, and then those new particles are used to search the global best position in the solution space. Finally the BP algorithm is used to search around the global optimum. In this way, this hybrid algorithm may find an optimum more quickly. The procedure for this PSO–BP algorithm can be summarized as follows:

Step 1: Initialize the positions and velocities of a group of particles randomly in the range of $[0, 1]$.

Step 2: Evaluate the fitness value of each initialized particle, and P_i is set as the positions of the current particles, while P_g is set as the best position of the initialized particles.

Step 3: If the maximal iterative generations are arrived, go to Step 8, else, go to Step 4.

Step 4: The best particle of the current particles is stored. The positions and velocities of all the particles are updated according to Eqs. (3) and (4), then a group of new particles are generated, If a new particle flies beyond the boundary $[X_{\min}, X_{\max}]$, the new position will be set as X_{\min} or X_{\max} ; if a new velocity is beyond the boundary $[V_{\min}, V_{\max}]$, the new velocity will be set as V_{\min} or V_{\max} . That is,

$$\begin{aligned} X_{i+1} &= X_i + V_i, & X_{\min} < X_i + V_i < X_{\max} \\ X_{i+1} &= X_{\max}, & X_i + V_i \geq X_{\max} \\ X_{i+1} &= X_{\min}, & X_i + V_i \leq X_{\min} \end{aligned} \quad (5)$$

Step 5: Evaluate the fitness value of each new particle, and the worst particle is replaced by the stored best particle. If the new position of the j th particle is better than P_i , P_i is set as the new position of the j th particle. If the best position of all new particles is better than P_g , then P_g is updated.

Step 6: Reduce the inertia weights w according to the selection strategy described in Section 3.

Step 7: If the current P_g is unchanged for ten generations, then go to Step 8; else, go to Step 3.

Step 8: Use the BP algorithm to search around P_g for some epochs, if the search result is better than P_g , output the current search result; or else, output P_g .

The parameter w , in the above PSO–BP algorithm also reduces gradually as the iterative generation increases, just like the APSO algorithm. The selection strategy for the inertial weight w is the same as the one described in Section 2.2, i.e., firstly reduce w linearly then reduce it nonlinearly. But the parameter $\max I$ generally is adjusted to an appropriate value by many repeated experiments, and then an adaptive gradient descending method is used to search around the global optimum P_g .

When the adaptive PSO algorithm is used in evolving weights of ANN, every particle represents a set of weights, and in the encoding strategy, each particle is encoded for a vector. For ANN involved, each particle represents all weights of a ANN structure. In the following experiments, the performances of BP algorithm, Adaptive PSO algorithm and PSO-BP algorithm in evolving the weights of the ANN have been compared. Supposed that every weight in the network was initially set in the range of $[-50, 50]$, and all thresholds in the network were 0 s. Supposed that every initial particle was a set of weights generated at random in the range of $[0, 1]$. Let the initial inertial weight w be 1.8, the acceleration constants, both c_1 and c_2 be 2.0, $rand()_1$ and $rand()_2$ be two random numbers in the range of $[0, 1]$, respectively. The maximum velocity assumed as 10 and the minimum velocity as -10. The initial velocities of the

initial particles were generated randomly in the range of [0, 1]. After each iteration, if the calculated velocity was larger or smaller than the maximum velocity or the minimum velocity, it would be reset to 10 or -10. The population size is 200.

3 Experimental Research

3.1 PMD Compensation System with ANN Based on PSO-BP Algorithm

In the experiment, as is shown in Fig.2, the pulses emitted from the tunable laser are scrambled by the polarization scrambler, thus different PMDs can be obtained in different channels. The searching and tracking of DOP is achieved the ANN based on PSO-BP algorithm, and through digital-to-analog (D/A) conversion, the corresponding voltages are performed on the polarization controllers. By this can a feedback control module be build up.

The PMD compensator can be separated into two parts, as is shown in Figure 4. One part is composed of PC1 and a piece of polarization-maintained fiber, in which, PC1 is under the control of three voltages from the D/A converter to change the polarization state of signal. The other part includes PC2 and a variable delay line made up of birefringence crystals, in which, PC2 is under the control of another three voltages from the D/A converter, and the birefringence crystal delay line can be controlled by SCM or PC to change the output differential group delay dynamically according to the feedback signal.

3.2 Experimental Results

Each particle represents a single intersection of multidimensional hyperspace. The position of the i th particle is represented by the position vector $X_i = (x_{i1}, x_{i2}, \dots, x_{i6})$. In the 6-DOF PMD compensation scheme, the components of the i th particle are represented by the combination of 6 voltages $(V_{i1}, V_{i2}, \dots, V_{i6})$. The PSO algorithm for adaptive PMD compensation should include two stages. First, the searching algorithm finds the global optimum from any initial PMD condition. Then the tracking algorithm starts to track the changed optimum because the PMD in the real fiber link always randomly changes, due to changes in the environment such as temperature fluctuations, etc. The response time of the compensator depends on the strategy of the chosen algorithms and the performance of the hardware. In a compensation loop, D/A converters write multi-voltages to the voltage-controlled PCs, the PCs adjust their wave plates to proper states, then perform A/D conversions, and process the data in the computer processor with the ANN based on PSO-BP algorithm. And as is shown in Fig.3, the maximum iteration number for each process is set to 50, and it is easy to observe that all the DOP values reach 0.9 within about 25 iterations and the maximum DOP available can be 0.96. Thus, PSO-BP algorithm can undertake the task of solving multidimensional problems well, and it is possible to achieve the searching of DOP for high-order PMD compensation. From the eye diagrams shown as Fig.4 and Fig.5, the compensation effectiveness can be easily observed. Compared the eye diagram before and after compensation, the opening of the eye diagram has been improved obviously. And the compensation response time can be attained to 70 ms.

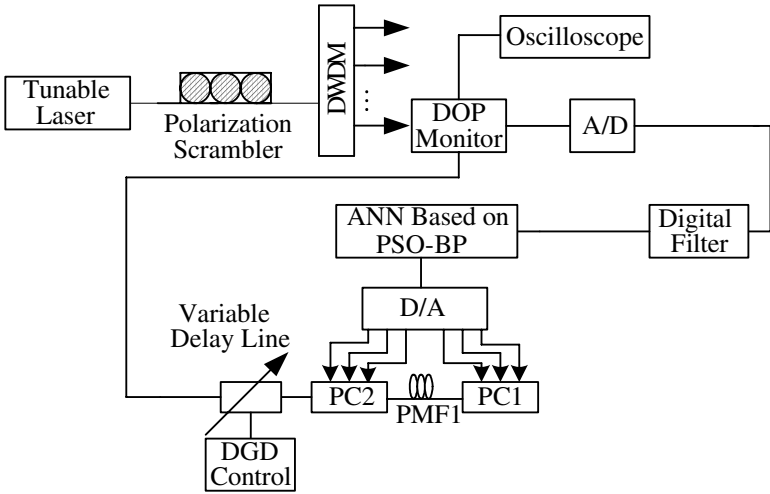


Fig. 2. Schematic diagram of experiment setup for adaptive PMD compensation

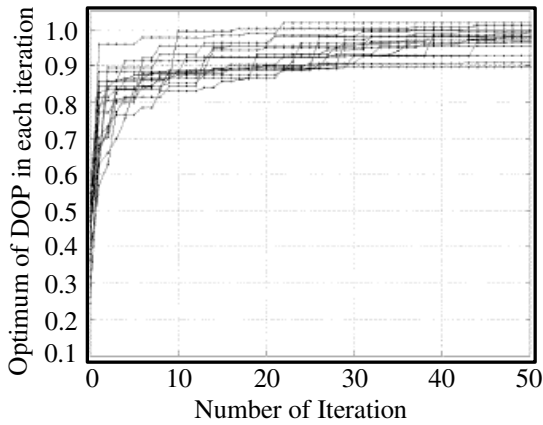


Fig. 3. Optimum of DOP in each iteration in 6-DOF PMD compensation

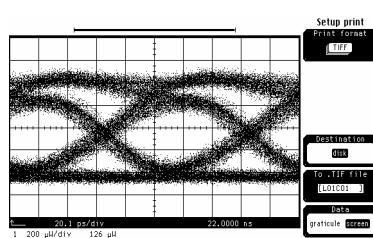
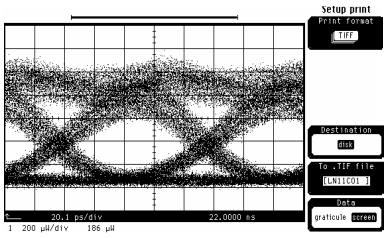


Fig. 4. Eye diagram before compensation

Fig. 5. Eye diagram after compensation

4 Conclusions

An ANN trained by hybrid PSO-BP algorithm has been proposed to achieve adaptive PMD compensation. From the simulation result, the PSO-BP algorithm spends less CPU time than the PSO algorithm and the BP algorithm, and it has been demonstrated that a random stochastic PMD can be successfully compensated by the ANN based on PSO-BP technique.

Acknowledgement

This work was supported by National Natural Science Foundation of China grant no. 60377002.

References

1. Zheng, Y., Zhang, X., Zhou, G.: Automatic PMD Compensation Experiment with Particle Swarm Optimization and Adaptive Dithering Algorithms for 10-Gb/s NRZ and RZ Formats. *IEEE J. Quantum Electron.* **40**(4) (2004) 427-435
2. Zhang, X., Zheng, Y.: Particle Swarm Optimization Used as A Control Algorithm for Adaptive PMD Compensation. *IEEE Photon. Technol. Lett.* **17**(1) (2005) 85-87
3. Shao, J., Zhang, L. X., Qian, F.: Soft Sensing Modeling via Artificial Neural Network Based on Pso-Alopex. *Proceedings of 2005 International Conference* **7** (2005) 4210-4215
4. Grimaldi, E.A., Grimaccia, F., Mussetta, M.: PSO as An Effective Learning Algorithm for Neural Network Applications. *Proceedings ICCEA 2004 3rd International Conference* (2004) 557-560
5. Zhang, C., Shao, H., Li, Y.: Particle Swarm Optimization for Evolving Artificial Neural Network. *Systems, Man, and Cybernetics, IEEE International Conference* **4** (2000) 2487 -2490
6. Gudise, V.G, Venayagamoorthy, G.K.: Comparison of Particle Swarm Optimization and Backpropagation as Training Algorithms for Neural Networks. *Swarm Intelligence Symposium* (2003) 110-117
7. Rasmussen, Jens C., Akihiko I., George, I.: Automatic Compensation of Polarization-Mode Dispersion for 40 Gb/s Transmission Systems. *IEEE J. Lightwave Technol.* **20** (12) (2002) 2101-2108
8. Kikuchi, N.: Analysis of Signal Degree of Polarization Degradation Used as Control Signal for Optical Polarization Mode Dispersion. *IEEE J. Lightwave Technol.* **19** (2001) 480-486
9. van den Bergh, F.: A Cooperative Approach to Particle Swarm Optimization. *IEEE Transactions on Evolutionary Computation* **8**(3) (2004) 225-239
10. Abdelbar, A. M., Abdelshahid, S.: Swarm Optimization with Instinct Driven Particles. *Proceedings IEEE Congress on Evolutionary Computation* (2003) 777-782
11. Robinson, J., Rahmat-Samii, Y.: Particle Swarm Optimization in Electromagnetism. *IEEE Transactions Antennas and Propagation* **52**(2) (2004) 591-594
12. Eberhart, R. C., Shi, Y.: Particle Swarm Optimization: Developments, Applications and Resources. *Proceedings IEEE International Conference on Evolutionary Computation*,(2001) 81-86

Stabilizing Lagrange-Type Nonlinear Programming Neural Networks

Yuancan Huang

Intelligent Robotics Institute, Beijing Institute of Technology
Nandajie 5, Zhongguancun, Haidian, 100081 Beijing, China
yuancanhuang@bit.edu.cn

Abstract. Inspired by the Lagrangian multiplier method with quadratic penalty function, which is widely used in Nonlinear Programming Theory, a Lagrange-type nonlinear programming neural network whose equilibria coincide with KKT pairs of the underlying nonlinear programming problem was devised with minor modification in regard to handling inequality constraints [1,2]. Of course, the structure of neural network must be elaborately conceived so that it is asymptotically stable. Normally this aim is not easy to be achieved even for the simple nonlinear programming problems. However, if the penalty parameters in these neural networks are taken as control variables and a control law is found to stabilize it, we may reasonably conjecture that the categories of solvable nonlinear programming problems will be greatly increased. In this paper, the conditions stabilizing the Lagrange-type neural network are presented and control-Lyapunov function approach is used to synthesize the adjusting laws of penalty parameters.

1 Problem Statement

Since an equality constraint can be equivalently decomposed into two coupled inequality constraints, without loss of generality we consider the following nonlinear programming problems with only inequality constraints:

$$\begin{aligned} & \text{minimize } f(x) \\ & \text{subject to } g(x) \leq 0, \end{aligned} \tag{1}$$

where $f(x) : R^n \rightarrow R$ and $g(x) : R^n \rightarrow R^m$ are scalar objective function and vector constraint function, respectively. Its feasible set is defined as $X = \{x : g(x) \leq 0, x \in R^n\}$.

Let x^* be a vector satisfying the constraint conditions, then $I(x^*)$ denotes a set of index i for which $g_i(x^*) = 0$, namely

$$I(x^*) = \{i \mid g_i(x^*) = 0, i = 1, 2, \dots, m\}. \tag{2}$$

If the gradients $\nabla g_i(x^*), i \in I(x^*)$ are linearly independent, then x^* is called regular point. The constraint whose value equals to zero at x^* is said to be active; otherwise, inactive.

If the Lagrangian function of the problem (II) is defined as

$$L(x, \lambda) = f(x) + \sum_{i=1}^m \lambda_i g_i(x) . \tag{3}$$

Karush-Kuhn-Tucker Theorem [7] furnishes the necessary condition for some x^* being a local minima of the problem (II):

Theorem 1. *Let x^* be a local minimum of the problem (I) and assume that x^* is a regular point. Then there exists a unique vector λ^* such that*

$$\nabla_x L(x^*, \lambda^*) = 0 , \tag{4}$$

$$\lambda_i^* g_i(x^*) = 0, i = 1, 2, \dots, m , \tag{5}$$

$$g(x^*) \leq 0 ,$$

$$\lambda^* \geq 0 .$$

The basic idea using neural network to solve nonlinear programming problem is to construct a continuous-time dynamical system whose equilibria satisfy the necessary condition in Karush-Kuhn-Tucker Theorem [1,2,3,4,5,6,10,11,12]. Inspired by the Henstein-Rockfellar-Powell Lagrangian Multiplier Method with quadratic penalty function [8,9], such a system is established:

$$\dot{x} = -\nabla_x L_c(x, \lambda) \tag{6}$$

$$\dot{\lambda}_i = 2\lambda_i g_i(x), i = 1, 2, \dots, m$$

in which the augmented Lagrangian function is defined as

$$L_c(x, \lambda) = f(x) + \sum_{i=1}^m \lambda_i^2 g_i(x) + \frac{1}{2} \sum_{i=1}^m c_i (\lambda_i g_i(x))^2 , \tag{7}$$

where c_i is positive penalty parameter. Note that square of multipliers is used to eliminate nonnegative constraint imposed on original multipliers in Karush-Kuhn-Tucker Theorem.

We may readily write the analytical expression of multipliers:

$$\lambda_i(t) = \lambda_i(0) e^{\int_0^t 2g_i(x) dt}, i = 1, 2, \dots, m , \tag{8}$$

where assume that $\lambda_i(0)$ is a nonzero initial value of $\lambda_i(t)$.

From this expression, it is obvious that, if not all inequality constraints are satisfied, the related multipliers will increase continuously so that the trajectory of $x(t)$ moves towards the feasible set. Therefore, if the constructed dynamical system is asymptotically stable, x will ultimately tend to the KKT pairs of the problem whereas the inactive multipliers approach to zero, and the active ones to some constants.

In [2], the asymptotic stability is proved under the assumption of second sufficient condition [7], and convergent mechanism is also analyzed using LaSalle

Invariance Principle [13,14]. Unfortunately, the asymptotic stability may be only guaranteed for some special nonlinear programming neural networks after cumbersome construction. However, if the positive penalty parameter c_i is considered as a control variable, we may reasonably conjecture that much more nonlinear programming neural networks will be endowed with desired asymptotic stability by devising appropriate adjusting laws of penalty parameters.

In conformity to the notation convention of control theory, let $F(x) = \nabla f(x)$, $G_i(x) = \nabla g_i(x)$, where ∇ represents the gradient of a function, and $c_i = u_i^2$. Then we have

$$\begin{aligned} \dot{x} &= -F(x) - \sum_{i=1}^m \lambda_i^2 G_i(x) - \sum_{i=1}^m \lambda_i^2 g_i(x) G_i(x) u_i^2 \\ \dot{\lambda}_i &= 2\lambda_i g_i(x), i = 1, 2, \dots, m . \end{aligned} \tag{9}$$

The system is standard input affine nonlinear system with positive control [15]. If a stabilizing feedback law $u_i = k_i(x, \lambda), i = 1, 2, \dots, m$, is synthesized for this system, it will eventually settle down to the KKT pair of the nonlinear programming problem. Now two questions, partly answered in this paper, are arising:

- (1) Under which conditions the system can be stabilized; and
- (2) How to design the stabilizing feedback law.

Our paper is organized as follows: Section 2 presents some conditions for existence of stabilizing control laws. In Section 3, we use control-Lyapunov function to synthesize the stabilizing control law. Finally, the conclusion and discussion are given.

2 Conditions for Existence of Stabilizing Control Laws

The stabilization problem in this paper can be regarded as a special case of the following general nonlinear finite-dimensional control systems:

$$\dot{x}(t) = f(x(t), u(t)) , \tag{10}$$

where states $x(t) \in R^n$ for all t , and controls are measurable locally essentially bounded maps

$$u(\cdot) : [0, \infty) \rightarrow \mathcal{U} \in R^m$$

into the control-value set. In particular, we often consider systems with no input:

$$\dot{x}(t) = f(x(t)) . \tag{11}$$

All definitions for such systems are implicitly applied as well to systems with input (10) by setting $u \equiv 0$.

Without loss of generality, we state all definitions and theorems for the case when the equilibrium point is at the origin of R^n since any equilibrium point can be shifted to the origin via a change of variables.

Here comparison functions are used to formulate the stability concepts. The relevant definitions are first recalled. The class of \mathcal{K}_∞ functions consist of all $\alpha : R_{\geq 0} \rightarrow R_{\geq 0}$ which are continuous, strictly increasing, unbounded, and satisfy $\alpha(0) = 0$. The class of \mathcal{KL} functions consists of those $\beta : R_{\geq 0} \times R_{\geq 0} \rightarrow R_{\geq 0}$ with the properties that (i) $\beta(\cdot, t) \in \mathcal{K}_\infty$ for all t , and (ii) $\beta(r, t)$ decreases to zero as $t \rightarrow \infty$. \mathcal{N} is used to denote the set of all nondecreasing functions $\sigma : R_{\geq 0} \rightarrow R_{\geq 0}$.

Expressed in this language, the property of global asymptotic stability(GAS) for the system $\dot{x} = f(x, 0)$ becomes:

$$(\exists \beta \in \mathcal{KL}) \quad |x(t, x_0, 0)| \leq \beta(|x_0|, t) \quad \forall x_0, \forall t \geq 0 .$$

A system is called to be (open loop, globally) asymptotically controllable, if for each initial state x_0 there exists some control $u = u_{x_0}(\cdot)$ defined on $[0, \infty)$, such that the corresponding solution $x(t, x_0, u)$ is defined for all $t \geq 0$, and converges to zero as $t \rightarrow \infty$, with "small" overshoot. Moreover, we wish to rule out the possibility that $u(t)$ becomes unbounded for x near zero. The precise formulation is as follows:

$$(\exists \beta \in \mathcal{KL})(\exists \sigma \in \mathcal{N}) \quad \forall x_0 \in R^n \quad \exists u, \|u\|_\infty \leq \sigma(|x_0|) , \\ |x(t, x_0, u)| \leq \beta(|x_0|, t) \quad \forall t \geq 0 .$$

Finally, we say that $k : R^n \rightarrow \mathcal{U}$ is a feedback stabilizer for the system with input (10) if k is locally bounded(that is, k is bounded on each bounded subset of R^n), $k(0) = 0$, and the closed-loop system

$$\dot{x}(t) = f(x(t), k(x)) \tag{12}$$

is GAS, i.e. there is some $\beta \in \mathcal{KL}$ so that $|x(t)| \leq \beta(|x(0)|, t)$ for all solutions and all $t \geq 0$.

In the sequel, some conditions for existence of continuously differentiable, continuous, and discontinuous stabilizing control laws are introduced.

2.1 Constant or Continuously Differentiable Feedback Stabilization

First, we present the results in [2]. The following theorem is well-known Lyapunov's indirect method[16,17]:

Theorem 2. *Let $x = 0$ be an equilibrium point for the nonlinear system with no input*

$$\dot{x}(t) = f(x(t)) , \tag{13}$$

where $f : D \rightarrow R^n$ is continuously differentiable and D is a neighborhood of the origin. Let

$$A = \frac{\partial f}{\partial x}(x) |_{x=0} .$$

Then the origin is asymptotically stable if $Re\lambda_i < 0$ for all eigenvalues of A .

If the second-order sufficient conditions for a x^* being minimum of the problem (II) is satisfied, we have shown that all eigenvalues of the linearized part of the system (9) has negative real part while the constant control laws u_i^2 greater than some positive real number are applied to (2). Hence there is

Proposition 1. *Let (x^*, λ^*) is the KKT pair for the problem (7). If the second-order sufficient condition is satisfied, then the system (9) is locally asymptotically stable while the constant control laws that guarantee u_i^2 greater than some positive real number are used.*

From Brockett's theorem in [18], the necessary condition for the existence of a continuously differentiable stabilizing feedback is written:

Theorem 3. *Let $\dot{x} = f(x, u)$ be given with $f(0, 0) = 0$ and $f(\cdot, \cdot)$ continuously differentiable in a neighborhood of $(0, 0)$. A necessary condition for the existence of a continuously differentiable control law which makes $(0, 0)$ asymptotically stable is that:*

- (i) *the linearized system should have no uncontrollable modes associated with eigenvalues whose real part is positive.*
- (ii) *there exists a neighborhood B of the origin such that for each $x_1 \in B$ there exists a control $u(\cdot)$ defined on $[0, \infty)$ such that this control steers the solution of $\dot{x} = f(x, u)$ from x_1 at $t = 0$ to 0 at $t = \infty$.*
- (iii) *the mapping $(x, u) \mapsto f(x, u)$ should be onto an open set containing $(0, 0)$.*

Remark: Mentioned as before the system (9) is a control affine nonlinear system with positive control. So the Lie algebra rank condition cannot be taken as a surrogate of condition (ii), but it can be used to rule out the systems where the rank condition is not satisfied.

2.2 Regular and Continuous Feedback Stabilization

Conventionally, it turns out that requirements away from zero, say asking whether k is continuous or smooth, are not very critical; it is often the case that one can "smooth out" a continuous feedback or, even, make it real-analytic via Grauert's Theorem, away from the origin. So, in order to avoid unnecessary complications in exposition due to nonuniqueness, a feedback k is called regular if it is locally Lipschitz on $R^n \setminus 0$. A necessary condition, originated from Brockett's Theorem, for existence of continuous feedback stabilization is presented [21]:

Theorem 4. *Let $\dot{x} = f(x, u)$ be given with $f(0, 0) = 0$ and $f(\cdot, \cdot)$ locally Lipschitz on (x, u) . If there is a stabilizing feedback which is regular and continuous at zero, then the map $(x, u) \mapsto f(x, u)$ is open at zero.*

2.3 Discontinuous Feedback Stabilization

Before we discuss the condition for existence of discontinuous feedback stabilization, some concepts are states as follows:

By a sampling schedule or partition $\pi = \{t_i\}_{t_i \geq 0}$ of $[0, +\infty)$ we mean an infinite sequence

$$0 = t_0 < t_1 < t_2 < \dots$$

with $\lim_{i \rightarrow \infty} t_i = \infty$. We call

$$d(\pi) := \sup_i (t_{i+1} - t_i)$$

the diameter of π . Suppose that k is a given feedback law for the system (10). For each π , the π -trajectory starting from x_0 of the system (10) is defined recursively on the interval $[t_i, t_{i+1}), i = 0, 1, \dots$, i.e., on each interval $[t_i, t_{i+1})$, the initial state is measured, the control value $u_i = k(x(t_i))$ is computed, and the constant control $u \equiv u_i$ is applied until time t_{i+1} ; the process is then iterated. In other words, we start with $x(t_0) = x_0$ and solve recursively

$$\dot{x}(t) = f(x(t), k(x(t_i))), \quad t \in [t_i, t_{i+1}), i = 0, 1, 2, \dots$$

using as initial value $x(t_i)$ the endpoint of the solution on the preceding interval. The ensuing π -trajectory, which is denoted as $x_\pi(\cdot, x_0)$, is defined on some maximal nontrivial interval; it may fail to exist on the entire interval $[0, \infty)$ due to a blow-up on one of the subintervals $[t_i, t_{i+1})$. We say that it is well defined if $x_\pi(\cdot, x_0)$ is defined on all of $[0, \infty)$.

Definition 1. *The feedback $k : R^n \rightarrow U$ stabilizes the system (10) if there exists a function $\beta \in \mathcal{KL}$ so that the following property holds: For each*

$$0 < \varepsilon < K,$$

there exists a $\delta = \delta(\varepsilon, K)$ such that, for every sampling schedule π with $d(\pi) < \delta$, and for each initial state x_0 with $|x_0| < K$, the corresponding π -trajectory of (10) is well-defined and satisfies

$$|x_\pi(t, x_0)| \leq \max\{\beta(K, t), \varepsilon\} \quad \forall t \geq 0.$$

In particular, we have

$$|x_\pi(t, x_0)| \leq \max\{\beta(|x_0|, t), \varepsilon\} \quad \forall t \geq 0,$$

whenever $0 < \varepsilon < |x_0|$ and $d(\pi) < \delta(\varepsilon, |x_0|)$ (just take $K := |x_0|$).

The definition of stabilization is physically meaningful, and is very natural in the context of sampled-data (computer control) systems. It says that a feedback k stabilizes the system if it drives all states asymptotically to the origin and with small overshoot when using any fast enough sampling schedule. A high enough sampling frequency is generally required when close to the origin, in order to guarantee small displacements, and also at infinity, so as to preclude large excursions or even blow-ups in finite-time. This is the reason for making δ depend on ε and K . If the admissible feedback stabilization is extended to the discontinuous case, then there is

Theorem 5. *The system (10) admits a stabilizing feedback if and only if it is asymptotically controllable (19).*

3 Control-Lyapunov Functions' Approach for Stabilizing Control Laws

In Section 2, the conditions for existence of stabilizing control laws are given. Now we demonstrate how to get them, although it is difficult for practical use.

According to the classical converse theorems of Massera and Kurzweil, whenever a system is GAS, there always exists a smooth Lyapunov function V . Hence, for each $x \neq 0$ there is some u so that $\dot{V}(x, u) < 0$. This is the characterizing property of control-Lyapunov functions.

We say that a continuous function

$$V : R^n \rightarrow R_{\geq 0}$$

is positive definite if $V(x) = 0$ only if $x = 0$, and it is proper if for each $\alpha \geq 0$ the set $\{x \mid V(x) \leq \alpha\}$ is compact, or, equivalently, $V(x) \rightarrow \infty$ as $|x| \rightarrow \infty$ (radial unboundedness). A property which is equivalent to properness and positive definiteness together is

$$(\exists \underline{\alpha}, \bar{\alpha} \in \mathcal{K}_\infty) \underline{\alpha}(|x|) \leq V(x) \leq \bar{\alpha}(|x|) \quad \forall x \in R^n .$$

3.1 Differentiable Control Lyapunov Functions

A differentiable control Lyapunov function (clf) is a differentiable function $V : R^n \rightarrow R_{\geq 0}$ which is proper, positive definite, and infinitesimally decreasing, meaning that there exists a positive definite continuous function $W : R^n \rightarrow R_{\geq 0}$, and there is some $\delta \in \mathcal{N}$, so that

$$\sup_{x \in R^n} \min_{|u| \leq \sigma(|x|)} \nabla V(x) \cdot f(x, u) + W(x) \leq 0 .$$

In principle, we could then stabilize the system by using the steepest descent feedback law:

$$k(x) := \arg \min_{u \in \mathcal{U}_\delta} \nabla V(x) \cdot f(x, u) ,$$

where $\arg \min$ means that we pick any u at which the minimum is attained; we restricted \mathcal{U} to be assured that $\nabla V(x) \cdot f(x, u)$ attains a minimum [21].

3.2 Nonsmooth Control Lyapunov Functions

Let $V : R^n \rightarrow R$ be any continuous function (or even, just lower semicontinuous and with extended real values). A proximal subgradient of V at the point $x \in R^n$ is any vector $\zeta \in R^n$ such that, for some $\sigma > 0$ and some neighborhood (\mathcal{O}) of x ,

$$V(y) \geq V(x) + \zeta \cdot (y - x) - \sigma^2 |y - x|^2 \quad \forall y \in \mathcal{O} .$$

In other words, proximal subgradients are the possible gradients of supporting quadratics at the point x . The set of all proximal subgradients at x is denoted $\partial_P V(x)$ [20].

A continuous (but not necessarily differentiable) $V : R^n \rightarrow R$ is a control Lyapunov function if it is proper, positive definite, and infinitesimally decreasing on the following generalized sense: there exists a positive definite continuous function $W : R^n \rightarrow R_{\geq 0}$ and a $\delta \in \mathcal{N}$ so that

$$\sup_{x \in R^n} \max_{\zeta \in \partial_P V(x)} \min_{|u| \leq \sigma(|x|)} \zeta \cdot f(x, u) + W(x) \leq 0 .$$

An equivalent property is to ask that V is a viscosity supersolution of the corresponding Hamilton-Jacobi-Bellman equation.

In [23], it is shown that the system (10) is asymptotically controllable if and only if it admits a continuous clf. Hence we may propose the procedure to synthesize the stabilizing feedback law.

Consider the Iosida-Moreau inf-convolution of V with quadratic function:

$$V_\alpha(x) := \inf_{y \in R^n} \left[V(y) + \frac{1}{2\alpha^2} |y - x|^2 \right] ,$$

where the number $\alpha > 0$ is picked constant on appropriate region. One has that $V_\alpha(x) \nearrow V(x)$, uniformly on compact sets. Since V_α is locally Lipschitz, Rademacher’s Theorem insures that it is differentiable almost everywhere. The feedback k is then made equal to a pointwise minimizer k_α of the Hamiltonian, at the points of differentiability:

$$k_\alpha(x) := \arg \min_{u \in \mathcal{U}_0} \nabla V_\alpha(x) \cdot f(x, u) ,$$

where α and \mathcal{U}_0 are chosen constant on certain compact sets.

One must define, on appropriate compact sets

$$k_\alpha(x) := \arg \min_{u \in \mathcal{U}_0} \zeta_\alpha(x) \cdot f(x, u) ,$$

where $\zeta_\alpha(x)$ is carefully chosen: At point x of nondifferentiability, $\zeta_\alpha(x)$ is not a proximal subgradient of $V_\alpha(x)$, since $\partial_P V_\alpha(x)$ may well be empty. One uses, instead, the fact that $\zeta_\alpha(x)$ happens to be in $\partial_P V_\alpha(x)$ for some $x' \approx x$.

4 Conclusions and Discussions

In order to broaden the categories of nonlinear programming problems which may be solved by the Lagrange-type nonlinear programming neural network, the penalty parameters are taken as positive control variables. If a control law is found to stabilize the Lagrange-type neural network, it is a reasonable hypothesis that the Lagrange-type neural network may be used to solve much more categories of nonlinear programming problems. In this paper, we investigate respectively sufficient, necessary or sufficient and necessary conditions for

the existence of the constant, continuously differentiable or nonsmooth stabilizing feedbacks. Then control-Lyapunov function approach is used to synthesize the adjusting laws of penalty parameters, and the computing formulae are presented. Of course, there still remain many difficult issues on this research direction, for example, it is not easy to find an appropriate control-Lyapunov function; the nonsmooth feedback law is extremely sensitive to measurement errors [21], etc.

Acknowledgments. The author would like to express the gratitude to Prof. Hou for his valuable comments on the earlier version of this paper.

References

1. Huang, Y.C.: A Novel Method to Handle Inequality Constraints for Convex Programming Neural Network. *Neural Processing Letters* **16** (2002) 17-27.
2. Huang, Y.C.: Lagrange-Type Neural Networks for Nonlinear Programming Problems with Inequality Constraints. *Proceeding of the 44th Conference on Decision and Control* (2005) 1578-1883.
3. Hou, Z.G., Gupta, M.M., Nikiforuk, P.N., Tan, M., Cheng, L.: A Recurrent Network for Hierarchical Control of Interconnected Dynamic Systems. *IEEE Transactions on Neural Networks* (in press).
4. Hou, Z.G., Wu, C.P., Bao, Wang: A Neural Network for Hierarchical Optimization of Nonlinear Large-scale Systems. *International Journal of Systems Science* **29** (1998) 159-166.
5. Hou, Z.G.: A Hierarchical Optimization Neural Network for Large-scale Dynamic Systems. *Automatica* **37** (2001) 1931-1940.
6. Hou, Z.G., Song, K.Y., Gupta, M.M., Tan, M.: Neural Units with Higher Order Synaptic Operations for Robotic Image Processing Applications. *Soft Computing* **11** (2007) 221-228.
7. Bertsekas, D.P.: *Nonlinear Programming*. 2nd edn. Massachusetts Athena Scientific, Belmont (1999).
8. Bertsekas, D.P.: *Constrained Optimization and Lagrange Methods*. Academic Press, New York (1982).
9. Rockafellar, R.T.: Lagrange Multiplier and Optimality. *SIAM Review* **35** (1993) 183-238.
10. Kennedy, M.P., Chuond, L.O.: Neural Networks for Nonlinear Programming. *IEEE Transaction on Circuits and Systems* **35** (1988) 554-562.
11. Cichocki, A., Unbehauen, R.: *Neural Networks for Optimization and Singal Processing*. Wiley, Chichester (1993).
12. Zhang, S., Constantinides, A.G.: Lagrange Programming Neural Networks. *IEEE Transaction on on Neural Networks* **39** (1992) 441-452.
13. Lakshmikantham, V., Matrosov, V.M., Sivasundaram, S.: *Vector Lyapunov Functions and Stability Analysis of Nonlinear Systems*. Kluwer Academic Publisher, Dordrecht (1991).
14. LaSalle, J.P.: *The Stability of Dynamical Systems*. Springer, New York (1976).
15. Kaliora, G., Astolfi, A.: Stabilization with Positive and Quantized Control. *Proceeding of the 41st Conference on Decision and Control* (2002) 1892-1897.
16. Marquez, H.J.: *Nonlinear Control Systems-Analysis and Design*. John Eiley & Sons, New Jersey (2003).

17. Khalil, H.K.: *Nonlinear Systems*. 3rd edn. Prentice Hall, New Jersey (2002).
18. Brockett, R.W.: *Asymptotic Stability and Feedback Stabilization*. In: Brockett, R.W., Millman, R.S., Sussmann, H.J.(eds.): *Differential Geometric Control Theory*. Birkhäuser, Boston (1983) 181-191.
19. Clarke, F.H., Ledyev, Yu.S., Sontag, E.D., Subbotin, A.I.: *Asymptotic Controllability Implies Feedback Stabilization*. *IEEE Trans. Automat. Control* **42** (1997) 1394-1407.
20. Clarke, F.H., Ledyev, Yu.S., Stern, R.J., Wolenski, P.: *Nonsmooth Analysis and Control Theory*. Springer-Verlag, New York (1998).
21. Sontag, E.D.: *Stability and Stabilization: Discontinuities and the Effect of Disturbances*. In: Clarke, F.H., Stern, R.J.(eds): *Nonlinear Analysis, Differential Equations and Control*. Kluwer Academ Publishers, Dordrecht (1999).
22. Sontag, E.D.: *Mathematical Control Theory, Deterministic Finite Dimensional Systems*. 2nd edn. Springer-Verlag, New York (1998).
23. Sontag, E.D.: *A Lyapunov-like Characterization of Asymptotic Controllability*. *SIAM J. Control Optim.* **21** (1983) 462-471.

Support Vector Machines and Genetic Algorithms for Soft-Sensing Modeling

Sang Haifeng¹, Yuan Weiqi¹, Wang Fuli², and He Dakuo²

¹ School of Information Science and Engineering,
Shenyang University of Technology 663[#], Shenyang, 110023, China
sanghaif@yahoo.com.cn

² School of Information Science and Engineering,
Northeastern University 131[#], Shenyang, 110004, China

Abstract. Soft sensors have been widely used in industrial process control to improve the quality of product and assure safety in production. This paper introduces support vector machines (SVM) into soft-sensing modeling. Building the models, on one hand we want to have the best set of input variables, on the other hand we want to get the best possible performance of the SVM model. So the Genetic Algorithms is used to choose the input variables and select the parameters of SVM. Moreover, training the model on data coming a real experiment process—*Nosiheptide* fermentation process and evaluating the model performance on the same process. Results show that SVM model optimized by Genetic Algorithms provides a new and effective method for soft-sensing modeling and has promising application in industrial process applications.

1 Introduction

Most of the bioprocesses are strongly characterized by nonlinear dynamics, time-varying parameters, lack of reproducibility of the experimental results, and lack of cheap and reliable sensors capable of providing reliable on-line measurement of the main process variables and parameters, such as the concentration of biomass, substrates or products in fermentation process. These peculiarities impede the application of high performance automatic control and monitoring strategies necessary for the production efficiency optimization, the products quality improvement or the disturbances detection in process operation [1]. The development of appropriate estimation algorithms using well-known information on the process dynamics and ‘robust’ about the missing information becomes a premise for the successful decision of this problem. So soft sensors have been developed and applied for the bioprocesses [2].

In recent years, SVM, which is a statistical learning theory based machine learning formalism is gaining popularity due to its many attractive features and promising empirical performance. For further details on SVM we refer to the following references [3], [4].

In order to build a good model, we must do the following two specific subtasks. (i) Selection of the model input variables. We want to find out the combination of the original input variables which contribute most to the model. (ii) Selecting of the

appropriate model parameters, such as the regularization parameter C and kernel parameters in the SVM algorithm. These parameters play a key role in the SVM performance, but they are sometimes guessed by users. This paper proposed an input variables and parameters selection methodology for soft-sensing modeling based on the genetic algorithms in order to guarantee a good performance of the model. And this method is employed to construct a soft-sensing model in *Nosiheptide* fermentation process.

2 Genetic Algorithms for Model Optimization

The selection of the input variables and parameters are the key factor to build a model with good performance. There is an interdependent relationship between the input variables and parameter selection to the SVM performance. But they are always selected separately. Thus it can not obtain the best model. So we developed a method for hybrid optimization of the input variables and parameter selection in order to obtain their integrated effect. Because the selection of input variables and parameters is time consuming, so the conventional optimization method can never find the satisfying results. This paper proposes a faster and more effective methodology for the hybrid optimization based on the genetic algorithms [5].

2.1 Selection of the Input Variables

If we assume that there are n variables to be selected as the input. We can define a set, $X = \{x_1, x_2, \dots, x_n\}$, then we can represent this search space by a standard binary encoding where a “1” indicates the selection of the variable and “0” indicates that the variable is not selected. We define the fitness functions as follows

$$E_{RMS} = \sqrt{\frac{1}{M} \sum_{i=1}^M (y_i - \hat{y}_i)^2}, \quad (1)$$

where y_i is the real value, \hat{y}_i is the output of the model with the test data. M is the number of sample data. Then the selection of input variable is formulated as the optimization problem:

$$\begin{aligned} & \min E_{RMS}(X) \\ & s.t. \quad X = \{x_1, x_2, \dots, x_n\} \quad x_i \in \{0, 1\}, i = 1, \dots, n. \end{aligned} \quad (2)$$

2.2 Optimizing Kernel Parameters with Genetic Algorithms

Many of the characteristics of the SVM model are determined by the type of kernel function being used. The level of non-linearity is determined by the kernel function. Numerous possibilities of kernels exist and it is difficult to explain their individual characteristics. However, there are two main types of kernels, namely Local and Global kernels. In local kernels only the data that are close or in the proximity of each other have an influence on the kernel values. An example of a typical local kernel is

the RBF kernel: $K(x, x') = \exp(-\|x - x'\|^2 / 2\sigma^2)$. In contrast, a global kernel allows data points that are far away from each other to have an influence on the kernel values as well. The Polynomial kernel: $K(x, x') = ((x \cdot x') + 1)^d$, is a typical example of a global kernel. For more information on the characteristics of local and global kernels see [6].

So we know that the polynomial kernel has better extrapolation abilities, but the interpolation abilities are bad. On the other hand, the RBF kernel has good interpolation abilities, but fails to provide longer range extrapolation. Preferably one wants to combine the ‘good’ characteristics of two kernels. We use a convex combination of the two kernels Ω_{poly} and Ω_{RBF} .

$$\Omega_{mix} = p\Omega_{poly} + (1 - p)\Omega_{RBF}, \tag{3}$$

where p is the optimal mixing coefficient. Before building the model, the parameter, C, σ, d, p must be selected. We also select the parameter based on Genetic Algorithms. The fitness function is Eqs.1. Then we can get the optimization problem as flowing,

$$\begin{aligned} \min E_{RMS}(\Omega) \\ \text{s.t. } \Omega = \{C, \sigma, d, p\}, p \in [0, 1], C, \sigma, d > 0. \end{aligned} \tag{4}$$

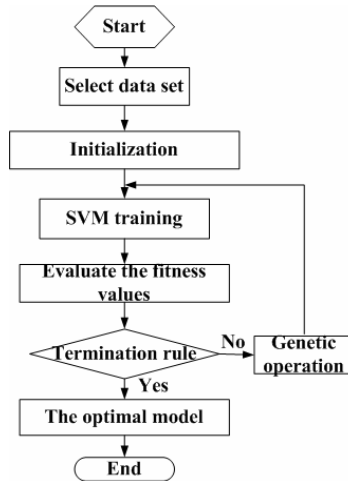


Fig. 1. Flow chart of input selection and parameters optimization

2.3 Hybrid Optimization

As before, there is an interdependent relationship between the input variables and parameter selection to the SVM performance. So they must be done as the same time. We represent input variable and parameter by a standard binary encoding together and

take the Eqs.1 as the fitness function. The iteration can stop when the fitness value of the best individual arrives at some value or there is no more obvious change for the best and average fitness values of the population. The flow chart of model based on GA is illustrated in Fig. 1.

3 Case Study: Biomass Concentration Soft-Sensing Modeling

3.1 Training Data and Pretreatment

Experimental data from *Nosiheptide* fermentation were used for the training and testing of the soft sensing model. The *Nosiheptide* production process is an aerobic fermentation in batch fermentor, which periods is about 96 hours. In *Nosiheptide* fermentation process, measurable variables on-line include physics and chemic parameters (see table 1). The biomass concentration can not measure on-line, so we take the biomass concentration as the output of the soft-sensing model. Prior to further processing, all of the data were normalized by conversion into the [0, 1] interval, using Eqs.5.

Table 1. Process variables

Variable	Units	Description	Min	Max
T	°C	Temperature	5	50
P	Pa	Pressure	0	35
DO	% sat.	Dissolved oxygen tension	0	100
pH	pH	pH value	3	10
φ_{O_2}	%	Off-gas oxygen concentration	12	22
φ_{CO_2}	%	Off-gas carbon dioxide concentration	0	7
OUR	mol·min ⁻¹	Oxygen uptake rate	0	5
CER	mol·min ⁻¹	Carbon dioxide production rate	0	5
R	r·min ⁻¹	Agitator rotate speed	0	375
Na	m ³ ·h ⁻¹	Aeration	0	0.42

$$x_{N,i} = \frac{x_i - x_{\min,i}}{x_{\max,i} - x_{\min,i}}, \quad (5)$$

where: x_i is the value of the I th process variable, $x_{N,i}$ is the normalized value of the I th process variable, $x_{\max,i}$ is the high-range value of the I th process variable, $x_{\min,i}$ is the low-range value of the I th process variable.

3.2 Result

Firstly, we select the input variables and parameters based on the optimization method as before. The size of initial population is 30, and the genetic operator is roulette wheel selection, adaptive crossover and Gauss mutation [7]. We take five batch data as the training data, one batch data as the validating data, and another batch as the

testing data. But we know that the growth of thalli is an accumulative effect of multi factor. And it can not be decided by the conditions of some times. That's to say, it is the outcome of a period of time. In order to improve the prediction capabilities of the model, the state variables at times $t-1, t-2$ were provided as inputs in addition to the states at time t . The variable at time $t-2$ is comparatively less influence than at time t and $t-1$. We can weight the variables at times $t, t-1, t-2$ by weighting factors λ_i . This leads to the input p_i at time t :

$$p_i = \lambda_1 x_{i,t} + \lambda_2 x_{i,t-1} + \lambda_3 x_{i,t-2}, \tag{6}$$

where $\sum_{i=1}^3 \lambda_i = 1, \lambda_1 > \lambda_2 > \lambda_3$, $x_{i,t}, x_{i,t-1}, x_{i,t-2}$ denote sampling value of i th variable at time $t, t-1, t-2$, respectively. We take $\lambda_1 = 0.7, \lambda_2 = 0.2$ and $\lambda_3 = 0.1$.

After optimization, we select the flowing five variables as the input: agitator rotate speed (R), dissolved oxygen (DO), CO₂ concentration, (φ_{CO_2}) O₂ concentration (φ_{O_2}) and pH value (pH). The parameters $\{C, \sigma, d, p\}$ in SVM are $\{18.4, 0.24, 1, 0.8\}$. At last, we use the testing data to test the soft-sensing model, the result is shown in Fig. 2 and mean square error between the output of the model and real value is 0.1324.

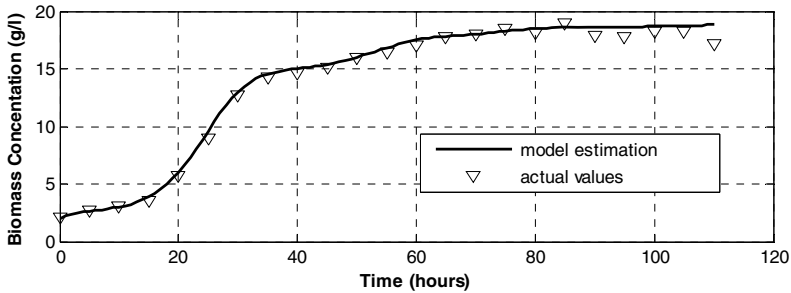


Fig. 2. Estimated and measured biomass concentration values VS time

To explain the performances of the SVM-based soft-sensing model furthermore, we also compare the results with the results with the NN-based estimation methods. The result is presented in Tab. 2.

Table 2. Comparison of different soft-sensing methods

Method	Training time (s)	Training error	Testing error
SVM	3.456	0.1035	0.1324
Neural Network	107	0.1022	0.5784

The results in Tab. 2 show that the NN model can achieve the high training error, but the generalization performance is bad, and cost much more time than SVM method.

4 Conclusions

Within this work a soft-sensing model for estimate of biomass concentration in *Nosi-heptide* fermentation was designed and examined. The model is based on the SVM approach. Because fermentation process is characteristic with nonlinear and time-varying, and there are many process variables. So the selection of input variables of the model becomes a key problem. It impacts not only the complexity but also the generalization performance of the model. Otherwise, the parameters in SVM play a key role in the SVM performance, and there is not a definite solution when SVM is constructed. So this paper introduces a hybrid optimization method of input variables selection and parameters selection based on GAs. It simplifies the complexity of the model and improves the SVM performance. Effective results indicate that SVM modeling method based on GAs provides a new tool for soft-sensing and has promising application in industrial process applications.

References

1. James, S., Budman, H.: On-Line Estimation in Bioreactors: A Review. *Reviews in Chemical Engineering* **56** (2000) 311-340
2. Adilson, J.D.A., Rubens, M.F.: Soft Sensors Development for On-Line Bioreactor State Estimation. **24** (2000) 1099-1103
3. Guozheng, L., Meng, W.: *An Introduction to Support Machines and Other Kernel-based Learning Methods*. Publishing House of Electronics Industry, Beijing (2004)
4. Suykens, J.A.K.: *Least Squares Support Vector Machines*. World Scientific Publishing, Belgium (2004)
5. Srinivas, M., Patnaik, L.M.: Genetic Algorithms: A Survey. *Computer* **27** (1994) 17-26
6. Smola, A.J.: *Learning with kernels*. Ph.D. Thesis, TU Berlin (1998)
7. Sang, H.F., Wang, F.L.: Detection of Element Content in Coal by Pulsed Neutron Method Based on an Optimized Back-propagation Neural Network. *Nucl. Instr. And Meth. B* **239** (2005) 202-208

Incremental Nonlinear Proximal Support Vector Machine

Qiuge Liu^{1,2}, Qing He^{1,2}, and Zhongzhi Shi^{1,2}

¹ The Key Laboratory of Intelligent Information Processing,
Department of Intelligence Software, Institute of Computing Technology, Chinese Academy
of Sciences, Beijing, 100080, China

² Graduate College of University of Chinese Academy of Sciences,
Beijing 100039, China
{liuqq, heq, shizz}@ics.ict.ac.cn

Abstract. Proximal SVM (PSVM), which is a variation of standard SVM, leads to an extremely faster and simpler algorithm for generating a linear or nonlinear classifier than classical SVM. An efficient incremental method for linear PSVM classifier has been introduced, but it can't apply to nonlinear PSVM and incremental technique is the base of online learning and large data set training. In this paper we focus on the online learning problem. We develop an incremental learning method for a new nonlinear PSVM classifier, utilizing which we can realize online learning of nonlinear PSVM classifier efficiently. Mathematical analysis and experimental results indicate that these methods can reduce computation time greatly while still hold similar accuracy.

1 Introduction

Recently some variations of standard support vector machines [7] have been presented, e.g. PSVM [1], RSVM [5], Least squares SVM [6], etc. These approaches make a quadratic problem that requires considerably longer computational time become merely requiring the solution of a single system of linear equations and clearly superior to classical SVM at speed while still hold comparable test correctness. In recent years tremendous growth in the amount of data gathered and the need of online learning have changed the focus of SVM classifier algorithms to not only provide accurate results, but to also enable incremental learning. An efficient incremental method for linear PSVM was introduced in [2], however this method can't apply to nonlinear PSVM classifier as the form of the solution of nonlinear PSVM is different from that of linear PSVM. To solve large data set learning problem [5] introduced reduced kernel techniques (RSVM), but it isn't suitable for online learning in which we don't know the entire data set at the beginning. Reference [4] introduces an incremental method for nonlinear PSVM [1], but it re-compute the classifier from beginning each time when new examples are added in. To solve online learning problem efficiently, we introduce a new nonlinear PSVM classifier which leads to shorter computing time and similar accuracy first, then based on this new model a new incremental learning method, utilizing the calculation formula for block

matrix’s inversion, is developed. Mathematical analysis and experimental results demonstrate that this incremental learning method fully utilizes the historical training results and leads to shorter training time and same correctness compared with learning in standard way. Similar work to regression least squares SVM can be found in [8].

We briefly summarize the contents of this paper now. In Section 2 we review the 2-category PSVM [1] and the incremental linear PSVM proposed in [2]. In Section 3 we introduce a new nonlinear PSVM based on which our incremental method is developed. In Section 4 we introduce a new incremental method and give a detailed complexity analysis. Some numerical testing results can be found in section 5 before the conclusion and future works in section 6.

2 Proximal Support Vector Machine (PSVM)

Some notations will be used in this paper: all vectors are column vectors, the 2-norm of the vector x is denoted by $\|x\| = \sqrt{x^T \cdot x}$, let matrix $A[m \times n]$ be the training points in the n -dimensional space R^n . The diagonal matrix $D[m \times n]$ of ± 1 contains the classes $+1$ and -1 of m training points. Let e be the column vector of 1, w, r be the coefficients and the scalar of the hyperplane. Let y be the slack variable and constant $v \geq 0$ is used to tune errors and margin size. The identity matrix is denoted by I .

The PSVM [1] proposed by Fung and Mangasarian try to find the “proximal” ($x^T w - r = \pm 1$ for class ± 1) planes, around which the points of each class are clustered and which are pushed as far apart as possible. The key idea in PSVM is replacing the inequality constraints in standard SVM [7] by equalities, and the solution with linear kernel is given by the following quadratic program:

$$\begin{aligned} \min_{(w,r,y) \in R^{m+1+m}} \quad & v \frac{1}{2} \|y\|^2 + \frac{1}{2} (w^T w + r^2), \\ \text{s.t.} \quad & D(Aw - er) + y = e. \end{aligned} \tag{1}$$

These modifications, even though very simple, change the nature of optimization problem significantly. In fact it turns out that we can write an explicit exact solution to the problem in terms of the problem data as we show below, whereas it is impossible to do that in standard SVM formulations. The Karush-Kuhn-Tucker optimality condition of (1) will give the linear equation system (2) of $(n+1)$ variables (w, r) :

$$[w, r]^T = (I/v + E^T E)^{-1} E^T D e, \quad E = [A, -e]. \tag{2}$$

Note that all we need to store in memory is the $m \times (n+1)$ training data matrix E , the $(n+1) \times (n+1)$ matrix $E^T E$ and the $(n+1) \times 1$ vector $E^T D e$. So if the dimension of the input space is small enough, PSVM can even classify millions data points. However the algorithm is limited by the storage capacity of the $m \times (n+1)$ training

data matrix E . In order to deal with very large (at least one billion points) data sets, as proposed in [2] we can split the training dataset E into blocks of lines E_i, D_i and compute $E'E$ and $E'De$ from these blocks:

$$E'E = \sum E_i'E_i, E'De = \sum E_i'D_i e . \tag{3}$$

For each step, we only need to load the $(blocksize) \times (n+1)$ matrix E_i and the $(blocksize) \times 1$ vector $D_i e$ for computing $E'E$ and $E'De$. Between two incremental steps, we need to store in memory $(n+1) \times (n+1)$ and $(n+1) \times 1$ matrices although the order of the dataset is one billion data points.

The nonlinear proximal classifier with a kernel $K(A, A')$ is given by the following quadric problem (NPSVM):

$$\begin{aligned} \min_{(u,r,y) \in R^{m+1+m}} \quad & v \frac{1}{2} \|y\|^2 + \frac{1}{2}(u'u + r^2), \\ \text{s.t.} \quad & D(K(A, A')Du - er) + y = e. \end{aligned} \tag{4}$$

Using the shorthand notation $K = K(A, A')$, through the KKT necessary and sufficient optimality conditions for our equality constrained problem (5) we can get the following solution:

$$Ds = \left(\frac{I}{v} + KK' + ee' \right)^{-1} De . \tag{5}$$

Here, $s \in R^m$ is the Lagrange multiplier associated with the equality constraint of (5). As the kernel matrix K is a square $m \times m$ matrix, the above incremental method isn't applicable to nonlinear PSVM. To make nonlinear classifier learning of large data set solvable, [5] proposed a reduced kernel techniques (RSVM), but RSVM isn't applicable when it comes to online learning where examples are added in different batches. With the focus on online learning problem next we introduce a new nonlinear PSVM model based on which a new incremental training method will be designed.

3 New Nonlinear PSVM (NNPSVM)

We obtain the new nonlinear PSVM model by making a simple change to the linear formulation (1) as follows:

$$\begin{aligned} \min_{(w,r,y) \in R^{m+1+m}} \quad & v \frac{1}{2} \|y\|^2 + \frac{1}{2}(w'w + r^2), \\ \text{s.t.} \quad & D(\phi(A)w - er) + y = e. \end{aligned} \tag{6}$$

Here $\phi(x)$ is a nonlinear transformation function, which maps a vector x in the real space R^n to a higher dimensional feature space $R^{\tilde{n}}$, similarly $\phi(A)$ maps $R^{m \times n}$ into

$R^{m \times \bar{n}}$, and there exist a kernel $K(A, A')$ satisfying $K(A, A') = \phi(A)\phi(A)'$. Apparently we can also get (6) from (8) by minimizing $w'w$ instead of $u'u$.

Using KKT necessary and sufficient optimality conditions for our equality constrained problem (7) we can get the following expressions:

$$w = \phi(A)'Du, r = -e'Du, vy = u, D(\phi(A)w - er) + y - e = 0. \tag{8}$$

Here, $u \in R^m$ is the Lagrange multiplier associated with the equality constraint of (7). Substituting the first three expressions in the last equality of (8) allows us to obtain an explicit expression for Du as follows:

$$Du = (I/v + \phi(A)\phi(A)' + ee')^{-1}De = (I/v + K + ee')^{-1}De. \tag{9}$$

The corresponding nonlinear classifier is then:

$$f(x) = \text{sgn}(K(x', A')Du - r). \tag{10}$$

Compared with (6) the solution of NNPSVM (9) eliminates two $m \times m$ matrix's product, furthermore based on NNPSVM we can develop an incremental training method. The effectiveness of this model is demonstrated in Section 5.

4 Incremental Nonlinear PSVM (INPSVM)

Assume that the current classifier is based on an input data set $A_1 \in R^{m_1 \times n}$ and a corresponding diagonal matrix $D_1 \in R^{m_1 \times m_1}$ of ± 1 and have the corresponding inversion $A_{11}^{-1} = (I/v + K(A_1, A_1) + ee')^{-1}$ stored in memory, now a "new" set of data points represented by a new matrix $A_2 \in R^{m_2 \times n}$ and a corresponding diagonal matrix $D_2 \in R^{m_2 \times m_2}$ of ± 1 needs to be added to the characterization of our new classifier. So the whole data set comes to be $A_t = [A_1', A_2']'$, according to formulation (9) to get the solution of Du we need to figure out the inversion of the following matrix:

$$A_t = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \tag{11}$$

where $A_{12} = (K(A_1, A_2) + ee')$, $A_{22} = (I/V + K(A_2, A_2) + ee')$ and $A_{21} = A_2'$.

Making use of the formula for computing block matrix's inversion we can compute the $(m_1 + m_2) \times (m_1 + m_2)$ matrix A_t 's inversion incrementally:

$$A_t^{-1} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}^{-1} = \begin{bmatrix} A_{11}^{-1} + X & Y \\ Y' & T \end{bmatrix}, \tag{12}$$

where T , X , and Y has the following expressions:

$$T=(A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1}, Y = -A_{11}^{-1}A_{12}T, X = -YA_{12}'A_{11}^{-1}. \tag{13}$$

As is shown in (13), to obtain the inversion of A_{11} we need to figure out A_{12} , A_{22} , T , X and Y which is a $m_1 \times m_2$ matrix, a $m_2 \times m_2$ matrix, a $m_2 \times m_2$ matrix's inversion, a $m_1 \times m_1$ matrix and a $m_1 \times m_2$ matrix respectively. Thus approximately we need $O(m_2^3) + 2m_1m_2^2 + 2m_1^2m_2$ operations to acquire A_{11}^{-1} through incremental method in contrast to $O((m_1 + m_2)^3)$ operations if we compute the whole data set from start. We turn now to our numerical experiments.

5 Experiments

All our computations were performed on a machine which utilizes a 731 MHZ Pentium III and 512 Megabytes of memory. Here we chose four data sets from the UCI Machine Learning Repository.

Table 1. INPSVM and NNPSVM recall and running time with some tuned values of ν

	Data Set $m \times n$	Ionosphere 351×34	Bupa Liver 345×6	Tic-Tac-Toe 958×9
INPSVM	Recall	99.72%	83.48%	100.00%
	Time(Sec.)	0.21	0.21	3.75
NNPSVM	Recall	99.72%	83.48%	100.00%
	Time(Sec.)	0.33	0.33	5.03

Table 2. NPSVM and NNPSVM average recall, ten-fold testing correctness and training time

	Data Set $m \times n$	Ionosphere 351×34	Bupa Liver 345×6	Tic-Tac-Toe 958×9
NPSVM	Recall	99.43%	80.23%	100%
	Test	93.17%	66.87%	99.06%
	Time(Sec.)	0.16	0.16	2.32
NNPSVM	Recall	99.94%	84.73%	100%
	Test	94.80%	67.27%	99.06%
	Time(Sec.)	0.14	0.13	1.56

To simulate the situation in online learning firstly split each data set into two parts and train the first part using NNPSVM, secondly add the second part of the data set into the classifier. There are two methods to do the second job (INPSVM and NNPSVM) distinguished by whether or not using the incremental method. The test results are recorded in Table 1, and it can be seen that INPSVM reduces the computation time evidently and has no influence on accuracy (because it gives the same solution as NNPSVM). It can be further expected easily that if the number of parts decomposed was more than 2, the reduction would be more distinct. We also

compared NNPSVM with standard nonlinear PSVM (5) (NPSVM) both with $\nu = 10^3$ in Table 2, it can be seen that they have similar ten-fold recall and test correctness but the average running time of NNPSVM is shorter.

6 Conclusions

In this paper we proposed an incremental learning method based on a new nonlinear PSVM model, utilizing which we can perform online learning of a new nonlinear PSVM classifier efficiently. Test results demonstrate that the computation time of NNPSVM is shorter than NPSVM and the accuracy of these two classifiers are similar, and that INPSVM reduce training time evidently while still hold same correctness as that of learning in standard way.

Acknowledgements

This work is supported by the National Science Foundation of China (No. 60435010, 90604017, 60675010), the 863 Project (No.2006AA01Z128), National Basic Research Priorities Programme (No. 2003CB317004) and the Nature Science Foundation of Beijing (No. 4052025).

References

1. Fung, G., Mangasarian, O.: Proximal Support Vector Machine Classifiers. Proceedings KDD-2001, Knowledge Discovery and Data Mining, San Francisco, CA (2001)
2. Fung, G., Mangasarian, O.: Incremental Support Vector Machine Classification. Data Mining Institute Technical Report 01-08, Computer Sciences Department, University of Wisconsin (2001)
3. Fung, G., Mangasarian, O.: Multicategory Proximal Support Vector Machine Classifiers. *Machine Learning* **59** (2005) 77-97
4. Wu, J., Zhou, J. G., Yan, P.L.: Incremental Proximal Support Vector Classifier for Multi-Class Classification. Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai (2004)
5. Lee, Y.J., Mangasarian, O.: RSVM: Reduced Support Vector Machines. Proceeding of the First SLAM International Conference on Data Mining (2001)
6. Suykens, J., Vandewalle, J.: Least Squares Support Vector Machine Classifiers. *Neural Processing Letters* **9** (1999) 293-300
7. Vapnik, V.: *The Nature of Statistical Learning Theory*. Springer-Verlag, New York (1995)
8. Zhang, H.R., Wang, X.D.: Incremental and Online Learning Algorithm for Regression Least Squares Support Vector Machine. *Chinese Journal of Computers* **29** (3) (2006) 400-406

Machinery Fault Diagnosis Using Least Squares Support Vector Machine

Lingling Zhao and Kuihe Yang

College of Information, Hebei University of Science and Technology
Shijiazhuang 050054, China
zll@hebust.edu.cn

Abstract. In order to enhance fault diagnosis precision, an improved fault diagnosis model based on least squares support vector machine (LSSVM) is presented. In the model, the wavelet packet analysis and LSSVM are combined effectively. The power spectrum of fault signals are decomposed by wavelet packet analysis, which predigests choosing method of fault eigenvectors. And then the LSSVM is adopted to realize fault diagnosis. The non-sensitive loss function is replaced by quadratic loss function and the inequality constraints are replaced by equality constraints. Consequently, quadratic programming problem is simplified as the problem of solving linear equation groups, and the SVM algorithm is realized by least squares method. It is presented to choose parameter of kernel function in definite range by dynamic way, which enhances preciseness rate of recognition. The simulation results show the model has strong non-linear solution and anti-jamming ability, and it can effectively distinguish fault type.

1 Introduction

The traditional neural networks method obtains many harvests in application research of fault diagnosis, but it has a lot of questions in network structure selecting, network training and enhancing network spread ability. The support vector machine (SVM) is a new machine study method which was established by Vapnik in base of statistical learning theory (SLT) [1][2]. The SVM stresses to study statistical learning rules under small sample. Via structural risk minimization principle to enhance extensive ability, the SVM preferably solves many practical problems, such as small sample, non-linear, high dimension number and local minimum points.

The least squares support vector machine (LSSVM) is an improved algorithm based on SVM [3]. The LSSVM is a kind of SVM under quadratic loss function. In LSSVM, the non-sensitive loss function is replaced by quadratic loss function and the inequality constraints are replaced by equality constraints. Via constructing loss function, the quadratic programming problem is changed as solving linear equation groups problem, which simplifies the complexity of calculation. In the paper, the wavelet packet analysis and least squares support vector machine (LSSVM) are combined effectively. The LSSVM is adopted to diagnose the diagnose machinery facility faults, which gets good diagnosis effect.

2 Eigenvectors Choosing

Fourier analysis can be used to analyze frequency bands energy, and received spectrum structure eigenvectors of different faults have obtained successful application. However, Fourier analysis only considers sine wave signals. Practical diagnosis signals usually contain non-sine wave signals. Strictly speaking, these signals cannot be described using sine signals as bases. If we use sine signals as bases to describe them, the energy will not be complete. Nevertheless, using wavelet to analyze signals can describe non-steady components. Especially using wavelet packet analysis technique can decompose signals into discrete accurate frequency bands. Doing energy calculation in these frequency bands to form eigenvectors is more reasonable.

As Fourier frequency spectrum analysis technique, the theory base of wavelet frequency charts analysis technique is Parseval energy integral equation.

The energy in $f(x)$ time field is expressed as follows.

$$\|f\|^2 = \int_{-\infty}^{+\infty} |f(x)|^2 dx \quad (1)$$

We can get wavelet transform of $f(x)$ as follows.

$$C_{j,k} = W(2^j, 2^j k) = 2^{-\frac{j}{2}} \int_R \overline{\psi}(2^{-j}x - k) f(x) dx \quad (2)$$

By relating the two items using Parseval's identical equation, we have

$$\int_{-\infty}^{+\infty} |f(x)|^2 dx = \int |C_{j,k}|^2 \quad (3)$$

From the above equation we can know that wavelet transform coefficient $C_{j,k}$ has the dimension of energy. Therefore it can be used for energy analysis.

In machinery equipment fault diagnosis technique, the study of rotating machinery fault diagnosis is most embedded and ideal, and its application is also most successful. Using excessive distinguishing analysis to characterize distilling of power spectrum, we can distill eigenvectors expediently and effectively. By means of wavelet packet analysis technique, we can do energy analysis to constantly appearing faults of

Table 1. Fault causes and sign corresponding

Fault	0	1	2	3	4	5	6	7	8
frequency bands	0.01~0.39f	0.40~0.49f	0.50f	0.51~0.99f	f	2f	3~5f	odd number f	> 5f
Fault 1	0.00	0.00	0.00	0.00	0.90	0.05	0.05	0.00	0.00
Fault 2	0.00	0.30	0.10	0.60	0.00	0.00	0.00	0.00	0.00
Fault 3	0.00	0.00	0.00	0.00	0.40	0.50	0.10	0.00	0.00
Fault 4	0.10	0.80	0.00	0.10	0.00	0.00	0.00	0.00	0.00
Fault 5	0.10	0.10	0.10	0.10	0.20	0.10	0.10	0.10	0.10
Fault 6	0.00	0.00	0.00	0.00	0.20	0.15	0.40	0.00	0.25

rotating machinery [7]. Via a lot of experiments, the rotating machinery fault causes and sign corresponding table can be built. Using rotating machinery ordinary 6 faults, such as imbalance, non-middle, oil film swirling and so on, as output of neural networks, and using 9 frequency bands different spectrum apex energy values in vibration signal spectrum, the training sample set of rotating machinery faults can be formed, as table 1 shows.

3 LSSVM Diagnosis Model

3.1 SVM Principle

The SVM is developed from hyperplane classifier under linear separable case [4]. Suppose having sample x_i and its sorts y_i , expressed as $\{(x_i, y_i)\}, x \in R^d, y_i \in \{1, -1\}, i = 1, \dots, N$. The d is dimension number of input space. For standard SVM, its classifying margin is $2 / \|w\|$. Making the classifying margin to maximum is equal to making the $\|w\|^2$ to minimum. Therefore, the optimization problem of making the classifying margin to maximum can be expressed the follow quadratic programming problem:

$$\min_{w, b, \xi} J(w, \xi) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^N \xi_i \tag{4}$$

$$\text{Subject to } y_i [w^T \Phi(x_i) + b] \geq 1 - \xi_i, i = 1, \dots, N \tag{5}$$

$$\xi_i \geq 0, i = 1, \dots, N \tag{6}$$

where $\xi_i \geq 0 (i = 1, \dots, N)$ is a slack variable, which ensures classification validity under linear non-separable case, and parameter C is a positive real constant which determines penalties to estimation errors, a larger C corresponding to assigning a higher penalty to errors. The function Φ is a non-linear mapping function, by which the non-linear problem can be mapped as linear problem of a high dimension space. In the transforming space, we can get optimal hyperplane.

Suppose non-linear function $\Phi : R^d \rightarrow H$ maps the sample of input space to the feature space H . When we construct optimal hyperplane in feature space H , the training algorithm only uses inner product of the space, namely $\Phi(x_i) \cdot \Phi(x_j)$, without individual $\Phi(x_i)$ appearing. Therefore, if a function K can be found to meet condition $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$, the inner product of high dimension space can be finished by the function in original space, and we needn't know the form of mapping Φ . Statistical learning theory points out that due to Hilbert-Schmidt principle,

function $K(x_i, x_j)$ will be corresponding inner product of a transforming space as long as it satisfies Mercer’s condition [5].

The SVM algorithm can realize right classification to sample by solve above quadratic programming problem.

3.2 LSSVM Classifying Algorithm

The LSSVM is an improved algorithm based on SVM. For LSSVM, the non-sensitive loss function is replaced by quadratic loss function and the inequality constraints are replaced by equality constraints. The optimization problem has been modified as follows:

$$\min_{w,b,\xi} J_{LS}(w, \xi) = \frac{1}{2} \|w\|^2 + \frac{1}{2} \gamma \sum_{i=1}^N \xi_i^2 \tag{7}$$

$$\begin{aligned} \text{Subject to } & y_i [w^T \Phi(x_i) + b] = 1 - \xi_i, \\ & i = 1, \dots, N \end{aligned} \tag{8}$$

where parameter γ is similar parameter C of SVM, which is used to control function $J_{LS}(w, \xi)$. In order to solve the optimization problem, the Lagrangian function is introduced as follows:

$$L_{LS}(w, b, \xi, \alpha) = J_{LS}(w, \xi) - \sum_{i=1}^N \alpha_i \{y_i [w^T \Phi(x_i) + b] - 1 + \xi_i\} \tag{9}$$

where α_i are Lagrangian multipliers that can either be positive or negative. In saddle points, differential coefficients for w , b , ξ and α_i are requested and let them equal to zero, and then the equation as follows can be obtained:

$$\frac{dL_{LS}}{dw} = 0, \quad w = \sum_{i=1}^N \alpha_i y_i \Phi(x_i) \tag{10}$$

$$\frac{dL_{LS}}{db} = 0, \quad \sum_{i=1}^N \alpha_i y_i = 0 \tag{11}$$

$$\frac{dL_{LS}}{d\xi} = 0, \quad \alpha_i = \gamma \xi_i \tag{12}$$

$$\frac{dL_{LS}}{d\alpha_i} = 0, \quad y_i [w^T \Phi(x_i) + b] - 1 + \xi_i = 0 \tag{13}$$

These conditions can be written as a linear system:

$$\begin{bmatrix} 0 & Y^T \\ Y & ZZ^T + \gamma^{-1}I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ \vec{1} \end{bmatrix} \tag{14}$$

Where $Z = [\Phi(x_1)^T y_1, \dots, \Phi(x_N)^T y_N]^T$, $Y = [y_1, \dots, y_N]^T$, $\vec{1} = [1, \dots, 1]^T$, $\xi = [\xi_1, \dots, \xi_N]^T$, $\alpha = [\alpha_1, \dots, \alpha_N]^T$, and $I \in R^{(N \times N)}$ is identity matrix.

We adopt RBF kernel function in common use as follows:

$$K(x_i, x_j) = \exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right) \tag{15}$$

Equation 14 can be solved by least squares method. In LSSVM, quadratic programming problem is changed as the problem of solving linear equation groups, which simplifies the complexity of calculation. After the optimization problem is solved, we can have the classifier of SVM as follows:

$$y(x) = \text{sign}\left[\sum_{i=1}^N \alpha_i y_i \Phi(x, x_i) + b\right] \tag{16}$$

3.3 The Improvement Algorithm of LSSVM

In LSSVM, regularization parameter γ and standardization parameter σ of RBF function are selected as constants commonly according to experience, but for different sample set, optimal parameter values is metabolic, which affects diagnosis precise rate of faults. By examination, we find that the value of parameter γ don't affect the result obviously, so we define $\gamma=10$. On the other hand, with different σ value, the results vary obviously.

Recognition error rate $y=f(\sigma)$ which is affected by parameter σ can not be expressed by simple function. The experiment results with parameter varying are showed as Fig.1.

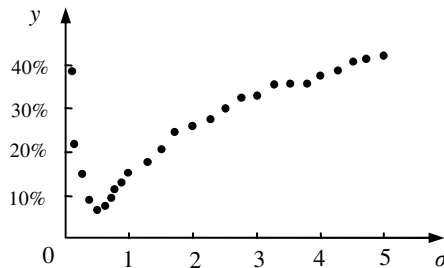


Fig. 1. Experiment results with parameter varying

However, the experiment results shows that although recognition error rate may appear small shake with parameter σ in a small section, the diagnosis error rate can be approximatively seen as a down single peak function in a big section $[a, b]$. In section $[a, b]$, function has exclusive minimum in t^* value and its neighboring section. If we take freewill two points a_1 and b_1 in the section, $a_1 < b_1$, and calculate function values $f(a_1)$ and $f(b_1)$, two instances as follows will appear.

- (1) $f(a_1) < f(b_1)$, then $t \in [a, b_1]$
- (2) $f(a_1) \geq f(b_1)$, then $t \in [a_1, b]$

This shows that we can shorten section $[a, b]$ to section $[a, b_1]$ or $[a_1, b]$ which still contain minimum as long as we take two different points belonging to $[a, b]$ and compare their function values. If we want to continue shortening section $[a, b_1]$ or $[a_1, b]$, one point value belonging to above section should be taken to calculate its function value which is use to compare with the values of $f(a_1)$ or $f(b_1)$. As long as the shortened section contains t^* value and its neighboring section, the section is smaller, the value of σ is nearer to the function minimum. However, when value of σ is in the range of t^* value neighboring section, recognition error rate y appears small shake. Thereby, section searching should stop at fist shake, and take σ value before shake as σ value. For briefness, we take $a=0.1$, $b=5$, and take a_1 and b_1 in three equality part points of section $[a, b]$.

4 Simulation Results

In order to check diagnosis success rates of LSSVM model, two diagnosis models are used to diagnose simulation faults. Model 1 adopts probabilistic neural networks (PNN) model [8][9]. PNN network is a back-propagation neural network developed form radial basic function network. Its theory base is Bayesian least risk rule of Bayesian decision making theory. PNN networks can be used to solve classifying question. When training sample is many enough, PNN networks is convergent to a Bayesian classifying machine, which has good spread capability.

Model 2 adopts LSSVM fault diagnosis model, which uses one-against-one method to realize multiclass classification. First the two models are trained by training sample set, and then we diagnosis the simulation faults using trained models.

Suppose D_1 is simulation sample data matrix before entering noises, and D_2 is simulation sample data matrix after entering noises. The containing noises input sample data required by simulation can gain by equation below.

$$D_2(i, j) = D_1(i, j) \times (1 + \alpha \times \text{rands}(1)) \quad (17)$$

where the α is noise control coefficient, $\alpha = 0, 0.2, 0.5$ respectively, and $\text{rands}(1)$ is a random function which can produce a number between -1 and 1.

Use above equation to produce 80 groups measure parameter containing noises to every fault, altogether 480 groups sample, 300 groups sample of which are used as training set and 180 groups sample of which are used test set. In not doing any pre-treatment, they are used to diagnose faults by PNN and LSSVM models. The average diagnosis results are showed in table 2.

Table 2. The diagnosis results

Noises control coefficient	$\alpha = 0.0$		$\alpha = 0.2$		$\alpha = 0.5$	
Diagnosis model	PNN	LSSVM	PNN	LSSVM	PNN	LSSVM
Diagnosis preciseness rate	100%	100%	98.3%	98.9%	92.8%	96.1%

Table 2 shows the diagnosis results of PNN and LSSVM models. The diagnosis results show that the diagnosis success rates are influenced by noise control coefficient α . When there are not any noises in sample data, such as $\alpha = 0.0$, the diagnosis success rates of both PNN and LSSVM are as high as 100%. When noises are comparatively small, such as $\alpha = 0.2$, the diagnosis success rate of LSSVM is 98.3%, and the diagnosis success rate of PNN is 98.9%. When noises are comparatively big, such as $\alpha = 0.5$, the LSSVM keeps as high as 96.1% diagnosis success rate, but the PNN only arrive at 92.8% diagnosis success rate. From above diagnosis success rates, although diagnosis success rates of both two models appear to decline along with increasing the sample noises, the declining speed of LSSVM model is lower than PNN model obviously, and the LSSVM shows robustness.

5 Conclusion

The Simulation results show that the fault diagnosis ability of LSSVM is much bigger than PNN. This reveals LSSVM has rather strongly robust diagnosis and it can be used for pattern classifying. Comparing with PNN, the primary merit of SVM is that it aims at small sample, and it can gain optimal answer according to information in existence instead of sample number going to infinity. The LSSVM changes classification question into quadratic programming problem and can gain optimal answer entirely in theory. For LSSVM, the non-sensitive loss function is replaced by quadratic loss function and the inequality constraints are replaced by equality constraints. Consequently, quadratic programming problem is simplified as the problem of solving linear equation groups, and the SVM algorithm is realized by least squares method, which predigests the complexity of calculation. The LSSVM model has strong non-linear solution and anti-jamming ability, and can effectively diagnose machinery facility faults.

Acknowledgment

This work is supported by China Postdoctoral Science Foundation (2005038515).

References

1. Cortes, C., Vapnik, V.: Support-Vector Network. *Machine Learning* **20** (1995) 273–297
2. Vapnik, V., Golowich, S., Smola, A.: Support Vector Method for Function Approximation, Regression Estimation, and Signal Processing. *Advances in Neural Information Processing Systems* **9** (1996) 281–287

3. Hsu, C. W., Lin, C. J.: A Comparison of Methods for Multiclass Support Vector Support Vector Machines. *IEEE Trans on Neural Networks* **13** (2002) 415-425
4. Guo, G., Li, S. Z., Chan, K. L.: Support Vector Machines for Face Recognition. *Image and Vision Computing* **19** (2001) 631-638
5. Lendasse, A., Simon, G., Wertz, V.: Fast Bootstrap for Least-Square Support Vector Machines. *Proceedings of European Symposium on Artificial Neural Networks, Bruges Belgium* (2004) 525-530.
6. Engel, Y., Mannor, S., Meir, R.: The Kernel Recursive Least-Squares Algorithm. *IEEE Trans on Signal Processing* **52** (2004) 2275-2285
7. Yu, H. J., Chen, C. Z., Zhang, S.: *Intelligent Diagnosis Based on Neural Networks*. Beijing, China, Metallurgy Industry Press (2002)
8. Ye, Z. F., Liu, J. G.: Probabilistic Neural Networks Based Engine Fault Diagnosis. *Aviation Transaction* **23** (2002) 155-157
9. Li, D. H., Liu, H.: Method and Application of Fault Diagnosis Based on Probabilistic Neural Network. *Systems Engineering and Electronics* **26** (2004) 997-999
10. Wei, X. P., Yu, X. B., Jin, Y. L.: Car Shape Recognition Based on Matrix Singular Value. *Journal of Image and Graphics* **8** (2003) 47-50

Support Vector Machine with Composite Kernels for Time Series Prediction

Tiejun Jiang¹, Shuzong Wang¹, and Ruxiang Wei²

¹ Research Institute of New Weaponry Technology & Application, Naval University of Engineering, Wuhan 430033, P.R. China
tiejunjiang@gmail.com

² Department of Equipment Economy Management, Naval University of Engineering, Wuhan 430033, P.R. China
weiruxiang@163.com

Abstract. In Support Vector Machine (SVM), Kernels are employed to map the nonlinear model into a higher dimensional feature space where the linear learning is adopted. The characteristics of kernels have great impacts on learning and predictive results of SVM. Considering the characteristics for fitting and generalization of two kinds of typical kernels—global kernel (polynomial kernel) and local kernel (RBF kernel), a new kind of SVM modeling method based on composite kernels is proposed. In order to evaluate the reasonable fitness of kernel functions, the particle swarm optimization (PSO) algorithm is used to adaptively evolve SVM to obtain the best prediction performance, in which each particle represented as a real vector corresponds to a set of the candidate parameters of SVM. Experiments in time series prediction demonstrate that the SVM with composite kernels has the better performance than with a single kernel.

1 Introduction

As a method of machine learning, support vector machine (SVM) can be traced to statistical learning theory introduced by Vapnik the late 1960s [1]. Unlike most of the traditional methods, SVM implements the Structural Risk Minimization (SRM) principal which seeks to minimize an upper bound of the generalization error rather than minimize the training error [2]. This eventually results in remarkable characteristics such as the good generalization performance, the absence of local minima and the sparse representation of solution. Recently, SVM has become a popular tool in time series prediction [3–5].

In the modeling of SVM, the choice of kernel function is quite important to the prediction performance. To solve the modeling problem of time series prediction, the modeling performance of SVM with different kernels is analyzed, which is related to the global and local characteristics of kernels [6]. In this paper, the mapping characteristics of two typical global (polynomial) and local (RBF) kernels are described, then a new kind of SVM modeling method based on composite kernels is proposed,

which not only has a good fitting accuracy, but also can avoid the fluctuation of the prediction outputs caused by the local kernel.

However, after a kernel is selected by SVM in practice, the main problem is how to determine the kernel parameter and the regularization parameter. In general, the direct setting and grid search are used. In this paper, the particle swarm optimization (PSO) algorithm is adopted to select the parameters of the SVM with composite kernels for time series prediction and numerical results show its validity.

The rest of this paper is organized as follows. In Section 2, composite kernels and PSO are described. Section 3 shows the experimental results, followed by the conclusions in the last section.

2 Methodology

2.1 Composite Kernels

In the application of SVM, the choice of kernel function is significantly important, which determines the performance of time series prediction. The characterization of a kernel function is done by means of the Mercer's theorem [7].

Every kernel has its advantages and disadvantages. Numerous possibilities of kernels satisfying Mercer's theorem exist, which can be simply divided into two types of kernels, local and global kernels [8].

In local kernels, only the data that are close or in the proximity of each other have obvious effects on the kernel values. An example of a typical local kernel is the radial basis function (RBF) kernel, $K(x, x_i) = \exp\{-|x - x_i|^2 / \sigma^2\}$, where the kernel parameter is the width σ of the radial basis function. Its mapping characteristics are shown in Fig.1 (a), where the points adjacent to the test input have great effects on the kernel values. In contrast, a global kernel allows data points that are far away from each other to have obvious effects on the kernel values as well. The polynomial kernel $K(x, x_i) = [(x \cdot x_i) + 1]^q$, where the kernel parameter q is the degree of the polynomial to be used, is a typical example of a global kernel. Its mapping characteristics are shown in Fig.1 (b), where the points far away from the test input have great effects on the kernel values. Fig.1 (c) shows the contrast between RBF kernel and polynomial kernel.

The composites of these two kernels may combine the good characteristics of two kernels and have better performances than any single kernel. The composites of the RBF and polynomial kernels can be defined as [9]:

$$K = \rho K_{poly} + (1 - \rho) K_{rbf} \quad , \quad (1)$$

where the value of the composite coefficient ρ is a constant scalar.

The characteristics of composite kernels are determined by different values of ρ for different regions of the input space. ρ is a vector. Through this approach, the relative contribution of both kernels to the model can be varied over the input space. In this paper, a uniform ρ over the entire input space is used.

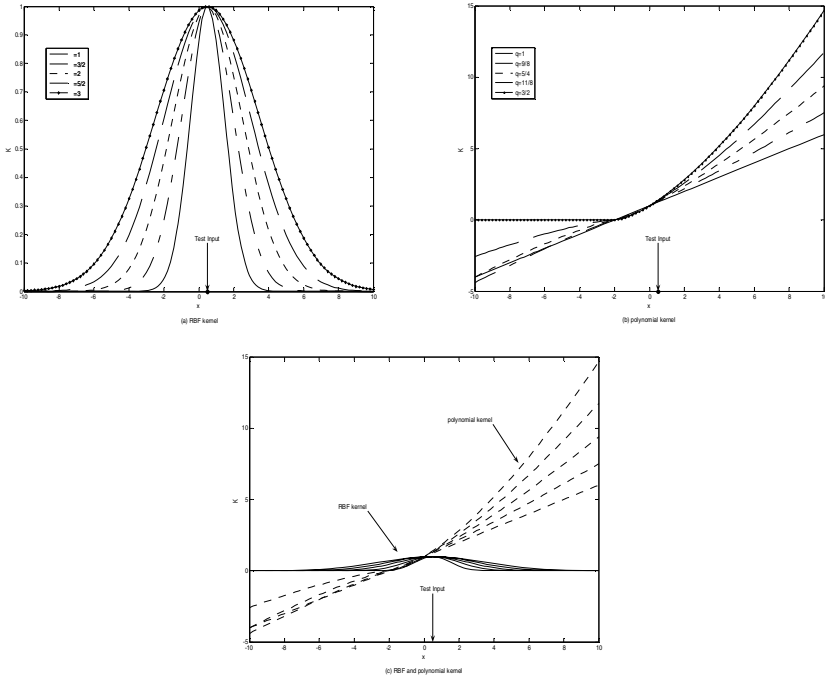


Fig. 1. The mapping characteristics of RBF kernel and polynomial kernel

2.2 PSO

Due to composite kernels introduced, the parameters to be tuned are much more than of a single kernel. In this paper, these parameters are automatically tuned using the particle swarm optimization (PSO) in the training phase.

Particle swarm optimization (PSO) is a parallel population-based computation technique that was invented by Kennedy and Eberhart (1995) [3], which has been motivated by the behaviour of organisms such as fish schooling and bird flocking.

In this paper, the global optimizing models are expressed as follows:

$$V_i(k+1) = w(k) \times V_i(k) + C_1 \times Rand \times (Pbest_i - X_i(k)) + C_2 \times rand \times (Gbest - X_i(k)) \quad (2)$$

$$X_i(k+1) = X_i(k) + V_i(K+1) \quad (3)$$

where $V_i(k+1)$ is the velocity of $(k+1)$ th iteration of i th individual, $V_i(k)$ is the velocity of k th iteration of i th individual, $X_i(k)$ represents the position of i th individual at k th iteration, $w(k)$ is the inertial weight. C_1 , C_2 are the positive constant parameters, $Rand$ and $rand$ are the random functions with a range $[0, 1]$, $Pbest_i$ is the best position of the i th individual and $Gbest$ is the best position among all individuals in the population.

In PSO algorithm, the population has l particles and each particle is an m -dimensional vector, where m is the number of optimized parameters. The computational flow of PSO can be described in the following steps:

- (1) Randomly initialize positions $X(0)$ and velocities of all particles $V(0)$.
- (2) Calculate the fitness value of current particle: $fit(X_i)$.
- (3) Compare the fitness value of $Pbest$ with $fit(X_i)$, if $fit(X_i)$ is better than the fitness value of $Pbest$, then $Pbest$ is set to the current position X_i .
- (4) Find the global best position of the swarm. If $fit(X_i)$ is better than the fitness value of $Gbest$ then set $Gbest$ to the position of the current particle X_i .
- (5) Calculate velocities V_i using Eq.(2). If $V_i > V_{max}$ then $V_i = V_{max}$. If $V_i < V_{min}$ then $V_i = V_{min}$.
- (6) Calculate positions X_i using Eq.(3). If a particle violates its position limits in any dimension, set its position at the proper limit.
- (7) If one of the stopping criteria (Generally, good enough fitness value or the preset maximal iteration k_{max}) is satisfied, then stop; else go to Step (2).

3 Experiments

To illustrate the effectiveness of using the SVM with composite kernels for time series prediction, the sunspot data were used as the evaluation of the prediction performance. The sunspot data set is believed to be nonlinear and non-stationary. The data set consists of a total of 280 yearly averaged sunspots recorded from 1700 to 1979. In this paper, the data points from 1700 to 1928 were used as the training cases, and the remaining data points from 1929 to 1979 were used as the testing cases.

The free parameters which produce the smallest sum of the validation errors were used in the experiment. The error was measured by the Root mean square error (RMSE) criterion. The RMSE is calculated as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}}, \quad (4)$$

where \hat{y}_i ($i=1,2,\dots,n$) is the predicted value, y_i ($i=1,2,\dots,n$) is the actual value, n is the number of data points.

In the paper, the population size of PSO was set at 50; The acceleration constant $C_1 = 2.0$, $C_2 = 2.0$; The maximal iteration $k_{max} = 200$; The inertia weight w is very important for the convergence behaviour of PSO. A suitable value usually provides a balance between global and local exploration abilities and consequently results in a better optimum solution. We used the following equation to adjust w to enable quick convergence:

$$w(k) = w_{max} - \frac{w_{max} - w_{min}}{k_{max}} \times k, \quad (5)$$

where w_{max} is the initial weight, w_{min} is the final weight, k is the current iteration number and k_{max} is the maximum iteration number. In this paper, $w_{max} = 1.2$, $w_{min} = 0.4$. Considering the randomness of the PSO algorithm, we did the experiments 30 times and chose the best results.

Eight previous sunspots were used to predict the current sunspot in this paper. In Table 1, the converged RMSE and the optimal parameters (including the width σ of the radial basis function, the degree q of the polynomial, the composite coefficient ρ and the regularization parameter γ of SVM) of three kernels in the sunspot data are shown. The best results obtained in the SVM by using three kernels are given in Table 1. The optimal parameters used in three different kernels are also illustrated in

Table 1. The converged RMSE and the optimal parameters of three kernels

Kernel function	RBF kernel	Polynomial kernel	Composite kernel
Parameters	$\sigma = 1.2878$ $\gamma = 27.0191$	$d = 5.1552$ $\gamma = 43.4788$	$d = 6.9605$ $\sigma = 4712.7783$ $\gamma = 55.8697$ $\rho = 0.011597$
RMSE	0.20699	0.2348	0.19499

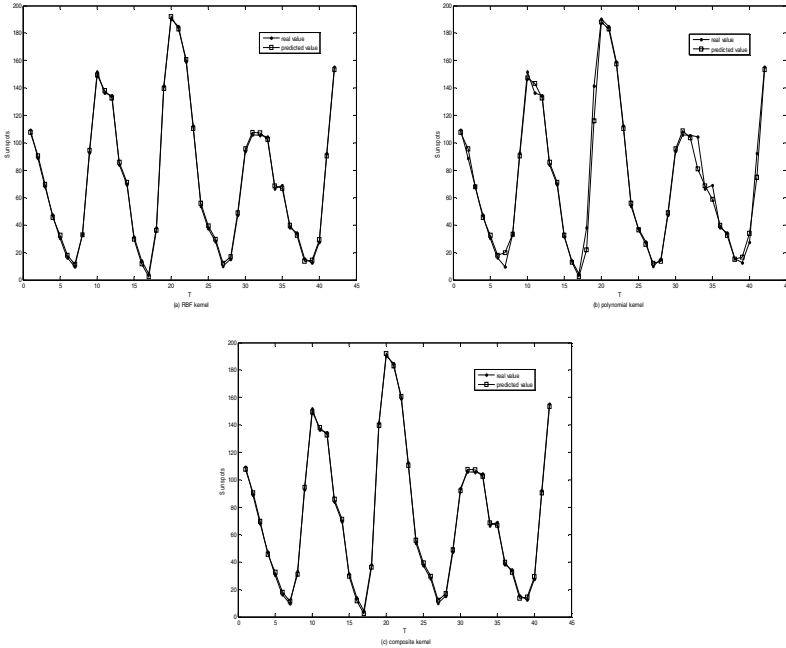


Fig. 2. One-step-ahead prediction results of the sunspot data by the SVM with three kernels

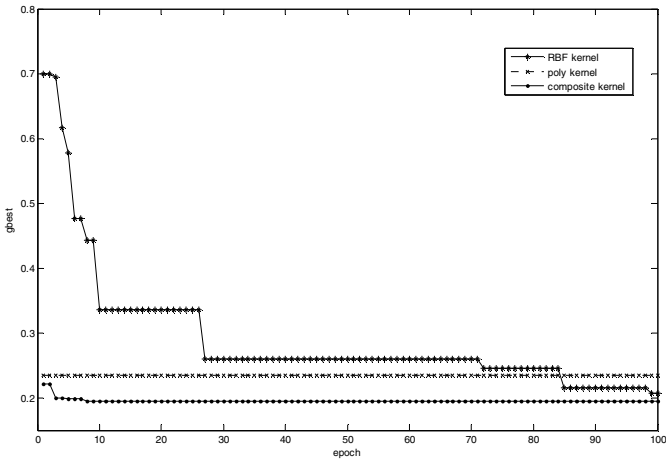


Fig. 3. Convergent cure of error

this table. From Table 1, the SVM with the RBF kernel or composite kernel can both converge to a smaller RMSE on the testing cases than with the polynomial kernel. Furthermore, there is the smallest error in the SVM with the composite kernel. Experiments show, through the appropriate composite of kernel functions, we can obtain better results than a simple kernel function.

In Fig.2, one-step-ahead prediction results of the sunspot data by the SVM with three different kernels, whose parameters are automatically tuned using PSO, are shown.

We can see that the prediction results are quite accurate. Fig.3 shows the constringency of PSO in the situation of three kernels. It can be seen the constringency of the SVM with the composite kernel is quicker and better than with a single kernel and the optimal results can be obtained through PSO.

4 Conclusions

Through the analysis on mapping characteristics of two kernels—polynomial and RBF kernels, a new kind of SVM modeling method based on composite kernels is proposed. Polynomial kernel can effectively restrain the fluctuation of prediction outputs. On the other hand, RBF kernel can provide a good way to improve the fitting accuracy. Experiments demonstrate the better performance of the proposed method in modeling analysis compared to a single kernel.

In this paper, the very efficient PSO optimization method has been used to enhance the SVM with composite kernels for improving the time series prediction efficiency. The sunspot data were used to evaluate the performance of the proposed method. The results show that PSO can efficiently obtain the optimal parameters needed in the SVM. The SVM with composite kernels optimized by PSO will be widely used in many other complex time series. In this paper, only polynomial and RBF kernels, as two typical global and local kernels, are discussed. Further work will discuss more complex composite kernels based on more single kernels.

References

1. Vapnik, V. N.: On the Uniform Convergence of Relative Frequencies of Events to Their Probabilities. *Soviet Mathematics: Doklady* **9** (1968) 915–918
2. Vapnik, V. N.: *The Nature of Statistical Learning Theory*. Springer-Verlag, New York (1995)
3. Mukherjee S., Osuna E., Girosi F.: Nonlinear Prediction of Chaotic Time Series Using Support Vector Machines. *Proc. of IEEE NNSP'97*, Amelia Island, FL (1997)
4. S. Mukherjee, E. Osuna, F. Girosi: Nonlinear Prediction of Chaotic Time Series Using Support Vector Machines. *NNSP' 97: Neural Networks for Signal Processing VII: Proceedings of the IEEE Signal Processing Society Workshop*, Amelia Island, FL, USA (1999) 511–520
5. Tay, F.E.H., Cao, L.J.: Application of Support Vector Machines in Financial Time Series Forecasting, *Omega* **29** (4) (2001) 309–317
6. Scholkopf, B., Mika, S., Burges, C.J.C., Knirsch, P., Muller, K.R., Ratsch, G., Smola, A.J.: Input Space Versus Feature Space in Kernel-Based Methods. *IEEE Trans. Neural Networks* **10** (1999) 1000–1017
7. Mercer, J.: Functions of Positive and Negative Type and their Connection with the Theory of Integral Equations. *Philos. Trans. Roy. Soc. London A* **209** (1909) 415–446
8. Smola, A.: *Learning with Kernels*. Ph.D. thesis, GMD, Birlinghoven (1999)
9. Smits, G.F., Jordaán, E.M.: Improved SVM Regression Using Mixtures of Kernels. In: *Proc. of IJCNN 02 on Neural Networks* **3** (2002) 2785–2790

Neuromorphic Quantum-Based Adaptive Support Vector Regression for Tuning BWGC/NGARCH Forecast Model

Bao Rong Chang^{1,*} and Hsiu Fen Tsai²

¹Department of Computer Science and Information Engineering
National Taitung University, Taitung, Taiwan 950
Phone: +886-89-318855 ext. 2607; Fax: +886-89-350214
brchang@nttu.edu.tw

²Department of International Business
Shu-Te University, Kaohsiung, Taiwan 824
soenfen@mail.stu.edu.tw

Abstract. A prediction model, called BPNN-weighted grey model and cumulated 3-point least square polynomial (BWGC), is used for resolving the overshoot effect; however, it may encounter volatility clustering due to the lack of localization property. Thus, we incorporate the non-linear generalized autoregressive conditional heteroscedasticity (NGARCH) into BWGC to compensate for the time-varying variance of residual errors when volatility clustering occurs. Furthermore, in order for adapting both models optimally, a neuromorphic quantum-based adaptive support vector regression (NQASVR) is schemed to regularize the coefficients for both BWGC and NGARCH linearly to improve the generalization and the localization at the same time effectively.

1 Introduction

Grey model (GM) [1] has been widely applicable for the purpose of short-term forecasting but has encountered the overshoot effect [2] which results in big residual errors in grey model prediction [3]. In order to reduce this effect, a cumulated 3-point least squared linear prediction (C3LSP) [2] yield the underestimated output at the same period to offset the overshoot result. A back-propagation neural net (BPNN) [4] is particularly introduced to combine GM and C3LSP linearly denoted by BWGC [3]. However, the volatility clustering effect [5] [6] actually deteriorates the performance of BWGC model due to the occurrence of big residual errors. In order to reduce the volatility clustering, a method called nonlinear generalized autoregressive conditional heteroscedasticity (NGARCH) [7] [8], which can overcome the problem of volatility clustering, has been incorporated into BWGC model to form a composite model. Neuromorphic quantum-based adaptive support vector regression (NQASVR) is applied to regularizing the linear combination of BWGC and NGARCH optimally.

* Corresponding author.

2 New Composite Model BWGC/NGARCH

ARMAX(r,m,Nx) [9] encompasses autoregressive (AR), moving-average (MA) and regression (X) models, in any combination, as expressed below

$$y_{armax}(t) = C^{armax} + \sum_{i=1}^r R_i^{armax} y(t-i) + e_{resid}(t) + \sum_{j=1}^m M_j^{armax} e_{resid}(t-j) + \sum_{k=1}^{N_x} \beta_k^{armax} \mathbf{X}(t,k) \quad (1)$$

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.

NGARCH(p,q) [10] [11] describes nonlinear time-varying conditional variances and Gaussian residuals $e_{resid}(t)$. Its mathematical formula is

$$\sigma_{ntvcv}^2(t) = K^{ng} + \sum_{i=1}^p G_i^{ng} \sigma_{ntvcv}^2(t-i) + \sum_{j=1}^q A_j^{ng} \sigma_{ntvcv}^2(t-j) \left[\frac{e_{resid}(t-j)}{\sqrt{\sigma_{ntvcv}^2(t-j)}} - C_j^{ng} \right]^2 \quad (2)$$

with constraints

$$\sum_{i=1}^p G_i^{ng} + \sum_{j=1}^q A_j^{ng} < 1, K^{ng} > 0, G_i^{ng} \geq 0, i=1,\dots,p, A_j^{ng} \geq 0, j=-1,\dots,q$$

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.

This proposed composite model is rewritten as BWGC/NGARCH. Formulation of the linear combination [12] is expressed as

$$y_{NewComposite Model}(t) = g(y_{BWGC}(t), y_{NGARCH}(t)) = Coef_1 \cdot y_{BWGC}(t) + Coef_2 \cdot y_{NGARCH}(t) \quad (3)$$

where g is defined as a linear function of the BWGC and NGARCH outputs, respectively, $y_{BWGC}(t)$ and $y_{NGARCH}(t)$, while $Coef_1$ and $Coef_2$ are respectively the coefficients of the linear combination of the BWGC and NGARCH outputs.

3 Neuromorphic Quantum-Based ASVR

3.1 Adaptive Support Vector Regression (ASVR)

Initially developed for solving classification problems, support vectors machines (SVMs) technology [13] can also be successfully applied in regression problems, i.e. functional approximation. We consider approximating functions $f(\cdot)$ solved by support vector regression (SVR) [14] with the form of

$$f(\mathbf{x}, \mathbf{w}) = \sum_{i=1}^l w_i \phi(x_i) + b, \tag{4}$$

where $\phi(\cdot)$, w_i , and b denote a nonlinear mapping, a weighted value, and a bias, respectively. Furthermore, Vapnik introduced a general type of loss function, namely the linear loss function with ε -insensitivity zone [13], as

$$|y - f(\mathbf{x}, \mathbf{w})|_{\varepsilon} = \begin{cases} 0 & \text{if } |y - f(\mathbf{x}, \mathbf{w})| \leq \varepsilon, \\ |y - f(\mathbf{x}, \mathbf{w})| - \varepsilon & \text{otherwise} \end{cases} \tag{5}$$

According to the learning theory of SVMs [13], this can be expressed by maximizing dual variables Lagrangian $L_d(\mathbf{a}, \mathbf{a}^*)$ where l , \mathbf{x}_i , \mathbf{y}_i , and $K(\cdot, \cdot)$ denote the number of vectors, an input vector, an output vector, and the kernel function, respectively.

$$L_d(\mathbf{a}, \mathbf{a}^*) = -\frac{1}{2} \sum_{i,j=1}^l (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) K(\mathbf{x}_i, \mathbf{x}_j) - \varepsilon \sum_{i=1}^l (\alpha_i + \alpha_i^*) - \sum_{i=1}^l (\alpha_i - \alpha_i^*) \mathbf{y}_i, \tag{6}$$

subject to the constraints

$$\sum_{i=1}^l \alpha_i = \sum_{i=1}^l \alpha_i^*, \tag{7}$$

$$0 \leq \alpha_i \leq C, \quad i = 1, \dots, l \tag{8}$$

$$0 \leq \alpha_i^* \leq C, \quad i = 1, \dots, l \tag{9}$$

After obtaining the Lagrange multipliers α_i and α_i^* , we find the optimal weights of regression

$$\mathbf{w}_0 = \sum_{i=1}^l (\alpha_i - \alpha_i^*) \phi(\mathbf{x}_i), \tag{10}$$

and an optimal bias

$$\mathbf{b}_0 = \frac{1}{l} \left(\sum_{i=1}^l (\mathbf{y}_i - \phi(\mathbf{x}_i)^T \mathbf{w}_0) \right). \tag{11}$$

A fast algorithm applied to constraint optimization for support vector regression (SVR) is called adaptive support vector regression (ASVR) [15]. It is designed for exploring three free parameters C , ε and σ_{rbkf} [14] such that the computation time of quadratic programming (QP) is significantly reduced and achieved rapid convergence to the near-optimal solution. In the ASVR algorithm, two scale factors, v and ν refer to [15], are evaluated in advance and then applied these evaluated values for calculating the free parameters ε in Eq. (12) and σ_{rbkf} in Eq. (13), respectively, where the vector \mathbf{x} stands for an input vector. An order n , see [15], is also predetermined and used in computing the free parameter C in Eq. (13) and Eq. (14). In this manner, a straightforward parameter-seeking is done rather than using a heuristic method due to a long time for searching. Note that Eq. (13) and Eq. (14) are based on the modified Bessel function of second kind with the order n [16] as follows:

$$\varepsilon = v \times \frac{|\max(\mathbf{X}) - \min(\mathbf{X})|}{2} \tag{12}$$

$$C = K_n(\varepsilon) = (-1)^{n+1} \{ \ln(\varepsilon/2) + \gamma \} I_n(\varepsilon) + \frac{1}{2} \sum_{k=0}^{n-1} (-1)^k (n-k-1) (\varepsilon/2)^{2k-n} + \frac{(-1)^n}{2} \sum_{k=0}^{\infty} \frac{(\varepsilon/2)^{n+2k}}{k!(n+k)!} \{ \Phi(k) + \Phi(n+k) \} \tag{13}$$

$$I_n(\varepsilon) = \sum_{k=0}^{\infty} \frac{(\varepsilon/2)^{n+2k}}{k! \Gamma(n+k+1)} \tag{14}$$

where $\gamma = 0.5772156\dots$ is Euler’s constant and $\Phi(p) = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{p}$, $\Phi(0) = 0$ [16].

Eq. (14) determines a free parameter σ_{rbkf} of radial basis kernel function for quadratic programming in SVR.

$$\sigma_{rbkf} = v \cdot \left(\sum_{i=1}^l (x_i - \bar{x})^2 / l - 1 \right)^{1/2}, \quad \bar{x} = \sum_{j=1}^l x_j / l \tag{15}$$

In short, adaptive support vector regression firstly finds three tunable free parameters C , ε , and σ_{rbkf} and subsequently utilizes these parameters in SVR optimization to search for the optimal weights and bias mentioned above.

3.2 Neuromorphic Quantum-Based ASVR (NQASVR)

The synaptic weights w_{ijkl} are given in a Hopfield network [17].

$$E_{HN} = -\frac{1}{2} \sum_{ij} \sum_{kl} w_{ijkl} o_{ij} o_{kl} - \sum_{ij} h_{ij} o_{ij}, \tag{16}$$

where h_{ij} is the external bias for a neuron. The synaptic weights are obtained as

$$w_{ijkl} = -2a \delta_{j,l} (1 - \delta_{i,k}) - 2b \delta_{i,k} (1 - \delta_{j,l}) - 2c \delta_{i+j,k+l} (1 - \delta_{i,k}) - 2d \delta_{i-j,k-l} (1 - \delta_{i,k}), \tag{17}$$

where δ_{ij} is the Kronecker delta. Let us consider that each qubit corresponds to each neuron of a Hopfield network. The state vector $|\psi\rangle$ of the whole system is given by the product of all qubit states.

The time evolution of the system is given by the following Schrödinger equation.

$$|\psi(t+1)\rangle = U(1) |\psi(t)\rangle = e^{-\frac{iH(t)}{\hbar}} |\psi(t)\rangle, \tag{18}$$

Here, the operator $U(1)$ is given by the Padé approximation [18].

The quantum computation algorithm utilizing adiabatic Hamiltonian evolution has been proposed by Farhi et al. [19]. Adiabatic Hamiltonian evolution is given as

$$H(t) = \left(1 - \frac{t}{T} \right) H_I + \frac{t}{T} H_F, \tag{19}$$

where H_I and H_F are the initial and final Hamiltonians, respectively. We assume that the quantum system starts at $t = 0$ in the ground state of H_I , so that all possible candidates are set in the initial state $|\psi(0)\rangle$. T denotes the period in which the Hamiltonian evolves and the quantum state changes, and we can control the speed of such changes to be suitable for finding the optimal solution among all candidates set in $|\psi(0)\rangle$. If a sufficiently large T is chosen, the evolution becomes adiabatic. The adiabatic theorem says that the quantum state will remain close to each ground state [20]. Therefore, the optimal solution can be found as the final state $|\psi(T)\rangle$. However, successful operation is not guaranteed in the case that there exists any degeneracy in energy levels or any energy crossing during the evolution [20]. The initial Hamiltonian H_I is chosen so that its ground state is given by the superposition of all states as

$$|\psi(0)\rangle = \frac{1}{\sqrt{2^n}} \sum_{i=0}^{2^n-1} |i\rangle, \tag{20}$$

where n is the number of qubits and $|i\rangle$ is the n -th eigenvector. The initial Hamiltonian H_I is given as

$$\begin{aligned} H_I &= \left(\sigma_x^{(0)} + \sigma_x^{(1)} + \dots + \sigma_x^{(2^{16}-1)} \right) \\ &= (\sigma_x \otimes I \otimes \dots \otimes I + I \otimes \sigma_x \otimes \dots \otimes I, \\ &\quad + \dots + I \otimes I \otimes \dots \otimes \sigma_x) \end{aligned} \tag{21}$$

where σ_x is the x-component of the Pauli spin matrix. One can choose any other H_I which satisfies that its ground state is expressed by a linear combination of all states. For example, σ_x can be replaced by σ_y . It may be possible to solve optimization problems by composing a new Hamiltonian H_F considering the synaptic weights w_{ijkl} . The Hamiltonian H_F for the target problem is obtained as shown in the following equation. The eigenvalue ε_i of a basis state $|i\rangle$ should be obtained from the cost function in Eq. (16). Therefore, H_F has ε_i as the diagonal elements as

$$H_F = \begin{pmatrix} \varepsilon_0 & & & 0 \\ & \varepsilon_1 & & \\ & & \ddots & \\ 0 & & & \varepsilon_{2^n-1} \end{pmatrix} = \sum_{n=0}^{2^n-1} \varepsilon_i |i\rangle\langle i|, \tag{22}$$

By choosing a proper H_F , it is possible to solve the problems. However, the calculation cost of 2^n , where n is the number of qubits, is necessary in order to obtain the above-mentioned Hamiltonian H_F , and there is no difference when compared with a heuristic search. Therefore, the key of application is how we choose a more effective H_F with less calculation cost.

The Hopfield net [21] is a recurrent neural network as shown in Fig. 1 in which all connections are symmetric. Invented by John Hopfield (1982), this network has the property that its dynamics are guaranteed to converge. If the connections are trained

using Hebbian learning then the Hopfield network can perform robust content-addressable memory, robust to connection alteration. Various functions of an artificial neural network (ANN) are realized by choosing suitable synaptic weights. A full connection neural network has n^2 synapses, where n is the number of neurons. In order to reduce the calculation cost of the Hamiltonian, we consider interactions between qubits and study a new method with a new H_F comprising nondiagonal elements considering its analogy to ANN. For convenience, we assume we have closely coupled 2-spin-1/2 qubits. The Hamiltonian of this quantum system is

$$H = J(\sigma_1, \sigma_2) = J_{12} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 2 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{23}$$

where J_{12} is the magnitude of the interactions, and σ_i is the Pauli spin matrix. The eigenvalues and eigenvectors of this Hamiltonian when $J_{12}=1$ are shown as -3 for $|01\rangle-|10\rangle$ and 1 for $|00\rangle, |11\rangle, |01\rangle+|10\rangle$, respectively. The possible states to be measured are $|10\rangle$ or $|01\rangle$ if the system is in the ground state $|01\rangle-|10\rangle$. It can be said that the interaction of two neurons is inhibitory if we consider the analogy with an ANN model. Excitatory interaction is also possible with another Hamiltonian. From the above consideration, we can design a new Hamiltonian by converting the synaptic weights in Eq. (16) to the interactions of qubits.

The neuromorphic quantum -based optimization, as shown in Fig. 1, can apply w_{kj} in Eq. (16) to calculate final Hamiltonian H_F like an example as shown in Eq. (23) providing for adiabatic evolution algorithm in Eq. (18)-(19) to train the constrained optimization for an cost function expressed in Eq. (6)-(9).

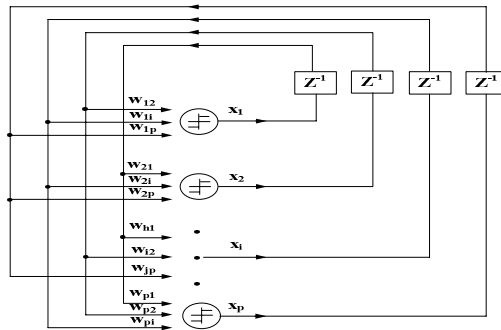


Fig. 1. A Hopfield network is exploited for neuronmorphic quantum-based optimization

4 Hybrid Forecasting System

For simplicity employed in [12], a linear weighted-average is devised in this study to combine both $\delta_{bavgc}(k)$ and $\hat{\sigma}_{ngarch}(k+1)$ as the best approach to resolve the problem

instead of the nonlinear one. Therefore, the overall result $\delta_{bwgcnarch}(k)$ will be proposed to be a linear combination of the outputs of BWGC and NGARCH as formulated:

$$\begin{aligned} \delta_{bwgcnarch}(k) &= w_{bwgc}(k) \cdot \delta_{bwgc}(k) + w_{ngarch}(k) \cdot \hat{\sigma}_{ngarch}(k) \\ \text{s.t. } w_{bwgc}(k) + w_{ngarch}(k) &= 1 \end{aligned} \tag{24}$$

where the weights, $w_{bwgc}(k)$ and $w_{ngarch}(k)$, are determined by applying neuromorphic quantum-based ASVR as shown in Fig. 2. A predicted result is determined by adding a predicted variation at next period into the current true value as expressed below.

$$\hat{y}_{bwgcnarch}(k) = y(k-1) + \delta_{bwgcnarch}(k) \tag{25}$$

A digital cost-function (DCF) [22] defined as

$$DCF = \sum_{k=1}^l \|y(k) - \hat{y}_{bwgcnarch}(k)\|^2 \tag{26}$$

which can be used for measuring the accuracy for each trial of quantum computing.

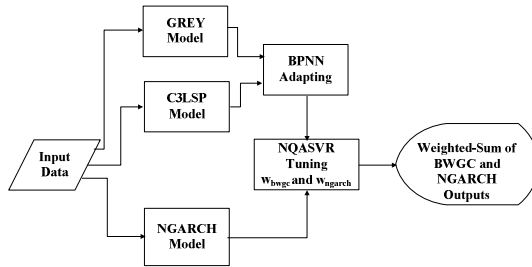


Fig. 2. A diagram depicts the composite model, BWGC/NGARCH, regularized by neuromorphic quantum-based adaptive support vector regression (NQASVR)

Neuromorphic quantum-based ASVR mentioned above is employed in this hybrid prediction for tuning the appropriate weights, $w_{bwgc}(k)$ and $w_{ngarch}(k)$, for the forecast $\delta_{bwgc}(k)$ and $\hat{\sigma}_{ngarch}(k)$ as per Eq. (24), respectively.

5 Experimental Results and Discussions

As shown in Fig. 3 to Fig. 6 or Fig. 7 to Fig. 8, the predicted sequences indicate the predicted results for the following competing methods: (a) grey model (GM), (b) auto-regressive moving-average (ARMA), (c) radial basis function neural network (RBFNN), (d) adaptive neuro-fuzzy inference system (ANFIS), (e) support vector regression (SVR), (f) real-coded genetic algorithm [23] tuning BWGC/NGARCH model (RGA-BWGC/NGARCH), (g) quantum-minimized BWGC/NGARCH model (QM- BWGC/NGARCH), and (h) neuromorphic quantum-based adaptive support vector regression tuning BWGC/NGARCH model (NQASVR-BWGC/NGARCH). In the experiments, the most recent four actual values is considered as a set of input data used for modeling to predict the next desired output. As the next desired value is

obtained, the first value in the current input data set is discarded and joins the latest desired (observed) value to form a new input data set for the use of next prediction. First, the international stock price indexes prediction for four markets (New York Dow Jones Industrials Index, London Financial Time Index (FTSE-100), Tokyo Nikkei Index, and Hong Kong Hang Seng Index) [24] have been experimented for 48 months (from Jan. 2002 to Dec. 2005) as shown in Fig. 3 to Fig. 6 and in Table 1. Criterion of mean square error for measuring the predictive accuracy is expressed.

$$MSE = \sum_{t=1}^l (y_{t_c+t} - \hat{y}_{t_c+t})^2 / l \tag{27}$$

where l = the number of periods in forecasting, t_c = the current period, y_{t_c+t} = a desired value at the $t_c + t$ th period and \hat{y}_{t_c+t} = a predicted value at the $t_c + t$ th period.

The goodness of fit on the first experiment is tested by Q-test successfully due to all of p-values greater than level of significance (0.05) [25]. Second, London International Financial Futures and Options Exchange (LIFFE) [26] provide the indices of volumes of equity products on futures and options, and further their indices forecast for 24 months (from Jan. 2001 to Dec. 2002) are shown in Fig. 7 and Fig. 8 and listed in Table 2. This is also tested by Q-test successfully due to all of p-values greater than level of significance (0.05).

Table 1. The comparison between different prediction models based on Mean Squared Error (MSE) on international stock price monthly indices (unit= 10^3)

Methods	NY-D.J. Industrials Index	London FTSE-100 Index	Tokyo Nikkei Index	HK Hang Seng Index	Average MSE
GM	1.9582	0.4006	3.2209	5.1384	2.6795
ARMA	1.8230	0.3883	2.9384	4.3407	2.3726
RBFNN	1.8319	0.3062	3.8234	3.1976	2.2898
ANFIS	1.4267	0.3974	3.5435	4.0279	2.3489
SVR	1.2744	0.3206	2.6498	3.1275	1.8431
RGA-BWGCNG	1.2642	0.2630	2.0175	3.1271	1.6680
QM-BWGCNG	1.1723	0.2156	1.8656	3.0754	1.5822
NQASVR-BWGCNG	1.1694	0.2168	1.8671	3.0362	1.5724

Table 2. The comparison between different prediction models based on Mean Squared Error (MSE) on futures and options volumes monthly indices of equity products

Methods	Futures Index of Equity Products	Options Index of Equity Products	Average MSE
GM	0.0945	0.0138	0.0542
ARMA	0.0547	0.0114	0.0331
RBFNN	0.0196	0.0087	0.0142
ANFIS	0.0112	0.0092	0.0102
SVR	0.0191	0.0085	0.0138
RGA-BWGCNG	0.0110	0.0082	0.0096
QM-BWGCNG	0.0098	0.0072	0.0085
NQASVR-BWGCNG	0.0101	0.0070	0.0086

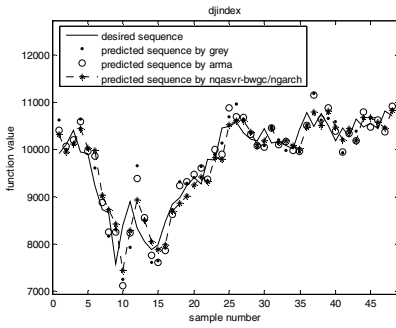


Fig. 3. Forecasts of N. Y. -D. J. Industrilas monthly-closing index

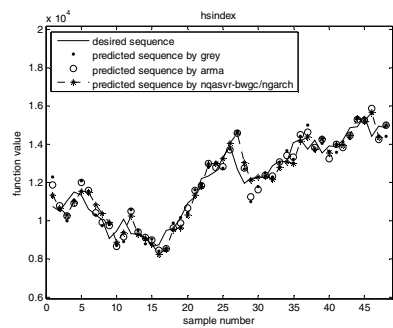


Fig. 6. Forecasts of Hong Kong Hang-Seng monthly-closing index

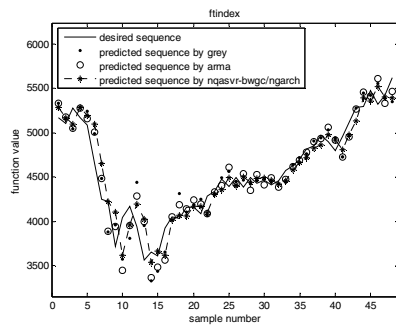


Fig. 4. Forecasts of London FTSE-100 monthly-closing index

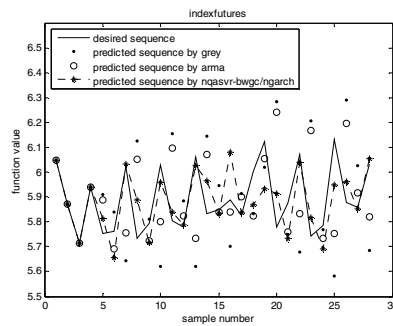


Fig. 7. Forecasts of monthly-closing equity volume index futures

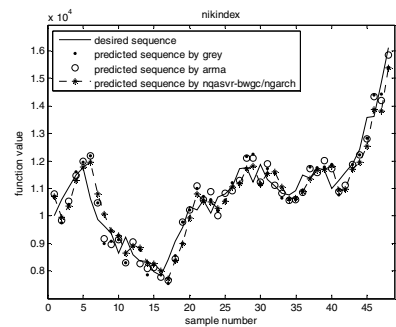


Fig. 5. Forecasts of Japan Nikkei monthly-closing index

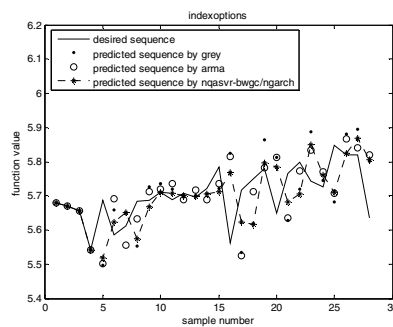


Fig. 8. Forecasts of monthly-closing equity volume index options

6 Concluding Remarks

Neuromorphic quantum-based ASVR optimization process tackles so-called NP-complete problem outperforms artificial neural network and RGA to attain optimal or

near-optimal coefficients over the search space. It follows that this proposed method can get the satisfactory results because its generalization is much improved.

Acknowledgement

This work is fully supported by the National Science Council, Taiwan, Republic of China, under grant number **NSC 94-2218-E-143-001**.

References

1. Deng, J. L.: Control Problems of Grey System. *System and Control Letter*. 1 **5** (1982) 288-294
2. Chang, B. R. and Tsai, S. F.: A Grey-Cumulative LMS Hybrid Predictor with Neural Network Based Weighting Mechanism for Forecasting Non-Periodic Short-Term Time Series. In *Proc. of IEEE SMC02*, **6** (2002) WA2P3
3. Chang, B. R.: Advanced Hybrid Prediction Algorithm for Non-Periodic Short-Term Forecasting. *International Journal of Fuzzy System*. 5 **3** (2003) 151-160
4. Haykin, S.: *Neural Networks*, 2nd Edition. Prentice Hall, New Jersey (1999)
5. Engle, R.: Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica*. **50** (1982) 987-1008
6. Gouriéroux, C.: *ARCH Models and Financial Applications*, Springer-Verlag, New York (1997)
7. Bollerslev, T.: Generalized Autoregressive Conditional Heteroscedasticity. *Journal of Econometrics*. **31** (1986) 307-327
8. Ritchken, P. and Trevor, R.: Pricing Options Under Generalized GARCH and Stochastic Volatility Process. *Journal of Finance*. **54** (1999) 337-420
9. Hamilton, J. D.: *Time Series Analysis*. Princeton University Press, New Jersey (1994)
10. Hentschel, L.: All in the Family: Nesting Symmetric and Asymmetric GARCH Models. *Journal of Financial Economics*. **39** (1995) 71-104
11. 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
12. Pshenichnyj, B. N. and Wilson, S. S.: *The Linearization Method for Constrained Optimization*. Springer, New York (1994)
13. Vapnik, V.: *The Nature of Statistical Learning Theory*. Springer-Verlag, New York (1995)
14. Cristianini, N. and Shawe-Taylor, J.: *An Introduction to Support Vector Machines (and other kernel-based learning methods)*. Cambridge University Press, London (2000)
15. 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
16. Kreyszig, E.: *Advanced Engineering Mathematics*, 8th Edition. Wiley, New York (1999)
17. Tank, D. W., Hopfield, J. J.: Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit. *IEEE Trans. Circuits Syst*. **36** (1986) 533-541
18. Golub, G. H., Loan, C. F. van.: *Matrix Computations*, 3rd Ed.. Johns Hopkins University Press, Baltimore (1996)

19. Farhi, E., Goldstone, J., Gutmann, S., Lapan, J., Lundgren, A. and Preda, D.: A Quantum Adiabatic Evolution Algorithm Applied to Random Instances of an NP-Complete Problem. *Science*. 292 (2001) 472-475
20. Messiah, A.: *Quantum Mechanics*. Dover, New York (1999)
21. Hopfield, J. J.: Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the USA*. 79 8 (1982) 2554-2558
22. Anguita, D., Ridella, S., Rivieccio, F., and Zunino, R.: Training Support Vector Machines: a Quantum- Computing Perspective. In *Proc. IEEE IJCNN (2003)* 1587-1592
23. Ono, I. and Kobayashi, S.: A Real-coded Genetic Algorithm for Function Optimization Using Unimodal Normal Distribution Crossover. In *Proc. 7th Int. Conf. Genetic Algorithms (1997)* 246-253
24. FIBV FOCUS MONTHLY STATISTICS, International Stock Price Index (2005)
25. Ljung, G. M., Box, G. E. P.: On a Measure of Lack of Fit in Time Series Models. *Biometrika*. 65 (1978) 67-72
26. London International Financial Futures and Options Exchange (LIFFE). <http://www.liffe.com/>, 2002

Modulation Classification of Analog and Digital Signals Using Neural Network and Support Vector Machine

Cheol-Sun Park¹ and Dae Young Kim²

¹ EW Lab., Agency for Defense Development, Korea
csun@add.re.kr

² Dept. of Info. Comm. Eng. Chungnam Nat'l Univ., Korea
dykim@cnu.ac.kr

Abstract. Most of the algorithms proposed in the literature deal with the problem of digital modulation classification and consider classic probabilistic or decision tree classifiers. In this paper, we compare and analyze the performance of 2 neural network classifiers and 3 support vector machine classifiers (i.e. 1-v-r type, 1-v-1 type and DAG type multi-class classifier). This paper also deals with the modulation classification problems of classifying both analog and digital modulation signals in military and civilian communications applications. A total of 7 statistical signal features are extracted and used to classify 9 modulation signals. It is known that the existing technology is able to classify reliably (accuracy $\geq 90\%$) only at SNR above 10dB when a large range of modulation types including both digital and analog is being considered. Numerical simulations were conducted to compare performance of classifiers. Results indicated an overall success rate of over 95% at the SNR of 10dB in all classifiers. Especially, it was shown that 3 support vector machine classifiers can achieve the probabilities of correct classification (Pcc) of 96.0%, 97.3% and 97.8% at the SNR of 5dB, respectively.

1 Introduction

An automatic radio signal classifier finds its use in military and civilian communications applications including signal confirmation, interference identification, spectrum monitoring, signal surveillance, electronic warfare, military threat analysis, and electronic counter-counter measure.

Signal recognition is a systematic design challenge which requires hierarchical signal processing from radio frequency (RF) to baseband in order to obtain comprehensive knowledge from the carrier to the information bit stream. Unlike conventional radios, the cognitive radios (CR) approach requires the receiver to be aware of its radio environment. In these applications, firstly the modulation type of radio signals must be automatically identified. The challenge is in the design of a universal receiver that can recognize various modulated waveforms with distinct properties [1].

Research on modulation classification (MC) has been carried out for at least two decades. Most of the algorithms proposed in the literature deal with the problem of digital MC [2], and a survey of such classification techniques is presented in [3]. Most of their schemes consider classic probabilistic or decision tree classifiers [4]. None of these approaches have been proven to work reliably with signals that have low SNR (below 10dB), or when a large range of modulation types including both digital and analog is being considered [4].

The aim of this paper is to recognize simultaneously analog and digital modulation types, although there is a tendency to move towards using digital modulation schemes, there are currently still analogue methods (i.e., legacy radios) in use [2]. It is also possible to extract the message from an analogue signal once it has been demodulated, which is more useful than the coded message which is received from a demodulated digital signal.

In this paper, we compare and analyze the performances of neural network classifiers (NNCs) and support vector machine classifiers (SVCs) with 7 features for 9 types of modulation signals. The performances of these classifiers are showed through numerical simulations. This paper showed that support vector machine classifiers can provide a robust approach to signal classification with SNR level as low as 5dB.

2 Neural Network Classifier

The neural network classifier (NNC) used in modulation classification (MC) is a feed-forward network commonly referred to as a multi-layer perceptron (MLP). In NNC, the threshold at each node (i.e., neuron) is chosen automatically and adaptively. In the NNC, all the key features are evaluated simultaneously. So, the time order of the key features does not affect on the probability of correct classification (Pcc) of on the modulation type of a signal.

The proposed structure of MLP in NNC is based on structures given in [5]. However a number of changes have been made in terms of network size, activation function, training algorithm and training data in order to improve the performance as well as reduce the complexity of the network.

In this paper, the NNC has two hidden layers and has 15-15-9 network structure as shown in Fig. 1. The two hidden layers of the MLP use the nonlinear log-sigmoid function. This approach contrasts with the log-sigmoid first hidden layer and linear second layer used in [5]. The output layer uses the linear activation function. Back-propagation is the learning algorithm used in this paper to train the MLP network using algorithm called Levenberg-marquardt (LM), which is one of the fastest training algorithms. The seven feature vectors were used as input vectors to the MLP network in NNC.

The selection of the network parameters is based on choosing the structure that gives the minimum sum square error and the maximum Pcc. In our NNC, we do not optimize the number of hidden layer nodes and do not use network hierarchical (tree) classification scheme for enhance the Pcc.

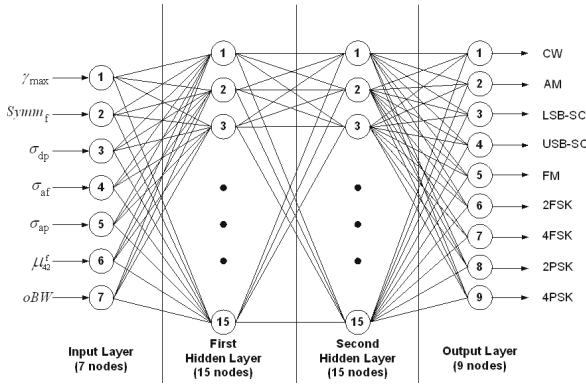


Fig. 1. Double hidden layer architecture (15-15-9) in neural network classifier

Key Features. The key features to be used for modulation classification must be selected so they are sensitive to the modulation types of interest. As shown in the Fig. 1, a set of seven features are commonly used in the evaluated other classifiers for classifying 9 modulated signals. The 6 key features that were used in [5,6] are γ_{max} , $symm_f$ (P), σ_{dp} , σ_{ap} , σ_{af} , and μ_{42}^f . The 1 key feature that were used in [7] is Occupied bandwidth (oBW), which is defined as ratio of the number of bins with the 90% of total PSD.

3 Support Vector Machine Classifier

Support Vector Machine (SVM) is known as a powerful method for pattern recognition and is the state-of-the-art for the existing classification methods. It has been reported that SVM can perform quite well in many pattern recognition problems. The SVM is basically a two-class classifier based on the ideas of large margin and mapping data into a higher dimensional space, and the kernel functions in the SVM.

The first objective of the Support Vector Machine Classifier (SVC) is the maximization of the margin between the two nearest data points belonging to two separate classes. The second objective is to constraint that all data points belong to the right class. It is a two-class solution which can use multi-dimensions features. The two objectives of the SVC problem are then incorporated into an optimization problem.

Since SVM is a binary classifier, the problem of multi-classification, especially for systems like SVM, does not present an easy solution [8]. The most typical method for multi-class problem is to classify one class from the other classes (refer 1-v-r), another typical method is to combine all possible two-class (pair wise) classifiers (refer 1-v-1). Platt et al. proposed DAG [8], which uses directed acyclic graph (DAG) to reduce the number of SVM that need to be used during the testing phase. In testing phase of N classes, 1-v-1 method conducts

$N(N-1)/2$ classifications as shown in Fig. 2a, while the DAG method reduces to $N-1$ classifications as shown in Fig. 2b.

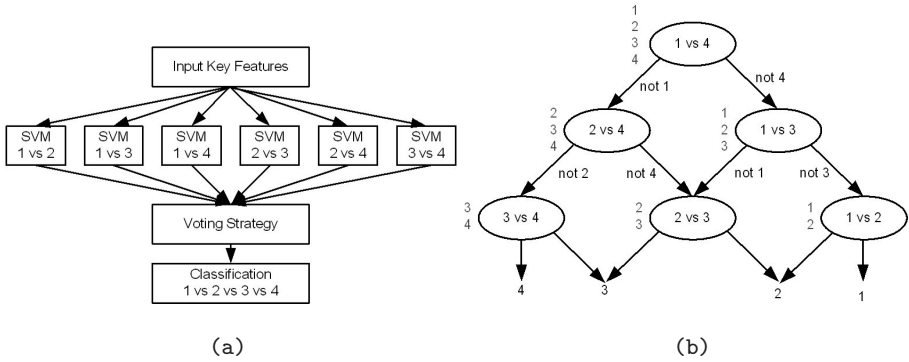


Fig. 2. Illustration of techniques for finding the best class out of 4-class modulation classification using SVM approaches (a) 1-v-1 method, and (b) directed acyclic graph (DAG) method

4 Performance Evaluations

In this section, in order to evaluate the reliability and robustness of the classifiers developed, the numerical simulations are performed for all type of modulation signals of interest. Existing technology is able to classify reliably (accuracy $\geq 90\%$) only at SNR above 10dB when a large range of modulation types including both digital and analog is being considered [4].

For performance comparison purpose, we developed the 2 NNCs without (refer NNC-1) and with (refer NNC-2) normalization using standard deviation instead peak value and the 3 SVCs according to the multi-class scheme, i.e., using 1-v-r method (refer SVC-1), 1-v-1 method (refer SVC-2) and DAG method (refer SVC-3).

To distinguish 9 modulation types, simulation runs were carried out with 4,096 samples at SNR ranging from 0 dB to 30 dB. The Pcc (Probability of correct classification) or Pe (Probability of classification error) obtained from 200 runs at each SNR are plotted in Fig. 3 and Fig. 4 for an AWGN condition. In Fig. 3a, the performance of NNC-2 showed Pcc of 9 modulated signals at SNR from 0 to 30dB. In Fig. 3b, the plot showed overall Pe of NNC-1 and NNC-2 under the same conditions. It was shown that Pcc of NNC-2, which was normalized using standard deviation for each class, much improved for a SNR of over 5dB. In Fig. 4a, the result indicated the Pe of 5 classifiers. The differences of performance of each SVC are shown clearly as in Fig. 4b. It was shown that the Pcc of SVC-2 and SVC-3 were almost same. Especially, it was shown that SVCs can achieve the Pcc of 95% for a SNR of over 5dB (see Fig. 4). The detailed results of DTC-2 and SVC-3 at the SNR of 7dB also are provided in the confusion matrix shown in Table 1-2.

Table 2. Confusion Matrix (SVC-3, Pcc=98.28%)

Actual Modulation	Estimated Modulation Type @SNR = 7dB								
	CW	AM	LSB	USB	FM	2FSK	4FSK	2PSK	4PSK
CW	100								
AM		99.5	0.5						
LSB			100						
USB				100					
FM				0.5	97.5		2.0		
2FSK				0.5		99.5			
4FSK				0.5	1.0	1.0	97.5		
2PSK								91.0	9.0
4PSK								0.5	99.5

5 Conclusion

Most of the algorithms proposed in the literature deal with the problem of digital modulation classification. In this paper, the neural network classifiers and support vector machine classifiers classifying algorithms to simultaneously recognize different analog and digital modulated signals were presented. It is known that the existing technology is able to classify reliably (accuracy $\geq 90\%$) only at SNR above 10dB when a large range of modulation types including both digital and analog is being considered. Numerical simulations were conducted to compare performance of classifiers. Results indicated an overall success rate of over 95% at the SNR of 10dB in all classifiers. Especially, it was shown that 3 support vector machine classifiers (i.e., SVC-1, SVC-2, and SVC-3) can achieve the probabilities of correct classification (Pcc) of 96.0%, 97.3% and 97.8% at the SNR of 5dB, respectively.

References

1. Le, B., et al.: Modulation Identification using Neural Network for Cognitive Radios. SDR forum technical conference, (2005)
2. Dobre, O.A., et al.: The Classification of Joint Analog and Digital Modulations. IEEE MILCOM, (2005) 3010–3015
3. Dobre, O.A., Abdi, A., Bar-Ness, Y., Su, W.: A survey of Automatic Modulation Classification Techniques: Classical Approaches and New Trends. IEE Proc. Communication, (1996)
4. Kremer, S.C., Sheils, J.: A Testbed for Automatic Modulation Recognition using Artificial Neural Networks. IEEE Canadian Conference on Electrical and Computer Engineering, (1997) 67–70
5. Nandi, A.K., Azzouz, E.E.: Algorithm for Automatic Modulation Recognition of Communication Signals. IEEE Trans. Communications, **46** (4) (1998) 431–436
6. Azzouz, E.E., Nandi, A.K.: Procedure for Automatic Recognition of Analogue and Digital Modulations. IEE Proc. Communications **143** (5) (1996) 259–266
7. Park, C.S., Kim, D.Y.: A Novel Robust Feature of Modulation Classification for Reconfigurable Software Radio. IEEE Trans. Consumer Electronics **52** (4) (2006)
8. Lingras, P., Buts, C.: Interval Set Representation of 1-v-r Support Vector Machine Multi-classifiers. IEEE Conf. Granular Computing, (2006) 193–198

A Facial Expression Recognition Approach Based on Novel Support Vector Machine Tree

Qinzheng Xu¹, Pinzheng Zhang², Luxi Yang¹, Wenjiang Pei¹, and Zhenya He¹

¹ School of Information Science and Engineering, Southeast University,
Nanjing, 210096, China

² School of Computer Science and Engineering, Southeast University,
Nanjing, 210096, China

{summer, luckzpz}@seu.edu.cn

Abstract. Automatic facial expression recognition is the kernel part of emotional information processing. This paper dedicates to develop an automatic facial expression recognition approach based on a novel support vector machine tree, which performs feature selection at each internal node, to improve recognition accuracy and robustness. After the Pseudo-Zernike moment features were extracted, they were used to train a support vector machine tree for automatic recognition. The structure of a support vector machine enables the model to divide the facial recognition problem into sub-problems according to the teacher signals, so that it can solve the sub-problems in decreased complexity in different tree levels. In the training phase, those sub-samples assigned to two internal sibling nodes perform decreasing confusion cross, thus, the generalization ability for recognition of facial expression is enhanced. The compared results on Cohn-Kanade facial expression database also show that the proposed approach appeared higher recognition accuracy and robustness than other approaches.

1 Introduction

Human facial expression contains important individual emotional and psychic information for human computer interface. Automatic Facial expression recognition has received great attention in many research fields such as emotion analysis, psychology research, image understanding, image retrieval, with the development of computer technique and its popular application [1]. Ekman and Friesen defined six basic emotions: happiness, sadness, fear, disgust, surprise, and anger (See fig.1) [2]. Most of the current automatic facial expression recognition systems are founded on the psychologic hypothesis of the six basic facial expressions.

Support vector machine (SVM) based approaches have been widely applied in pattern recognition and function fitting. The predominant performance of SVM has been studied and validated in both theory and experiment [3]. Currently, we addressed the problems associated with complex pattern recognition and presented a confusion-crossed Support Vector Machine tree (CSVMT) [4]. A CSVMT is a binary decision tree with SVMs embedded in internal nodes. Those patterns assigned to two internal sibling nodes perform confusion cross. It is developed to achieve a better performance

for complex distribution problems with lower dependence on the two parameters of SVM and better robustness on unbalanced classification problems. One problem remained is that the trained internal nodes may be high complex for those high-dimensional feature space problems due to undesirable complexity added to the underlying probability distribution of the concept label for learning algorithm to capture. A feature selection based CSVMT (FS-SCVMT) learning approach, in which the input space for each internal node is adaptively dimensionality reduced by sensitivity based feature selection, were studied in our recent work [11]. In this paper we introduce FS-CSVMT in facial expression recognition phase.

The rest of the paper is organized as follows. The Pseudo-Zernike moments based facial expression feature extraction approach is illustrated in section 2. FS-CSVMT learning approach is introduced in section 3. Experimental results and discussions are described in section 4. Eventually, conclusions are made in section 5.



Fig. 1. Six basic emotions: (a) Surprise (b) Sadness (c) Fear (d) Anger (e) Disgust (f) Happiness

2 Pseudo-Zernike Moments Based Facial Expression Feature Extraction

Facial expression recognition deals with the classification of facial motion and facial feature deformation into abstract classes [6], and thus facial feature extraction and learning of these features are of great importance for automatic facial expression. A number of developed feature extraction approaches are grouped into two types: shape-based features and image-based features. The shape-based approaches describe the facial features on the variance of parameters of face models [7]. These approaches usually require robust face tracking, hence high computation cost [1]. The imaged-based approaches express features on the pixel intensities of the whole face image or certain

regions of the face image. These approaches are popular researched and applied. Here, we introduce Pseudo-Zernike moments to extract imaged-based features for its denoising ability. Moments are used to depict the distribution of random variables in statistics. The images can be treated as two-dimensional or three-dimensional density distribution functions. In this manner, moments are introduced in image analysis.

Pseudo-Zernike moment is defined by the orthogonal complex-value polynomials in the unit circle. The Pseudo-Zernike moment of two-dimensional image $f(\rho, \theta)$ is defined as [8]:

$$C_{nm} = \frac{n+1}{\pi} \int_0^{2\pi} \int_0^1 f(\rho, \theta) W_{nm}^*(\rho, \theta) \rho d\rho d\theta, \tag{1}$$

where $|m| \leq n$ are the orders of Pseudo-Zernike polynomial. Pseudo-Zernike moment at small values of m and n represents the global image information. Correspondingly, Pseudo-Zernike moment denotes the detailed image information when m and n are large. $W_{nm}^*(\rho, \theta)$ is the conjugate of basis function of Pseudo-Zernike moment. It is defined as:

$$W_{nm}(\rho, \theta) = S_{nm}(\rho) e^{jm\theta}, \tag{2}$$

where $S_{nm}(\rho)$ is Pseudo-Zernike polynomial:

$$S_{nm}(\rho) = \sum_{s=0}^{n-|m|} \frac{(-1)^s (2n-s+1)! \rho^{n-s}}{s!(n-|m|-s)!(n-|m|-s+1)!}. \tag{3}$$

To improve the computation efficiency, the recursive formulas of Pseudo-Zernike moment are applied for fast computation of discretized Pseudo-Zernike moment values [9].

3 Feature Selection Based Support Vector Machine Tree

3.1 Confusion Cross

The construction of a CSVMT model is actually a process of applying the divide-and-conquer idea to solve tough problems. Let SVM_j and SVM_{j+1} be the two internal sibling nodes derived from the common parent node SVM_p , and S_p be the training set assigned to node p . Consider the spacing variable on the trained SVM at node p : $\gamma_p(x) = \sum_{i=l_1}^{l_N} \alpha_i y_i K(x_i, x) + b$, where α_i are the Lagrange multipliers, $x_i, i \in \{l_1, \dots, l_N\} \subseteq \{1, \dots, l\}$ are the support vectors. The confused set is defined as those examples, which are close to the decision hyperplane and accordingly more likely to be misclassified as depicted by $S_c = \{x | x \in S_p, |\gamma_p(x)| \leq C_0 \bar{\gamma}_p\}$, where $\bar{\gamma}_p = (1/|S_p|)$, $\sum_{x_i \in S_p} \gamma_p(x_i)$ and $0 < C_0 < 1$ is a small valued positive number. Instead of partitioning the training examples to node j and $j+1$ by the symbol function $\text{sign}(\gamma_p)$, the confusion cross

process is presented to implement decreasing cross between the two training subsets partitioned by $\text{sign}(\gamma_p)$ to keep those confused patterns in both training subsets of the two internal sibling nodes. The reassignment process is controlled by confusion cross factor

$$C_{\gamma_p, m} = \rho_0 \exp(-\lambda \cdot m) \bar{\gamma}_p, \tag{4}$$

where m is the node level of internal nodes j and $j+1$, $\rho_0 \in [0,1]$ is the initial confusion cross rate, and λ controls the convergence of the cross and terminates the cross process at the deep tree levels. When confusion cross is performed, the subset of training examples S_p reassigned to those two child nodes j and $j+1$ are $S_j = \{x | x \in S_p, \gamma_p(x) \geq -C_{\gamma_p, m}\}$ and $S_{j+1} = \{x | x \in S_p, \gamma_p(x) \leq C_{\gamma_p, m}\}$. In this way, the set of crossed examples $S_{cm} = S_j \cap S_{j+1} = \{x | x \in S_p, |\gamma_p(x)| \leq C_{\gamma_p, m}\}$, which are close to the decision hyperplane and accordingly more likely to be misclassified, are kept in both two child nodes for further construction of decision hyperplane at a fine node level, i.e., those confused examples are made a validation of their contribution to the fine decision hyperplane.

The property of the tree-structure approach allows the models to divide the problems in different levels according to teacher signals constructed by the heuristic approach and then conquer the sub-problems with decreased complexity. Further, the classification accuracy of an SVM with Gaussian kernel $k(x_i, x_j) = \exp[-\|x_i - x_j\| / (2\sigma^2)]$ is dependent on the kernel width σ and the penalty parameter. Inappropriately selected values of these two parameters may cause overfitting or underfitting problems. Some approaches, such as cross-validation approach [10], were developed to solve this problem. CSVMT can achieve better performance for complex distribution problems with lower dependence on the two parameters than single SVM model [4].

3.2 Feature Selection Based CSVMT

Unnecessary features add undesirable complexity to the underlying probability distribution of the concept label for learning algorithm to capture, in which case the learning period is increased and those learned internal node may be high complex. A CSVMT model with high complex internal nodes is likely to fall into a depressed test efficiency and performance. A feature selection based CSVMT (FS-SCVMT) learning approach is proposed to address this problem [11]. Since the generalization performance of an SVM is deeply related to its margin, the degree of influence of a feature on an SVM is computed by the sensitivity of the margin of the SVM to the feature [12], i.e., the derivative based sensitivity to feature k is defined as

$$D_k(w^2) = \sum_i \left| \sum_j \alpha_i \alpha_j y_i y_j \frac{\partial K(\bar{x}_i, \bar{x}_j)}{\partial \bar{x}_j^k} \right|. \tag{5}$$

In the tree-structured learning approach, subsets are assigned to internal nodes to build local decision hyperplans. The contribution degree of a feature for local decision

is variable at different nodes. We accordingly credit feature k by derivative based feature sensitivity given by eq. (2) for each node split instead of for the overall information proposed in [12]. The input features for current internal node are adaptively selected according to the measure of feature selection ratio computed by $Sr = \sum_{k=1}^n D_k / \sum_{k=1}^N D_k$, where $\{D_k\}_{k=1,2,\dots,N}$ are the feature credits sorted in descending order, and N is the feature dimension. The first n features with larger credits are selected for current node if Sr reaches a given threshold. The construction process of FS-CSVMT for binary complex problems is sketched in Table 1. A simplified heuristic method was introduced to extend the FS-CSVMT for binary complex classification problems to a multi-classification one [4]. It defines the teacher signals for the data assigned to internal nodes.

Table 1. Construction process of FS-CSVMT

FSCSVMT($S, C_{\gamma_p, m}, Sr^*$)

Input:

training set S ;
 confusion cross factor $C_{\gamma_p, m}$;
 threshold of feature selection ratio Sr^* .

Output:

FS-CSVMT for binary classification.

Procedure:

Initialize($T, S, C_{\gamma_p, m}, Sr^*$) ;

if *BuildLeaf*(S) = *TRUE*
 return(T) ;

%Build a leaf node if training patterns
 belong to one class.

else *currentSVM* = *TrainSVM*(S) ;

%Train an SVM for current subsample

$S_{selected} = FS(\text{currentSVM}, S, Sr^*)$

%Build feature space on selected features
 for current node

currentNode = *TrainNode*($S_{selected}$)

%Build current internal node

 [*leftS*, *rightS*]

 = *Cross*(*currentNode*, $S, C_{\gamma_p, m}$) ;

%Confusion cross is performed.

$C_{\gamma_{p_son}, m+1} = \text{DecreaseOn}(C_{\gamma_p, m})$;

%Decrease confusion cross factor as depicted
 in Eq. (4)

FSCSVMT(*leftS*, $C_{\gamma_{p_son}, m+1}, Sr^*$) ;

%Construct left sub-tree of FS-CSVMT

FSCSVMT(*rightS*, $C_{\gamma_{p_son}, m+1}, Sr^*$) ;

%Construct right sub-tree of FS- CSVMT

4 Experimental Results

In this section we reported the experimental results on Cohn-Kanade facial expression database [13]. It includes facial expressions of age 18 to 30 of different race. The data has been referenced many times in different facial expression research work. 66 facial features were extracted for the order $n \leq 11$ of Pseudo-Zernike polynomial. 1427 facial expression patterns were included in the experiment, i.e., 460 for surprise, 464 for

happiness, 156 for sadness, 127 for anger, 126 for disgust and 94 for fear. 1/2 of the patterns were for training and the rest for testing. The initial confusion cross rate $\rho_0 = 1.0$, and $\lambda = 0.3$. The numerical results were the average of 8 runs of the recognition process. The test recognition accuracy of different recognition approaches trained on Pseudo-Zernike moment facial features were listed in tab. 2. The compared approaches included three SVM modes studied in Hsu and Lin's work [14] (i.e. DAGSVM, 1-V-1 SVM and 1-V-R SVM), SVMT introduced in [5], linear discriminant analysis, and k-nearest neighbors.

Table 2. Facial expression recognition accuracy based on CSCMT

Methods \ Accuracy	DAG SVM	1-V-1 SVM	1-V-R SVM	SVMT	LDA	KNN	CSVMT
Surprise(%)	98.86	99.42	97.68	97.39	90.82	92.46	96.96
Happiness(%)	99.65	100.00	98.85	100.00	91.77	95.61	97.74
Sadness(%)	90.00	88.32	64.10	90.17	75.06	72.27	94.87
Anger(%)	89.52	86.77	76.04	92.59	70.52	75.01	91.27
Disgust(%)	84.44	87.89	75.26	83.60	66.11	60.99	91.27
Fear(%)	82.97	85.82	71.01	82.98	76.02	66.45	91.49
Total(%)	95.01	95.36	88.76	94.86	84.45	85.21	95.62

It was observed in the experiments that the training recognition accuracy of DAGSVM, 1-V-1 SVM, 1-V-R SVM, SVMT, and FS-CSVMT reached 100.0%. The average number of features selected at each internal node for FS-CSVMT is 56.50 instead of 66 for other learning approaches. Tab. 2 showed that the FS-CSVMT based facial expression recognition approach achieved relative better test recognition accuracy than other approaches in total. Further, the proposed approach trained on the unbalanced training samples reached not only high recognition accuracy for large sample facial expressions, but also better accuracy than other approaches for small sample facial expressions, such as sadness, anger, disgust and fear. It indicated that the proposed approach might achieve better generalization ability and higher recognition robustness on unbalanced facial expression recognition problems.

We also compared the experimental results with that of other approaches adopted the Cohn-Kanade facial expression database on six basic emotions: AdaBoost approach was introduced in the feature extraction phase and a set of binary SVMs were used to recognize 6 basic facial expressions in [15], and its best recognition accuracy was 92.9%; A multistream hidden Markov based recognition model was presented in [1], which reached recognition accuracy of 93.66%; The Kanade-Tucas-Tomasi feature track approach was applied to compute the distances and angles between feature points in extraction phase and the expressions were classified by a rough-set based approach in [16], and the recognition accuracy reached 79.00%. The compared results showed that the proposed approach appeared higher recognition accuracy than the other approaches.

5 Conclusions

This paper dedicates to develop an automatic facial expression recognition approach based on FS-CSVMT learning approach to improve recognition accuracy and robustness. After the Pseudo-Zernike moment features were extracted, they were used to train a FS-CSVMT for automatic recognition. The structure of FS-CSVMT enables the model to divide the facial recognition problem into sub-problems according to the teacher signals, so that it can solve the sub-problems in decreased complexity in different tree levels. In the training phase, the input space for each internal node is adaptively dimensionality reduced by sensitivity based feature selection; those sub-samples assigned to two internal sibling nodes perform decreasing confusion cross, thus, the generalization ability of FS-CSVMT for recognition of facial expression is enhanced. The experiments are conducted on Cohn-Kanade facial expression database. Competitive recognition accuracy 95.62% is achieved by the FS-CSVMT based approach. The compared results on Cohn-Kanade facial expression database also show that the proposed approach appeared higher recognition accuracy and robustness than other approaches.

Acknowledgement

This work was supported by a Scientific Research Starting Foundation of Southeast University (4004001041) and a Foundation of Excellent Doctoral Dissertation of Southeast University (YBJJ0412).

References

1. Aleksic, P.S., Katsaggelos, A.K.: Automatic Facial Expression Recognition using Facial Animation Parameters and Multistream HMMs. *IEEE Trans. Information Forensics and Security* **1** (2006) 3–11
2. Ekman, P., Friesen, W.V.: *Facial Action Coding System: A technique for the Measurement of Facial Movement*. Palo Alto, CA: Consulting Psychologists Press (1978)
3. Vapnik, V.N.: An Overview of Statistical Learning Theory. *IEEE Trans. Neural Networks* **10** (1999) 988–999
4. Xu, Q.Z., Song, A.G., Pei, W.J., Yang, L.X., He, Z.Y.: Tree-Structured Support Vector Machine with Confusion Cross for Complex Pattern Recognition Problems. In: *Proceedings of 2005 IEEE International Workshop on VLSI Design and Video Technology* (2005) 195–198
5. Pang, S.N., Kim, D.J., Bang, S.Y.: Face Membership Authentication using SVM Classification Tree Generated by Membership based LLE Data Partition. *IEEE Trans. Neural networks* **16** (2005) 436–446
6. Fasel, B., Luetttin, J: Automatic Facial Expression Analysis: A Survey. *Pattern Recognition* **36** (2003) 259–275
7. Pardàs, M., Bonafonte, A.: Facial Animation Parameters Extraction and Expression Detection using HMM. *Signal Processing: Image Communication* **17** (2002) 675–688
8. Teh, C.H., Chin, R.T.: On Image Analysis by the Methods of Moments. *IEEE Trans. Pattern Analysis and Machine Intelligence* **10** (1988) 496–513

9. Chong, C.W., Raveendran, P., Mukundan, R.: An Efficient Algorithm for Fast Computation of Pseudo-Zernike Moments. *Int. J. Pattern Recognition and Artificial Intelligence* **17** (2003) 1011–1023
10. Foody, G. M., Mathur, A.: A Relative Evaluation of Multiclass Image Classification by Vector Machines. *IEEE Trans. Geoscience and Remote Sensing* **42** (2004) 1335–1343
11. Xu, Q.Z., Pei, W.J., Yang, L.X., He, Z.Y.: Support Vector Machine Tree Based on Feature Selection. 13th International Conference on Neural Information Processing, Part I, LNCS **4232** (2005) 856–863
12. 12.Sindhvani, V., Rakshit, S., Deodhare, D., Erdogmus, D., Principe, J. C., Nivogi P.: Feature Selection in MLPs and SVMs based on Maximum Output Information. *IEEE Trans. Neural Networks* **15** (2004) 937–948
13. Kanade, T., Cohn, J.F., Tian, Y.I.: Comprehensive Database for Facial Expression Analysis. Proc of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition (2000) 46–53
14. Hsu, C.W., Lin, C.J.: A Comparison of Methods for Multiclass Support Vector Machines. *IEEE Trans. Neural Networks* **13** (2002) 415–525
15. Littlewort, G., Bartlett, M., Fasel, I., Susskind, J., Movellan, J.R.: Dynamics of Facial Expression Extracted Automatically from Eideo. Proc. Of IEEE Conference on Computer Vision and Pattern Recognition, Workshop on Face Processing in Video, Orlando, FL **5** (2004) 80
16. Chibelushi, C.C., Bourel, F.: Hierarchical Multistream Recognition of Facial Expressions. *IEE Proceedings on Vision, Image and Signal Processing* **151** (2004) 307–313

Universal Steganalysis Using Multiwavelet Higher-Order Statistics and Support Vector Machines

San-ping Li¹, Yu-sen Zhang¹, Chun-hua Li², and Feng Zhao¹

¹ Institute of Command Automation, PLA University of Sci. and Tech, Nanjing 210007, China

sandeli1273@163.com

² College of Computer Sci. and Tech., Huazhong University of Sci. and Tech., Wuhan, 430074, China

li.chunhua@163.com

Abstract. In this paper, a new universal steganalysis algorithm based on multiwavelet higher-order statistics and Support Vector Machines(SVM) is proposed. We follow the philosophy introduced in Ref[7] in which the features are calculated from the stego image's noise component in the wavelet domain. Instead of working in wavelet domain, we calculate the features in multiwavelet domain. We call this Multiwavelet Higher-Order Statistics (MHOS) feature. A nonlinear SVM classifier is then trained on a database of images to construct a universal steganalyzer. The comparison to the current state-of-the-art universal steganalyzers, which was performed on the same image databases under the same testing conditions, indicates that the proposed universal steganalysis offers improved performance.

1 Introduction

It is not surprising that with the emergence of steganography, that the development of a counter-technology, steganalysis, has also emerged (see[1]for a review). The goal of steganalysis is to determine if an image (or other carrier) contains an embedded message. Current steganalysis methods fall broadly into one of two categories: embedding specific or universal. While universal steganalysis attempts to detect the presence of an embedded message independent of the embedding algorithm and, ideally, the image format, embedding specific approaches to steganalysis take advantage of particular algorithmic details of the embedding algorithm. Given the ever growing number of steganography tools, universal approaches are clearly necessary in order to perform any type of generic, large-scale steganalysis.

The concept of universal steganalysis appeared for the first time in the work of Avcibas et al.[2]. Farid et al.[3][4] proposed a 72-dimensional feature space (for grayscale images) consisting of the first four statistical moments of wavelet coefficients and their prediction errors. Harmsen et al.[5] used a simple three-dimensional feature vector obtained as the center of gravity of the three-dimensional Histogram Characteristic Function (HCF). While this method gives good results for steganalysis of color images with a low noise level, such as previously compressed

JPEG images, its performance is markedly worse for grayscale images and raw, never compressed images from digital cameras or scanners. Ker[6] substantially improved this method by introducing the concept of calibration. Holotyak et al.[7] used an approach similar to Farid's except they calculate the features from the noise component of the image in the wavelet domain. The authors also advocate usage of high statistical moments and show that a substantial benefit can be obtained by considering higher order moments. Instead of working with very high order normalized even moments of the noise residual as in [7], M. Goljan et al. [8] used absolute non-normalized moments of order 1 to 9. A closer look reveals that the first four moments are conceptually the same as the prediction errors used by Farid. I. Avcibaş et al.[9] present a novel technique for steganalysis of images that have been subjected to embedding by steganographic algorithms. The basic idea is that, the correlation between the bit planes as well the binary texture characteristics within the bit planes will differ between a stego-image and a cover-image. These telltale marks are used to construct a classifier that can distinguish between stego and cover images. Xuan et al.[10] proposed features calculated as the first three absolute moments of the HCF of all 9 three level subbands in a Haar decomposition. Almost all above method work in the wavelet domain for the feature extracting.

Recently, multi-wavelets have been introduced as a more powerful multi-scale analysis tool. A scalar wavelet system is based on a single scaling function and mother wavelet. On the other hand, a multi-wavelet uses several scaling functions and mother wavelets[11], [12]. This adds several degrees of freedom in multi-wavelet design and makes it possible to have several useful properties such as symmetry, orthogonality, short support, and a higher number of vanishing moments simultaneously. The usefulness of these properties is well known in wavelet design. Symmetric property allows symmetric extension when dealing with the image boundaries. This prevents discontinuity at the boundaries and therefore a loss of information in these points would be prevented. Orthogonality generates independent sub-images. A higher number of vanishing moments result in a system capable of representing high-degree polynomials with a small number of terms. For example, in a wavelet system with two vanishing moments, locally constant and locally linear functions reside in v_1 (the first level of multiresolution approximation). In particular, this property is very useful in image processing since images can be best described by locally constant and locally linear functions.

Since multi-wavelet is basically a multi-filter, it needs several streams of input rather than one. This necessitates a prefiltering operation on the input stream to produce the required multiple streams. This prefiltering operation is also called multi-wavelet initialization and can be performed in a critically sampled or an over-sampled fashion [11]. Although a unified framework for application of multi-wavelet to image processing has not been developed yet, all the stated useful properties offer a great potential for applying multiwavelet to image-processing applications.

In this paper, a new universal steganalysis algorithm based on multiwavelet higher-order statistics and Support Vector Machines(SVM) is proposed. We follow the philosophy introduced in [7] in which the features are calculated from the stego image's noise component in the wavelet domain. Instead of working in wavelet domain, we calculate the features in multiwavelet domain. We call this Multiwavelet

Higher-Order Statistics (MHOS) feature. A nonlinear SVM classifier is then trained on a database of images to construct a universal steganalyzer. The performance of this method is compared to the current state-of-the-art universal steganalyzers.

In Section 2, we describe the MHOS features proposed for steganalysis. In Section 3, We give some overviews SVM classifier. Then the performance of proposed method in this paper is compared to the current state-of-the-art universal steganalyzers on exactly the same databases in Section 4. Finally, conclusion is drawn in Section 5.

2 Multiwavelet Higher Order Statistics Feature Extraction

In this section, we focus on the proposed MHOS features vector calculated in multiwavelet domain.

2.1 Steganalysis as a Task of Pattern Recognition

Based on whether an image contains hidden message, images can be classified into two classes: the image with no hidden message and the corresponding stego-image (the same image with message hidden in it). Steganalysis can thus be considered as a task of pattern recognition to decide which class a test image belongs to. The key issue for steganalysis just like for pattern recognition is feature selection. The features should be sensitive to the data hiding process. In other words, the features should be rather different for the image without hidden message and for the corresponding stego-image. The larger the difference, the better the features are. The features should be as general as possible, i.e., they are effective to all different types of images and different data hiding schemes. Almost all prior method work in the wavelet domain for the feature extracting, but the result is unsatisfied yet. Therefore, MHOS feature vectors calculated in multiwavelet domain are proposed. Steganalysis has thus become a pattern classification process in the MHOS feature space.

2.2 Multiwavelet

Unlike a scalar wavelet, a multi-wavelet uses several scaling functions and mother wavelets [12,13]. This adds several degrees of freedom in multi-wavelet design and generates useful properties such as symmetry, orthogonality, short support, and a higher number of vanishing moments simultaneously. The usefulness of these properties is well known in wavelet design.

As mentioned above, in a multi-resolution representation generated by multi-wavelet, we have more than one scaling function. A multi-wavelet system of multiplicity r consists of r scaling functions, ϕ_1, \dots, ϕ_r , written as a vector $\phi = (\phi_1, \dots, \phi_r)^T$, which satisfies the following matrix dilation equation:

$$\phi(x) = \sum_k L_k \sqrt{2} \phi_k(2x - k) \quad (1)$$

where L is a low-pass matrix quadrature mirror filter (QMF) and $\sqrt{2}$ maintains the norm of the r scaling functions with the scale of 2. Corresponding to each scaling function ϕ_i , there is a wavelet ψ_i forming a multi-wavelet vector as $\psi = (\psi_1, \dots, \psi_r)^T$. Multi-wavelet vector satisfies the following wavelet matrix equation:

$$\psi(x) = \sum_k H_k \sqrt{2} \phi_k(2x - k) \tag{2}$$

where H is a high-pass matrix QMF.

Like the scalar wavelet case, the multiwavelet decomposition of a 1-dimensional signal is performed by Mallat algorithm. However, since the lowpass filterbank and highpass filterbank are $r \times r$ matrices in the multiwavelet case, the signal must be preprocessed to be a vector before the multiwavelet decomposition. As for the multiwavelet decomposition of a 2-dimensional image, the 1-dimensional algorithm can be performed in each dimension. After one cascade step, the result can be realized as the fig.1(a)

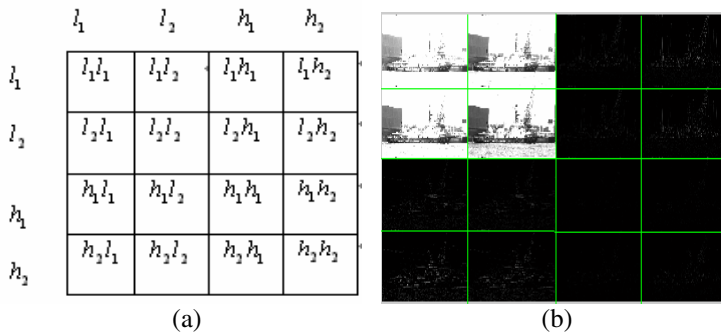


Fig. 1. (a) the multiwavelet decomposition of a 2-dimensional image. (b) a stego image decomposed to the first level using the GHM multiwavelet.

Note that a typical block $l_1 h_2$ contains lowpass coefficients corresponding to the first scaling function in the horizontal direction and highpass coefficients corresponding to the second wavelet in the vertical direction.

Geronimo, Hardin, and Massopust constructed one of the most well-known multi-wavelets, called GHM [14]. GHM scaling functions and multiwavelets are shown in Fig.2. From this figure it can be seen that GHM scaling functions have short support and are symmetric about their centers. Two other important features are the orthogonality of integer translates of scaling functions and an approximation order of two. The usefulness of these features was discussed in the Introduction section. In this paper, we decomposed each image to 16 sub-images using the multi-wavelets. Fig.1 (b) shows the 16 sub-images of a stego image decomposed to the first level using the GHM multi-wavelet. There are 4 lowpass sub-images and 12 highpass sub-images.

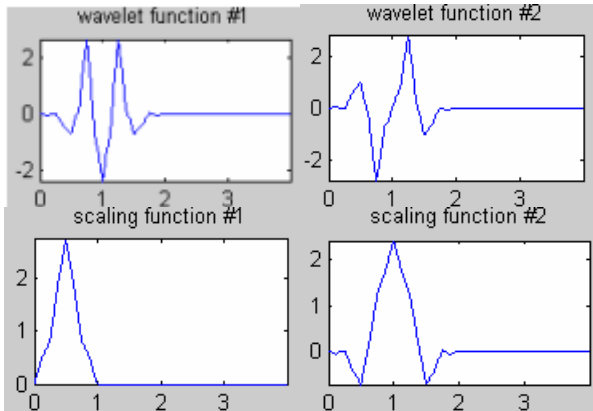


Fig. 2. GHM scaling functions and wavelet functions

2.3 MHOS Feature Extracting

In [7] the authors proposed a new idea to calculate the features for steganalysis only from the noise component of the stego image in the wavelet domain. The noise component was obtained using the denoising filter due to Mihcak et al.[15] We reiterate that the denoising step increases the SNR between the stego signal and the cover image, thus making the features calculated from the noise residual more sensitive to embedding and less sensitive to image content. The denoising filter is designed to remove Gaussian noise from images under the assumption that the stego image is an additive mixture of a non-stationary Gaussian signal (the cover image) and a stationary Gaussian signal with a known variance (the noise). Here We calculate statistical features of the noise residual in the Multiwavelet domain. The procedure for calculating the MHOS features in a gray scale image is shown below.

- Step 1. Calculate the first level multiwavelet decomposition of the stego image with the GMH, Denote the horizontal, vertical, and diagonal subbands of highpass sub-images as $l_m h_n(i, j), h_m l_n(i, j), h_m h_n(i, j), m = 1$ or $2, n = 1$ or 2 . And (i, j) runs through some index set J .
- Step 2. In each subband, estimate the local variance of the cover image for each wavelet coefficient using the MAP estimation for 4 sizes of a square $N \times N$ neighborhood, for $N \in \{3, 5, 7, 9\}$

$$\tilde{\sigma}_N^2(i, j) = \max\left(0, \frac{1}{N^2} \sum_{(i, j) \in N} w^2(i, j) - \sigma_0^2\right), (i, j) \in J. \tag{3}$$

Take the minimum of the 4 variances as the final estimate,

$$\tilde{\sigma}^2(i, j) = \min(\tilde{\sigma}_3^2(i, j), \tilde{\sigma}_5^2(i, j), \tilde{\sigma}_7^2(i, j), \tilde{\sigma}_9^2(i, j)), (i, j) \in J. \tag{4}$$

- Step 3. The denoised wavelet coefficients are obtained using the Wiener filter

$$\tilde{l}_m \tilde{h}_n(i, j) = l_m h_n(i, j) \frac{\tilde{\sigma}^2(i, j)}{\tilde{\sigma}^2(i, j) + \sigma_0^2}, (i, j) \in J, m = 1 \text{ or } 2, n = 1 \text{ or } 2 \quad (5)$$

and similarly for $h_m l_n(i, j), h_m h_n(i, j), (i, j) \in J, m = 1 \text{ or } 2, n = 1 \text{ or } 2$.

Step 4. Calculate the noise residual in each subband

$$r_{l_m h_n}(i, j) = l_m h_n(i, j) - \tilde{l}_m \tilde{h}_n(i, j), \quad (6)$$

and similarly $r_{l_m l_n}$ and $r_{h_m h_n}$ for $h_m l_n(i, j)$ and $h_m h_n(i, j), (i, j) \in J, m = 1 \text{ or } 2, n = 1 \text{ or } 2$.

Step 5. Denoting the mean value with a bar, calculate the absolute central moments of each noise residual for $p = 1, 2, \dots, nmom$

$$m_p = \frac{1}{|J|} \sum_{(i, j) \in J} |r_{l_m h_n}(i, j) - \bar{r}_{l_m h_n}|^p, \quad (7)$$

In this paper, we use $nmom = 9$ and set the parameter $\sigma_0^2 = 0.5$, which is the same value as in[7] and corresponds to the variance of the stego signal for an image fully embedded with ± 1 embedding. So the total number of MHOS features for a grayscale image is $12 \times 9 = 108$ (we only calculate 12 highpass subbands of the first level multiwavelet decomposed). For color images, the features are calculated for each color channel, bringing the total number of MHOS features to $36 \times 9 = 324$.

3 SVM Classifier

SVM is a statistical classification method proposed by Vapnik[16].The main advantage of SVM is that it can serve better in the processing of small-sample learning problems by the replacement of Experiential Risk Minimization by Structural Risk Minimization. Moreover, SVM can treat a nonlinear learning problem as a linear learning problem by mapping the original data to the kernel space in which we only solve the linear problems. Because SVM have these theoretical properties as well as the considerable performance of learning, they become a new research hotspot in recent years and attract more attentions after the research of Artificial Neural Networks. Given a labeled training set:

$$S = \{(x_i, y_i) \mid x_i \in R, y_i \in \{-1, 1\}, i = 1, \dots, m\}, \quad (8)$$

where x_i stands for input vector i and y_i is the desired category, positive or negative, SVM can generate a separation hyperplane H that separates the positive and negative examples. If any point x which lies on the hyperplane must satisfy $w \cdot x + b = 0$,

where w is normal to the hyperplane and b is the bias. Finally, the optimal hyperplane: $H : w_0 \cdot x + b_0 = 0$ can be determined by

$$w_0 = \sum_{i=1}^m \alpha_i y_i x_i, \tag{9}$$

where α_i and b_0 are Lagrange multipliers and bias that determined by SVM’s training algorithm. In Eq. (9), those points x_i with $\alpha_i = 0$ can be ignored and those with $\alpha_i > 0$ are called “support vectors”. After the training of SVM is completed, H is thus determined, then any data x will be classified according to the sign of the decision function. The decision function is defined as:

$$d(x) = \text{sgn}\left(\sum_{i=1}^m \alpha_i y_i K(x_i, x) + b_0\right), \tag{10}$$

where $K(x_i, x)$ is the kernel function which maps the training samples to a higher dimensional feature space as shown in Figure.3.

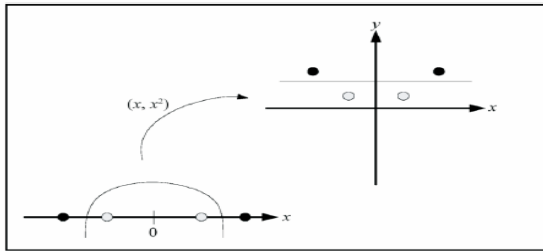


Fig. 3. A mapping function from input space to feature space

Three kinds of kernel functions are commonly adopted in SVM as indicated in Table 1. In general, the RBF network is preferred to train the classifier, because it is more powerful and more efficacious than Polynomial and Two-layer [17]. In this paper, the non-linear SVM (using Radial-basis function network) classify images based on the MHOS feature vector as described in Section 2.

Table 1. Common Kernel Functions

Polynomial learning machine	$K(x_i, x) = ((x_i \cdot x) + 1)^p$
Radial-basis function network	$K(x_i, x) = \exp(-\ x - x_i\ ^2 / 2\sigma^2)$
Two-layer perception	$K(x_i, x) = \tanh(v(x_i \cdot x) + \delta)$

4 Evaluation of the Proposed Steganalysis Method

We now compare the performance of the proposed Steganalysis methods with the results reported for current state-of-the-art classifiers[3,6,7,9]. To make the comparison fair, we always compare on the same image database and the same testing methodology. So we collect the following two classes of image database.

Set #1 consists of 2000 images from [18]. To be able to compare our results to [3,9]. All images were preprocessed as in [9] and converted to grayscale, black borders around them were cropped, and finally the images were recompressed with a quality factor of 75.

Set #2 includes high resolution (1500×2100 pixels) 32 bit CMYK color TIFF images (2375 images) from [19]. To be able to compare our results to [6,7], similar as [6] we converted all color images to grayscale and applied bicubic downsampling.

The receiver operating characteristics (ROCs) for Set #1 and #2 with different classifiers are presented in Fig.4. The embedding method tested was ± 1 embedding also called LSB matching with different embedding capacity. Fig.4.(a), Fig.4.(b) and Fig.4.(c) are the result of Set #1 with proposed classifier, classifier[9] and classifier[3] respectively. The embedding capacity is 0.01, 0.05, 0.1 and 0.15 bits per pixel (bpp) which is the same value as in Ref[9]. Fig.4.(d), Fig.4.(e) and Fig.4.(f) are the result of Set #2 with proposed classifier, classifier[7] and classifier[6] respectively. The embedding capacity is 0.25, 0.5, 0.75 and 1.0 bpp which is the same value as in Ref[7]. From Fig.4, we can see that the performance of proposed classifier is better than classifier[9]'s and classifier[3]'s for Set #1, and also superior than classifier[7]'s and classifier[6]'s for Set #2.

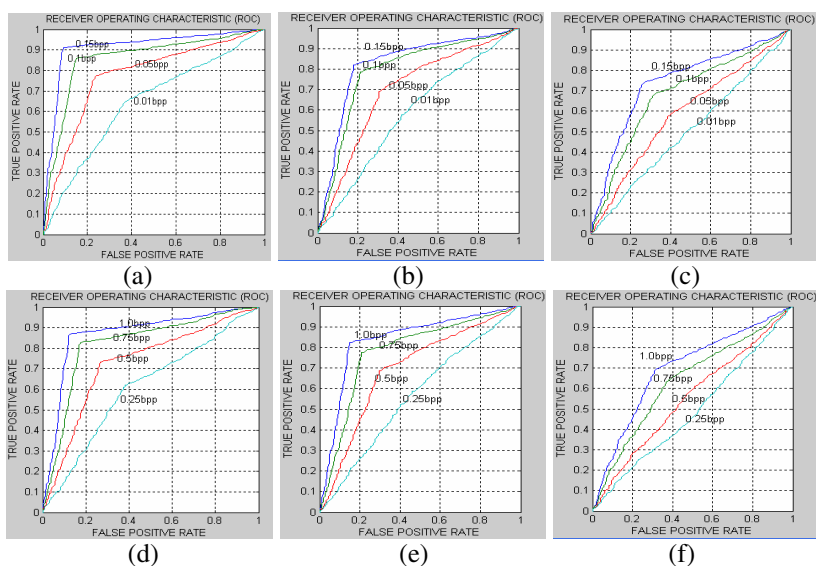


Fig. 4. ROC for Set #1: (a) proposed classifier; (b) classifier[9]; (c) classifier[3]. ROC for Set #2: (d) proposed classifier; (e) classifier[6]; (f) classifier[7].

We also report the false positives for 50% and 80% detection rates (as in [6,7]) in Table 2. The results for the two methods referenced above are taken from [7]. From Table2 we can make direct comparison of performance. The false positives for the proposed classifier for detection rate 50% and 80% were 1.47% and 6.75% compared to about 3.45% and 16.25% reported in [7] and 7% and 27% reported in [6].

Table 2. Percentage of false positives at 50% and 80% true detection rates compared to two published methods

Classifier	false positive rate at 50% detection rate	false positive rate at 80% detection rate
Proposed classifier	1.47	6.75
Classifier[7]	3.45	16.25
Classifier[6]	7	27

5 Conclusions

In this paper, we built a new approach to universal steganalysis based on MHOS features and SVM and compared its performance to four previously proposed universal steganalyzers. The comparison, which was performed on the same image databases under the same testing conditions, indicates that the proposed universal steganalysis offers improved performance.

References

1. Fridrich, J., Goljan, M.: Practical steganalysis: State of the art. SPIE Photonics West, Electronic Imaging, San Jose, CA, (2002)
2. Avcıbaşı, I., Memon, N., Sankur, B.: Steganalysis Using Image Quality Metrics. E. Delp et al. (eds.): Proc. SPIE Electronic Imaging, Security and Watermarking of Multimedia Contents II **4314** (2001) 523-531
3. Farid, H., Lyu, S.: Detecting Hidden Messages Using Higher-Order Statistics and Support Vector Machines. F.A.P. Petitcolas (ed.): 5th International Workshop on Information Hiding, LNCS **2578** (2002) 340-354
4. Lyu, S., Farid, H.: Steganalysis Using Color Wavelet Statistics and One-Class Support Vector Machines. Proc. SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents VI **5306** (2004) 35-45
5. Harmsen, J.J., Pearlman, W.A.: Steganalysis of Additive Noise Modelable Information Hiding," in E. Delp et al. (eds.): Proc. SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents V (2003) 131-142

6. Andrew, D.K.: Steganalysis of LSB Matching in Grayscale Images. *IEEE Signal Processing Letters* **12** (2005) 441-444
7. Holotyak, T., Fridrich, J., Voloshynovskiy, S.: Blind Statistical Steganalysis of Additive Steganography Using Wavelet Higher Order Statistics. 9th IFIP TC-6 TC-11 Conference on Communications and Multimedia Security, LNCS **3677** (2005) 273-274
8. Goljan, M., Holotyak, T.: New Blind Steganalysis and its Implications. *Proc. SPIE Electronic Imaging, Photonics West*, January 2006
9. Avcıbaşı, I., harrazib, M., Memon, N., Sankur, B.: Image Steganalysis with Binary Similarity Measures. *EURASIP JASP* **17** (2005) 2749-2757
10. Xuan, G., Shi, Y.Q., Gao, J., Zou, D., Yang, C., Zhang, Z., Chai, P., Chen, C., Chen, W.: Steganalysis Based on Multiple Features Formed by Statistical Moments of Wavelet Characteristic Functions *Proc. 7th Information Hiding Workshop, Barcelona, Spain, June 6-8, 2005*
11. Strela, V., Heller, P.N., Strang, G., Topiwala, P., Heil, C.: The application of multiwavelet filter banks to image processing. *IEEE Trans. Imag. Process.* **8** (1999) 548-563
12. Shen, L.X., Tan, H.H., Tham, J.Y.: Symmetric-antisymmetric orthogonal multiwavelets and related scalar wavelets. *Appl. Comput. Harmonic Anal.* **8** (2000) 258-279
13. Tham, J.Y., Shen, L.X., Lee, S.L., Tan, H.H.: A general approach for analysis and application of discrete multiwavelet transform. *IEEE Trans. Signal Process* **48** (2000) 457-464
14. Geronimo, J.S., Hardin, D.P., Massopust, P.R.: Fractal functions and wavelet expressions based on several scaling functions. *J. Approx. Theory* **78** (1994) 373-401
15. Mihcak, M.K., Kozintsev, I., Ramchandran, K.: Spatially Adaptive Statistical Modeling of Wavelet Image Coefficients and its Application to Denoising. *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing* **6** (1999) 3253-3256
16. Vapnik, V.: *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995
17. Hsu, C.W., Chang, C.C. Lin, C.J.: *A Practical Guide to Support Vector Classification*. <http://www.Csie.ntu.edu.tw/~cjlin/papers/>
18. Images downloaded from <http://philip.greenspun.com/>
19. NRCS Photo Gallery, <http://photogallery.nrcs.usda.gov>

Tree-Structured Support Vector Machines for Multi-class Classification*

Siyu Xia¹, Jiuxian Li¹, Liangzheng Xia¹, and Chunhua Ju²

¹ School of Automation, Southeast University, Nanjing 210096, China
xia081@gmail.com

² College of Computer Information and Engineering, Zhejiang Gongshang University,
Hangzhou 310035, China

Abstract. In this paper, a non-balanced binary tree is proposed for extending support vector machines (SVM) to multi-class problems. The non-balanced binary tree is constructed based on the prior distribution of samples, which can make the more separable classes separated at the upper node of the binary tree. For an k class problem, this method only needs $k-1$ SVM classifiers in the training phase, while it has less than k binary test when making a decision. Further, this method can avoid the unclassifiable regions that exist in the conventional SVMs. The experimental result indicates that maintaining comparable accuracy, this method is faster than other methods in classification.

1 Introduction

The support vector machine (SVM) [1], rooted in the statistical learning theory, has been successfully applied to pattern recognition problems. The main idea of a conventional SVM is to construct a optimal hyper-plane that maximize the margin between two classes. Originally, the SVM approach was developed for two-class or binary classification, and its extension to multi-class problems is still an ongoing research issue.

The most common way to build a k -class SVM is to combine several sub-problems that involve only binary classification. This approach is used by methods as one-against-all or one-against-one: in the one-against-all case k SVMs are trained to separate the point of each class from all the others; in the one-against-one case, instead, $k(k-1)/2$ binary classifiers are trained on all the class pairs. Platt et al. [2] proposed another algorithm in which Directed Acyclic Graph is used to combine the results of one-against-one classifiers (DAGSVM).

In this paper, we introduce a new multi-class method, which is based on non-balanced binary tree. The construction of the binary tree is based on the prior distribution of the training data. For an k class problem, this method only needs $k-1$ SVM classifiers in the training phase, while it has less than k binary test when making a decision. Additionally, this can resolve the unclassifiable regions that exist in the conventional SVMs.

* This work was supported by United Project of Yang Zi delta integration (2005E60007).

This paper is organized as follows. In section 2, we review several multi-class SVM methods. In section 3, we describe how to construct the non-balanced binary tree, and analyze the time complexity of our method. This approach is validated experimentally in Section 4. Section 5 states the main conclusions.

2 Multi-class SVM Methods

2.1 One-Against-all Method

For a k -class problem, the one-against-all method constructs k SVM models. The i th SVM is trained with all of the training examples in the i th class with positive labels and all other examples with negative labels. The final output of the one-against-all method is the class that corresponds to the SVM with the highest output value, i.e.

$$\text{the class of } \mathbf{x} = \arg \max_{i=1,2,\dots,k} f_i(\mathbf{x}) \tag{1}$$

Where $f_i(\mathbf{x})$ is the decision function of the i th SVM.

2.2 One-against-one Method

The one-against-one method constructs all possible pairwise hyperplanes, where each hyperplane is constructed using the training examples from two classes chosen out of k classes. The decision function for class pair ij is defined by

$$f_{ij}(\mathbf{x}) = \text{sgn} \left[\sum_{h=1}^{n_{ij}} \alpha_h^{ij} y_h K(\mathbf{x}, \mathbf{x}_h) + b_h^{ij} \right], \quad i = 1, 2, \dots, k \quad i < j \tag{2}$$

Since $f_{ij}(\mathbf{x}) = -f_{ji}(\mathbf{x})$, there exist $k(k-1)/2$ different decision functions for a k -class problem. This method fits perfectly to the known characteristics of the SVM, where the borderlines between two classes are computed directly.

The most popular method for the class identification of the one-against-one method is the “max wins” algorithm [3]. In the “max win” algorithm each classifier casts one vote for its preferred class, and the final result is the class with the most votes.

2.3 DAGSVM Method

The DAGSVM method uses a rooted binary DAG to define a class in the classification tasks. A rooted binary DAG has nodes connected by arcs where each node has either 0 or 2 arcs leaving it. For a k -class problem, a rooted binary DAG is used that has $k(k-1)/2$ internal nodes and k leaves. Each node is associated with a binary SVM, and the leaves are labeled by the classes. The i th node in the $(N-j+i)$ layer distinguishing the classes i and j provide $i < j$. To define the class of a point x , starting at the root node, the binary decision function at this node is evaluated. Then, it moves to either left or right depending on the output value. The next node’s binary function is then evaluated.

Therefore, one goes through a path before reaching a leaf that indicates the class. Thus for a k -class problem, $k-1$ decision nodes are evaluated to define a class. The training phase of this method is the same as the one-against-one method but its testing time is less than that of the one-against-one method. The disadvantage of this method over the “ma win” algorithm is that it cannot resolve the tie regions and results may vary according to the decision functions at nodes with different pairs of the classes.

2.4 Comparison of the Above Multi-class Methods

In [4], Hsu and Lin had compared the performance of the above methods with a large set of different problems. Their observations are that the accuracy rate of all the methods is very similar. That is, no one is statistically better than the others. The one-against-all method uses all training data for learning, i.e. the number of variables of each optimization problem equals the number of training data. Therefore, a long training time is necessary for training a problem with a large number of training data. For the training time, one-against-one and DAGSVM methods are the best. The testing time of the DAGSVM method is less than that of the one-against-one method.

3 Proposed Method

The above methods are all based on a consideration: all classes appear in an equal probability. However, it may happen for some problems that classes are aligned approximately or some classes are possibly distant from other classes. If all classes are recognized equal probably, the efficiency may be very low. On the contrary, if the classes with high probability are recognized preferentially, the speed of recognizing all classes can be quickened greatly. According to this idea, a non-balanced binary tree is introduced to combine multiple binary-SVMs for multi-class classification problems.

It is known that the classification error depends on the structure of a binary tree. Especially, if the classification performance is not good at the upper node of the decision tree, the overall classification performance becomes worse. Therefore, more separable classes should be separated at the upper node of the binary tree. To determine the binary tree, we use the fact that the physical relationship of data in input space is kept in the feature space. However, for measuring the distance of classes in input space, an appropriate definition of the distance between the classes should be introduced. We will discuss this in the next subsection.

3.1 Distance Calculation Method

The Euclidian distance is chosen frequently for calculating the distance between two classes. However, the Euclidian distance of centers of two classes cannot represent the separability of two classes. Fig.1 shows an example of the Euclidian distance. The Euclidian distances are equal in (a) and (b), but the two classes in (b) are apparently more separable than classes in (a).

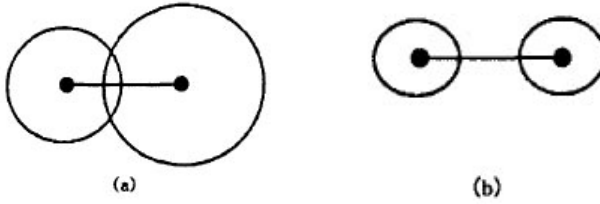


Fig. 1. Comparison of the separability of two classes

Thus, we used a distance calculation method [5] based on the distribution of classes, which is given as

$$\delta_{ij} = \frac{d_{ij}}{(\delta_i + \delta_j)} \tag{3}$$

Where d_{ij} is the Euclidian distance of the class centers, δ_i and δ_j is the covariance of the class, i.e.

$$\delta_i = \frac{1}{n_i - 1} \sum_{x \in X_i} \|x_i - c_i\|, \quad i = 1, 2, \dots, k \tag{4}$$

Here X_i is a set of training data included in class i , and n_i is the number of elements included in X_i , and c_i is the center of class i .

When $\delta_{ij} \geq 1$, it implies that the two classes have no intersection. The bigger the value of δ_{ij} is, more separable the two classes are.

3.2 Algorithm Description

For a k -class problem, the construction of non-balanced binary tree is described by the following steps:

1. Divide the training data set into all the pairwise subsets;
2. Calculate the distances of all class pairs using the method as discussed in the previous subsection;
3. Combine the nearest two classes into a new class X' . Then $k-1$ classes are constructed and k is turned to be $k-1$;
4. Respectively calculate the distances of class X' and all other classes. Select the class with the nearest distance, and combine it with class X' . Update the center and covariance of class X' , then $k-1$ is turned to be $k-2$.
5. If k classes are combined into two classes, the constructing of the binary tree will be completed. Else go to step 4.

In training, we select the SVM of the last two classes as root node of binary tree. One of the two classes, namely X' , contains $k-1$ classes, and the other contains one class. Following that, we select the SVM of the last two classes in X' as second node. We repeat this procedure until there is only one class in X' . Fig.2 illustrates the structure of non-balanced binary tree.

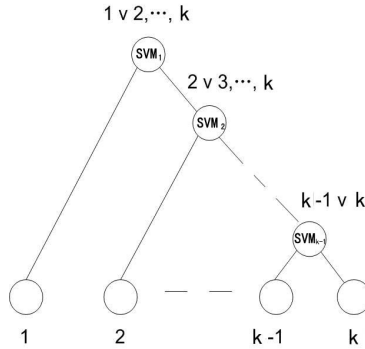


Fig. 2. Non-balanced binary tree

In classification, starting from the top of the binary tree, the input data x is classified by SVM. If x is classified into class 1, the classification is completed, if not, x is processed by next SVM. The best case occurs if we find the class at the first node, and the worst case occurs if we find the class after applying all the $k-1$ SVMs. Table.1 compares the number of classifiers needed in four multi-class methods. For a k -class problem, the number of classifiers in our method, either training phase or testing phase, is less than other methods.

Table 1. Comparison of the number of classifiers needed in four multi-class methods

	Training Phase	Testing Phase
One-against-all	k	k
One-against-one	$k(k-1)/2$	$k(k-1)/2$
DAGSVM	$k(k-1)/2$	$k-1$
Our Method	$k-1$	$< k$

3.3 Time Complexity

The quadratic optimization problem in the training phase of SVM, slows down the training process. Platt introduced a fast algorithm, which is called SMO [6], for training

support vector machines. Using SMO, training a single SVM is observed to scale in polynomial time with the training set size m :

$$T_{single} = cm^\gamma \tag{5}$$

With this relation, we can find the training time for one-against-all as

$$T_{one-against-all} = ckm^\gamma \tag{6}$$

From Equation 5, the training time for one-against-one is found as

$$T_{one-against-one} = T_{DAGSVM} = 2^{\gamma-1} ck^{2-\gamma} m^\gamma \tag{7}$$

assuming that the classes have the same number of training data samples. With the same assumption, the training time of our method would be

$$T = \sum_{i=1}^{k-1} c \left[\frac{m}{k} (k - i + 1) \right]^\gamma \tag{8}$$

In [6], Platt assumed that the typical value for γ is 2. In this case, when k , namely the number of classes, is less than 6, our method is relatively faster than other methods. If k is large, the one-against-one method is fastest. But our method is still faster than one-against-all method.

4 Experimental Results

In this section we present the experimental results and compare the performance of the proposed method with the one-against-one, one-against-all, and DAGSVM. The data set comes from the ORL face database, in which there are 400 face images (10 samples per person).

The whole data set has been divided into a set of 200 training samples and a set of 200 test samples. The training set has been used to design the classifiers, and the test set for testing the performance of the classifiers. All the algorithms are realized with MATLAB6.5 and VC6.0. Experiment platform is P4-2.4G PC, 512M RAM and operating system is Windows XP professional. The experiments are executed to compare four methods based on SMO on testing classification precision, testing time. The experimental results are as Table.2 shows.

Table 2. Result of comparison on the ORL face database

Multi-class Method	Accurate Rate	Testing Time(seconds)
One-against-all	97%	15.2
One-against-one	97.5%	296.4
DAGSVM	97.5%	14.82
Our Method	97%	7.41

The experimental results show that maintaining comparable accuracy, our method is faster than other methods in classification. Because the more separable classes are separated at the upper node of the binary tree, the error of the whole classification is under control. Since less classifiers are needed for many test samples, compared to other methods, the training time of our method is greatly reduced.

5 Conclusions

We have introduced a new method as a solution to multi-class problems. This method is to construct a non-balanced binary tree based on a prior distribution of training samples. Applying a distance calculation method, we make the more separable classes separated at the upper node of the binary tree. We compare the proposed method to the one-against-all, the one-against-one, and the DAGSVM methods with the ORL face database. Experimental results show that the testing time of the proposed method is the least. The training time of proposed method is less than that of one-against-all method. Further, the proposed method can resolve the unclassifiable regions.

References

1. Vapnik, V.N.: *Statistical Learning Theory*, John Wiley & Sons (1998)
2. Platt, J.C., Cristianini, N., Shawe-Taylor, J.: Large Margin DAGs for Multiclass Classification, *Advances in Neural Information Processing Systems* **12** (2000) 547-553
3. Friedman, J.: Another approach to polychotomous classification, Stanford University, Department of Statistics (1996)
4. Hsu, C.W., Lin, C.J.: A comparison of methods for multiclass support vector machines, *IEEE Trans Neural Networks* **13** (2002) 415–425
5. Shi, Z.H., Wang, X.D., Zhao, S.M., Yang, J.X.: An Improved Algorithm for SVM Decision Tree, *Journal of Air Force Engineering University* **7** (2006) 32-35
6. Platt, J.: Sequential minimal optimization: A fast algorithm for training support vector machines, Technical Report 98-14, Microsoft Research, Redmond, Washington, April (1998)

Identification of MIMO Hammerstein Models Using Support Vector Machine

Hua Liang and Bolin Wang

College of Electrical Engineering,
HoHai University, Nanjing 210098, P.R. China
hliangsmile@163.com

Abstract. The least squares support vector machines (LS-SVMs) regression is presented for the purpose of nonlinear dynamic system identification. LS-SVMs are used for system identification of Hammerstein models with memoryless nonlinear blocks and linear dynamical blocks. LS-SVMs achieves higher generalization performance than a hybrid neural network (HNN) which consist of a multi-layer feed-forward neural network (MFNN) in cascade with a linear neural network (LNN). The identification procedure is illustrated using two simulated examples. The results indicate that this approach is effective even in the case of additive noise to the system.

1 Introduction

The Hammerstein models are special kinds of nonlinear systems, Hammerstein models are composed of a memoryless static nonlinearity followed by a linear dynamical system. Such models have been widely used in many areas, e.g., nonlinear filtering, actuator saturations, audio-visual processing, signal analysis, biologic systems, chemical processes, and so on [1]. Therefore, there exist a large amount of work on identification of these models exploring different approaches and frameworks [2]-[8]. For Hammerstein and Wiener models, Bai reported some validity results: a two-stage identification algorithm based on the recursive least-squares and on the singular value decomposition and a blind identification approach [8]. Two popular approaches are prediction error minimization and subspace identification. Bart De Moor and his research group developed the subspace identification algorithms [6][7].

Throughout the last few decades, artificial neural network have proved to be a powerful methodology in a wide range of fields and applications. Most of the ANN-based system identification techniques are based on multilayer feed-forward networks such as multilayer perceptron (MLP) trained with back propagation or more efficient variation of this algorithm [10]-[15]. Narendra and Parthasarathy have proposed effective identification and control of dynamic systems using MLP networks [11]. Li present a method, it uses a hybrid neural network which consists of a multi-layer feed-forward neural network in cascade with a linear neural network [10]. Of course these networks are robust and effective in identification and control of complex dynamic plants. Despite many advances, there

still remain a number of weak points, including local minimum, over learning, the difficulties of choosing network structure, and so on. To overcome those problems, major breakthroughs are obtained at this point with a new class of ANN called support vector machines (SVMs), SVMs were developed in a pattern classification context as an implementation of Vapnik’s Structural Risk Minimisation principle. Regression estimation can be performed by an extended form of SVM and has been shown to perform well in areas such as identifying chaotic time series models. Least squares support vector machines (LS-SVMs) have been proposed by Sukens and Vandewalle for solving pattern recognition and nonlinear function estimation problem.

This paper focuses on the LS-SVMs in identification of Hammerstein models. This paper is organized as follows: In Section II, we describe the problem formulation related to the Hammerstein systems. In section III, we introduce LS-SVMs system for identification. Simulation and application examples are included in Section IV. Finally, in Section V, conclusions are drawn.

2 Basic Method Description

Hammerstein models consist of a nonlinear memoryless element followed by a linear dynamical system, where the true output (namely, the noise-free output) $w(t)$ and the inner variable $x(t)$ (namely, the output of the nonlinear block) are unmeasurable, $u(t)$ is the system input, $y(t)$ is the measurement of $w(t)$ but is corrupted by the disturbance $v(t)$, $G(z)$ is the transfer function of the linear part in the model. The nonlinear part in the Hammerstein model is a polynomial of a known order in the input as follows:

$$x(t) = r_1 u(t) + r_2 u^2(t) + \dots + r_p u^p(t). \tag{1}$$

Assume that the linear dynamical block in Fig. 1 is described by a difference



Fig. 1. Hammerstein Models

equation model, which has the following input-output relationship:

$$A(q^{-1})y(t) = b(q^{-1})x(t), \tag{2}$$

where $A(q^{-1}) = 1 + a_1q^{-1} + \dots + a_nq^{-n}$ and $B(q^{-1}) = b_0 + b_1q^{-1} + \dots + b_nq^{-n}$. If the noise $v(t)$ is obvious, we modify the equation (2):

$$A(q^{-1})y(t) = b(q^{-1})x(t) + A(q^{-1})v(t), \tag{3}$$

we merge the equation (1) and (3), and obtain the Hammerstein models:

$$A(q^{-1})y(t) = b(q^{-1})[u(t) + \sum_{i=2}^p r_i u^i(t)] + e(t), \tag{4}$$

where $e(t) = A(q^{-1})v(t)$ The description of MIMO linear dynamic systems can be written as follows:

$$Y(t) = -A_1(t)Y(t-1) - \dots - A_{na}(t-na) + B_0X(t) + \dots + B_{nb}X(t-nb) + V(t), \tag{5}$$

where $Y(t)$ is the $ny \times 1$ output vector; $X(t)$ is the $nu \times 1$ input vector; $V(t)$ is the $ny \times 1$ noise vector; A_1, \dots, A_{na} are $ny \times ny$ matrices; B_0, \dots, B_{nb} are $ny \times nu$ matrices; and na, nb represent the order of the model. The input vector is defined as $U(t) = [u_1(t) \dots u_{nu}(t)]^T$. The objective of this paper is to present identification algorithms to estimate the system parameters a_i, b_i and r_i of the model by using the available input-output data $\{u(t), y(t)\}$.

3 Least Squares Support Vector Machines for Identification

In the following, we briefly introduce LS-SVMs regression, which can be used for nonlinear system identification.

Usually a typical regression problem is defined as follows: Given a training data set of N points $\{x_k, y_k\}_{k=1}^N$ with the input data $x_k \in R^n$ and the corresponding target $y_k \in R$. In feature space SVM models take the form

$$y(x) = \omega^T \varphi(x) + b, \tag{6}$$

where the nonlinear mapping $\varphi(\cdot)$ maps the input vector into a higher dimensional feature space, b is the bias and ω is a weight vector of the same dimension as the feature space. In LS-SVMs for function estimate, one considers the following optimization problem

$$\min_{\omega, e} J(\omega, e) = \frac{1}{2} \omega^T \omega + \frac{1}{2} \gamma \sum_{k=1}^N e_k^2, \tag{7}$$

subject to the equality constraints

$$y_k = \omega^T \varphi(x) + b + e_k, k = 1, \dots, N, \tag{8}$$

here γ is the regularization parameter.

This problem can be solved by using the optimization theory. One can define the Lagrangian for this problem as follows:

$$L(\omega, b, e, \alpha) = j(\omega, e) - \sum_{k=1}^N \alpha_k (\omega^T \varphi(x_k) + b + e_k - y_k). \tag{9}$$

In this equation, the α_k s are called the Lagrangian multipliers. The saddle point can be found by setting the derivatives equal to zero.

$$\begin{cases} \frac{\partial L}{\partial \omega} = 0 \rightarrow \omega = \sum_{k=1}^N \alpha_k \varphi(x_k), \\ \frac{\partial L}{\partial b} = 0 \rightarrow \sum_{k=1}^N \alpha_k = 0, \\ \frac{\partial L}{\partial e_k} = 0 \rightarrow \alpha_k = \gamma e_k, \\ \frac{\partial L}{\partial \alpha_k} = 0 \rightarrow \omega^T \varphi(x_k) + b + e_k - y_k = 0, \end{cases} \tag{10}$$

for $k = 1, \dots, N$. Elimination of e_k and ω through substitution in the following set of linear equations

$$\begin{bmatrix} 1 & e1^T \\ e1^T & \Omega + \gamma^{-1}I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}, \tag{11}$$

where $y = (y_1, \dots, y_l)^T, e1 = (1, \dots, 1)^T, \alpha = (\alpha_1, \dots, \alpha_l)^T, \Omega_{ij} = \varphi(x_i)^T \varphi(x_j) = k(x_i, x_j), i, j = 1, \dots, l$.

The resulting LS-SVMs model for function estimation becomes

$$y(x) = \sum_{k=1}^N \alpha_k k(x, x_k) + b, \tag{12}$$

$k(x, x_k)$ is a symmetric function which satisfies Mercer conditions. Some useful kernels are as follows:

- (1)Polynomial kernel of order p: $y(x) = \sum_{k=1}^N \alpha_k k(x, x_k) + b;$
- (2)Gaussian kernel function: $k(x, x^*) = exp(-\frac{\|x-x^*\|_2^2}{2\sigma^2});$
- (3)Hyperbolic kernel: $k(x, x^*) = tanh(\beta x^T x^* + k).$

The method for performing Hammerstein models identification is to induce the function $f(\cdot)$ in Equation (5) using LS-SVMs regression, which is trained with a set of finite data of past inputs and outputs as the target value of the LS-SVMs regression. A two-layer LS-SVMs is used to estimate the linear and nonlinear elements simultaneously in a framework. The first layer is to identify the nonlinear memoryless system, the second layer is to identify the linear dynamical system. Where α_k, b re the solution to the linear system, $k(\cdot)$ represents the high dimensional feature spaces that is nonlinearly mapped from the input space . The LS-SVMs approximates the function using the Equation (12). It is a good practice to represent the linear dynamical and static nonlinearity using the Gaussian kernel function. Gaussian kernel function tends to give good performance under general smoothness assumptions. Consequently, they are especially useful if no additional knowledge of the data is available.

4 Simulation Results

In this section, the two simulated examples are used as evaluation of the identification power of LS-SVMs. We used noise levels of 0.5 to investigate the quality of

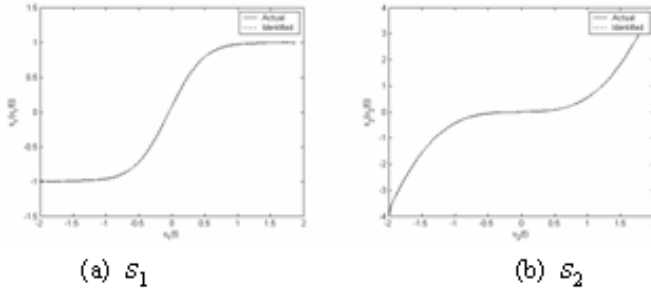


Fig. 2. Actual and identified nonlinearities of example 1

proposed identification method in comparison with the results of the noise-free identification.

Example 1 Consider a process with two inputs and two outputs described by the following equation [9] and [10]:

$$\begin{aligned} \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} &= \begin{bmatrix} 0.5 & -0.1 \\ 0.8 & -0.7 \end{bmatrix} \begin{bmatrix} y_1(t-1) \\ y_2(t-1) \end{bmatrix} + \begin{bmatrix} -0.3 & 0.2 \\ 0.9 & -0.5 \end{bmatrix} \begin{bmatrix} y_1(t-2) \\ y_2(t-2) \end{bmatrix} \\ &+ \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} s_1(u_1(t)) \\ s_2(u_2(t)) \end{bmatrix} + \begin{bmatrix} \zeta_1(t) \\ \zeta_2(t) \end{bmatrix}, \end{aligned}$$

where the nonlinearities are given by

$$\begin{aligned} s_1(u_1(t)) &= (1 - e^{-4u_1(t)}) / (1 + e^{-4u_1(t)}), \\ s_2(u_2(t)) &= 0.5u_2^3(t), \end{aligned}$$

$\zeta_1(t), \zeta_2(t)$ are white, zero-mean Gaussian distributed noises of variances 0.5. The inputs to the plant are random signals whose amplitude is uniformly distributed in the interval $[-2, 2]$. Two identification procedures of the plant, with noise or not, are implemented respectively. The numerical results of example 1 are located in the fifth row and sixth row of the Tab.1. The numerical results of the method in [9] are the second row. The numerical results of the method in [10] are the third row and the fourth row. Fig.2 shows the actual and identified nonlinearities when the noise is not added to the plant. Example 2 Consider a process the same as the process of example 1 except that the nonlinearities are given by [10]

$$\begin{aligned} s_1(u_1(t), u_2(t)) &= \frac{(1 - e^{-(u_1(t)+u_2(t))})}{(1 + e^{-(u_1(t)+u_2(t))})}, \\ s_2(u_1(t), u_2(t)) &= 0.25u_1^2(t)u_2(t). \end{aligned}$$

The input to the plant is random signals whose amplitude is uniformly distributed in the interval $[-2, 2]$. Numerical results are also shown in the ninth row

Table 1. Numerical results of example 1 and example 2

	A_1	A_2	B_0
Desired value	$\begin{bmatrix} 0.5 & -0.1 \\ 0.8 & -0.7 \end{bmatrix}$	$\begin{bmatrix} -0.3 & 0.2 \\ 0.9 & -0.5 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Result of [9]	$\begin{bmatrix} 0.5027 & -0.0951 \\ 0.7958 & -0.7015 \end{bmatrix}$	$\begin{bmatrix} -0.3034 & 0.2044 \\ 0.8963 & -0.5018 \end{bmatrix}$	$\begin{bmatrix} -0.3155 & 0.1846 \\ -0.0679 & 0.2464 \end{bmatrix}$
Result of [10] Example1			
(Without noise)	$\begin{bmatrix} 0.5000 & -0.1001 \\ 0.7982 & -0.7011 \end{bmatrix}$	$\begin{bmatrix} -0.3002 & 0.2000 \\ 0.8993 & -0.5005 \end{bmatrix}$	$\begin{bmatrix} -0.3187 & 0.0000 \\ 0.0012 & -0.5102 \end{bmatrix}$
Result of [10] Example1			
(With noise)	$\begin{bmatrix} 0.5029 & -0.1088 \\ 0.7724 & -0.6903 \end{bmatrix}$	$\begin{bmatrix} -0.3111 & 0.2185 \\ 0.9005 & -0.4887 \end{bmatrix}$	$\begin{bmatrix} -0.3126 & -0.0039 \\ 0.0016 & 0.7575 \end{bmatrix}$
Example1			
(Without noise)	$\begin{bmatrix} 0.5 & -0.1 \\ 0.8 & -0.7 \end{bmatrix}$	$\begin{bmatrix} -0.3 & 0.2 \\ 0.9 & -0.5 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Example1			
(With noise)	$\begin{bmatrix} 0.5006 & -0.0941 \\ 0.8006 & -0.6941 \end{bmatrix}$	$\begin{bmatrix} -0.3053 & 0.2134 \\ 0.8987 & -0.4986 \end{bmatrix}$	$\begin{bmatrix} 0.9272 & 0.0159 \\ -0.0028 & 1.0159 \end{bmatrix}$
Result of [10] Example2			
(Without noise)	$\begin{bmatrix} 0.5001 & -0.1002 \\ 0.7997 & -0.7003 \end{bmatrix}$	$\begin{bmatrix} -0.3000 & 0.2001 \\ 0.9004 & -0.5019 \end{bmatrix}$	$\begin{bmatrix} -0.3197 & 0.0141 \\ 0.0078 & 1.0460 \end{bmatrix}$
Result of [10] Example2			
(With noise)	$\begin{bmatrix} 0.4577 & -0.1172 \\ 0.8066 & -0.7292 \end{bmatrix}$	$\begin{bmatrix} -0.2807 & 0.1972 \\ 0.8765 & -0.4939 \end{bmatrix}$	$\begin{bmatrix} 0.6455 & 0.0242 \\ 0.0492 & 6589 \end{bmatrix}$
Example2			
(Without noise)	$\begin{bmatrix} 0.5 & -0.1 \\ 0.8 & -0.7 \end{bmatrix}$	$\begin{bmatrix} -0.3 & 0.2 \\ 0.9 & -0.5 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
Example2			
(With noise)	$\begin{bmatrix} 0.5534 & -0.0860 \\ 0.8358 & -0.6760 \end{bmatrix}$	$\begin{bmatrix} -0.3215 & 0.2174 \\ 0.8785 & -0.4826 \end{bmatrix}$	$\begin{bmatrix} 0.9790 & -0.0179 \\ -0.0210 & 0.9821 \end{bmatrix}$

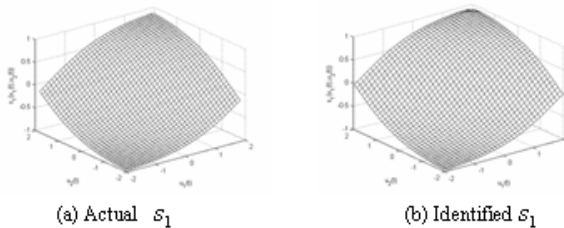


Fig. 3. Actual and identified nonlinearities of example 2(s1)

and the tenth row of Tab.1. The numerical results of the method in [10] are the seventh row and the eighth row. From the Tab.1, if the training data are corrupted with noise of 0.5 levels, it is verified that additive noise has an influence on the system identification. Fig.3 and Fig.4 show the actual and identified nonlinearities when the noise is not added to the plant.

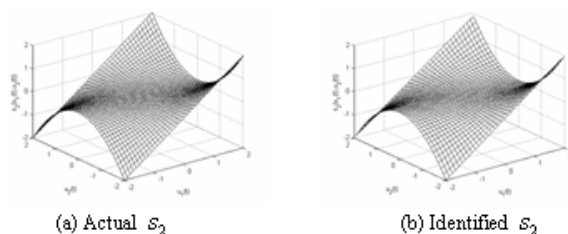


Fig. 4. Actual and identified nonlinearities of example 2(s2)

5 Conclusion

In this paper, we investigate the problem of nonlinear system dynamic identification using the LS-SVMs regression. We also consider the influence of noise on the identification. The simulation results show the efficiency of the proposed methods. This approach is found to be superior to that of the neural networks for the complex task of nonlinear system identification even in the case of additive noise to the system. And the results of this research suggest that LS-SVMs are a better method to identification of the Hammerstein models, and appear to have potentials for future applications in identifying the other models, and also for analysis of other models.

References

1. Ding, F., Chen, T.W.: Identification of Hammerstein Nonlinear ARMAX Systems. *Automatica* (2005) 1479-1489
2. Cerone, V., Regruto, D.: Parameter Bounds for Discrete-Time Hammerstein Models with Bounded Output Errors. *IEEE Transactions on Automatic Control* (2003) 1855-1860
3. Haber, R., Unbehauen, H.: Structure Identification of Nonlinear Dynamic Systems- A Survey Ofinput-Output Approaches. *Automatica* (1990)651-677
4. Greblicki, W.: Stochastic Approximation in Nonparametric Identification of Hammerstein Systems. *IEEE Transactions on Automatic Control* (2002)1800-1810
5. Wigren, T., Nordsjo, A.E.: Compensation of the RLS Algorithm for Output Nonlinearities. *IEEE Transactions on Automatic Control* (1999) 1913-1918
6. Espinoza, M., Suykens, J.A.K., De Moor, B.: Partially Linear Models and Least Squares Support Vector Machines. 43rd IEEE Conference on Decision and Control (2004) 3388-3393
7. Goethals, I., Pelckmans, K., Suykens, J.A.K., De Moor, B.: Subspace Identification of Hammerstein Systems Using Least Squares Support Vector Machines. *IEEE Transactions of Automatic Control* (2005) 1509-1518
8. Bai, E.W.: An Optimal Two-Stage Identification Algorithm for Hammerstein-Wiener Nonlinear Systems. *Automatica* (1998) 333-338
9. Duwaish, H.A., Karim, M.N.: A New Method for the Identification of Hammerstein Model. *Automatica* (1997)1871-1875
10. Li, S.H., Li, Q., Li, J.: Identification of Hammerstein Model Using Hybrid Neural Network. *Journal of Southeast University (English Edition)* (2001) 26-30

11. Narendra, K.S., Parhasarathy, K.: Identification and Control of Dynamical Systems Using Neural Networks. *IEEE Trans. Neural Networks* (1990) 4-26
12. Nguyen, D.H., Widrow, B.: Neural Networks for Self-Learning Control System. *Int. J. Contr* (1991)1439-1451
13. Cembrano, G., Wells, G., Sarda, J., Ruggeri, A.: Dynamic Control of a Robot Arm Based on Neural Networks. *Contr. Eng. Practice* (1997)485-492
14. Lu, S., Basar, T.: Robust Nonlinear System Identification Using Neural Network Models. *IEEE Trans. Neural Networks* (1998) 407-429
15. Wang, X.D., Ye, M.Y.: Nonlinear Dynamic System Identification Using Least Square Support Vector Machine Regression. *Proceedings of the Third International Conference on Machine Learning and Cybernetics* (2004) 941-945

Extraction of Filled-In Items from Chinese Bank Check Using Support Vector Machines

Liangli Huang¹, Shutao Li¹, and Liming Li²

¹ College of Electrical and Information Engineering, Hunan University, Changsha, 410082, China
shutao_li@hnu.cn

² Department of Operation Command, Naval Arms Command Academy, Guangzhou 510430, China

Abstract. In this paper, a novel bank check filled-in-items extraction method based on support vector machines (SVM) is proposed. After preprocessing, the bank logos are cropped and recognized by SVMs. Then the bank type characters, in Chinese, “支”, “现”, or “转”, are cropped and classified with another SVM. After check type is recognized, real check is registered to the corresponding standard blank check. Finally the filled-in items in real check are extracted. The experimental results demonstrate the presented method is effective.

1 Introduction

The widespread use of bank check in most economic activities prompts the development of automatic bank check processing, which refers to extraction and recognition of user entered information from different data fields on the check [1-2].

In the past few years, some automatic check recognition methods have been developed [2-5]. Lee et al. described a prototype for automatic Brazilian bank check recognition that recognizes or identifies both printed and filled information on a bank check automatically [3]. Ye et al. presented a technique for extracting the user-entered information from bank check images based on a layout-driven item extraction method [4]. The baselines of checks are detected and eliminated by using gray-level mathematical morphology. Yu et al. described a system for the recognition of legal amounts on bank checks written in the Chinese language [5].

In this paper, we present a novel method for the extraction of filled-in items from Chinese bank check. The idea is based on the extraction and classification of the name and logo of different banks, and the types of different check. Firstly, check logo is extracted and classified, and then check type character is extracted and classified. Finally we can extract handwritten or printed items in real bank check with subtraction.

This paper is organized as follows. In section 2 the characteristics of Chinese check are introduced. Then in section 3 the whole system including check pre-processing, logo and character cropping, features extraction, logo and character recognition,

extraction of the handwritten financial character and stamps fields is described. The experimental results are presented in section 4. Finally the conclusions are given in section 5.

2 Characteristics of Chinese Check

In this paper, we have assumed that the layout structure of bank check is standardized, such as size, position and the type of information field, etc.

A typical Chinese bank check structure can be divided into nine blocks as shown in Fig.2. On the left top row (1), bank logo and name, check type name are found. On the top right (2), bank check number is filled. Then date and recipient name are on the left of the next row (3). On the right (4) the payer’s bank name and amount number are found. Just below the second row (5), there is the area of amount, which is written by digital and Chinese. On the 4th row (6) is reserved for the described of check’s use. The next row (7), all stamps are sealed, in general 3 stamps are essential. On the right (8) are subject of debit and credit, date of transfer and names of operators. The last row (9), the special codes are filled. A sample of bank check is as followed in Fig.1.

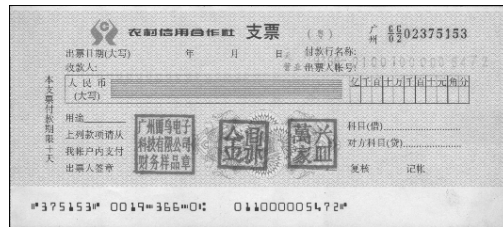


Fig. 1. A Sample of Bank Check

(1) Bank logo and name, check type name	(2) Check number
(3) Date and the recipient name area	(4) Payer’s bank name Payer’s amount number
(5) Filled amount area with digital and Chinese	
(6) Describe of check’s use	
(7) Payer’s stamps area	(8) Subject of debit and credit Date of transfer Names of Operators
(9) Special codes about check number	

Fig. 2. Block Division of a Bank Check

3 Check Recognition Systems

The overview of our system is given in Fig.3. The first step is to scan the check, and then some pre-processing steps are applied to improve the quality of the scanned image. Then check logo is cropped and recognized with SVMs. The next step is to crop check type character and extract its features, then another SVM is used to recognize characters. After getting check layout, the field of filled-in items can be obtained with subtraction of blank layout. The following sections describe the each step in detail.

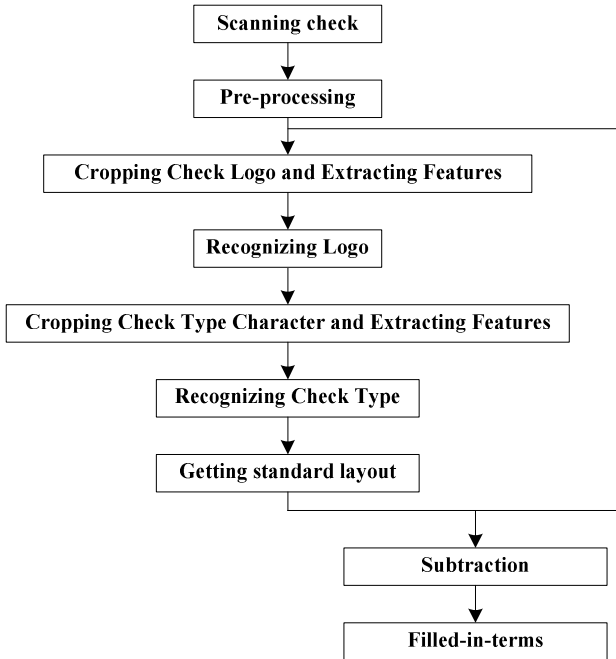


Fig. 3. Scheme of the proposed method

3.1 Pre-processing

The scanned bank check images are pre-processed before recognition. The first step in preprocessing is cropping. Cropping achieves the real check image area, gets rid of the background around the check block.

Once the real check is obtained, it is then passed to the skew detection and rectification stage. The skew detection method using projection profile analysis is adapted [6]. The next step is enhancement with local histogram equalization [7]. Finally the check image is normalized to a standard size of 320×720 pixels in gray mode.

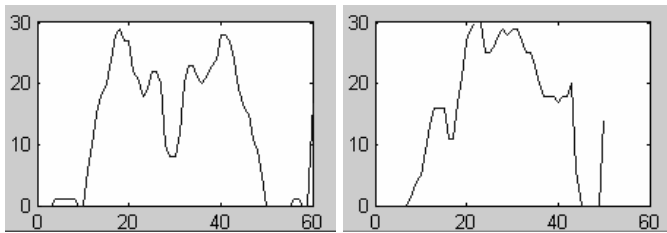
3.2 Cropping Check Logo and Extracting Features

Based on the features of Chinese check, we can extract check logo quickly. By examining the layout of the check images, we found that the logo is located in an approximation region from the 13th row to 63rd row, and from the 107th column to 167th column. One approximation logo region is shown in Fig. 4(a). Then it is binarized using threshold method as shown in Fig. 4(b). Then two projection profiles are created as shown in Fig. 4(e) and (f). The former is the vertical projection profile. Choose M as the maximum value of the projection. Ver_i is the vertical projection value; the threshold value is set to 0.05 . Then we scan the region from left to right, if $Ver_i > M \times 0.05$, the left boundary of logo is obtained, then by using $Ver_i < M \times 0.05$, the right boundary is got. It is shown in Fig.4(c). The horizontal projection profile is shown in Fig. 4(f). With the same approach, the bottom and top boundaries are obtained, which are shown in Fig. 4(d).

After the logos are cropped, they are resized to 12×12 . Then the gray values of the logos are extracted as the feature vectors.



(a) Approximated region (b) Binarized region (c) Vertical boundary (d) Horizontal boundary



(e) Vertical projection profile (f) Horizontal projection profile

Fig. 4. Cropping bank check logo

3.3 Logo Recognition

Support Vector Machines (SVM) are learning systems that use a hypothesis space of linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory[8]. SVM can be trained to solve the binary cases or multiple classes, and the structure of multiple classes can be one against one, one against the rest or DAGSVM, etc. In this case one-against-the-rest SVM is used for logo recognition.

3.4 Cropping Check Type Character

As Chinese check only has three types, including open check, cash check and check for transfer (see Fig.5), which are different at the first character, that is “支”, “现”, or “转”. After the check logo is detected and classified, the bank name is obtained. Then we can get the approximation area of the first check type character in an area which is from the 339th row to 368th row and from the 107th column to 167th column.

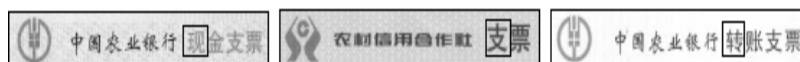
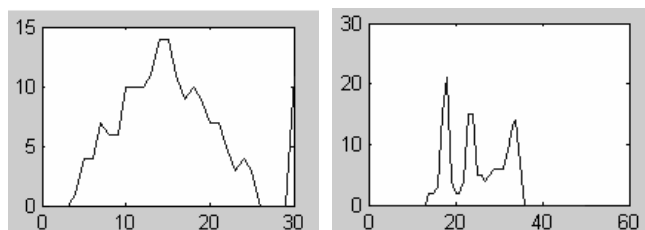


Fig. 5. Chinese bank check logos

The China Construction Bank is used for demonstrating the processing of the first check type character. Firstly, an approximation region of the type character is segmented (Fig.6 (a)). After binarization (Fig.6 (b)), the cropping procedure is similar to the cropping of check logo. Here the threshold value is 0.05 too. The cropped type character image is resized to 10×10. And the normalized intensity values are used as the input feature for another SVM classifier.



(a) Approximated region (b) Binarized region (c) Vertical boundary (d) Horizontal boundary



(e) Vertical projection profile (f) Horizontal projection profile

Fig. 6. Cropping check type character

3.5 Check Type Recognition

Similar to the section 3.3, another SVM also is constructed here. The structure of one against the rest is chosen. Half of samples which are the character image pixel vector of 100×1 are used for training and the rest for testing. In section 4.2 the corresponding experimental results are given.

3.6 Extracting Filled-In-Items

The extraction of filled-in information in the real checks is described as follows:

Step 1: Save all types of blank bank checks as references.

Step 2: Input the real checks, get the type of them through above methods which are illustrated in section 3.1-3.5.

Step 3: Registration by Maximum Mutual Information between the real check and the corresponding blank check.

Step 4: Subtraction of blank layout from the real check form. In this paper, the threshold of difference between their gray levels is choosed as 50.

Step 5: Median filtering post-processing.

4 Experimental Results

4.1 Logo Recognition with SVMs

For the SVM applications, more training samples will give a better result. So we propose a virtual training sample method to generate more samples. For every cropped logo images, some virtual images are generated using N geometric transformations. The number N is calculated as follows: $N = A \times B \times C$, where A = number of shifting pixels $\times 8$ is the total number of shifts, in 8 directions. B = number of scales $\times 2$ is the total number of scales, in 2 directions (zooming-in and zooming-out). $C = 2$ is the number of rotating methods (anticlockwise and clockwise). In the experimental setup, 2 shift and 2 scales are used. This produces 128 ($2 \times 8 \times 2 \times 2 \times 2$) virtual images for each original image. Some virtual samples are shown in Fig.7. In this way, we can have many training samples and the experimental results show it can improve the recognition accuracy.

We selected 18 types of domestic bank checks as the original experimental samples. Then 2304 (128×18) virtual samples are generated using virtual sample generating method. In the experimental setup, half of the virtual samples randomly selected from the 2304 virtual samples, plus the 18 original samples are used for training. The remained 50% virtual samples are used for testing. In order to gain fair classification performance, the training and testing are performed 50 times, and the averaged accuracy as the final accuracy.



Fig. 7. Some virtual logo samples

Before forwarded to SVMs, all logo samples are resized into 12×12 . Then the 144×1 vector is obtained as the input features. In this paper we choose one against the rest method. That's to say select one type input vectors as the positive examples, the remains as the negative. So 18 SVMs classifiers are needed for the recognition of 18 type domestic bank checks. Table 1 shows the experiment results. Gamma is the

parameter of the radial based kernel. C represents the cost of the constrain violation. From these data, we can obtain the better recognition accuracy, if Gamma from 2^{-7} to 2^{-5} and C from 1 to 1000.

Table 1. Logo Recognition Performance (%)

Gamma	C			
	1	10	100	1000
2^{-4}	99.31	99.1	99.31	99.31
2^{-5}	100	100	100	100
2^{-6}	100	100	100	100
2^{-7}	99.31	100	100	100
2^{-8}	97.40	99.31	99.31	99.31

4.2 Recognition of Check Type Character

After three check type characters is cropped (see Fig.8), they are resized into 10×10 . Then we adopt the similar method to get 384 (128×3) samples. Half of them are used for training, and the remained half are used for testing. In order to obtaining fair recognition performance, the experiment is run 50 times. 100 normalized intensity values of each sample are used directly as the input features. In this case, three SVM classifiers are designed and trained. From Table 2, we can see that SVM classifiers with Gamma from 2^{-7} to 2^{-3} and C from 1 to 1000 have the recognition accuracy of 100%.



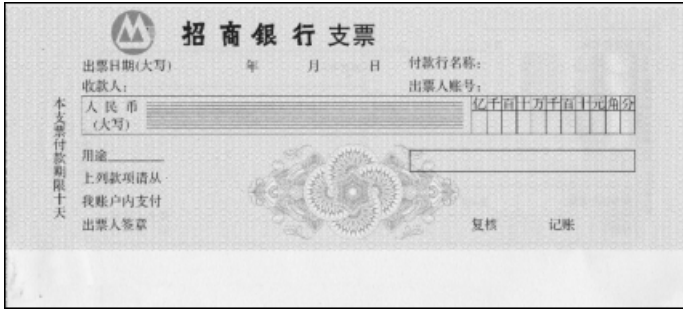
Fig. 8. The original check type character samples

Table 2. Performance of the SVMs for Type Character Recognition (%)

Gamma	C			
	1	10	100	1000
2^{-3}	100	100	100	100
2^{-4}	100	100	100	100
2^{-5}	100	100	100	100
2^{-6}	100	100	100	100
2^{-7}	100	100	100	100

4.3 Subtraction Results Against Blank Check

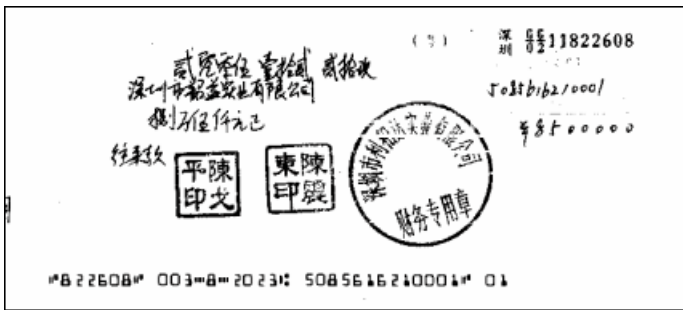
Fig. 9(a) and (b) show one blank check and filled in check from China Merchants Bank. After the above extraction and recognition, we can get the subtraction result shown in Fig. 9(c), from which we can see that the filled in items are extracted clearly and correctly.



(a) A sample of blank check



(b) A sample of the real check



(c) Extracted filled-in items

Fig. 9. Example of extracted filled in items

5 Conclusions

In this paper, an automatic recognition system for Chinese bank check is proposed. By the application of the projection profile technique logo and type character are cropped efficiently and accurately. Then the SVM classifiers are used to classify the check logos and check type characters. After achieved layout of check, the filled-in items in check are obtained with subtraction of blank check. Experimental results demonstrate that the proposed method can recognize the check type and extract filled-in data correctly.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (No.60402024) and Program for New Century Excellent Talents in University.

References

1. Suen, C.Y., Lam, L., Cheriet, M., Saïd, J.N., Fan, R.: Bank Check Processing System. *Int. J. Imaging Systems and Technology* **7** (1996) 392–403
2. Suen, C.Y., Xu, Q., and Lam, L.: Automatic Recognition of Handwritten Data on Checks - Fact or fiction?. *Pattern Recognition Letters* **20** (1999) 1287-1295
3. Lee, L., Lizarraga, M., Gomes, N., Koerich, A.: A Prototype for Brazilian Bank-check Recognition. *International Journal of Pattern Recognition and Artificial Intelligence* **11** (1997) 549-570
4. Ye, X., Cheriet, M., Suen, C.Y., Liu, K.: Extraction of Bank-check Items by Mathematical Morphology. *International Journal on Document Analysis and Recognition* **2** (1999) 53-66
5. Yu, M., Kwok, P., Leung, C.H., Tse, K.W.: Segmentation and Recognition of Chinese Bank Check Amounts. *International Journal of Document Analysis and Recognition* (2001) 207-217
6. Li, S., Shen, Q., Sun J.: Skew Detection using Wavelet Decomposition and Projection Profiles Analysis. *Pattern Recognition Letters* (2006), doi:10.1016/j.patrec.2006.10.002
7. Gonzalez, R.C., Woods, R.E.: *Digital Image Processing*, 2nd edition. Addison-Wesley Longman Publishing Co. (2002)
8. Cristianini, N., shawe-Taylor, J.: *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press (2000)

Online Least Squares Support Vector Machines Based on Wavelet and Its Applications

Qian Zhang, Fuling Fan, and Lan Wang

School of Electronic Information, ZhongYuan Institute of Technology
Zhengzhou 450007, China
qianzhang91@163.com

Abstract. As the conventional training algorithms of least squares support vector machines (LS-SVM) are inefficient in online applications, an online learning algorithm is proposed. The online algorithm is suitable for the large data set and practical applications where the data come in sequentially. Aiming at the characteristics of signals, a wavelet kernel satisfying wavelet frames is presented. The wavelet kernel can approximate arbitrary functions in quadratic continuous integral space, hence the generalization ability of LS-SVM is improved. To illustrate its favorable performance, the wavelet based online LS-SVM (WOLS-SVM) is applied to nonlinear system identification. The simulation results show that the WOLS-SVM outperforms the existing algorithms with higher learning efficiency as well as better accuracy, and indicate its effectiveness.

1 Introduction

Support vector machines (SVM) [1,2] technology is a powerful machine learning method for both classification and regression problems, which has become an exciting research area in machine learning community because of its remarkable generalization performance and elegant statistical learning theory. SVM are based on the principle of structural risk minimization (SRM), rather than on empirical risk minimization (ERM) as do many other methods. SRM is intended to improve generalization performance for small sample-size learning problems, where ERM is likely to overfit the training data [3]. Moreover, SVM training can be reduced to a quadratic programming problem. Consequently, the solution is always globally optimal. SVM use a kernel function to map the input data into a high-dimensional feature space, and then constructs an optimal separating hyperplane in that space. The kernel function must satisfy the condition of Mercer [1].

As an interesting variant of the standard SVM, least squares support vector machines (LS-SVM) have been proposed for classification [4,5,6] and function estimation [7,8]. LS-SVM differ from classical SVM in the fact that the inequality constraints in the original convex quadratic programming problem are replaced by the equality constraints, so that a set of linear equations is solved to obtain the

optimum solution. Conventionally LS-SVM are used for regression estimation of input data that are supplied in batch. In many application problems, such as system identification, chaotic time series prediction and signal processing, data are obtained in a sequence and learning has to be done from scratch. Therefore, it is time consuming to achieve the regression using the conventional LS-SVM and it is not possible to apply the LS-SVM for real-time regression problems. Hence, online learning algorithms are preferred over the batch learning algorithms. In this paper, an online learning algorithm for LS-SVM (OLS-SVM) is proposed.

OLS-SVM are a kernel based approach, which allows the use of linear, Gaussian, polynomial and RBF kernels and so on that satisfy Mercer’s condition. As the wavelet has the property of time-frequency localization and is a powerful tool for arbitrary function approximation, an allowed support vector kernel function based on the wavelet is proposed. The OLS-SVM adopts the wavelet kernel, so its generalization ability is improved. Finally, the wavelet based OLS-SVM is applied to system identification to test the efficiency.

This paper is organized as follows. Section 2 briefly reviews some basic notions of LS-SVM for function estimation. Section 3 presents a wavelet kernel and online LS-SVM. Section 4 applies wavelet based OLS-SVM to system identification. Section 5 summarizes the results of this paper.

2 Least Squares Support Vector Machines [9],[10]

Consider a training data set of N points $\{x_k, y_k\}_{k=1}^N$, where $x_k \in R^n$, $y_k \in R$, k is the samples number, n is the number of input dimension. In LS-SVM for function estimate, the following optimization problem is considered

$$\min_{\omega, e} J(\omega, e) = \frac{1}{2}\omega^T\omega + \frac{1}{2}\gamma \sum_{k=1}^N e_k^2, \tag{1}$$

subject to the equality constraints

$$y_k = \omega^T \varphi(x) + b + e_k, \quad k = 1, \dots, N, \tag{2}$$

where the nonlinear mapping $\varphi(\cdot)$ maps the input space into a so-called higher dimensional feature space, ω is a weight vector of the same dimension as the feature space, b is the bias, and γ is the regularization parameter. The the positive real constant γ relative importance of these terms is determined.

In primal feature space, the models take the form

$$y(x) = \omega^T \varphi(x) + b. \tag{3}$$

This problem can be solved in the dual space instead of the primal space. The solution is obtained by constructing the Lagrangian function

$$L(\omega, b, e; \alpha) = J(\omega, e) - \sum_{k=1}^N \alpha_k (\omega^T \varphi(x_k) + b + e_k - y_k). \tag{4}$$

with Lagrange multipliers $\alpha_k \in R$.

The conditions for optimality are given by

$$\begin{aligned} \frac{\partial L}{\partial \omega} = 0 &\rightarrow \omega = \sum_{k=1}^N \alpha_k \varphi(x_k), \\ \frac{\partial L}{\partial b} = 0 &\rightarrow \sum_{k=1}^N \alpha_k = 0, \\ \frac{\partial L}{\partial e_k} = 0 &\rightarrow \alpha_k = \gamma e_k, \\ \frac{\partial L}{\partial \alpha_k} = 0 &\rightarrow \omega^T \varphi(x_k) + b + e_k - y_k = 0, \end{aligned} \tag{5}$$

for $k = 1, \dots, N$. The condition $\alpha_k = \gamma e_k$ is the important differences between standard SVM and LS-SVM, which greatly simplifies the problem.

According to Karush-Kuhn-Tucker (KKT) conditions, e_k and ω are eliminated, and the solution is given by the following set of linear equations:

$$\left[\begin{array}{c|c} 0 & \bar{1}^T \\ \hline \bar{1} & \Omega + \gamma^{-1}I \end{array} \right] \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}, \tag{6}$$

where $y = [y_1; \dots; y_N]$, $\bar{1} = [1; \dots; 1]$, $\alpha = [\alpha_1; \dots; \alpha_N]$ and $\Omega_{kl} = \varphi(x_k)^T \varphi(x_l)$ for $k, l = 1, \dots, N$. According to Mercer's condition, there exists a mapping φ and an expansion

$$K(x, y) = \sum_i \varphi_i(x) \varphi_i(y), \quad x, y \in R^n. \tag{7}$$

For any $g(x)$ such that $\int g(x)^2 dx$ is finite, we get

$$\int K(x, y) g(x) g(y) dx dy \geq 0, \tag{8}$$

and then choose a kernel $K(., .)$

$$K(x_k, x_l) = \varphi(x_k)^T \varphi(x_l), \quad k, l = 1, \dots, N. \tag{9}$$

The resulting LS-SVM model for function estimation becomes

$$y(x) = \sum_{k=1}^N \alpha_k K(x, x_k) + b, \tag{10}$$

where α, b are the solutions of (6). In this paper, Gaussian kernel can be chosen as the kernel function, $K(x, x') = \exp(-\|x - x'\|^2 / 2\sigma^2)$.

The standard SVM solutions are characterized by convex optimization problems, and model complexity follows from this convex optimization problem. The formulation of the optimization problem involves inequality constraints. LS-SVM

uses equality instead of inequality constraints and works with a least squares cost function. The solution follows from a linear equations instead of a quadratic programming problem. Therefore, the solution of the standard SVM is more complex and time consuming than that of LS-SVM.

3 Wavelet Based Online Least Squares Support Vector Machines

3.1 Conditions for Support Vector Kernel [11], [12], [13]

The kernel function is a dot product in the feature space, such as $K(x, x') = K(\langle x \cdot x' \rangle)$. The kernel function must satisfy the conditions in Mercer condition.

Theorem 1. *In $L_2(R^N)$ space, $\forall g(x) \in L_2(R^N), \int g^2(x)dx < \infty, g \neq 0$ and $K(x, x') \in L_2(R^N \times R^N)$, if*

$$\int \int_{L_2 \otimes L_2} K(x, x')g(x)g(x')dx dx' \geq 0, \tag{11}$$

holds, $K(x, x')$ is a dot product in the feature space. Then $K(x, x')$ is a kernel function.

Theorem 1 is a simple method for constructing support vector kernel function.

Theorem 2. *A translation invariant kernel $K(x, x') = K(x - x')$ is an admissible support vector kernels if and only if the Fourier transform of $K(x)$ must satisfy the condition as follows*

$$F[k](\omega) = (2\pi)^{-N/2} \int_{R^N} exp(-j\omega x)K(x)dx \geq 0. \tag{12}$$

3.2 Wavelet Kernels

According to [16], if $\psi(x) \in L_2(R)$ is a mother wavelet, then we can get the dot product wavelet kernels

$$K(x, x') = \prod_{i=1}^N \psi\left(\frac{x_i - b_i}{a_i}\right)\psi\left(\frac{x'_i - b'_i}{a^i}\right), \tag{13}$$

and the translation invariant wavelet kernels that satisfy Theorem 2

$$K(x, x') = \prod_{i=1}^N \psi\left(\frac{x_i - x'_i}{a_i}\right). \tag{14}$$

Choosing an appropriate wavelet function as a wavelet kernel is a critical problem. We take into account not only the wavelet function satisfying the Mercer’s

condition, but also the properties of the wavelet function. We choose Littlewood-Paley wavelet as a wavelet kernel [15]. Littlewood-Paley wavelet function is defined as follows

$$\psi(x) = \frac{\sin(2\pi x) - \sin(\pi x)}{\pi x}. \tag{15}$$

The Fourier transform of Littlewood-Paley wavelet is not negative, and the value of the transform is

$$\hat{\psi}(\omega) = \begin{cases} 1, & \pi \leq |\omega| \leq 2\pi, \\ 0, & \text{other.} \end{cases} \tag{16}$$

Theorem 3. *According to the Littlewood-Paley wavelet, a wavelet kernel can be constructed as*

$$\begin{aligned} K(x, x') &= \prod_{i=1}^N \psi\left(\frac{x_i - x'_i}{a_i}\right) \\ &= \prod_i \frac{\sin 2\pi\left(\frac{x_i - x'_i}{a_i}\right) - \sin \pi\left(\frac{x_i - x'_i}{a_i}\right)}{\pi\left(\frac{x_i - x'_i}{a_i}\right)}, \end{aligned}$$

which is an admissible support vector kernel.

Proof: According to the theorem 2, we only need to prove

$$F[k](\omega) = (2\pi)^{-N/2} \int_{R^N} \exp(-j(\omega \cdot x))K(x)dx \geq 0, \tag{17}$$

then

$$\begin{aligned} F[k](\omega) &= (2\pi)^{-\frac{N}{2}} \int_{R^N} \exp(-j(\omega \cdot x))K(x)dx \\ &= (2\pi)^{-\frac{N}{2}} \int_{R^N} \exp(-j(\omega \cdot x)) \prod_i \frac{\sin 2\pi\left(\frac{x_i}{a_i}\right) - \sin \pi\left(\frac{x_i}{a_i}\right)}{\pi\left(\frac{x_i}{a_i}\right)} dx \\ &= (2\pi)^{-\frac{N}{2}} \prod_i \int_{-\infty}^{+\infty} \exp(-j\omega_i x_i) \frac{\sin 2\pi\left(\frac{x_i}{a_i}\right) - \sin \pi\left(\frac{x_i}{a_i}\right)}{\pi\left(\frac{x_i}{a_i}\right)} dx_i \\ &= (2\pi)^{-\frac{N}{2}} \prod_i \int_{-\infty}^{+\infty} \exp(-j(a_i \omega_i) \cdot \left(\frac{x_i}{a_i}\right)) \frac{\sin 2\pi\left(\frac{x_i}{a_i}\right) - \sin \pi\left(\frac{x_i}{a_i}\right)}{\pi\left(\frac{x_i}{a_i}\right)} dx_i. \end{aligned}$$

According to [16], $F[k](\omega) \geq 0$.

Substituting (15) into (8), we can obtain function estimation of LS-WSVM

$$y(x) = \sum_{k=1}^N \alpha_k \prod_i \frac{\sin 2\pi(\frac{x_k - x_k^i}{a_j^i}) - \sin \pi(\frac{x_k - x_k^i}{a_j^i})}{\pi(\frac{x_k - x_k^i}{a_j^i})} + b, \tag{18}$$

where x_k^i represents the i th component of the k th training example.

Now, LS-SVM can adopt the wavelet kernel as its kernel function. As LS-SVM cannot optimize the parameters of kernels, it is difficult to determine $N \times N$ parameters $a_k^i, i = 1, \dots, N$. For the sake of simplicity, one lets $a_k^i = a$, so the number of parameters becomes 1 [16].

3.3 Online Learning Algorithm for LS-SVM

The online learning algorithm in [17] for classification is extended for nonlinear system identification. The samples arrive sequentially in online learning. For online learning, the samples is windows roll and we set the windows size n . Consider the training samples $\{X(i), Y(i)\}$, in which $X(i) = [x_i, \dots, x_{i+n-1}], Y(i) = [y_i, \dots, y_{i+n-1}]^T, x_i \in R^n, y_i \in R$. Therefore, at time i the kernel Ω, α , and b may be represented as $\Omega_{kl}(i) = K(x_{k+i-1}, x_{l+i-1}), k, l = 1, \dots, n, \alpha(i) = [\alpha_i; \dots; \alpha_{i+n-1}], b(i) = b_i$, and $y(i) = y_i$, then the function estimation (10) becomes

$$y(i) = \sum_{k=i}^{i+n-1} \alpha_k K(x, x_k) + b(i). \tag{19}$$

Let $U(i) = \Omega(i) + \gamma^{-1}I$, then (6) can be represented as

$$\begin{bmatrix} 0 & \vec{1}^T \\ \vec{1} & U(i) \end{bmatrix} \begin{bmatrix} b(i) \\ \alpha(i) \end{bmatrix} = \begin{bmatrix} 0 \\ y(i) \end{bmatrix}. \tag{20}$$

Let $P(i) = U(i)^{-1}$, then according to the equation (20)

$$b(i) = \frac{\vec{1}^T P(i) y(i)}{\vec{1}^T P(i) \vec{1}}, \tag{21}$$

$$\alpha(i) = P(i)(y(i) - \vec{1} b(i)), \tag{22}$$

and according to [18]

$$\begin{aligned} P(i) &= U(i)^{-1} = [\Omega(i) + \gamma^{-1}I]^{-1} \\ &= \begin{bmatrix} h(i) & H(i)^T \\ H(i) & D(i) \end{bmatrix}^{-1} \\ &= \begin{bmatrix} 0 & 0 \\ 0 & D(i)^{-1} \end{bmatrix} + s_h(i) s_h^T c_h(i), \end{aligned} \tag{23}$$

where I is a identity matrix, $h(i) = K(x_i, x_i) + \frac{1}{\gamma}, H(i) = [K(x_{i+1}, x_i) + \frac{1}{\gamma}, \dots, K(x_{i+n-1}, x_i)]^T,$

$$D(i) = \begin{bmatrix} K(x_{i+1}, x_{i+1}) + \frac{1}{\gamma} & \dots & K(x_{i+n-1}, x_{i+1}) \\ \vdots & \ddots & \vdots \\ K(x_{i+1}, x_{i+n-1}) & \dots & K(x_{i+n-1}, x_{i+n-1}) + \frac{1}{\gamma} \end{bmatrix},$$

$$s_h(i) = [-1, H(i)^T D(i)^{-1}]^T, c_h(i) = 1/(h(i) - H(i)^T D(i)^{-1} H(i)).$$

At the time $i + 1$, a new sample (x_{i+1}, y_{i+1}) is inserted, then the old sample (x_i, y_i) is deleted, and the kernel is $\Omega_{kl}(i + 1) = K(x_{k+i}, x_{l+i})$, $P(i + 1) = U(i + 1)^{-1} = [\Omega(i + 1) + I/\gamma]^{-1}$.

According to the above discussion, the online algorithm is obtained [17]:

1. Initialization: $i=1$;
2. Get the new data $(X(i), Y(i))$ and delete the old data;
3. Compute the kernel $\Omega(i)$ and $P(i)$;
4. Compute $b(i)$ and $\alpha(i)$, then predicate $y(i+1)$;
5. $i = i+1$, and return to 2.

Each update of the algorithm runs in $O(i^2)$.

4 Application of Wavelet Based OLS-WSVM

We validate the performance of wavelet based OLS-SVM by a nonlinear system identification and compare its performance with the batch LS-SVM and the incremental learning SVM [19]. In the batch LS-SVM and incremental learning SVM, the kernel is Gaussian function, $K(x_i, x_j) = \exp(-\lambda \|x_i - x_j\|)$, $\lambda = 1$.

A nonlinear system [20] to be identified is governed by

$$y(t + 1) = 0.3y(t) + 0.6y(t - 1) + f(u(t)), \tag{24}$$

in which

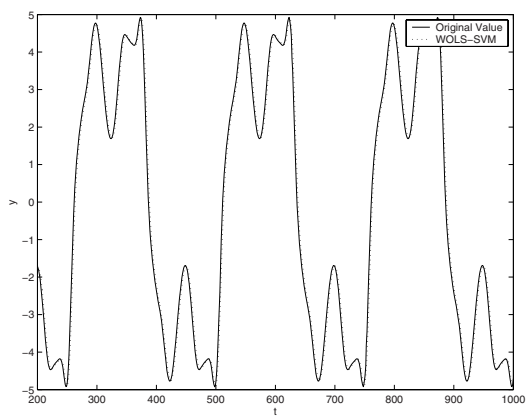
$$f(t) = 0.6 \sin(\pi u) + 0.3 \sin(3\pi u) + 0.1 \sin(5\pi u), \tag{25}$$

and the input

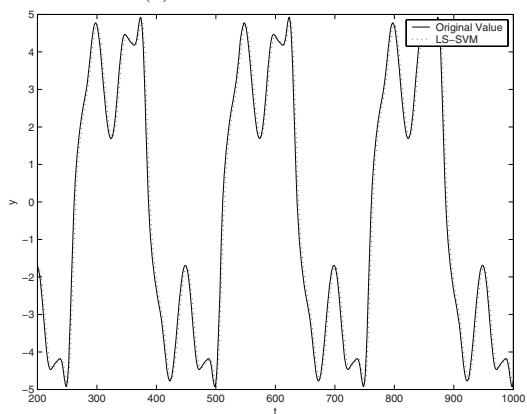
$$u(t) = \sin(2\pi t/250). \tag{26}$$

We take 1000 points as the training samples in which 200 points are training samples and the others are prediction samples. Figure 1 shows simulation results.

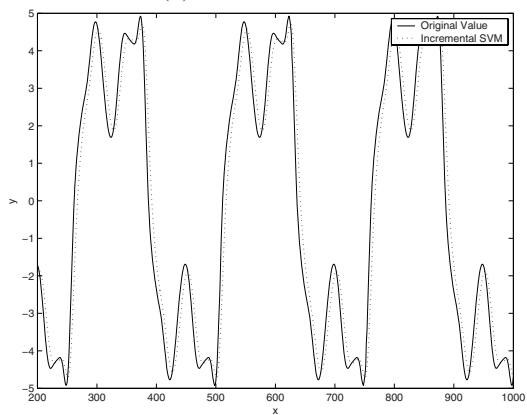
From the simulation experiments, we can find the proposed WOLS-SVM outperforms the incremental SVM and batch LS-SVM. WOLS-SVM is generally much faster than the batch LS-SVM algorithms when applied to online prediction. The accuracy of WOLS-SVM is higher than the incremental SVM for only providing an approximation solution and the batch LS-SVM has to be done from scratch when data are obtained in a sequence, so its speed is rather slow. The wavelet kernel is not only a kind of multidimensional wavelet functions, but also inherit the characters of Littlewood-Paley wavelet orthonormal capacity.



(a) Performance of WOLS-SVM.



(b) Performance of LS-SVM.



(c) Performance of Incremental SVM.

Fig. 1. Simulation results in nonlinear system identification

5 Conclusions

Least squares support vector machine is a powerful machine learning method for pattern recognition and regression problems. LS-SVM has been used to model nonlinear system identification. As the training samples are online obtained, conventional LS-SVM suffer from the problem of large memory requirement when trained in batch mode on large-scale online sample sets, so an online learning algorithm for LS-SVM is presented. As the wavelet has the property of time-frequency localization and can approximate any complicated functions in $L_2(R)$ space, a wavelet kernel satisfying the wavelet framework is proposed, which enhances the generalization ability of LS-SVM. Finally, the wavelet based online LS-SVM is applied to system identification. Simulation shows that the WOLS-SVM has a much faster convergence and a better generalization performance in comparison with the existing algorithms.

References

1. Vapnik, V.: The Nature of Statistical Learning Theory. New York: Springer-Verlag (1995)
2. Burges, C.J.C.: A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery* **2** (1998) 955-974
3. Drezet, P.M.L., Harrison, R.F.: A New Method for Sparsity Control in Support Vector Classification and Regression. *Pattern Recognition* **34** (2001) 111-125
4. Suykens, J.A.K., Vandewalle, J.: Least Squares Support Vector Machine Classifiers. *Neural Processing Letters* **9** (1999) 293-300
5. Chua, K.S.: Efficient Computations for Large Least Square Support Vector Machine Classifiers. *Pattern Recognition Letters* **24** (2003) 75-80
6. Tsujinishi, D., Abe, S.: Fuzzy Least Squares Support Vector Machines for Multi-class Problems. *Neural Networks* **16** (2003) 785-792
7. Saunders, C., Gammerman, A., Vovk, V.: Ridge Regression Learning Algorithm in Dual Variables. *Proceedings of the 15th International Conference on Machine Learning*, Morgan Kaufmann (1998) 515-521
8. Gestel, T.V., Suykens, J.A.K., Baestaens, D. et, al: Financial Time Series Prediction Using Least Squares Support Vector Machines within the Evidence Framework. *IEEE Trans. on Neural Networks* **12** (2001) 809-821
9. Suykens, J.A.K., Brabanter, J.D., Lukas, L., Vandewalle, J.: Weighted Least Squares Support Vector Machines Robustness and Sparse Approximation. *Neurocomputing* **48** (2002) 85-105
10. Kruif, B.J.d., Vries, T.J.A.D.: Pruning Error Minimization in Least Squares Support Vector Machines. *IEEE Trans. on Neural Networks* **14** (2003) 696-702
11. Mercer, J.: Functions of Positive and Negative Type and Their Connection with the Theory of Integral Equation. *Transactions of the London Philosophical Society*. A-209 (1909) 415-446
12. Smola, A., Schölkopf, B., Müller, K.R.: The Connection between Regularization Operators and Support Vector Kernels. *Neural Networks* **11** (1998) 637-649
13. Burges, C.J.C.: Geometry and Invariance in Kernel Based Methods. *Advance in Kernel Methods-Support Vector Learning*. Cambridge, MA: MIT Press. London (1999) 89-116

14. Zhang, Q.: Using Wavelet Network in Nonparametric Estimation. *IEEE Trans. on Neural Networks* **8** (1997) 227-236
15. Wu, F., Zhao, Y.: Least Squares Littlewood-Paley Wavelet Support Vector Machine. *Lecture Notes in Computer Science* 3789 (2005) 462-472
16. Zhang, L., Zhou, W.D., Jiao, L.C.: Wavelet Support Vector Machine. *IEEE Trans. on Systems, Man, and Cybernetics-Part B: Cybernetics* **34** (2004) 34-37
17. Kuh, A.: Adaptive Least Square Kernel Algorithms and Applications. *Proceedings of International Joint Conference on Neural Networks* (2002) 2104-2107
18. Kuh, A.: Adaptive Kernel Methods for CDMA Systems. *Proceedings of International Joint Conference on Neural Networks* (2001) 1404-1409
19. Ralaivola, L., d'Alche-Buc, F.: Incremental Support Vector Machine Learning: a Local Approach. *Lecture Notes in Computer Science* 2130 Springer-Verlag, Berlin Heidelberg New York (2001) 322-330
20. Narendra, K., Parthasarathy, K.: Identification and Control of Dynamical Systems Using Neural Networks. *IEEE Trans. on Neural Networks* **1** (1990) 4-27

Ultrasound Estimation of Fetal Weight with Fuzzy Support Vector Regression

Jin-Hua Yu¹, Yuan-Yuan Wang¹, Ping Chen², and Yue-Hua Song²

¹ Department of Electronic Engineering, Fudan University, 200433 Shanghai, China
{jhyu, yywang}@fudan.edu.cn

² The First Maternity and Infant Health Hospital, 200433 Shanghai, China
mellonping@yahoo.com.cn

Abstract. Accurate acquisition of expected fetal weight (EFW) based on ultrasound measurements is important to antenatal care. The accuracy of EFW is disturbed by random error of measurements and impropriety of regression method. There have been several studies using neural networks to improve estimation validity, but these methods are all on the premises of measurements accuracy. This paper utilizes the fuzzy logic to deal with the measurements inconsistency, while combines with the support vector regression (SVR) to pursue generalization ability. By this way, the suspect inaccurate measurements can have relatively less contributions to the learning of new fuzzy support vector regression (FSVR). Tests on a clinical database show that proposed algorithm can achieve 6.09% mean absolute percent error (MAPE) for testing group while the back-propagation algorithm and classical SVR achieve 8.95% and 7.23% MAPE respectively. Experimental results show the effectiveness of the proposed algorithm over traditional methods based on neural network.

1 Introduction

In obstetrics, estimation of fetal weight based on ultrasound images plays a key role in prenatal care. Obtaining accurate expected fetal weight (EFW) is of paramount importance in prediction of fetal compromise and in management of labor. Ultrasound is a major tool for fetal weight estimation, due to its noninvasiveness, portability and relatively low cost. In clinical applications, the fetal weight is estimated based on several ultrasound measurements with the regression analysis. The accuracy of EFW is disturbed by two main factors, the one is the random errors in measurements, and the other is the impropriety of regression equations.

On commercial equipment, measurements are made by manually tracing on-screen with a mouse-like device. The poor quality of ultrasound images and the variability of the human observer influence the manual measurement accuracy and consistence. Measurements are then combined to estimate fetal weight by mathematically based non-linear regression analysis. Variety of formulas, incorporating different ultrasonic measurements, has been studied extensively. Dudley ^[1] have evaluated 11 different formulas in a systematic review on the ultrasound estimation of fetal weight, and concluded that no consistently superior method has emerged.

The Artificial neural network (ANN), has been used to estimate the fetal weight^[2, 3]. Compared with the fixed formulas using the regression analysis, the ANN model that develops nonlinear relationship between input variables and output outcomes can reduce the estimate errors. However, the constructed architecture of most ANN models are using back-propagation algorithm which is based on Empirical Risk Minimization (ERM) principle. ERM pursues minimizing the error on the training data, so it may fall into a local optimal solution due to the overtraining problem. Support vector machine (SVM), which is developed by Vapnik *et al*^[4], are gaining popularity because of many attractive features. SVM is based on statistic learning theory and the Structural Risk Minimization (SRM) principle which pursues minimizing the upper bound on the expected risk. The SRM principle equips SVM with a good ability to generalize and better classification precision. Thus, at present, support vector regression (SVR) has been applied to estimate fetal weight^[5]. Song *et al*^[5] indicate that, compared with another classical neural network—Radial-Basis Function (RBF), SVR is superior in both fitting and testing accuracy. The principle of SRM makes the algorithm lay particular stress on the data close to boundary (support vector). Because data close to the class boundary is most affected by outliers, the classical SVM formulation is not robust to noise^[6]. As mentioned above, in real clinical applications, fetal measurements based on ultrasound images are often inaccuracy and inconsistency. Previous works of [2], [3] and [5] are all on the premises of measurements accuracy.

If we take the considerations of data inconsistencies into model training, the accuracy of fetal weight estimation may be prospectively improved. Fuzzy logic is a powerful tool to deal with uncertain, nonlinear and ill-posed problem. Fuzzy support vector regression (FSVR) with a new membership calculation method is presented in this paper to deal with fetal weight estimation.

The remainder of this paper is organized as follows: Section 2 provides an introduction about the standard SVR and FSVR. In Section 3, we present in detail the calculation method of membership function for each training data and the implementation of FSVR. The data acquisition and the experimental results are presented in Section 4. Section 5 concludes the paper.

2 SVR and FSVR

This section provides an introduction about standard SVR and FSVR.

2.1 SVR^[4]

The basic idea of SVM is mapping data into a high-dimensional space and constructing optimal hyperplane with maximum margin between the two classes. The successful applications of SVM in pattern recognition make the same principle utilized in the nonlinear regression. Suppose that there is a m -dimensional training data set containing n data points $\{\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n\}$ and the corresponding expected values

of the model output $\{d_1, d_2, \dots, d_n\}$. The estimate of d , denoted by y , is expanded in terms of a set of nonlinear basis function as follow

$$y = \bar{w}^T \bar{\phi}(\bar{x}). \tag{1}$$

where \bar{w} is the weight vector, and ϕ is the nonlinear mapping from the input space to the high dimensional feature space. The constrained optimization problem can be defined by introducing two sets of nonnegative slack variables,

$$d_i - \bar{w}^T \bar{\phi}(\bar{x}_i) \leq \varepsilon + \xi_i, \quad i = 1, 2, \dots, N; \tag{2}$$

$$\bar{w}^T \bar{\phi}(\bar{x}_i) - d_i \leq \varepsilon + \xi'_i, \quad i = 1, 2, \dots, N; \tag{3}$$

$$\xi_i, \xi'_i \geq 0, \quad i = 1, 2, \dots, N. \tag{4}$$

The slack variables ξ_i and ξ'_i describe the ε -insensitive loss function. This constrained optimization problem can be solved by minimizing the cost functional

$$\min_{\bar{w}, b, \xi, \xi'} C \left(\sum_{i=1}^n (\xi_i + \xi'_i) \right) + \frac{1}{2} \bar{w}^T \bar{w}. \tag{5}$$

where C is a regularization constant, controlling a compromise between maximizing the margin and minimizing the number of training set errors.

2.2 FSVR

In order to decrease the effect of outliers or noise, in FSVR ^[7], the fuzzy membership is associated with data points such that different data points can have different contributions to the learning of regression function. If one data point is detected as an outlier, it is assigned with a low membership, so its contribution to total error term decreases.

Suppose u_i represents the attitude of the corresponding training point toward the error term. The FSVR is modeled by the following programming

$$\min_{\bar{w}, b, \xi, \xi'} C \left(\sum_{i=1}^n u_i (\xi_i + \xi'_i) \right) + \frac{1}{2} \bar{w}^T \bar{w}. \tag{6}$$

subject to the constraints of Eq. (2) to (4). By introducing Lagrange multipliers α, α', γ and γ' , the Lagrangian function is defined to solve the Eq. (6), and leads to a solution of the form

$$y = \sum_{i=1}^n (\alpha_i - \alpha'_i) \bar{\phi}(\bar{x}) + b; \tag{7}$$

$$\gamma_i = C - \alpha_i, \quad \gamma'_i = C - \alpha'_i. \tag{8}$$

In Eq. (7), α and α' are obtained by maximizing the dual function which has the following form

$$Q(\alpha_i, \alpha'_i) = \sum_{i=1}^n d_i(\alpha_i - \alpha'_i) - \varepsilon \sum_{i=1}^n (\alpha_i + \alpha'_i) - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n (\alpha_i - \alpha'_i)(\alpha_j - \alpha'_j)K(\bar{x}_i, \bar{x}_j). \tag{9}$$

subject to the following constraints

$$\sum_{i=1}^n (\alpha_i - \alpha'_i) = 0; \tag{10}$$

$$0 \leq \alpha_i, \alpha'_i \leq \mu_i C. \tag{11}$$

The above problem is a quadratic programming problem. There are several classical methods [8] to solve this problem. $K(\bar{x}_i, \bar{x}_j) = \bar{\phi}(\bar{x}_i)^T \bar{\phi}(\bar{x}_j)$ is the kernel function. There are several typical kernel functions are usually used in SVR. In this work, the radial basis function (RBF) is used,

$$K(\bar{x}_i, \bar{x}_j) = \exp\left(-\frac{\|\bar{x}_i - \bar{x}_j\|^2}{\sigma^2}\right). \tag{12}$$

3 Implementation of FSVR for EFW

Fetal weight is estimated based on several ultrasound measurements. The standard measurements which have been well standardized include: fetal biparietal diameter (BPD), head circumference (HC), abdominal circumference (AC) and femur length (FL) [1]. The accuracy of measurements is compromised by poor quality of ultrasound images and the large intra- or interobserver variabilities. Thus, the data for model constructing contains much inaccurate and inconsistent information which can be viewed as outliers. To reduce the effect of the outliers in model forming, the low membership value should be assigned to the suspect outliers. The key steps before solving the quadratic programming in Eq. (6) are detecting the outliers and generating membership which falls in the unit interval [0, 1]. Membership generating method itself is required to have a good data regression ability, which means the method itself can be used to estimate the fetal weight. Then the estimation performance can be improved by using the FSVR.

It is relatively straightforward to come to classical SVR or back-propagation neural network (BPNN). The pre-training results of SVR or BPNN can be used for outlier detection. Once the termination of FSV or BPNN, a primary estimation (denoted by y_i) is obtained for each input data \bar{x}_i . Compute the distance (denoted by $dist_i$) between y_i and its corresponding goal d_i , then use following formula to compute the membership:

$$\mu_i = \frac{dist_i}{\max_{i=1,2,\dots,n} dist_i + D}. \tag{13}$$

Here, D is a constant with the range $(0, +\infty)$, which is used to control the effect of fuzziness to the model training. When D is large enough, the FSVR degenerates to classical SVR. The procedure of FSVR can be summarized in the following steps:

Step 1: Choose the training and testing data set. Normalize the data.

Step 2: Obtain the membership for each training data.

- a) Training the SVR or BPNN.
- b) Compute the distance between every output and its corresponding goal.
- c) Compute the membership using Eq. (13).

Step 3: FSVR

- a) Solving the quadratic programming problem of Eq. (9) under the constraints of (10) and (11).
- b) Obtain the EFW using Eq. (7).

4 Data Acquisition and Experimental Results

The data for this study consists of 100 normal fetuses, who delivered with a birth-weight between 1850~4265 g, in Shanghai First Maternity and Infant Health Hospital from 3 July to 13 October 2006. In order to analyze the accuracy of our model, the actual birth weights (ABW) are immediately obtained after delivery and recorded. The commercially available 2-D ultrasound scanner (PHILIPS 2540A) with a 3.5 MHz trans-abdominal probe is used in this study. We perform experiments on a PC with 2.0 GHz Pentium processor using Matlab 7.0. The 100 samples are randomly separated into five groups, and four groups are randomly chosen as training set, one as testing set.

Next, we discuss comparisons of the prediction results obtained by BPNN, classical SVR and FSVR. The architecture of the BPNN model in our work is composed of three layers, i.e. input layer with four inputs, hidden layer with five neurons and output layer. The BPNN is constructed as shown in Fig.1. In order to avoid the unsteadiness of BPNN, training procedure is carried out five times and the output values are averaged. The convergence condition of BPNN is the neuron weight variation is less than 0.001 and the upper bound of iteration are 2000. The activation function used in BPNN is sigmoidal nonlinearity. We use general RBF as the kernel

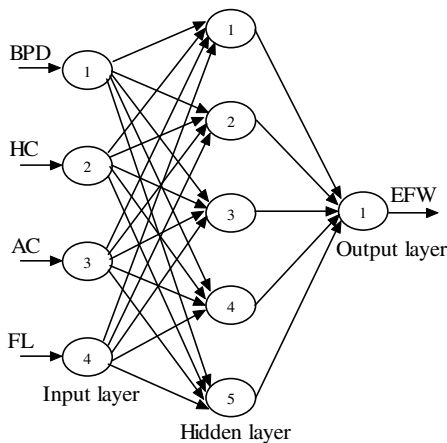


Fig. 1. The architecture of BPNN

function in SVR and FSVR with $\sigma=10$. The D in (13) equals to 0.02. The values of C in (11) and ε in (9) are chosen based on the cross-validation results. The final chosen C is 10^4 for SVR and 10^2 for FSVR respectively, and the chosen ε is 10^{-2} in this study.

The distribution of the actual birth weight and the estimate weight of the training group and the testing group are shown in Fig.2 to 4. In these figures, upper row is the distribution on the training group and lower row is on the testing group. To evaluate the performance of the proposed algorithm quantitatively, we calculate the absolute percent error (APE) of three algorithms on five testing groups. Table.1 gives the mean absolute percent error (MAPE) and the error ranges.

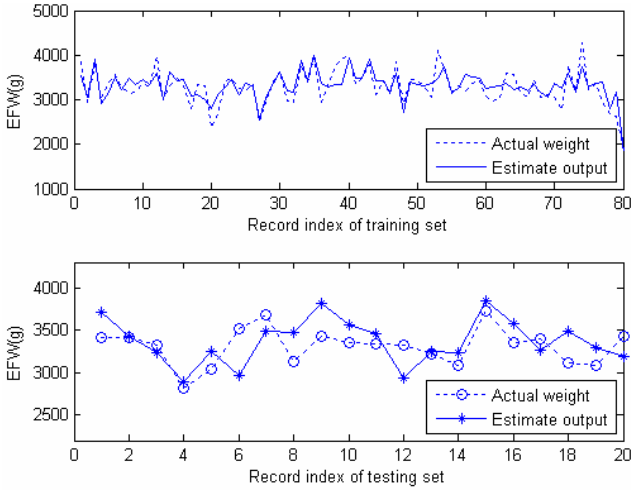


Fig. 2. BPNN results

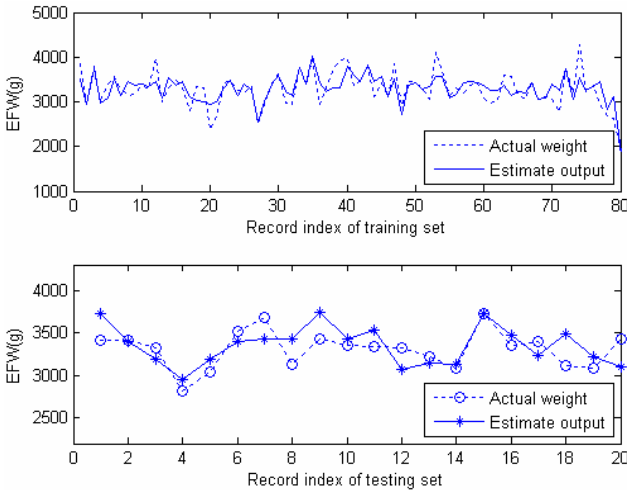


Fig. 3. SVR results

From the results illustrations, we can see that the BPNN has poorer generalization ability than SVR and FSVR. The overtraining problem makes the BPNN can almost memorize every input data, thus it has the relatively high fitting accuracy while the low predicting accuracy. In FSVR, the contributions of some training data to the regression learning have been inhibited over the membership function. In this way, the fitting accuracy of FSVR decreases slightly, but FSVR achieves higher generalization abilities than SVR. The proposed method produces less accurate estimation at the extremes of the weight range (<2000 g or >4000 g, the most of fetal weight ranges from 2000 g to 4000 g). The main reason for lower accuracy might be the scarcity of cases in extreme fetal weight. FSVR model also can be build to improve the estimation accuracy in extreme fetal weight if enough cases are collected.

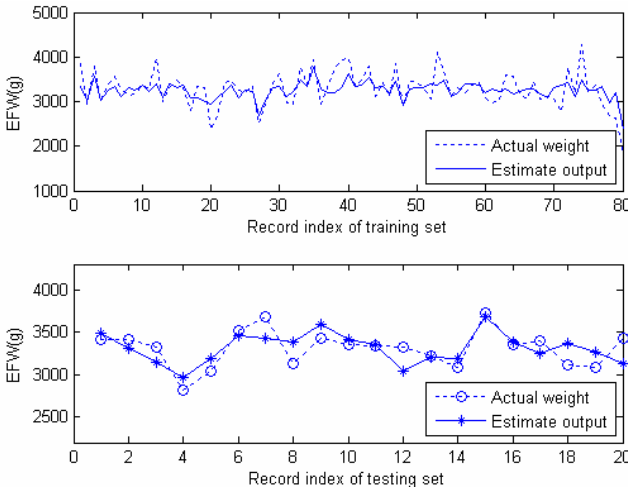


Fig. 4. SFVR results

Table 1. APE comparisons among three algorithms

APE(%)	BP	SVR	FSVR
mean	8.95	7.23	6.09
maximum	22.69	15.08	11.89
minimum	0.65	0.29	0.19

From above experimental results, we can conclude that the FSVR model can improve ultrasound estimation of fetal weight over other methods based on BPNN and classical SVR model.

5 Conclusion

This paper presents a new fetal weight estimation algorithm based on a fuzzy support vector regression in which every training data is pre-estimated and evaluated, then the

contribution degree (membership) of each training data to the final estimation training is obtained. By utilizing the fuzzy logic, each of the input training samples has different contribution to the learning of regression function. By this means, the inaccuracy and inconsistency of ultrasound measurements are taken into account in the FSVR model construction. Because the proposed algorithm alleviates the effect of two main disturbing factors in fetal weight estimation, random error in measurements and the impropriety of regression equations, it can achieve more accurate fetal weight estimation compared with BPNN and classical SVR. For improving the estimation accuracy on extreme weight fetus, much more cases and further researches will be needed.

Acknowledgement

This work was supported by the National Basic Research Program of China (No. 2006CB705700), Natural Science Foundation of China (No.30570488) and Shanghai Science and Technology Plan (No.054119612).

References

1. Dudley, N.J.: A Systematic Review of the Ultrasound Estimation of Fetal Weight. *Ultrasound Obstet. Gynecol.* **25** (2004) 80-89
2. Farmer, R.M., Medearis, A.L., Hirata, G.I., Platt, L.D.: The Use of A Neural Network for the Ultrasound Estimation of Fetal Weight in the Macrosomic Fetus. *AM J Obstet Gynecol.* **166** (1992) 1467-1472
3. Chuang, L., Hwang, J.Y., Chang, C.H., Yu, C.H., Chang, F.M.: Ultrasound Estimation of Fetal Weight with the Use of Computerized Artificial Neural Network Model. *Ultrasound in Med. & Biol.* **28** (2002) 991-996
4. Vapnik, V.N.: *Statistical learning theory*, Wiley, New York (1998)
5. Song, X.F., Han, P., Zou, L., Chen, D.Z., Hu, S.X.: A New Method for Estimation of Fetal Weight Using Support Vector Machine. *Chinese Journal of Biomedical Engineering.* **23** (2004) 516-522
6. Zhang, X.: Using Class-center Vectors to Build Support Vector Machines. *Proc. IEEE NNSP.* (1999) 3-11
7. Lin, C.H., Wang, S.D.: Fuzzy Support Vector Machines. *IEEE Trans on Neural Networks.* **13** (2002) 464-471
8. Mangasarian, O.L., David R.: Successive Over-relaxation for Support Vector Machines. *IEEE Trans on Neural Networks.* **10** (1999) 1032-1037

Hybrid Support Vector Machine and General Model Approach for Audio Classification

Xin He¹, Ling Guo¹, Xianzhong Zhou², and Wen Luo¹

¹ School of Automation, Nanjing University of Science and Technology,
Nanjing, Jiangsu, 210094, P.R. China

[kernel_he, guoling876, luowen203}@hotmail.com](mailto:{kernel_he, guoling876, luowen203}@hotmail.com)

² School of Management Engineering, Nanjing University,
Nanjing, Jiangsu, 210093, P.R. China

zhouxz@nju.edu.cn

Abstract. In recent years, the searching and indexing techniques for multimedia data are getting more attention in the area of multimedia databases. As many research works were done on the content-based retrieval of image and video data, less attention was received to the content-based retrieval of audio data. Audio is one of important multimedia information and there is a growing need for automatic audio indexing and retrieval techniques in recent years. Audio data contain abundant semantics and the audio signal processing can reduce computational complexity, so effective and efficient indexing and retrieval techniques for audio data are getting more attention. In this paper, problems of audio retrieval are discussed firstly. Then, main audio characteristics and features are introduced. Finally the combination of Support Vector Machine and General Model is described and the hybrid model is used in audio retrieval. Experiments show that the hybrid model is effective for audio classification.

1 Introduction

The recent explosive growth in multimedia and capacity of the internet and computer has allowed the amount of data becomes more and more. It's significance for information utility to find useful information from huge database and use them to classification and retrieval, which requires advanced techniques for content-based multimedia information retrieval. Audio data is one of most important information in multimedia. Audio retrieval is becoming one of important methods in multimedia information retrieval [1].

There are two types in audio information retrieval. One is based on threshold and the other is based on statistical model [2]. There is no threshold problem in the method based on statistical model and it can be used to retrieval refined levels. So the method based on statistical model is used in this paper. General Model(GM) [3] and Support Vector Machine(SVM) both belong to the method based on statistical model. GM is equal to deal with continuous signal and the semblance among same class data can be reflected in the result. SVM is fit for

classification problems and the result of SVM can be shown the difference of different data. It needs enough data in GM, but in application data are finite. SVM is a powerful new machine learning algorithm, which is rooted in statistical learning theory and can solve classification problems in small sample. Furthermore, audio data are not fixed length signal, SVM is a statistic classifier and can solve fixed length problems. But GM can be used to solve dynamic characteristics of audio data. In this paper, virtues of GM and SVM are combined and the hybrid model of GM and SVM is presented to solve audio classification. The rest of this paper is organized as follows. In Section 2 some useful audio features are introduced. Some basic theories of General Model are introduced in Section 3. Section 4 describes basic theories of Support Vector Machines. In Section 5, the method of hybrid General Model and Support Vector Machine approach for Audio Classification is presented. Finally, in Section 6, experiments and evaluations on audio data are given.

2 Audio Features

Audio data is one of most important information in multimedia. There are two main methods for audio feature extraction. One is to extract perception features, such as pitch and loudness, etc. The other is non perception features or called physical features, such as frequency cepstral coefficients and linear prediction coefficients, etc.

2.1 Zero-Crossing Rate(ZCR)

In the context of discrete-time signals, a zero-crossing is said to occur if successive samples have different signs. The rate at which zero-crossings occur is a simple measure of the frequency content of a signal, which is described as below:

$$ZCR = \frac{1}{2} \sum_{m=1}^{N-1} |sgn[x(m+1)] - sgn[x(m)]|, \quad (1)$$

where N is denoted as the frame length; $x(m)$ is the value of m -th sampled signal; $sgn[x(m)] = \begin{cases} 1, & x(m) \geq 0; \\ -1, & x(m) < 0. \end{cases}$

In a general way, ZCR of consonant signals are lower than those of vowel signals. And there are many consonant signals at the beginning or end of speech signals, then there are correspondingly many vowel signals after the beginning or before the end of speech signals. So ZCR are changed remarkably at parts of the beginning and the end of speech signals, and they can be used to judge the beginning or the end of speech signals. Moreover, structures of music signals are different from these, so changes of music signals ZCR are smaller than those of speech signals.

2.2 Frequency Energy

The calculation of the frequency energy is described as:

$$E = \log\left(\int_0^{\omega_0} |F(\omega)|^2 d\omega\right), \tag{2}$$

where E is presented as the frequency energy of audio frame; $F(\omega)$ is the transformation coefficient of the Fast Fourier Transform Algorithm; ω_0 is half of the audio sampling frequency.

The frequency energy can be used to describe the intension of audio signals, so it's one of effective features to distinguish between speech and music.

2.3 Loudness

It is one of perception features concerned with frequency. Especially in speech recognition, it is one of important parameters to distinguish male and female.

2.4 Mel-Scaled Frequency Cepstral Coefficients

Audio data are processed with Z-transform and logarithm processing, and then MFCCs can be obtained. 12-order MFCCs are usually used because of its good discriminating ability.

3 Basic Theory of GENERAL MODEL(GM)

In methods of signal processing, Hidden Markov Model(HMM) and Dynamic Time Warping(DTW) are very important. In fact, there are internal relations between HMM and DTW [3-4]. In paper [3] and [4], relations between HMM and DTW are described in detail, and they are unified into one model, named as General Model(GM). Experiments of these papers are indicated that GM can solve problems which can be solved by HMM and DTW.

The GM is defined as below:

$$\Psi = \{D, A, f\}, \tag{3}$$

where Ψ is a triple, including a directed graph D , a phalanx A which is composed of weighted values of every directed arc, and a matching mapping.

The acme set of D is $P = \{p_0, \dots, p_m, \dots, p_M\}$, where p_0 is denoted as starting acme with output directed arc and no input directed arc; p_M is termination acme with input arc and no output arc; p_m is traverse acme with both input and output directed arc, just shown in Fig. 1.

A is a phalanx in $M+1$ dimension, which is composed of weighted values of every directed arc in D , namely a_{ij} is the weighted value between p_i and p_j ($i, j = 0, 1, \dots, M$). If there is no directed arc between p_i and p_j , then $a_{ij} = \infty$. f is a matching mapping:

$$f : P \times X \rightarrow R, \tag{4}$$

where $f(p_i, x_j)$ is denoted the matching weighted values of acme p_i and feature vector x_j .

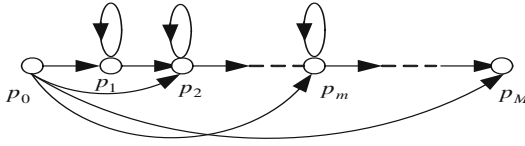


Fig. 1. Directed graph D

4 Basic Theory of SUPPORT VECTOR MACHINE (SVM)

The SVMs are proposed recently by Vapnik[5], which is rooted in statistical learning theory[6]. By constructing a decision surface hyper-plane which yields the maximal margin between position and negative examples, SVM approximately implements the Structure Risk Minimization(SRM) Principle. There are three layers in the structure of SVM: input layer, which gets data just as classification characters; hidden layer, with two functions, namely mapping the input data from low-dimension space to high-dimension by non-linear map and calculating the inner production of character vector and support vector; output layer, just showing the classification results. In the practical application, the function of hidden layer is achieved by the kernel function. The kernel function can be formulated as follows:

$$K(x, y) = (\Phi(x) \bullet \Phi(y)), \quad (5)$$

where K, Φ, \bullet denotes respectively kernel function, non-linear mapping in high-dimension and inner production.

As well known, there are three main types in kernel functions: Polynomial: $K(x, y) = (x \bullet y + 1)^d$, where d is the degree of the polynomial; Gaussian Radial Basis Function: $K(x, y) = \exp(-\frac{\|x-y\|^2}{2\sigma^2})$, and the parameter σ is the width of Gaussian Function; Sigmoid Function: $K(x, y) = \tanh(k(x \bullet y) - \mu)$, k and μ are the scale and offset parameters. In this paper, the Sigmoid Function is used, just because it includes only one hidden multi-layer perception, the number of hidden-layer nodes is confirmed by the algorithm and there is no local minima problem, which exists in the method of Network Neural.

5 Audio Classification Method Based on SVM/GM

The hybrid model includes two parts: training and testing. Firstly, audio features of training sets can be obtained and used to classification by SVM; secondly, SVM classification results are as inputs of GM and trained by GM; finally, GM of every audio class can be obtained and labelled with semantic information, which just is template of every audio class. For observation sequence $\bar{O} = o_1, o_2, \dots, o_M$, which are audio data for classification, Firstly, audio features of training sets can be obtained and used to classification by SVM; secondly, SVM classification

results are trained by GM; finally, these results are compared with those available template and classification results can be obtained.

But the output of SVM is numerical value. For combined with GM easily, it should transform the value to probability form[7-10]. In the processing of calculating, it should make training sample normalization, namely $|g(x)| = 1$. Then the sample point can be described as $g(x) = \pm dw$, where d denotes the distance between sample point x and classification plane, and the sign denotes which side the point position. The probability output form is formulated as follows:

$$P(C_{\pm 1}|x_i) = \frac{1}{1 + \exp(\mp g(x_i))}, \quad (6)$$

5.1 Training of SVM

SVM is a two-class classifier, it should be used to a multi-class classification in audio classification. There are two common schemes for this purpose: one-against-all and the one-against-one. In this paper, the one-against-all is used. The detailed processing is described as follows:

Supposing there are m classes audio in audio sample database, and there are n training sample of each class.

(1) Selecting all sample of the first audio from sample database, labelling as class I and other audio classes as class II. Using these samples as input to training a SVM and obtaining the corresponding support vector and optimal classification plane. Then labelling the SVM as ①, which can distinguish class I and other audio classes.

(2) Selecting all sample of the second audio from sample database, labeling as class I and rest audio classes as class II. Using these samples as input to training a SVM and obtaining the corresponding support vector and optimal classification plane. Then labelling the SVM as ②, which can distinguish the second class and other audio classes.

(3) Repeating steps described above, training all audio classes and achieving m SVMs.

5.2 Training of GM

As said above, there are m SVMs. Supposing the state number of GM is M , then setting $M - 1$ initial state center register and vector counter respectively. Segmenting observation sequences by Fisher Algorithm, accumulating center register and vector counter of every part and getting corresponding weighted value matrix. Finally, achieving model of all training audio classes.

5.3 Audio Classification of the Hybrid Model

Supposing there are m classes audio in sample database, which is for classification. And there are trained templates according to each audio class. When a new

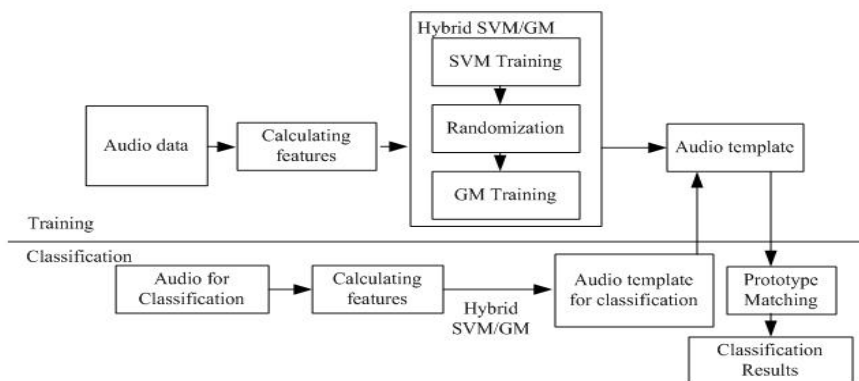


Fig. 2. Hybrid Model of SVM/GM

sample for classification is input, it can be classified by the method described as above. It's formulated as Fig. 2.

6 Experiments

In order to test effects of the use of the SVM/GM, small sample database are used in experiments, including background music, silence and speech. In experiments, these audio data are collected from TV program or video clips, saved as ".wav" and sample rate at 16 kHz. The database is partitioned into a training set and a testing set, including 60 clips for each audio class, which 40 for training and 20 for testing. Classification results can be obtained, which is described in Table 1. And in Table 2, there are some compares between SVM/GM and HMM.

Table 1. Experiment Results of Classification

Audio Class	Pure speech	Speech with music	Silence	Music	Acc.(%)
Pure speech	19	1	0	0	95
Speech with music	1	17	0	2	85
Silence	0	0	20	0	100
Music	0	2	0	18	90

Table 2. Compare Results

Acc.(%)	Pure speech	Speech with music	Silence	Music
SVM/GM	95	85	100	90
HMM	95	80	95	90

7 Conclusions

Audio data is one of most important information in multimedia. It's always based on audio features for audio classification. Audio feature extraction becomes one of emphases in audio classification. In this paper, classification problems of audio data are discussed. In this paper, it presents in detail the approach that uses SVM/GM for classification of an audio clip. And experiments have shown that the hybrid SVM/GM achieves high classification accuracy. As for future research, we will improve this method for more audio classes and take emphases on feature set definition.

Acknowledgments. This research was sponsored by the Nature Science Foundation of Jiangsu Province, P.R. China (BK2004137). The authors are very grateful to Dr. Bing Huang who gives many valuable comments and the anonymous referees for their helpful comments.

References

1. Erling Wold, et al.: Content based classification search and retrieval of audio, *IEEE Multimedia* (1996) 27-36
2. Zhuang, Y., Pan, Y. and Wu F.: *Web-based Multimedia Information Analysis and Retrieval*, Beijing: Tsinghua University Press (2002)
3. Ma, M.: Study on the Several Problems in Speech Recognition, Nanjing University of Science and Technology Ph.D. Dissertation (1996)
4. Juang, B.H.: On the Hidden Markov Model and Dynamic Time Warping for Speech Recognition-A Unified view, *AT&T Bell Laboratories Technical Journal* 63 (7) (1984) 1213-1243
5. Vapnik, V.: *Statistical Learning Theory*, New York: Springer Verlag (1998)
6. Burges, J.C.: A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery* 2(2) (1998) 121-167
7. Xin, D., Yang, Y. and Wu, Z.: Speaker Verification with the Hybrid Use of Support Vector Machine and Hidden Markov Model, *Journal of Computer-Aided Design and Computer Graphics* 14 (2002) 1080-1082
8. Ganapathiraju, A., Hamaker, J. and Picone, J.: Hybrid SVM/HMM architectures for speech recognition, [http://www.isip.msstate.edu/publications/conferences/icslp/2000/hybrid asr/paper v0. pdf](http://www.isip.msstate.edu/publications/conferences/icslp/2000/hybrid%20asr/paper%20v0.pdf) (2002)
9. Kwok, J.T.: Moderating the outputs for support vector machine classifiers, *Proceedings of International Joint Conference of Neural Networks*, Washington DC (1999) 943-948
10. Platt, J.: Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods, In *Advances in Large Margin Classifiers*, MIT Press, Cambridge, MA, USA (2000)

An Improved SVM Based on 1-Norm for Selection of Personal Credit Scoring Index System

Xin Xue^{1,2} and Guoping He¹

¹ College of Information Science and Engineering, Shandong University of Science and Technology, Qingdao Shandong 266510, China

xuexin04@163.com

² Department of Mathematics & System Science, Taishan University, Taian Shandong 271021, China

Abstract. The selection of evaluating index system is the key to personal credit scoring, which is a feature selection problem. By improving the typical SVM based on 1-norm, which can select the important and necessary feature of samples, an improved SVM based on 1-norm adapted to the selection of personal credit scoring index system is proposed. Experimental results show that the new improved method can select evaluating index system with small scale and enhance the generality ability and reduce the arithmetic complexity of the classification machine.

1 Introduction

Personal credit scoring is the base of personal credit consume. Originally, personal credit scoring was conducted on personal experience. Subsequently, the 3C evaluating principle (character, capacity, collateral) was introduced in personal credit scoring. Lately, more and more mathematic methods are applied in personal credit scoring [1]. There are k-nearest-neighbor [2], discriminant analysis [3], expert system [4], mathematic programming [5], regression analysis [6], neural network [7], etc. Now, more and more experts and researchers are applying SVM (support vector machine) in personal credit scoring [8, 9].

SVM is a new and very promising classification technique developed by Vapnik and his groups at AT&T Bell Laboratories on statistics learning theory [10]. The main idea behind the technique is to separate the classes with a surface that maximizes the margin between them. An important property of this approach is that it is an approximate implementation of the Structural Risk Minimization (SRM) induction principle. SVM classification is currently an active research area and successfully solves classification problem in many domains.

The selection of evaluating index system is the key to personal credit scoring. A good classification machine comes from an appropriate evaluating index system. There are a number of different approaches to feature selection, such as the filter approach, the wrapper approach [11], etc. This paper introduces another approach for feature selection, SVM based on 1-norm [12], which can select the

important and necessary feature of samples effectively and rapidly. In order to reduce the risk of loan, we propose an improved SVM based on 1-norm adapted to the selection of personal credit scoring index system.

This paper is organized as follows. In section 2, we give the Structural Risk Minimization induction principle. In section 3, we introduce the SVM based on 1-norm. In section 4, by using different penalty number to different class, we propose an improved SVM based on 1-norm adapted to the selection of personal credit scoring index system and gives the algorithm of the selection of personal credit scoring index system via the improved SVM based on 1-norm. In section 5, the property of the method as above is testified with numerical testing.

2 Structural Risk Minimization Induction Principle

In the case of two-class pattern recognition, the task of learning from examples can be formulated in the following way [10, 12, 13] : given a set of examples:

$$\{(x_1, y_1), \dots, (x_l, y_l)\}, x_i \in X \subset R^n, y_i \in Y = \{-1, 1\}, \quad (1)$$

drawn from some unknown probability distribution $P(x, y)$ (These data is independently drawn and identically distributed), and a loss function $c(x, y, f(x))$, we want to find a hypothesis f from a set of hypotheses (decision functions):

$$F \subset \{f : X \subset R^n \longrightarrow Y = \{-1, 1\}\}, \quad (2)$$

which provides the smallest possible value for the expected risk:

$$R[f] = \int_{X \times Y} c(x, y, f(x)) dP(x, y) \quad (3)$$

Since the probability distribution $P(x, y)$ is unknown, we are unable to compute the expected risk $R[f]$. We can compute the stochastic approximation of $R[f]$, the so called empirical risk:

$$R_{emp}[f] = \frac{1}{l} \sum_{i=1}^l c(x_i, y_i, f(x_i)), \quad (4)$$

Since the law of large numbers guarantees that the empirical risk converges on probability to the expected risk, a common approach consist in minimizing the empirical risk rather than the expected risk. Empirical Risk Minimization induction principle is to find a hypothesis f from the hypothesis space F , which provides the smallest possible value for the empirical risk $R_{emp}[f]$.

Vapnik and Chervonenkis showed that necessary and sufficient condition for consistency of the Empirical Risk Minimization induction principle is the finiteness of the VC-dimension h of the hypothesis space F . The VC-dimension of the hypothesis space F is a natural number, possibly infinite, which is the largest number of training data points that can be shattered in all possible ways by

the hypothesis space F . When the number of samples converges to infinite, and the VC-dimension of the hypothesis space F is finite, the minimum of $R_{emp}[f]$ converges to the minimum of $R[f]$. When the number of samples is finite, the Empirical Risk Minimization induction principle can be replaced by a better Structural Risk Minimization induction principle as the following:

If $l > h$, and $h(\ln\frac{2l}{h} + 1) + \ln\frac{4}{\delta} \geq \frac{l}{4}$, for some probability distribution $P(x, y)$, some given δ such that $0 \leq \delta \leq 1$, and some hypothesis $f \in F$, the typical and uniform Vapnik and Chervonenkis bound, with probability $1 - \delta$, has the following form:

$$R[f] \leq R_{emp}[f] + \sqrt{\frac{8}{l}(h(\ln\frac{2l}{h} + 1) + \ln\frac{4}{\delta})}, \tag{5}$$

The second term on the right hand side of inequality (5) is called the VC confidence. The right hand side of inequality (5) is called the structural risk bound. The structural risk bound is independent of $P(x, y)$. If we known h , we can easily compute the the structural risk bound. From this bound, it is clear that, in order to achieve small expected risk, both the empirical risk and the VC confidence have to be small, which is the Structural Risk Minimization induction principle.

3 SVM Based on 1-Norm(1-SVM)

We consider the problem of classifying l points in the n -dimensional real space R^n , represented by the $l \times n$ matrix A , according to membership of each point x_i in the class 1 or class -1 as specified by a given $l \times l$ diagonal matrix D with plus ones or minus ones along its diagonal¹⁰. In the separable case, see Figure 2, the SVM simply looks for the separating hyperplane $x^T w + b = 0$, with the largest margin $2/\|w\|'$. $\|\cdot\|'$ denotes the dual norm. In the non-separable case, see Figure 3, the bound plane $l_3 : x^T w + b = 1$ bounds the class 1 points, possibly with some error, and the bound plane $l_2 : x^T w + b = -1$ bounds the class -1 points, also possibly with some error. So, we introduce the nonnegative slack variant $\xi_i, i = 1, \dots, l, \xi = (\xi_1, \dots, \xi_l)^T$. The support vector machine is given by the following quadratic program with penalty number $C > 0$:

$$\begin{aligned} \min_{w,b,\xi} \quad & \|w\|' + Ce^T \xi, \\ \text{s.t.} \quad & D(Aw + eb) + \xi \geq e, \\ & \xi \geq 0. \end{aligned} \tag{6}$$

Arbitrary norm can be used in model(6)[14]. If we use ∞ -norm, since $\|w\|'_\infty = \|w\|_1$, the optimization problem based on 1-norm is as follows:

$$\begin{aligned} \min_{w,b,\xi} \quad & \|w\|_1 + Ce^T \xi, \\ \text{s.t.} \quad & D(Aw + eb) + \xi \geq e, \\ & \xi \geq 0. \end{aligned} \tag{7}$$

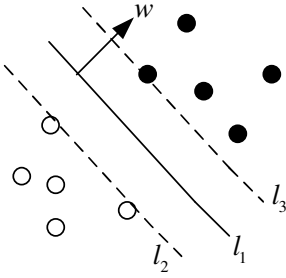


Fig. 1. The separable case

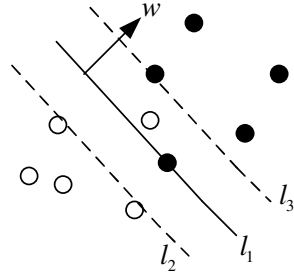


Fig. 2. The non-separable case

Model (7) can select and suppress the problem feature efficiently [15]. If we use 2-norm in model (6), since $\|w\|_2' = \|w\|_2$, the optimization question of SVM based on 2-norm is as follows:

$$\begin{aligned}
 \min_{w,b,\xi} & \frac{1}{2}\|w\|_2^2 + Ce^T\xi, \\
 \text{s.t.} & D(Aw + eb) + \xi \geq e, \\
 & \xi \geq 0.
 \end{aligned}
 \tag{8}$$

Real-world problem are usually nonlinear separable in nature. In order to generalize the above methods to the case where the decision tree is nonlinear function of data, now suppose we first mapping the data x in the input space X to some other (possibly infinite dimension) Euclidean space H , using a mapping which is called $\varphi:R^n \rightarrow H$, where we can find the linear separating hyperplane. If there are a kernel function K [10] such that $K(x_i, x_j) = (\varphi(x_i), \varphi(x_j))$, we would only need to use K in the training algorithm and would never need to explicitly even known what φ is. The kernels commonly used is the followings:

$$k(x, y) = (x \cdot y + 1)^d, \tag{9}$$

$$k(x, y) = e^{-\|x-y\|^2/2\sigma^2}, \tag{10}$$

$$k(x, y) = \tanh(\kappa(x \cdot y) - v), \kappa > 0, v < 0. \tag{11}$$

Eq.(9) results in a classifier that is a polynomial of degree p in the data; Eq.(10) gives a Gaussian radial basis function classifier, and Eq.(11) gives a particular kind of two-layer sigmoidal neural network.

By replacing the $(x_i \cdot x_j)$ of the dual problem of model(8) with $K(x_i, x_j) = (\varphi(x_i), \varphi(x_j))$, the typical nonlinear SVM is given by the following quadratic program:

$$\begin{aligned}
 \min_{\alpha} & \frac{1}{2}\alpha^T DK(A, A^T)D\alpha - e^T\alpha, \\
 \text{s.t.} & e^T D\alpha = 0, \\
 & 0 \leq \alpha \leq C.
 \end{aligned}
 \tag{12}$$

By using the model(12) and the equation, $b = y_j - \sum_{i=1}^l \alpha_i y_i K(x_i, x_j)$ (Here, (x_j, y_j) is some support vector for which $0 < \alpha_j < C$), we get the classification machine:

$$f(x) = \text{sgn}\left(\sum_{i=1}^l \alpha_i y_i K(x_i, x) + b\right). \tag{13}$$

Although Model (12) cannot select and suppress the problem feature, it owns better generalization and stronger classification than model (7).

4 An Improved SVM Based on 1-Norm Adapted to the Selection of Personal Credit Scoring Index System (1 – SVM $^\rho$)

It is worse to classify a customer as good when they are bad than it is to class a customer as bad when they are good. So, we improve the model (7) by adjusting penalty number with parameter $\rho > 0$, and propose an improved SVM based on 1-norm adapted to the selection of the personal credit scoring index system.

At first, we introduce the nonnegative slack variant $\xi^+ = (\xi_1^+, \dots, \xi_l^+)^T$ to class 1(Good Credit), $\xi^- = (\xi_{k+1}^-, \dots, \xi_l^-)^T$ to class -1(Bad Credit), $\xi_i^+ \geq 0, \xi_j^- \geq 0, i = 1, \dots, k, j = (k + 1), \dots, l$. Using different penalty number to class 1 and class -1, an improved SVM based on 1-norm adapted to the selection of the personal credit scoring index system is as follows:

$$\begin{aligned} \min_{w, b, \xi^+, \xi^-} & \|w\|_1 + C(e^T \xi^+ + \rho e^T \xi^-), \\ \text{s.t.} & A^+ w + eb \geq e - \xi^+, \\ & A^- w + be \leq -e + \xi^-, \\ & \xi^+ \geq 0, \xi^- \geq 0. \end{aligned} \tag{14}$$

Because $\|w\|_1 = \sum_{i=1}^m |w_i|$, which is non-smooth, we introduce the nonnegative variant s . Accordingly, the equivalent problem of(14) is the following:

$$\begin{aligned} \min_{w, b, s, \xi^+, \xi^-} & e^T s + C(e^T \xi + \rho e^T \xi), \\ \text{s.t.} & A^+ w + eb \geq e - \xi^+, \\ & A^- w + be \leq -e + \xi^-, \\ & \xi^+ \geq 0, \xi^- \geq 0, \\ & -s \leq w \leq s. \end{aligned} \tag{15}$$

Introducing the nonnegative Lagrange multipliers $\alpha^+, \alpha^-, \beta^+, \beta^-, \gamma, \delta$, and letting $\alpha = (\alpha^{+T}, \alpha^{-T})^T$, $\beta = (\beta^{+T}, \beta^{-T})^T$, and $A = (A^{+T}, A^{-T})^T$, the dual question of(15) is as follows:

$$\begin{aligned} \min_{\alpha} & e^T \alpha, \\ \text{s.t.} & e^T D \alpha = 0, \\ & -e \leq A^T D \alpha \leq e, \\ & 0 \leq \alpha^+ \leq C, 0 \leq \alpha^- \leq C\rho. \end{aligned} \tag{16}$$

By solving(16), we get the optimal value of α . By using the equations, $\gamma = \frac{1}{2}(e - A^T D\alpha)$, $\delta = \frac{1}{2}(e + A^T D\alpha)$, $\alpha^+ + \beta^+ = C$, $\alpha^- + \beta^- = C\rho$ and the slack condition:

$$\begin{aligned} \alpha^T(e - DAw - Deb - \xi) &= 0, \\ \beta^T \xi &= 0, \\ \gamma^T(s + w) &= 0, \\ \delta^T(w - s) &= 0, \end{aligned} \tag{17}$$

we get the optimal value of w . At last, we select all the indexes x_i for which w_i is nonzero as the personal credit scoring index system.

Now, giving the algorithm of the selection of evaluating index system of personal credit scoring via $(1 - SVM^\rho)$.

Algorithm 1:

Step 1. Clean up training set as $X=(x_1^+, 1), \dots, (x_k^+, 1), (x_{(k+1)}^-, 1), \dots, (x_m^-, -1)$.

Step 2. Look for the optimal value of α by solving(16).

Step 3. Look for the optimal value of w .

Step 4. Select all the indexes corresponding to the nonzero factor of w as the personal credit evaluating index system.

5 Numerical Testing

Our numerical testing is carried on a part of the personal credit database of a bank in China. The data sets X including 600 Good Credit data and 300 Bad Credit data with 19 Indexes. The testing utilizes Matlab7.0, supposing $\rho = 5$. Because the personal credit scoring is not a linear problem, we utilize the model(14) to search for the classification machine, supposing $K(x_i, x_j) = \exp(-\|x_i - x_j\|^2/(2\sigma^2))$.

The classification machine based on the evaluating index system selected by $1 - SVM^\rho$ has higher classification ability to class -1(bad credit) than the classification machine based on the evaluating index system selected by 1-SVM. The comparative results of numerical testing are shown in Table 1.

Table 1. Comparative results of numerical testing

selected method of evaluating index system	Number of evaluating index selected	Classification accuracy to class good credit	Classification accuracy to class bad credit
$1 - SVM$	13	80.35%	76.41%
$1 - SVM^\rho$	12	78.22%	80.10%

6 Conclusions

With the rapid growth in the personal credit consume industry , all kinds of mathematics models have been extensively used for the personal credit scoring. Because 1-SVM can select and suppress the problem feature efficiently, and

2-SVM owns better generalization and stronger classification, SVM, the new leaning machine, now is applied in the personal credit scoring. This paper proposes an improved SVM based on 1-norm adapted to the selection of the personal credit scoring index system. Experimental results show that the $1 - SVM^p$ can select evaluating index system with small scale and improve the classification ability of classification machine to class bad credit. The next work is to compare results of this paper with some other techniques: for example Neural Net with PCA (Principal Component Analysis) feature extraction or SOM (Self Organizing Maps) categorization.

Acknowledgments

This work is supported by national science foundation of China (10571109).

References

1. Guo, B., Liang, S.D., Fang, Z.B.: A Survey for Consumer Credit Scoring. *Systems Engineering* **19** (6) (2001) 9-15
2. Henley, W.E., Hand, D.J.: A k-nearest-neighbor Classifier for Assessing Consumer Credit Risk. *Statistician* **45** (1965) 77-95
3. Eisenbeis, R.A.: Problems in Applying Discriminant Analysis in Credit Scoring Models. *Journal of Banking and Finance* **2** (1978) 205-219
4. Zocco, D.P.: A Framework for Expert System in Bank Loan Management. *J. Commercial Bank Lend.* **67** (1985) 47-54
5. Joachimsthaler, E.A., Stam, A.: Mathematical Programming Approaches for the Classification Problem in Two-group Discriminant Analysis. *Multivariate Behavioral Research* **25** (1990) 427-454
6. Joanes, D.C.: Reject Inference Applied to Logistic Regression for Credit Scoring. *IMA J. Math. Appl. Bus. Industry* **5** (1993) 35-43
7. West, D.: Neural Network Credit Scoring Models. *Computers and Operations Research* **27** (2000) 1131-1152
8. Shen, C.H., Deng, N.Y., Xiao, R.Y.: Personal Credit Scoring on Support Vector Machines. *Computer Engineering and Application* **30** (23) (2004) 198-199
9. Yao, Y., Ye, Z.X.: The Credit Scoring System Based on Support Vector Machines. *Journal of SystemSimulation* **16** (4) (2004) 783 -786
10. Burges, C.J.C.: A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery* **2** (1998) 121-167
11. Kohavi, R., John, G.: Wrappers for Feature Subset Selection. *Artificial Intelligence* **12** (1997) 273-324
12. Vapnik, V.: *Statistical Learning Theory*. New York, Wiley (1998)
13. Deng, N.Y., Tian, Y.J.: *A New Method of Data Mining-Support Vector Machine*. Beijing, Science Press (2004)
14. Mangasarian, O.L.: Arbitrary-norm Separating Planes. *Operations Research letters* **10** (1999) 1032-1037
15. Bradley, P.S., Mangasarian, O.L.: Feature Selection via Concave Minimization and Support Vector Machine. *Machine Learning Proceedings of the Fifteenth International Conference (ICML98)* (1998) 801-807

Combining Weighted SVMs and Spectrum-Based k NN for Multi-classification

Ling Ping^{1,2}, Lu Nan¹, Wang Jian-yu¹, and Zhou Chun-Guang^{1,*}

¹ College of Computer Science, Jilin University, Key Laboratory of Symbol Computation and Knowledge Engineering of the Ministry of Education, Changchun 130012, China

² School of Computer Science, Xuzhou Normal University, Xuzhou, 221116, China
cgzhou@jlu.edu.cn

Abstract. This paper presents a Multi-Classification Schema (MCS) which combines Weighted SVMs (WSVM) and Spectrum-based k NN (Sk NN). Basic SVM is equipped with belief coefficients to reveal its capacity in identifying classes. And basic SVM is built in individual feature space to bring adaptation to diverse training data context. Coupled with a weighted voting strategy and a local informative metric, Sk NN is used to address the case rejected by all basic classifiers. The local metric is derived from most discriminant directions carried by data spectrum information. Two strategies of MCS benefit computational cost: training dataset reduction, and pre-specification of Sk NN working set. Experiments on real datasets show MCS improves classification accuracy with moderate cost compared with the state of the art.

1 Introduction

Support Vector Machine (SVM) [1] is a well-developed technique for classification. In spite of the outstanding performance in binary classification, it, however, cannot present an easy solution for multi-classification. Currently, two SVM-based approaches to multi-classification are on-going research [1]. One is the “decomposition-reconstruction” idea that uses the combination of SVMs, and the other is the “all-together” idea that formulates a single SVM optimization procedure to identify all classes. Usually the former idea is more popular and its many realizations appear: 1-vs-1 method [2]; 1-vs-r method [3]; DAGSVM [4] and error-correcting codes [5].

This paper presents a Multi-Classification Schema (MCS) consisting of weighted 1-vs-r SVMs (WSVM) and Spectrum-based k NN (Sk NN). WSVM weights its basic classifiers according to their decision confidence and this confidence is integrated with decision values to assign label. Sk NN is a spectrum-based version of k NN [6] to address the query rejected by WSVM. It works in query’s neighborhood that is developed by a locally informative metric and it makes decisions with a weighted voting strategy. MCS has computation ease in self tuning SVM hyper parameters and training dataset reduction.

* Corresponding author.

2 Related Knowledge

Firstly, we review SVM. For l samples: $(x_1, y_1) (x_2, y_2) \dots (x_l, y_l)$ sampling from $X \times Y$, where $X = R^n, Y = \{1, -1\}$. The optimal classification interface is determined by:

$$g(x) = \sum_i \alpha_i y_i K(x_i, x) - b. \tag{1}$$

The orientation vector α and offset vector b are obtained by optimizing:

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j K(x_i, x_j) \quad \text{s.t. } 0 \leq \alpha_i \leq C_{svm}, \sum_i \alpha_i y_i = 0 \tag{2}$$

Then it proceeds to Support Vector Clustering (SVC) [7]. It aims to find the smallest hyper sphere containing all data in feature space. The produced SVs form cluster contours. It corresponds to below optimization problem:

$$\max_{\gamma} \sum_i \gamma_i K(x_i, x_i) - \sum_{i,j} \gamma_i \gamma_j K(x_i, x_j) \quad \text{s.t. } \sum_i \gamma_i = 1, 0 \leq \gamma_i \leq C_{svc} \tag{3}$$

3 MCS

3.1 WSVM

In this paper basic classifiers are equipped with confidence weights. Set β_{IA} for basic SVM (I -vs- r) to show its decision capacity on class A , and all weights form a matrix $\beta = (\beta_{IA})_{M \times M}$. Set $\beta_{II} = 1$, which is natural that SVM (I -vs- r) is absolutely confident to declare query's membership to class I . For point x , its memberships to all classes form a row vector: $F_x = (f^1(x), f^2(x), \dots, f^M(x))$, where f^I corresponds to basic SVM (I -vs- r). Then global decision is made as:

$$label(x) = \max_A \{F_x \cdot \beta_{\bullet A}\}. \tag{4}$$

with

$$\beta_{\bullet A} = \sum_{I=1}^M \beta_{IA}, \quad \beta_{IA} \geq 0.$$

$$\beta_{IA} = \begin{cases} 1 & I=A \\ -\frac{\exp[dis(Center_A, f^I)]}{\sum_{J=1, J \neq I}^M \exp[dis(Center_J, f^I)]} & I \neq A \end{cases} \tag{5}$$

$$dis(Center_A, f^I) \approx \frac{1}{\|w^I\|^2} + \min_{s \in nbSVs} \|Center_A - s\|. \tag{6}$$

In (7), minus of case $I \neq A$ is introduced to match the negative value of $f^I(x)$ when x belongs to the rest classes. $dis(Center_A \text{ to } f^I)$ computes the distance between class A and decision function f^I . $Center_A$ is the average of representatives of class A .

3.2 Weighted Voting of Sk NN

For the rejected query Q , Sk NN works in its neighborhood. Neighborhood size k is designed as the size estimate of the natural dense region around Q . Sort distance list

of Q to other points $Sdis(Q, x_j)$ in the ascending order. Then $k = \max_j \{ Sdis(Q, x_j) - Sdis(Q, x_{j-1}) \}$. $Sdis(Q, x_j)$ is a spectrum-based metric discussed in Section 3.3. Let occurring frequencies of M class be: $t_1 \dots t_M$. Then $SkNN$ labels Q in a weighted way:

$$label(Q) = \max_A \{ \mu_A t_A \}. \tag{7}$$

$$\mu_A = 1 - \frac{Sdis(Q, A_{NEI})}{\sum_{I=1}^M Sdis(Q, I_{NEI})}. \tag{8}$$

$$Sdis(Q, A_{NEI}) = ave\{ Sdis(Q, x_A) | x_A \in A_{NEI}, A_{NEI} \text{ collects } Q\text{'s neighbors of class } A \}. \tag{9}$$

3.3 Neighborhood Formulation for SkNN

Spectrum analysis is used to define a new metric able to capture discriminant directions. Generate Kernel matrix H with Gaussian Kernel. Details are: 1) Normalize H into $H' = D^{-1/2} H D^{-1/2}$, where diagonal-shaped D has $D_{ii} = \sum_{j=1}^n H_{ij}$. 2) Conduct eigen-decomposition $H' = V \Lambda V^T$, where columns of V are eigenvectors, and Λ is the diagonal matrix of eigenvalues λ_i . 3) Select top p eigenvectors $V_1 \dots V_p$ to form matrix HS by stacking p eigenvectors in columns. Rows of HS are points' spectral coordinates. Then the new metric is defined according to the magnitude of each eigenvalue, as shown in (10). New distance based on spectrum coordinates x^s and y^s is (11).

$$\mu_i = \frac{|\lambda_i|}{\sum_{j=1}^p |\lambda_j|}. \tag{10}$$

$$Sdis(x^s, y^s) = \sqrt{(x^s - y^s)^T \mu (x^s - y^s)}. \tag{11}$$

Nystrom method [8] is used to yield spectrum coordinate of Q . Given existing data size m , new eigenvector \tilde{V}_i of matrix $HS_{m+1, m+1}$ is approximated by eigenvector V_i of matrix $HS_{m, m}$, and the matrix among existing data and query $HS_{m+1, m}$ in the way:

$$\tilde{V}_i = \sqrt{\frac{m}{m+1}} \frac{1}{\lambda_i} HS_{m+1, m} V_i. \tag{12}$$

4 MCS Implementation

4.1 Self Tuning Basic SVM Hyper Parameters

Tuning Kernel Scale. For each class I , a scale factor is formulated in the way:

$$\tau_I = ave\{ \|x - x_r\| \} \quad x \in I. \tag{13}$$

Here, x_r is the r th nearest point of x . Given r , if $\|x - x_r\| < \|y - y_r\|$, it indicates the density of x 's neighborhood is denser than that of y . r is probed as: $r = \max_j \{ \|x - x_j\| - \|x - x_{j-1}\| \}$, where Euclidean distance list $\|x - x_j\|$ has been sorted in the ascending order. Then Gaussian Kernel of SVM I -vs- r sets its scale as:

$$\tau_I^2 = \tau_I \cdot \tau_{rest(I)} \quad \text{with } \tau_{rest(I)} = ave\{ \tau_J | J \neq I \}. \tag{14}$$

Penalty Tuning Individually. Consider various demands of points to slack variables. To those outliers or SVs, they hope a big C_{svm} to emphasis the slack, but to inner-class-points, they need a small one to highlight maximum margin. So $C_{svm(x)}$ is:

$$C_{svm(x)} = \|x - x_r\| / r. \quad (15)$$

Tuning Strategies Effect. Employing above approaches, our SVM procedure is performed on real datasets. And its accuracy and time cost is compared with traditional SVM, which parameterizes hyper parameters by 5-fold cross validation. From Table 1, our SVM is competitive with the optimal results with less computation time.

Table 1. Comparison between our SVM and traditional SVM. 20% data are randomly sampled for training. Error number and time is the average of 20 runs.

Data	Our SVM		Traditional SVM	
	Error (%)	Time (s)	Error (%)	Time (s)
Iris (1-vs 2, 3)	0	0.672	0	1.108
Iris (2-vs 1, 3)	4.2	0.702	4.1	1.131
Iris (3-vs 1, 2)	3.27	0.691	3.26	1.107
Breast Cancer	2.52	3.87	2.41	7.05

4.2 Training Dataset Reduction and Sk NN

A tuning-scaled SVC procedure is conducted on each class respectively to select data representatives. In each SVC procedure, C_{svc} is set as 1. Kernel function integrates scale factors of individual points to give data-specific affinity. That is:

$$k(x, y) = \exp\left(-\frac{\|x-y\|^2}{\sigma_x \sigma_y}\right) = \exp\left(-\frac{\|x-y\|^2}{\|x-x_r\| \|y-y_r\|}\right). \quad (16)$$

r is the same meaning as above. This setting produces a little more SVs than traditional SVC. These SVs are located on both boundaries and important positions where sharp changes of density happen, to form a brief sketch of dataset.

5 Experimental Results

Real datasets are taken from [9]. In Table 2, MCS are compared on the average of 30 runs with following classifiers: Simple k NN method; SVMs schema in 1-vs-r version (SVM_{1r}); SVMs schema in 1-vs-1 version (SVM_{11}); C4.5 decision tree method [10]; Machete [11]; Scythe [11]; DANN [12]; Adamenn [13]. In all datasets, 50% data are sampled randomly for training. From Table 2, it can be seen that MCS outperforms other approaches in three of the five sets, and is rather competitive in other cases. Among SVM-based approaches, MCS outperforms its peers usually. And SVM_{1r} takes advantages over SVM_{11} usually. C4.5 and Machete work poorly in some sets due to their greedy idea. Scythe modifies the greedy nature and thereby achieves better accuracy. The metric employed by DANN approximates the weighted Chi-squared distance, which causes it fails in datasets of non-Gaussian distribution. Adamenn also works well in some cases, but it requires huge cost to tune six parameters.

Table 2. Comparisons on classification error (%) (The optimal result is shown in bold style)

Data	kNN	SVM _{lr}	SVM _{l1}	C4.5	Machete	Scythe	DANN	Adamenn	MCS
Iris	6.0	4.1	4.0	7.7	5.1	4	5.8	3	3
Wine	7.79	5.83	6.15	8.92	7.11	6.03	6.2	5.28	5.33
Sonar	12.5	11	12	23.1	21.2	16.3	9.7	8.7	8.11
Liver	31.5	28.4	26.2	38.3	28.5	29.1	30.1	26.7	26.67
Vote	7.8	3	2.6	3.4	3.3	3.3	3	3	3

Table 3. Comparisons on classification error on News Group (%) (Number in the bracket is the number of data sampled from training set, and the optimal result is shown in bold style)

Dataset	kNN	SVM _{lr}	SVM _{l1}	C4.5	Machete	Scythe	DANN	Adamenn	MCS
{NG2(150), NG3(50), NG4(200)}	33.79	30.8	31.2	34.92	31.07	29.03	31.46	31.2	30.38
{NG6(200), NG7(150), NG8(350)}	15.95	15.1	16.3	18.3	16.5	15.2	14.8	14.17	14.67
{NG1, NG2, NG7, NG8} (200)	15.8	13.2	12.98	14.4	13.83	13.74	13.65	12.5	12.6
{NG1 (50) NG2(100), NG7(150), NG8(50)}	14.9	13.8	13.9	13.55	13.72	12.68	13.0	13.8	13.0
{NG7(100), NG8(50), NG12(200), NG16(50), NG17(100)}	15.6	12.3	12.3	14.7	12.9	12.0	11.64	12.5	12.1

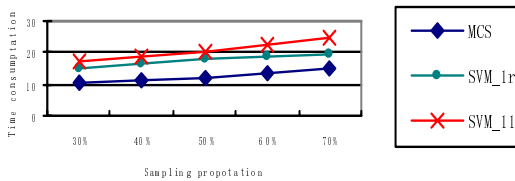


Fig. 1. Time consumption of three SVM-based approaches

Then News group [14] is tested. This dataset contains about 20,000 articles divided into 20 newsgroups. We label each newsgroup as follows: NG1, NG2...NG20. We apply the usual *tf.idf* weighting schema to express documents. Words that appear too few times are deleted and document vectors are normalized. MCS is compared with other approaches on the average classification error rates of 10 runs in Table 3. Clearly, MCS takes an advantage or a competitive performance over its peers. On the classification task: {NG6(200), NG7(50), NG8(150)}, time consumption of three SVM-based schemas are observed in Fig 1. Clearly MCS pays less time.

6 Conclusion

A novel method MCS is described in this paper for multi classification issue, which integrates WSVM and Sk NN. Basic classifiers are created in diverse feature spaces adaptively, and they are weighted according to their decision confidence. Sk NN deals with the rejected case with a weighted voting strategy. The spectrum-based metric helps to explore informative neighborhood and class identification. Training dataset is reduced by the tuning-scaled SVC. Experiments on real datasets demonstrate the improved performance and efficiency of MCS.

Acknowledgement

This work was supported by the National Natural Science Foundation of China under Grant No. 60433020、60673099, and the Key Laboratory for Symbol Computation and Knowledge Engineering of the National Education Ministry of China; 985 Project, Technological Creation Support of Computation and Software Science.

References

1. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines. Cambridge University Press, London (2000)
2. Hastie T.J., Tibshirani, R.J.: Classification by Pairwise Coupling. In: M.I. Jordan, M.J. Kearns and S.A. Solla (eds.): Advances in Neural Information Processing Systems. MIT Press, Cambridge, MA **10** (1998) 507–513
3. Vapnik, V.: Statistical Learning Theory. Wiley New York (1998)
4. Platt, J., Cristianini, N., Shawe-Taylor, J.: Large Margin DAGs for Multi-Class Classification. Advances in Neural Information Processing Systems. MIT Press, Cambridge, MA **12** (2000) 547–553
5. Crammer, K., Singer, Y.: On the Learn Ability and Design of Output Codes for Multi-Class Problems. Machine Learning **47** (2002) 201–233
6. Bottou, L., Vapnik, V.: Local Learning Algorithm Neural Computation. **4** (1992) 888–900
7. Ben-Hur, A., Horn, D., Siegelmann, H.T.: Support Vector Clustering. Journal of Machine Learning Research **2** (2001) 125-137
8. Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P.: Neunmrical Recipies in C. Second ed. Cambridge Univ. Press (1992)
9. <http://www.ics.uci.edu/~mlearn/MLSummary.html>
10. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan-Kaufmann Publishers (1993)
11. Friedman, J.H.: Flexible Metric Nearest Neighbor Classification. Tech. Report, Dept. of Statistics, Stanford University (1994)
12. Hastie, T., Tibshirani, R.: Discriminant Adaptive Nearest Neighbor Classification. In: IEEE Trans. on Pattern Analysis and Machine Intelligence. **18**(6). (1996) 607-615
13. Domeniconi, C., Peng, J., Gunopulos, D.: An Adaptive Metric Machine for Pattern Classification. Advances in Neural Information Processing Systems 13, MIT Press (2000)
14. <http://www.cs.cmu.edu/afs/cs/project/theo-11/www/naive-bayes.html>

Rotation Invariant Texture Classification Algorithm Based on DT-CWT and SVM*

Shuzhen Chen, Yan Shang, Bingyi Mao, and Qiusheng Lian

Department of Electronics and Communication,
Yanshan University, Hebei 066004, China
lianqs@ysu.edu.cn

Abstract. A rotation invariant texture classification algorithm based on dual-tree complex wavelet transform (DT-CWT) and support vector machines (SVM) is proposed. First, the texture image is transformed by Radon transform to convert the rotation to translation, the rotation invariant feature vector is composed of the energies of the subbands acquired by DT-CWT which is shift invariant to the transformed texture image, the SVM algorithm is used to the texture classification at last. This algorithm is compared with the classifier of probabilistic neural network (PNN) and other rotation invariant texture classification algorithm, the experiment results show that it can improve the classification rate effectively.

1 Introduction

Texture is an important feature of image that reflects the information of gray statistic, space distribution and structure synthetically. It's regarded as the inherent characteristic of the surface of image and plays an important role in image processing. Texture classification has been an active research topic for several decades that is extremely useful in numerous areas such as object recognition, remote sensing, content-based image retrieval and so on. Researchers working in this area have proposed a number of texture classification algorithms, these methods have achieved fine results, but most of them are on the assumption that textures are in the invariant directions that is very unpractical in practical applications, so efficient classification of rotated texture images is a topic to be researched deeply.

The dual-tree complex wavelet transform (DT-CWT) proposed recently not only has the same virtue as the discrete wavelet transform (DWT) but also has some important additional properties: approximately shift invariance, better directional selection, lower redundancy and perfect reconstruction, so it can character the textures more precisely [1]. Support vector machines (SVM) developed recently can achieve better classification performance and has been successfully used in many areas[2]. In this paper, the rotation invariant feature vectors are extracted using Radon transform and DT-CWT, then the SVM algorithm is used to the classification at last. This algorithm is compared with others, the experiment results show that it can improve the correct classification rate (CCR) effectively.

* This work is supported by Hebei education bureau under Grant 2004124.

2 Rotation Invariant Feature Extracting

2.1 Radon Transform

Radon transform reflect the projection along the radial in certain angle. The Radon transform of a 2D function $f(x,y)$ is defined as[3]:

$$R(r, \theta)[f(x, y)] = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y) \delta(r - x \cos \theta - y \sin \theta) dx dy, \tag{1}$$

where r is the perpendicular distance of a line from the origin and θ is the angle between the line and the y -axis.

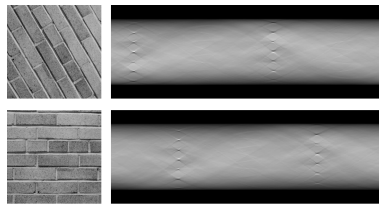


Fig. 1. Different orientation texture images and their corresponding Radon transforms

Fig. 1 shows how the Radon transform changes as the image rotates, the above figure shows the original image and its Radon transform, the other one shows the image rotated 60° and its Radon transform. As Fig.1 shown, rotation of the input image corresponds to the translation of the Radon transform along θ (x -axis).

2.2 The Dual-Tree Complex Wavelet Transform

The 1-D DT-CWT is implemented using two filter banks in parallel operating on the input data, it is consist of two parallel 1-D discrete wavelet transform (DWT) labeled tree A and B. Assuming that $h_0(n)$ and $h_1(n)$ are the low-pass and high-pass filter respectively in tree A as well as $g_0(n)$ and $g_1(n)$ corresponding to tree B. $H(e^{j\omega})$ and $G(e^{j\omega})$ are the discrete Fourier transform of $h(n)$ and $g(n)$, then the two low-pass/high-pass filter banks should satisfy the following desired properties[1]:

(1) Perfect reconstruction property

$$H_0(e^{j\omega})\tilde{H}_0(e^{j\omega}) + H_1(e^{j\omega})\tilde{H}_1(e^{j\omega}) = 2, \tag{2}$$

(2) $\psi_g(t)$ and $\psi_h(t)$ form an approximate Hilbert transform pair, so $g_0(n)$ should be approximately a half-sample shift of $h_0(n)$:

$$g_0(n) \approx h_0(n - 0.5). \tag{3}$$

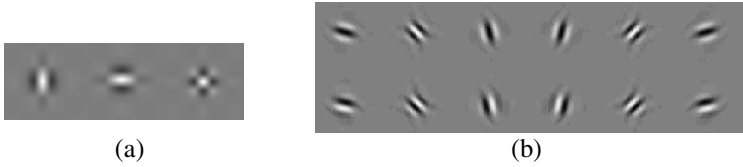


Fig. 2. The two set of wavelet bases (a) 2-D discrete wavelet bases (b) 2-D dual-tree complex wavelet bases

The properties above mentioned make the DT-CWT have the performance of perfect reconstruction and approximate shift invariance. Figure 2(a) shows the 2-D discrete wavelet bases while (b) shows the 2-D dual-tree complex wavelet bases where the first row shows the real part and the second one shows the imaginary part. As shown, the 2-D discrete wavelet bases have the three orientations of horizontal, vertical and diagonal while the 2-D dual-tree complex wavelet bases have six orientations: $\pm 15^\circ$, $\pm 45^\circ$, $\pm 75^\circ$, so the 2-D DT-CWT has better directional selection than 2-D DWT. The practical textures have abundant directional information, so the 2-D DT-CWT can describe the textures more effectively.

2.3 Rotation Invariant Feature Extracting

The steps of rotation invariant feature extracting are as follows:

(1) The gray levels of the image are normalize to $[0,1]$, then calculate the Radon transform for the biggest circular area of the texture image so as to convert the rotation to translation.

(2) Transform the resultant image in 1. using four levels of 2-D DT-CWT which is translation invariant to avoid the translation produced by Radon transform. 52 subbands are exported in all.

(3) Get the magnitudes of the real and imaginary parts of the complex subband coefficients, then compute the energies of the magnitudes as the rotation invariant feature vectors. The three kinds of energies used in experiments are defined as formulas (4) where $I(x,y)$ is the subband with size of $M \times N$. Each of them produces feature vectors with size of 1×26 .

$$\begin{aligned}
 e_1 &= \frac{1}{M \times N} \sum_{x=1}^M \sum_{y=1}^N \sqrt{|I(x,y)|}, \quad e_2 = \frac{1}{M \times N} \sum_{x=1}^M \sum_{y=1}^N |I(x,y)|, \\
 e_3 &= \frac{1}{M \times N} \sum_{x=1}^M \sum_{y=1}^N |I(x,y)|^2
 \end{aligned} \tag{4}$$

3 Classification Algorithm

3.1 Classification Algorithm Based on Support Vector Machines

Support vector machines (SVM) as a classifier to the texture analysis is relatively new and offers several typical advantages that are not found in traditional classifiers:

- (1) The optimal separating hyperplane can make it achieve higher CCR.
- (2) It's a convex quadratic programming problem, so it avoids local extremea.
- (3) The theory of kernel function makes it overcome the problem of 'dimension disaster' and can be developed to the nonlinear case easily.

The fundament of SVM is constructing an optimal classification function for two-class linear problem that maximizes the margin of examples belonged to different classes in order to classify the two classes as precise as possible. Solve the quadratic programming problem by Lagrange theorem, then the weight vector $w^* = \sum_{i=1}^l y_i \alpha_i^* x_i$ as the key can produce optimal separating hyperplane at last.

For the nonlinear classification, a kernel function $K(x_i, x_j)$ is used to map the input space into a higher dimensional feature space such that the nonlinear hyperplane becomes linear. The polynomial kernel and Gaussian kernel are both used in this paper.

The one-against-one decomposition algorithm is adopted in the multi-texture classification. That is, we decompose the q -class ($q > 2$) problem into $q(q-1)/2$ two-class cases.

3.2 Classification Algorithm Based on Probabilistic Neural Networks

The probabilistic neural network [4] is a kind of radial basis network suitable for pattern classification and it is commonly used to classification problems. It has the advantages of simple principle and fast convergence. The network converges to a Bayesian classifier under the condition of enough train samples and it can be generalized easily. However, too many inputs will lead to high computing complexity and low speed of calculation.

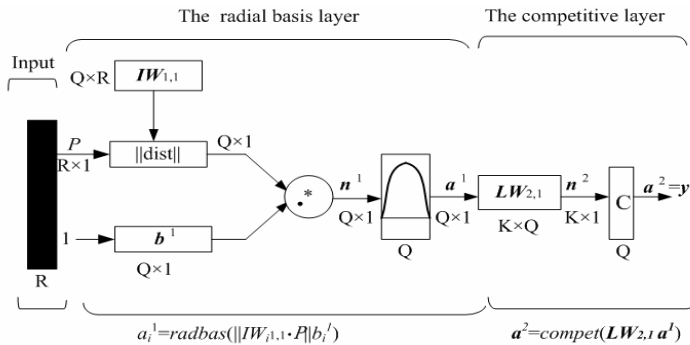


Fig. 3. The structure of PNN

As shown in figure 3, the structure of PNN is composed of the radial bases layer with Q neurons and the competitive layer with K neurons. R is the dimension of pattern vector. The competitive layer determines the final outputs by computing the probabilities of the exports of the radial basis layer.

4 Experiment Results

Two texture databases are used in the experiments:

(1) Texture database 1. As shown in figure 4, it contains thirteen classes of texture images with size of 512×512. Each class is a single texture and rotated in seven angles, i.e. 0°,30°,60°,90°,120°,150°,200°. Each of the images is partitioned into sixteen nonoverlapping small images with size of 128×128. Therefore, 16×7=112 small images are obtained for each of the thirteen classes, out of which eleven training images are chosen from the 0° images and all of the rest are used to be tested as in paper [5]. That is, the training set contains 11×13=143 small images in all.

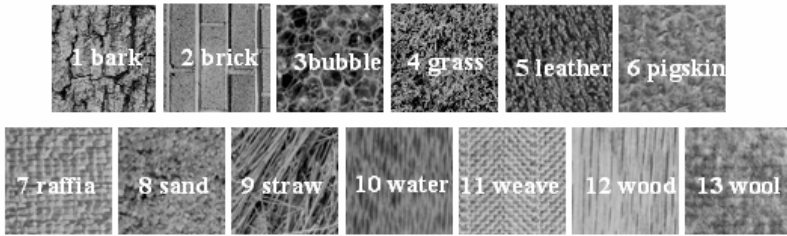


Fig. 4. Texture database 1

Extract the rotation invariant feature vector for each of the images sized 128×128 by Radon transform and 2-D DT-CWT first, then train the features in training set using SVM with polynomial kernel function, classify the feature vectors in test sets by SVM at last. Table 1 shows the average correct classification rates (ACCR)with different kinds of energies, as shown, the highest ACCR (85.15%) is produced by e_1 and the rate produced by e_2 is following. Test the classification performance of SVM with Gaussian kernel function under the condition of energy e_1 , the correct rate is 84.39%, so the performance of polynomial kernel function is better than the Gaussian kernel. Classify the same data using probabilistic neural network as the classifier and e_1 as the energy, the result is shown in table 1, too. From table 1 we can see that the ACCR obtained by SVM was significantly higher than that obtained by PNN ,that is, the SVM algorithm can improve the CCR effectively. The classification algorithm in paper [5] is based on the directional filter bank and its result for the same data is also shown in table 1, as shown, the result produced by algorithm proposed is higher by 13.94% than the other one.

Table 1. Experiment results for database 1

Algorithm	Algorithm proposed			PNN(e_1)	Algorithm in [5]
	e_1	e_2	e_3		
ACCR	0.8515	0.7616	0.7951	0.7989	0.7121

The optimal classification results of each texture are shown in table 2. As shown, the results of textures without ‘straw’ and ‘wool’ are generally well, out of which the CCR of ‘weave’ achieved 100%.

Table 2. Classification results of each texture in database 1

texture	bark	brick	bubble	grass	leather	pigskin	raffia
CCR	0.9208	0.7723	0.9703	0.7921	0.9307	0.8614	0.8812
texture	sand	straw	water	weave	wood	wool	ACCR
CCR	0.8812	0.5050	0.9307	1.0000	0.9802	0.6436	0.8515

(2) Texture database 2. As shown in figure 5, it contains sixteen classes of texture images with size of 180×180. Each class is rotated in ten angles, i.e. 0°, 20°, 30°, 45°,60°,70°,90°,120°,135°,150°.Each of them is partitioned into sixteen nonoverlapping small images with size of 45×45, then small images in 0°,30°,45°, 60°are used to train and all of the rest are used to be tested.

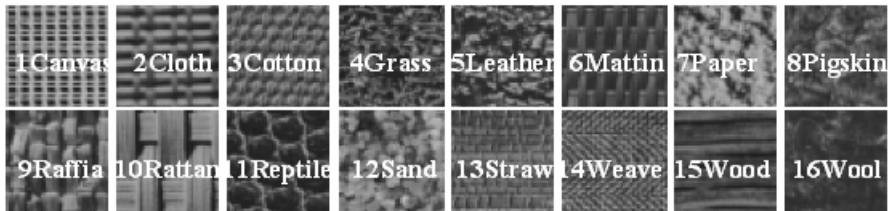


Fig. 5. Texture database 2

The energy ‘ e_1 ’ and polynomial kernel SVM are applied in the experiment because of their best performance in the first experiment. The CCRs of each texture are shown in table 3. From table 3 we can see that all of the textures can be nicer classified and half of textures are classified with the CCR 100%. The ACCR can achieve 98.86% which is much better than that in database 1, this is mainly because of the increase of texture angles used in training. The algorithm in paper [6] is based on the autocorrelation measures of subbands acquired by DT-CWT and its result for the same data is also shown in table 3, as shown, the result produced by algorithm proposed is higher by 5.01% than the other one.

Table 3. Classification results of each texture in database 2

texture	Canvas	Cloth	Cotton	Grass	Leather	Mattin
CCR	1.0000	1.0000	1.0000	0.9792	1.0000	0.9792
texture	Paper	Pigskin	Raffia	Rattan	Reptile	Sand
CCR	1.0000	1.0000	1.0000	0.9792	0.9896	0.9792
texture	Straw	Weave	Wood	Wool	ACCR	ACCR in [6]
CCR	0.9896	1.0000	0.9792	0.9271	0.9876	0.9375

5 Conclusions

A rotation invariant texture classification algorithm based on the dual-tree complex wavelet transform and support vector machine is proposed. Extract the rotation invariant feature vectors using Radon transform and the 2-D dual-tree complex wavelet transform first and the support vector machines algorithm is used to the texture classification at last. The dual-tree complex wavelet transform which has the advantages of approximately shift invariance and better directional selection could character the textures more effectively as well as the principle of optimal separating hyperplane can make the support vector machines achieve higher classification rate. Both the experiment results for database1 and database 2 demonstrate that the algorithm proposed can improve the classification rate effectively.

References

1. Selesnick, Ivan W., Baraniuk, Richard G., Kingsbury, Nick G.: The Dual-Tree Complex Wavelet Transform. *IEEE Signal Processing* **22** (6) (2005) 123-151
2. Burges, C.J.C.: A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery* **2** (2) (1998) 1-47
3. Kouros, J.K., Hamid, S.Z.: Rotation-Invariant Multiresolution Texture Analysis Using Radon and Wavelet Transforms. *IEEE Transactions on Image Processing* **14** (6) (2005) 783-795
4. Specht, D.F.: Probabilistic Neural Networks. *Neural Networks* **3** (1) (1990) 109-118
5. Duan, R., Man, H., Chen, L.: Rotation Invariant Texture Classification Based on A Directional Filter Bank. *IEEE International Conference on Multimedia and Expo (ICME'04)* **2** (2004) 1291-1294
6. Hill, P.R., Bull, D.R., Canagarajah, C. N.: Rotationally Invariant Texture Classification. *IEE Seminar on Time-scale and Time-Frequency Analysis and Applications* **20** (2000) 1-5

A Two-Pass Classification Method Based on Hyper-Ellipsoid Neural Networks and SVM's with Applications to Face Recognition

Chengan Guo, Chongtao Yuan, and Honglian Ma

School of Electronic and Information Engineering,
Dalian University of Technology, Dalian, Liaoning 116023, China
cguo@dlut.edu.cn, sheva810712@gmail.com, mhl@dlut.edu.cn

Abstract. In this paper we propose a two-pass classification method and apply it to face recognitions. The method is obtained by integrating together two approaches, the hyper-ellipsoid neural networks (HENN's) and the SVM's with error correcting codes. This method realizes a classification operation in two passes: the first one is to get an intermediate classification result for an input sample by using the HENN's, and the second pass is followed by using the SVM's to re-classify the sample based on both the input data and the intermediate result. Simulations conducted in the paper for applications to face recognition showed that the two-pass method can maintain the advantages of both the HENN's and the SVM's while remedying their disadvantages. Compared with the HENN's and the SVM's, a significant improvement of recognition performance over them has been achieved by the new method.

1 Introduction

Face recognition has been emerging as a very active research field over past few years [1, 2]. Two issues are essential in face recognition: the first is what features are to be used to represent a face. The second is how to design an efficient classifier to recognize a new face image. In this paper we focus on the classifier design problem. The feature selection issue is also discussed through a comparison study of experimental results in the paper.

Many good classification methods have been proposed and applied to face recognition in recent years. Among them, the Support Vector Machine (SVM) [3] is an efficient one. The SVM was originally designed for binary classification, and for multiclass problems, as in image recognition, one needs an appropriate combination of a number of binary SVM's. Several approaches to the multiclassification problems using binary SVM's have been proposed, including the M-ary algorithm [4], the One-against-one [5], the One-against-the-others [4], and the Error-correcting Output Codes (ECOC) [6]. The SVM's with ECOC has the error control ability that can correct a certain number of intermediate misclassifications by training some extra SVM's, and it has been successfully applied to face recognition in our previous work [7]. According to the experiment results given in [7], the SVM's with ECOC classifier

outperforms the other SVM-based classifiers by taking both the recognition accuracy and the computation complexity into account. However, there still leaves room for improving on the method. For example, it always maintained a certain amount of false acceptance rates (FAR) with this method. For a practical recognition problem such as an authentication system (e.g., an image identification system), the FAR is an important index that should be made as small as possible.

The Biomimetic Pattern Recognition (BPR) proposed by Wang et al [8] is a topological pattern recognition method that can overcome the drawback of the high FAR problem effectively. In the BPR method [8,9], the hyper-ellipsoid neural network (HENN) or the double synaptic weight neurons (DSWN) [9] are trained to implement the classifier in which the classification hypersurfaces are constructed by a number of connected hyper-ellipsoid spheres as shown in Fig. 1. It was showed by [8,9] in the application to image recognitions that the HENN method can reduce the FAR significantly (almost to zero). It is also noticed, however, that the method yields quite a high false rejection rate (FRR) and therefore its correct recognition rates are quite low.

In this paper we propose a two-pass classification method for face recognition based on the HENN's and the SVM's with ECOC. In the new method, the HENN's and the SVM's with ECOC are integrated together that can maintain the advantages of the two methods on the one hand, and can overcome the defects of them on the other hand. Simulation experiments conducted in the paper show that, a significant improvement on recognition performance can be achieved by using the two-pass method.

In Section 2, the two-pass classification method is presented following brief descriptions of the HENN's and the SVM's with ECOC method. Experiment results with application to face recognition are given in Section 3. Section 4 gives the summary and further directions of the paper.

2 The Two-pass Classifier Based on HENN's and SVM's

2.1 The HENN Classification Method

The hyper-ellipsoid neural network (HENN), also called the double synaptic weight neurons (DSWN), was proposed by Wang et al [8,9] based on the Biomimetic Pattern Recognition (BPR) theory [8]. The neuron model of the HENN is shown in Fig. 2 and its general formula [9] is given by

$$y = \varphi \left[\sum_{i=1}^n \left(\frac{w_i(x_i - w'_i)}{|w_i(x_i - w'_i)|} \right)^q |w_i(x_i - w'_i)|^p - \theta^r \right] \tag{1}$$

If the parameters in (1) are set such as $q=0$, $p=2$ and $r=2$, this model can describe a closed n dimensional hyper-ellipsoid surface by $\sum_{i=1}^n w_i^2(x_i - w'_i)^2 - \theta^2 = 0$. The classification hypersurface of a HENN is composed of a number of these connected hyper-ellipsoidal spheres, as illustrated in Fig. 1.

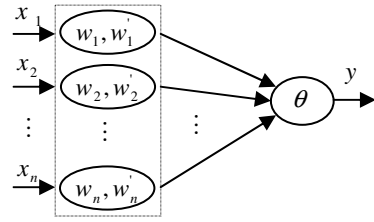
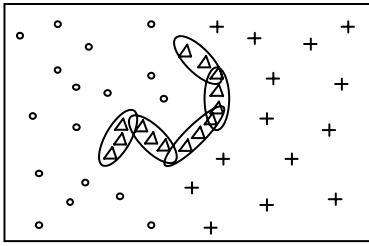


Fig. 1. The classification hypersurface of a HENN **Fig. 2.** The model of the HENN neuron

In order to design a HENN classifier for an m -class problem, we need to design m HENN's. For each HENN, e.g., for the i -th HENN, we need to train k_i HENN neurons with training samples. The k_i neurons need to be trained to such a result that they can construct the closed surface of k_i connected hyper-ellipsoid spheres, as illustrated in Fig. 1, which will be both able to include all the training samples of the i -th class inside the surface, and also able to exclude all the other class samples outside the surface. In this way, one can obtain the HENN classifier after the training stage for m HENN's has been accomplished.

As for the learning algorithm for training the HENN neurons to estimate the weights, w_i 's, W_i 's, and θ 's, a gradient descent algorithm can be induced that is omitted here due to the limitation of the paper length.

Having the HENN classifier obtained, one can use it to classify a new sample in the following algorithm:

- (i) Input the feature vector of the sample into the i -th HENN, for $i = 1, \dots, m$.
- (ii) Classify the sample to the j -th class, if the feature vector of the sample is only inside the j -th closed surface. Otherwise, the sample is rejected.

2.2 The SVM's with ECOC Classification Method

It is well known that the SVM is an optimal classifier in terms of structural risk minimization based on VC theory [3]. As the SVM was originally designed for binary classification, for multiclass problems, one needs an appropriate combination of a number of binary SVM's.

For an m -class problem, k binary SVM's, where $k = \lceil \log_2 m \rceil$, are enough in theory for classifying the m classes. However, the classifier with this number of SVM's has no robustness (or error tolerance), and once an SVM gives a wrong intermediate classification, it leads to a final mistake. As a direct scheme for gaining robustness, one can use more SVM's. But how many SVM's should be used and how much robustness can be obtained by these more SVM's? In order to solve this problem, the error-correcting output codes (ECOC) algorithm was proposed by Dietterich and Bakiri [6] and its main idea is as follows:

The classification procedure of an m -class problem using binary classifiers can be viewed as a digital communication problem and the classification errors made by some binary classifiers are viewed as transmission errors of a binary string over a channel. In this way the errors may be corrected by adding some redundant SVM's using an error control coding scheme. According to coding theory [10], for a n -bit code with the minimum Hamming distance d , it is able to correct l errors, where $n \geq \lceil \log_2 m \rceil + d$ and $l = \lceil (d-1)/2 \rceil$. Therefore, in the ECOC approach [6], it was proposed that some error control coding scheme can be incorporated into solving an m -class learning problem, in which l intermediate misclassifications can be corrected by using n binary classifiers.

In order to implement the SVM's with ECOC classifier, two stages are included: the encoding and training stage for obtaining a coded n binary SVM's with a properly selected error control coding scheme and a learning algorithm, and the decoding and classification stage for getting error correction and performing classification based on the coded n SVM's. Details for these implementing algorithms can be found in [6] and [7], that are omitted here.

2.3 An Integration Scheme of HENN's and SVM's —The Two-Pass Classification Method

As pointed out in Section 1, both the classifiers of the HENN's and the SVM's with ECOC have their own advantages and disadvantages. For the SVM's with ECOC, one can always gain the higher correct recognition rates by using this method than by using the HENN's. However, its false acceptance rates (FAR) are always retained to a certain level that is not negligible. For the HENN method, it can reduce its FAR significantly (almost to zero) while it always gives quite a high false rejection rates (FRR).

Then the question is raised that if we can design a new classifier by integrating the two methods together in such a way that it will maintain the advantages of the two methods and overcome the drawbacks of them meantime. The answer to the question is that this is possible by suitably incorporating the two methods. However, it should also be noted that some integrations of two methods may result in accumulation or enlargement of errors made by them. In this paper we propose a two-pass classification approach by integration the HENN's and the SVM's together that is going to be proved effective by applications of the method to face recognitions.

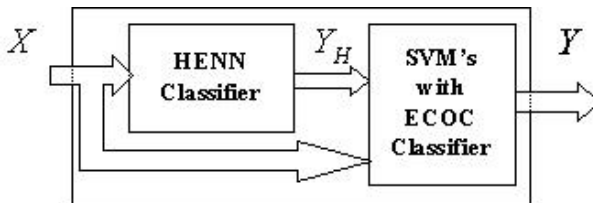


Fig. 3. Block diagram of the two-pass classification approach

The idea of the two-pass classification approach is shown in Fig. 3 in which X is the feature vector of a sample to be classified, Y is the classification result for the input sample, and Y_H is the intermediate classification result given by the HENN classifier.

The classification algorithm for the two-pass classifier for an m - class problem includes two passes of classification steps as follows:

- (i) Input the feature vector X for the sample to be classified. It is assumed that the sample belongs to one of the m given classes.
- (ii) The first pass of the method is to get Y_H by using the HENN classifier, which is then sent to the SVM's with ECOC classifier. There are two possible cases for the intermediate classification result Y_H to indicate: one case is that the input is already classified to one of the m classes by the HENN classifier if X is only inside one closed surface of the HENN classifier, and the other case is that the input is rejected if X is inside more than one closed surfaces, or outside all the closed surfaces, of the HENN classifier.
- (iii) The second pass is to use the SVM's with ECOC classifier to get the final classification result Y based on X and Y_H , in which the input sample is re-classified to one of the m classes if Y_H indicates that the input is rejected by the HENN classifier, and otherwise Y_H is taken as the final classification result.

We make some remarks on this classification algorithm in the following:

Remark 1. The HENN classifier and the SVM's with ECOC classifier are assumed already established before performing the two-pass classification algorithm. This can be done by training the two classifiers separately using the learning methods given in Section 2.1 and 2.2 respectively with the same training data.

Remark 2. The performance of the two-pass classifier is also related to the input features that are extracted for expressing the samples to be classified. Usually, using different kind of features will result in different performances. A comparison study of simulations for using different features in face classifications will be given in the next section of the paper.

3 Applications to Face Recognition and Experiment Results

Many simulation experiments have been conducted for the two-pass classification method with applications to face recognition in the paper. Meantime these simulations were also conducted for the HENN classifier and the SVM's with ECOC classifier.

The simulations are performed on the Cambridge ORL face database which contains 400 face images of 40 distinct persons in total with 10 images for each person. In the simulations, the eigenfaces based on PCA [11] and the Fisherfaces [12] are used respectively as feature templates to represent face images, and the input feature vectors for the classifiers are extracted by projecting the sample images onto

the eigenfaces or the Fisherfaces. In each simulation experiment, 200 samples (5 samples for each person) selected randomly from the ORL database are used as the training set to train the classifiers and the remaining 200 samples are used as the testing set.

In each simulation experiment for establishing the HENN classifier, 40 HENN's are trained, with each HENN consisting of 4 hyper-ellipsoid neurons. Having the training stage done, 40 classification hypersurfaces are constructed by the 40 HENN's, with each hypersurface consisting of 4 connected hyper-ellipsoid surfaces, as illustrated in Fig. 1. The learning algorithm given in Section 2.1 was used to train the networks.

For establishing the SVM's with ECOC classifier in the simulation, the (31,6) BCH code [10] was used for the encoding and decoding algorithm. The (31,6) BCH code contains 31 bits in total with 6 information bits and 15 error control bits that has the minimum Hamming distance of 15, and the code is able to correct 7 error bits according to coding theory [10]. In correspondence with this coding scheme, 31 SVM's were trained each time using the learning algorithm given in Section 2.2 for obtaining the SVM's with ECOC classifier to recognize the 40 classes of face images in the simulations.

Having the training stage done both for the HENN classifier and the SVM's with ECOC classifier, we constructed the two-pass classifier with the two classifiers in the way as shown in Fig. 3. The two-pass classifier was then used to realize the classification algorithm given in Section 2.3 in the simulations.

In the paper, experiment simulations were conducted at the same time for the HENN classifier, the SVM's with ECOC classifier and the two-pass classifier, respectively, by using the same data. In order to study the influence of different features on the classifiers, both the eigenface and the Fisherface features were used as inputs to perform the training and testing processes. Table 1 gives the testing results that were obtained by averaging over 10 simulations in the paper. In the table, three kinds of recognition rates are used for performance evaluation including the false rejection rates (FRR), the false acceptance rates (FAR), and the correct recognition rates (CRR).

It can be seen from Table 1 that for the face recognition problem, the two-pass method proposed in the paper always gives the highest CRR among the three methods. From Table 1, it can also be seen that, the advantage of the HENN classifier is significant that it yields very low FAR, but its defect is also notable that it has very high FRR. For the SVM's with ECOC classifier, it is superior to the HENN classifier by giving much higher CRR, while it always keeps a certain amount of not negligible FAR. As for the two-pass classifier, one can see that it carries on the advantages, and meantime remedies the defects, of both the HENN classifier and the SVM's with ECOC, in which its CRR are improved significantly while keeping its all error rates, including FRR and FAR, quite low.

The influence of different features on the performance of the classifiers can also be observed from Table 1, in which using Fisherfaces can get better result than using eigenfaces. The influence of the feature dimensions was also studied in the

Table 1. Simulation results for performance evaluation of different classification methods

Feature	Eigenface			Fisherface		
	Recognition Rate (%)	FRR	FAR	CRR	FRR	FAR
HENN Classifier	11.7	0.00	88.3	8.05	0.00	91.95
SVM's with ECOC	0.00	5.10	94.9	0.00	2.25	97.75
Two-pass Classifier	0.00	1.90	98.1	0.00	0.75	99.25

simulations. It has been observed that the difference of performances is slight with the dimensions from 35 to 39. The experiment results shown in Table 1 were obtained with 39 dimensions.

4 Summary and Further Directions

In this paper we proposed a two-pass classification method and applied it to face recognitions. The new method was obtained by integrating together two methods, the HENN classifier and the SVM's with ECOC classifier. For this method, a classification operation is realized in two passes: the first one is to get an intermediate classification result for an input sample by using the HENN classifier, and the second one is followed by using the SVM's with ECOC classifier to re-classify the sample based on both the input data and the intermediate result. The simulation results obtained in the paper for applications to face recognition showed that the two-pass method can retain the advantages of both the HENN classifier and the SVM's with ECOC while remedying their disadvantages. In comparison with both the HENN classifier and the SVM's with ECOC, a significant improvement on the recognition performance has been gained by the two-pass method, which is a result of "1+1>2".

In the paper we have showed that a significant improvement can be obtained by applying the two-pass method to face recognitions. Then, applying the method to other classification problems may also get improvements, which is the problem for further study of the paper.

In the paper we also conducted a preliminary experiment study on using different input features for the classification method, which shows that better performances can be obtained by using Fisherfaces than by using eigenfaces. Therefore, getting more suitable features for the method is another problem for further study.

Finally, the problem under consideration is that, since a good combination of two methods can make an improvement over them as shown here, we may also try some other combinations based on other methods for better results.

Acknowledgement

This work was supported by Liaoning Province of China Science and Technology Foundation Grant (20022139).

References

1. Zhao, W., Chellappa, R., Phillips, P. J., Rosenfeld, A.: Face Recognition: A Literature Survey. *ACM Computing Surveys* **35**(4)(2003) 399-458
2. Chellappa, R., Wilson, C. L., Sirohey, S.: Human and Machine Recognition of Faces: A Survey. *Proceedings of the IEEE* **83** (5) (1995) 705-741
3. Vapnik, V.: *Statistical Learning Theory*. John Wiley and Sons Inc., New York (1998)
4. Sebald, D. J., Bucklew, J. A.: Support Vector Machines and Multiple Hypothesis Test Problem. *IEEE Trans. on Signal Processing* **49**(11)(2001) 2865-2872
5. Kreßel, U.: Pairwise Classification and Support Vector Machines. In: Schölkopf, B., Burges, J. C. and Smola, A. J. (eds): *Advances in Kernel Methods: Support Vector Learning*. MIT Press, Cambridge, MA (1999) 255-268
6. Dietterich, T., Bakiri G.: Solving Multiclass Learning Problems via Error-Correcting Output Codes. *Journal of Artificial Intelligence Research* **2** (1995) 263-286
7. Wang, C.B., Guo, C.A.: An SVM Classification Algorithm with Error Correction Ability Applied to Face Recognition. In: Wang, J., Yi, Z., et al (Eds): *Advances in Neural Networks – ISNN 2006*. Lecture Notes in Computer Science, Vol. 3971. Springer-Verlag, Berlin Heidelberg (2006) 1057–1062
8. Wang, S.J., Chen, X.: Biomimetic (Topological) Pattern Recognition—a New Model of Pattern Recognition Theory and Its Application. *Proceedings of the International Joint Conference on Neural Networks*, **3** (2003)2258~2262
9. Wang, S.J., Chen, X., Qi, H., et al: Double Synaptic Weight Neuron Theory and Its Application. In: Wang, L., Chen, K., Ong, Y.S. (Eds): *The First International Conference on Neural Computation*. Lecture Notes in Computer Science, Vol. 3610. Springer-Verlag, Berlin Heidelberg (2005) 264–272
10. Lin, S., Costello D. J.: *Error Control Coding: Fundamentals and Applications*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey (1983)
11. Turk, M. A., Pentland, A.: Eigenfaces for Recognition. *Journal of Cognitive Neuroscience* **3**(1) (1991) 71-86
12. Belhumeur, P.N., Hespanha, J.P., and Kriegman, D.J.: Eigenface vs. Fisherfaces: Recognition Using Class Specific Linear Projection. *IEEE Trans on PAMI* **19**(7)(1997) 711-720

SVM Based Adaptive Inverse Controller for Excitation Control

Xiaofang Yuan and Yaonan Wang

College of Electrical and Information Engineering, Hunan University,
Changsha 410082, P.R. China

yuanxiaof@21cn.com, yaonan@hnu.cn

Abstract. An adaptive inverse controller based on support vector machines (SVM) was designed for excitation control. Two SVM networks were utilized in the controller, one is SVM identifier (SVMI) and the other is SVM inverse controller (SVMC). The plant was identified by SVMI, which provided the sensitivity information of the plant to SVMC. SVMC was established using inverse system method as the pseudo-inverse model. Both SVMI and SVMC are offline learned firstly and are online trained using back propagation algorithm. To guarantee convergence and for faster learning, adaptive learning rates and convergence theorems are developed. Simulations show that this controller has better performance in system damping and transient improvement.

1 Introduction

Application of nonlinear control methods in excitation control to enhance transient stability has been given much attention in various literatures [1-4]. Since nonlinear controllers have a more complicated structure and are difficult to implement relative to linear controllers. In addition, feedback linearization methods require exact system parameters to cancel the inherent system nonlinearities, and this contributes further to the complexity of the stability analysis. Recently the use of artificial neural networks (ANN) as neuro-controllers offers a possibility to overcome this problem. The feasibility is based on the universal approximation properties as well as strong learning ability of ANN. Several literatures have been focus on ANN based excitation controller [5-6], which is an effective alternative to nonlinear controller. Unfortunately, most ANN using gradient-based training method like back-propagation, often suffer from local minima, and it is also not easy to choose a suitable network structure.

As a novel breakthrough to ANN, support vector machines(SVM) [7-8] have proved to be a powerful alternative in many areas. Here, SVM based adaptive inverse excitation controller is presented. Two SVM networks are utilized in this control system, one is the plant identifier providing plant information as learning signal for the controller, the other is an inverse model identifier acting as an inverse controller. General learning algorithm is employed in the offline learning of SVM networks and they are online trained using back-propagation algorithm.

2 Problem Formulation

We consider the third order model of a generator connected to an infinity-bus, called a single machine infinite bus (SMIB) system, which is as follows [9]

$$\dot{\delta} = \omega - \omega_0, \dot{E}'_q = -\frac{x_{d\Sigma}E'_q}{x'_{d\Sigma}T'_{d0}} + \frac{V_s(x_d - x'_d)\cos\delta}{x'_{d\Sigma}T'_{d0}} + \frac{V_f}{T'_{d0}}, \quad (1)$$

$$\dot{\omega} = \frac{\omega_0 P_m}{H} - \frac{D(\omega - \omega_0)}{H} - \frac{\omega_0 E'_q V_s \sin\delta}{H x'_{d\Sigma}} - \frac{\omega_0 V_s^2 (x'_d - x_q) \sin 2\delta}{2H x'_{d\Sigma} x_{q\Sigma}}, \quad (2)$$

where variables are presented in detail in [9]. If the power angle δ is as the output y and the excitation voltage V_f as the input u , the following third order differential equation can be deduced from (1) and (2).

$$y^{(3)} = \frac{D\ddot{y}}{H} - \frac{\omega_0 E'_q V_s \dot{y} \cos y}{H x'_{d\Sigma}} - \frac{\omega_0 V_s^2 \dot{y} (x'_d - x_q) \cos 2y}{H x'_{d\Sigma} x_{q\Sigma}} - \frac{(x'_d - x_q) V_s \cos y - x_{q\Sigma} E'_q}{x'_{d\Sigma} T'_{d0}} * \frac{\omega_0 V_s \sin y}{H x'_{d\Sigma}} - \frac{\omega_0 V_s \sin y * u}{H x'_{d\Sigma} T'_{d0}}, \quad (3)$$

$$\dot{E}'_q = \frac{2(\omega_0 P_m x'_{d\Sigma} - D(\dot{y} - \omega_0) - H\ddot{y} x'_{d\Sigma}) - \omega_0 V_s^2 (x'_d - x_q \sin 2y)}{\omega_0 V_s \sin y}. \quad (4)$$

Therefore, (3) can be denoted in the form of nonlinear mapping

$$y = f(y^{(3)}, \dot{y}, \ddot{y}, u). \quad (5)$$

During the operation range of the power angle ($0 < \delta < \pi$) there exists a $\frac{\partial y^{(3)}}{\partial u} \neq 0$. In other words, the excitation control system is invertible and so the following inverse system exists [9].

$$u = g(y^{(3)}, \dot{y}, \ddot{y}, y). \quad (6)$$

3 SVM Based Function Approximation

Compared with ANN and standard SVM, least squares SVM (LS-SVM) [8] has the following advantages: no number of hidden units has to be determined, no centers has to be specified for the Gaussian kernel, and fewer parameters have to be prescribed, so LS-SVM is employed here for the model identification.

Let $\{x_t, y_t\}_{t=1}^N$ be the set of input/output training data with input x_t and output y_t . Consider the regression model $y_t = f(x_t) + e_t$ where x_t are deterministic points, f is a smooth function and e_t are uncorrelated errors. For the purpose of estimating the nonlinear f , the following model is assumed

$$f(x) = \omega^T \varphi(x) + b, \quad (7)$$

where $\varphi(x)$ denotes a infinite dimensional feature map. The regularized cost function of the LS-SVM is given as

$$\min J(\omega, e) = \frac{1}{2}\omega^T\omega + \gamma\frac{1}{2}\sum e_t^2, y_t = \omega^T\varphi(x_t) + b + e_t, t = 1, \dots, N. \quad (8)$$

In order to solve this constrained optimization, a Lagrangian is constructed

$$L(\omega, b, e; \alpha) = J(\omega, e) - \sum \alpha_t(\omega^T\varphi(x_t) + b + e_t - y_t), \quad (9)$$

with α_t the Lagrange multipliers. The conditions for optimality are given by

$$\frac{\partial L}{\partial \omega} = 0, \frac{\partial L}{\partial b} = 0, \frac{\partial L}{\partial e_t} = 0, \frac{\partial L}{\partial \alpha_t} = 0. \quad (10)$$

Substituting (7-9) into (10) yields the following set of linear equations

$$\begin{bmatrix} 0 & 1_N^T \\ 1_N & \Omega + \gamma^{-1}I_N \end{bmatrix} \cdot \begin{bmatrix} b \\ a \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}, \quad (11)$$

with $y = [y_1, \dots, y_N]^T, 1_N = [y_1, \dots, 1]^T, \alpha = [\alpha_1, \dots, \alpha_N]^T, \Omega_{ij} = K(x_i, x_j)$.

The resulting LS-SVM model can be evaluated at a new point x_* as:

$$\hat{f}(x_*) = \sum \alpha_t K(x_*, x_t) + b, \quad (12)$$

where M is the number of support vectors (SVs), $K(\cdot, \cdot)$ is kernel function, α_t, b are the solutions to (11). Here Gaussian kernel function $K(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2})$ is selected. As the training of SVM is equivalent to a linear programming problem, it can realize global optimization effectively. Moreover, the learning results decide the number of SVs and this selects the nodes of hidden layer of SVM networks.

4 Plant Identifier and Inverse Model Identifier

From (6), in the viewing of inverse system method, the control signal u can be determined by $(y^{(3)}, \ddot{y}, \dot{y}, y)$. Here y is the power angle δ , and u is the excitation voltage V_f . By using the n -order approximation [10], one has $T\dot{y} = y(k+1) - y(k), T\dot{u} = u(k+1) - u(k), T^2\ddot{y} = y(k+1) - 2y(k) + y(k-1)$, and $T^3y^{(3)} = y(k+1) - 3y(k) + 3y(k-1) - y(k-2)$ with T is the sampling period. In this way, (6) can then be described in the discrete system as:

$$u(k) = G(y(k), y(k-1), y(k-2), y(k-3)). \quad (13)$$

The plant is modeled by a SVM identifier (SVM I) as Fig.1(a), which provides information on the plant to the controller. SVM is also used in identifying the inverse model of the plant called SVM inverse identifier (SVM II) in Fig.1(b), which serves as an inverse controller. The inputs of SVM I are $[u(k), y(k-1)$,

$y(k-2), y(k-3)]^T$, the output of SVMI is $\hat{y}(k)$ corresponding to the desired output $Y_I(k)$ be $[u(k), y(k-1), y(k-2), y(k-3)]^T$, then

$$\hat{y}(k) = \hat{F}(Y_I(k)) = \sum \alpha_t K(Y_I(k), Y_I(t)) + b = W_I^T \cdot \Phi_I(k) + b, \quad (14)$$

where $W_I = [\alpha_1, \dots, \alpha_M]^T$ are the weight vectors of LS-SVM networks as in (12), $\Phi_I(k) = [K(Y_I(k), Y_I(1)), K(Y_I(k), Y_I(M))]^T$ are the outputs of the kernel function. The inputs of SVMII are $[y(k), \dots, y(k-3)]^T$, the output of SVMII is $\hat{u}(k)$. Let $Y_c(k)$ be $[y(k), \dots, y(k-3)]^T$, then

$$\hat{u}(k) = \hat{G}(Y_c(k)) = \sum \alpha_t K(Y_c(k), Y_c(t)) + b = W_c^T \cdot \Phi_c(k) + b, \quad (15)$$

where $W_c = [\alpha_1, \dots, \alpha_M]^T$ are the weight vectors of LS-SVM networks, $\Phi_c(k) = [K(Y_c(k), Y_c(1)), K(Y_c(k), Y_c(M))]^T$ are the outputs of the kernel function.

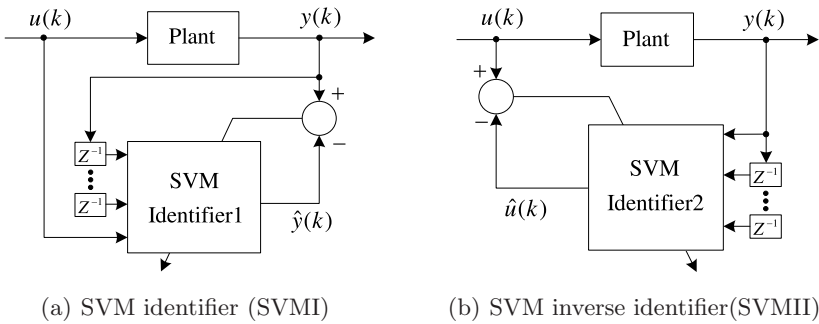


Fig. 1. Structure of the SVM identifiers

5 Adaptive Inverse Controller Design

5.1 The Structure of Adaptive Inverse Controller

Fig. 2 shows the block diagram of the control system which includes two SVM networks, that is, SVMC and SVMI. Here, SVMC is just SVMII in Fig.1(b), and SVMI is just the same as in Fig.1(a). The inputs to SVMC are the reference input $y_d(k)$, the previous plant output $[y(k), \dots, y(k-3)]$, the output of SVMC is the control signal $u(k)$. Using the back-propagation (BP) algorithm, the weights of SVMC are online adjusted such that the error $e_c(k)$, $e_c(k) = y_d(k) - y(k)$, approaches a small value. When SVMC is in training, the information on the plant is needed and SVMI is used to estimate the plant sensitivity y_u . The current control signal $u(k)$ and previous plant output $[y(k), \dots, y(k-3)]$ are the inputs to SVMI, and the output of SVMI is $\hat{y}(k)$. Let $y_d(k)$ and $y(k)$ be the desired and actual responses of the plant, then an error function is defined as

$$E_c = 0.5 * (y_d(k) - y(k))^2. \quad (16)$$

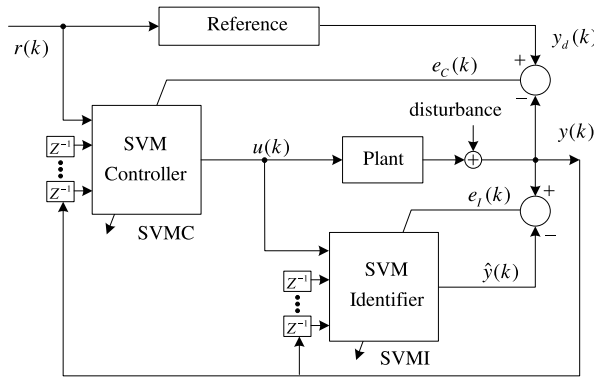


Fig. 2. Structure of the proposed adaptive inverse controller

The error function in (16) is also modified for the SVM I as

$$E_I = 0.5 * (y(k) - \hat{y}(k))^2. \tag{17}$$

The gradient of error in (16) with respect to vector W_c is given by

$$\frac{\partial E_c}{\partial W_c} = -e_c(k) \frac{\partial y(k)}{\partial W_c} = -e_c(k) y_u(k) \frac{\partial u(k)}{\partial W_c} = -e_c(k) y_u(k) \Phi_c(k), \tag{18}$$

with $y_u(k) = \frac{\partial y(k)}{\partial u(k)}$ denotes the sensitivity of the plant with respect to its input.

In the case of the SVM I, the gradient of error in (17) simply becomes

$$\frac{\partial E_I}{\partial W_I} = -e_I(k) \frac{\partial \hat{y}(k)}{\partial W_I} = -e_I(k) \frac{\partial O_I(k)}{\partial W_I} = -e_I(k) \Phi_I(k). \tag{19}$$

5.2 Learning Algorithm of Adaptive Inverse Controller

(1) Back-propagation for SVM I. From (19), the negative gradient of the error with respect to a weight vector is $-\frac{\partial E_I}{\partial W_I} = e_I(k) \frac{\partial O_I(k)}{\partial W_I} = e_I(k) \Phi_I(k)$. The weights can now be adjusted following a gradient method as

$$W_I(n + 1) = W_I(n) + \eta \left(\frac{\partial E_I}{\partial W_I} \right), \tag{20}$$

where η is a learning rate.

(2) Back-propagation for SVM C: From (18), the negative gradient of the error with respect to a weight vector is $-\frac{\partial E_c}{\partial W_c} = e_c(k) y_u(k) \frac{\partial O_c(k)}{\partial W_c} = e_c(k) y_u(k) \Phi_c(k)$. This unknown value $y_u(k)$ can be estimated by using the SVM I. When the SVM I is trained, the behavior of the SVM I is close to the plant, i.e., $y(k) \approx \hat{y}(k)$. Once the training process is done, the sensitivity can be approximated as

$$y_u(k) = \frac{\partial y(k)}{\partial u(k)} \approx \frac{\partial \hat{y}(k)}{\partial u(k)}. \tag{21}$$

Applying the chain rule to (21), and noting that $\hat{y}(k) = O_I(k)$ of (14).

$$\frac{\partial \hat{y}(k)}{\partial u(k)} = \frac{\partial O_I(k)}{\partial u(k)} = -u(k) \frac{W_I^T \cdot \Phi_I(k)}{\sigma^2}, \tag{22}$$

$$y_u(k) \approx \frac{\partial \hat{y}(k)}{\partial u(k)} = -u(k) \frac{W_I^T \cdot \Phi_I(k)}{\sigma^2}. \tag{23}$$

5.3 Convergence and Stability Based on Lyapunov Function

The update rule of (20) calls for a proper choice of η . This section develops a adaptive learning rate in selecting η properly. A Lyapunov function is given by

$$V(k) = 0.5 * e^2(k). \tag{24}$$

Thus, the change of the Lyapunov function is obtained by

$$\Delta V(k) = V(k+1) - V(k) = \frac{1}{2}(e^2(k+1) - e^2(k)) = \Delta e(k)(e(k) + \frac{1}{2}\Delta e(k)). \tag{25}$$

The error difference due to the learning can be represented by

$$e(k+1) = e(k) + \Delta e(k) = e(k) + [\frac{\partial e(k)}{\partial W}]^T \Delta W. \tag{26}$$

Now, we will give the convergence of SVM learning. From (19) and (20)

$$\Delta W_I = -\eta_I e_I(k) \frac{\partial e_I(k)}{\partial W_I} = \eta_I e_I(k) \frac{\partial O_I(k)}{\partial W_I}. \tag{27}$$

Theorem 1. *Let η_I be the learning rate for the weights of SVM and $g_{I,max}$ be defined as $g_{I,max} := \max_k \|g_I(k)\|$ where $g_I(k) = \frac{\partial O_I(k)}{\partial W_I}$, and $\|\cdot\|$ is the usual Euclidean norm. Then the convergence is guaranteed if η_I is chosen as*

$$0 < \eta_I < \frac{2}{g_{I,max}^2}. \tag{28}$$

Proof. From (25)-(27), $\Delta V(k)$ can be represented as

$$\Delta V(k) = [\frac{\partial e_I(k)}{\partial W_I}]^T \eta_I e_I(k) \frac{\partial O_I(k)}{\partial W_I} [e_I(k) + \frac{1}{2}[\frac{\partial e_I(k)}{\partial W_I}]^T \eta_I e_I(k) \frac{\partial O_I(k)}{\partial W_I}]. \tag{29}$$

For SVM, $\frac{\partial e_I(k)}{\partial W_I} = -\frac{\partial O_I(k)}{\partial W_I}$, we obtain

$$\Delta V(k) = -\eta_I e_I^2(k) \|\frac{\partial O_I(k)}{\partial W_I}\|^2 + \frac{1}{2} \eta_I^2 e_I^2(k) \|\frac{\partial O_I(k)}{\partial W_I}\|^4 \equiv -\lambda_I e_I^2(k). \tag{30}$$

Let $g_I(k) = \frac{\partial O_I(k)}{\partial W_I}$, $g_{I,max} := \max_k \|g_I(k)\|$, and let $\eta_1 = \eta_I g_{I,max}^2$. Then

$$\lambda_I = \frac{1}{2} \|g_I(k)\|^2 \eta_I (2 - \eta_1 \frac{\|g_I(k)\|^2}{g_{I,max}^2}) \geq \frac{1}{2} \|g_I(k)\|^2 \eta_I (2 - \eta_1) > 0. \tag{31}$$

From (31), we obtain $\Delta V(k) = -\lambda_I e_I^2(k)$ and $0 < \eta_1 < 2$, and (28) follows.

Remark 1. The convergence is guaranteed as long as (31) is satisfied, i.e., $\eta_I(2 - \eta_1) > 0$ or $\frac{\eta_1(2-\eta_1)}{g_{I,max}^2} > 0$. This implies that any η_1 , $0 < \eta_1 < 2$, guarantees the convergence. However, the maximum learning rate which guarantees the most rapid or optimal convergence is corresponding to $\eta_1 = 1$, i.e., $\eta_I^* = \frac{1}{g_{I,max}^2}$, which is the half of the upper limit in (28). This shows an interesting result that any other learning rate larger than η_I^* does not guarantee the faster convergence.

Now, we will give the convergence of SVMC learning. From the update (20)

$$\Delta W_c = -\eta_c e_c(k) \frac{\partial e_c(k)}{\partial W_c} = \eta_c e_c(k) y_u(k) \frac{\partial O_c(k)}{\partial W_c}. \tag{32}$$

Theorem 2. Let η_c be the learning rate for the weights of SVMC and $g_{c,max}$ be defined as $g_{c,max} := \max_k \|g_c(k)\|$, where $g_c(k) = \frac{\partial O_c(k)}{\partial W_c}$, and $S_{max} = \max_k \|y_u(k)\|$. Then the convergence is guaranteed if η_c is chosen as

$$0 < \eta_c < \frac{2}{S_{max}^2 g_{c,max}^2}. \tag{33}$$

Proof. From (25), (26), (27) and (32), $\Delta V(k)$ can be represented as

$$\Delta V(k) = \frac{\partial e_c(k)}{\partial W_c} \eta_c e_c^2(k) y_u(k) \frac{\partial O_c(k)}{\partial W_c} + \frac{1}{2} \left[\frac{\partial e_c(k)}{\partial W_c} \right]^2 \eta_c^2 e_c^2(k) y_u^2(k) \left[\frac{\partial O_c(k)}{\partial W_c} \right]^2. \tag{34}$$

For SVMC, $\frac{\partial e_c(k)}{\partial W_c} = -y_u(k) \frac{\partial O_c(k)}{\partial W_c}$, we obtain

$$\Delta V(k) = \frac{1}{2} \eta_c^2 e_c^2(k) y_u^4(k) \left\| \frac{\partial O_c(k)}{\partial W_c} \right\|^4 - \eta_c e_c^2(k) y_u^2(k) \left\| \frac{\partial O_c(k)}{\partial W_c} \right\|^2 \equiv -\lambda_c e_c^2(k). \tag{35}$$

Conditions of (32) and (27) are similar except $y_u(k)$ needs to be incorporated in the SVMC. Therefore, it remains to find the limit on $y_u(k)$. From (21) and (23)

$$S_{max} = |y_u(k)|_{max} = \left| \frac{u(k) W_I^T \cdot \Phi_I(k)}{\sigma^2} \right| \leq \frac{|W_I^T \cdot \Phi_I(k)|_{max} |u(k)|_{max}}{\sigma^2}, \tag{36}$$

where S_{max} is the limit on sensitivity and is estimated from (36). σ is a parameter seen from (14) and we can determine σ from the learning data-sets. Thus following the proof of Theorem 1, we obtain $0 < \eta_c < \frac{2}{S_{max}^2 g_{c,max}^2}$.

Remark 2. In the case of SVMI, the optimal convergence rate is $\eta_c^* = \frac{1}{S_{max}^2 g_{c,max}^2}$.

6 Simulation

The performances of the proposed adaptive excitation controller (AIC) are compared with other two controllers, one is the conventional AVR and governor

controllers [5], and the other is the neuro-controller [6]. Parameters of the plant are given as: $x_d = 2.156$; $x_q = 2.101$, $x'_d = 0.265$; $x_T = 0.1$; $x_L = 1.46$; $D = 5$; $H = 8$; $T'_{d0} = 10$; $P_m = 0.6$.

For the off-line learning, we first generate random signals as inputs $u(k)$ to the plant. Here we selected multi-amplitude varying-step square wave as testing signals. By sampling the inputs and outputs at high speed and after computing the derivatives off-line, we obtain training data $[u(k), y(k), \dots, y(k - 3)]$. The training data-set consisted of 400 samples, and LS-SVM parameters are: $\sigma = 0.42$, $\gamma = 200$. Results are displayed in Fig.3-4, the conventional controllers (CON1) are indicated by dashed lines, neuro-controller (CON2) response by solid lines, and the adaptive excitation controller (AIC) response by thick solid lines.

Example 1. Step changes in the reference voltage of the exciter. The plant is operating at a steady-state condition ($P_t = 1.0pu$, $Q_t = 0.18pu$). At $t = 2s$, a 5% step increase in the reference voltage of the exciter is applied. At $t = 10s$, the 5% step increase is removed, and the system returns to its initial operating point. Fig. 3 show that AIC improve the transient system damping compared to other two controllers.

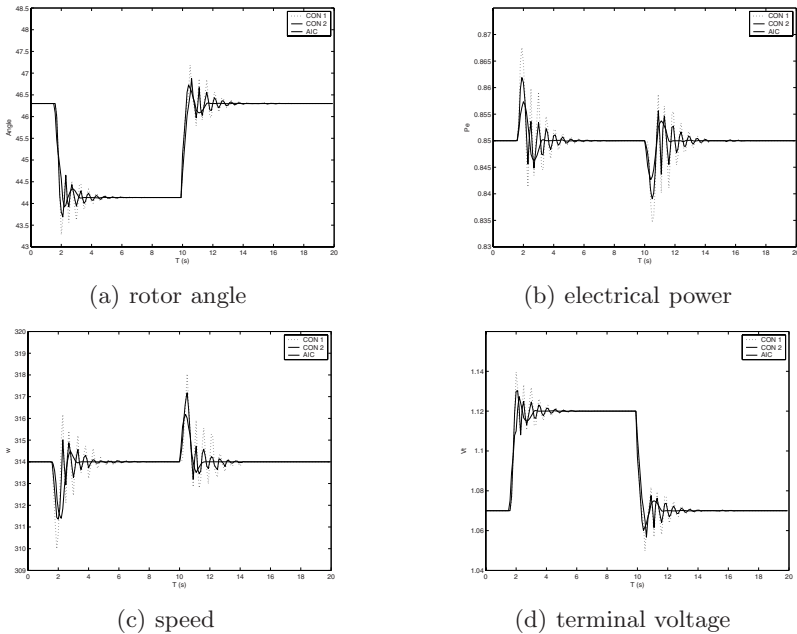


Fig. 3. Response to step changes in the reference voltage of the exciter

Example 2. Three phases short circuit test. A severe test is carried out to evaluate the performances of controllers under a large disturbance. At $t = 2s$, a temporary three-phase short circuit is applied at the infinite bus for $100ms$ from $2s$ to $2.1s$ for the plant operating at the same steady state condition as previous test. Fig.4

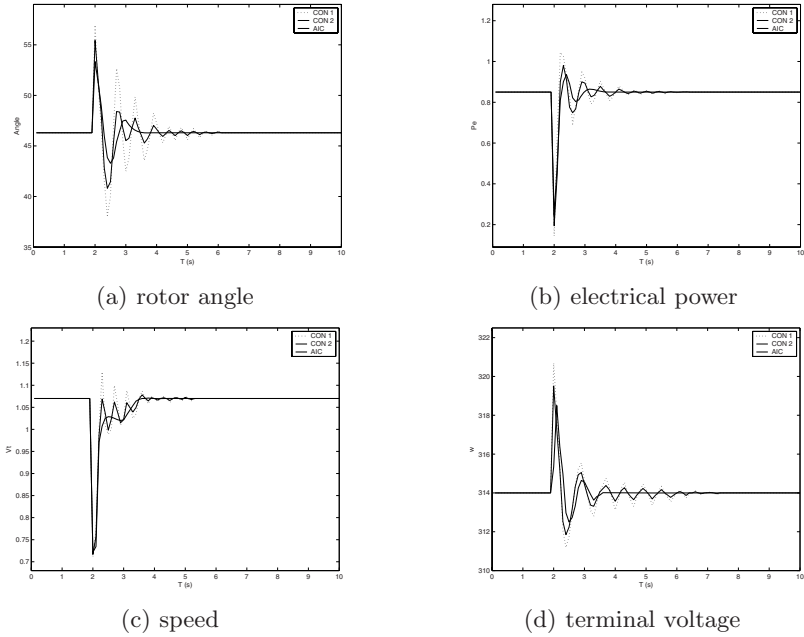


Fig. 4. Response to three phases short circuit test

show that the AIC damp out the low frequency oscillations for the rotor angle and terminal voltage more effectively than other two controllers.

In these two tests, SVM based adaptive excitation controller returns the system to stable conditions with a much better damping compared to other controllers. While adaptive excitation controller still gives a good result because it can be learned from all kinds of operating states.

7 Conclusion

This paper presents SVM based adaptive inverse controller, the superior performance of SVM over ANN is due to the following reasons: (1) SVM implements the structural risk minimization principle which minimizes an upper bound for the generalization error rather than minimizing the training error in ANN. (2) ANN may not converge to global solutions due to its inherent algorithm design. In contrast, finding solutions in SVM is equivalent to solving a linearly constrained quadratic programming, which leads to a global optimal solution. (3) In choosing parameters and structure, SVM are less complex than ANN. Results show that SVM adaptive inverse controller is very promising for future real-time applications. Not only do it improves the system damping and dynamic transient stability more effectively than other two controllers for the large disturbance, but also it has a faster transient response for a small disturbance.

References

1. Lu, Q., Sun, Y.Z.: Nonlinear Stabilization Control of Multimachine Systems. *IEEE Trans. Power Systems* **4** (1989) 236-241
2. Tan, Y., Wang, Y.: Augmentation of Transient Stability Using a Supper Conduction Coil and Adaptive Nonlinear Control. *IEEE Trans. Power Systems* **13** (1998) 361-366
3. Wang, Y., Hill, D.J.: Robust Nonlinear Coordinated Control of Power Systems. *Automatica* **32** (1996) 611-618
4. Shen, T., Mei, S., Lu, Q.: Adaptive Nonlinear Excitation Control with L2 Disturbance Attenuation for Power Systems. *Automatica* **39** (2003) 81-89
5. Venayagamoorthy, G.K., Harley, R.G., Wunsch, D.C.: Dual Heuristic Programming Excitation Neurocontrol for Generators in a Multimachine Power System. *IEEE Trans. Industry Applications* **39** (2003) 382-394
6. Fan, S., Mao, C., Lu, J.: Real-time Excitation Controller Using Neural Networks. *Engineering Intelligent Systems* **11** (2003) 151-156
7. Suykens, J.A.K.: Support Vector Machines: A Nonlinear Modeling and Control Perspective. *European Journal of Control* **7** (2001) 311-327
8. Suykens, J.A.K., Lukas, L., Vandewalle, J.: Sparse Approximation Using Least Squares Support Vector Machines. In: *Proc. 2000 IEEE International Symposium on Circuits and Systems* (2000) 757-760
9. Dai, X., He, D., Zhang, T., Zhang, K.: ANN Generalized Inversion for the Linearization and Decoupling Control of Nonlinear Systems. *IEE Proc.-Control Theory Application* **150** (2003) 267-277
10. Ge, S.S., Zhang, J., Lee, T.H.: Adaptive MNN Control for a Class of Nonaffine NARMAX Systems with Disturbances. *Syst. Contr. Lett.* **53** (2004) 1-12

An Adaptive Internal Model Control Based on LS-SVM

Changyin Sun^{1,2} and Jinya Song²

¹ School of Automation,
Southeast University, Nanjing 210096, P.R. China

² College of Electrical Engineering,
Hohai University, Nanjing 210098, P.R. China
cysun@seu.edu.cn

Abstract. Based on least squares support vector machines regression algorithm, reverse model of system model is constructed, and adaptive internal model controller is developed in this paper. First, least squares support vector machine (LS-SVM) regression model and its training algorithm are introduced, provides SMO-based on pruning algorithms for LS-SVM. Then it is used in adaptive internal model control (IMC) for constructing internal model and designing the internal model controller. At last, LS-SVM regression based adaptive internal model control is used to control a benchmark nonlinear system. Simulation results show that the controller has simple structure, good control performance and robustness.

1 Introduction

Internal model control (IMC) is a powerful technique for design of robust controllers for linear systems, which has several attractive features when compared with the conventional feedback control. The IMC structure also facilitates the analytical treatment of unconstrained linear model predictive control (MPC) algorithms in the framework of classical frequency domain techniques. The nonlinear extension of linear IMC formulation was proposed by Economou for open loop stable MIMO nonlinear systems with stable inverses. Similar to the linear IMC strategy, the nonlinear IMC controller was designed as a right inverse of the nonlinear model operator. The IMC has been solved by neural network (NN) [1], the robust stabilization and performance of the control system may then be a problem. Wang solves the problem by support vector machines (SVM) [2]. Compared with NN, SVM has well generalization ability, and is especially fit for machine learning in small sample condition. The training algorithm of SVM will not run into local minimum point. SVM is a new machine learning method and has been used for classification, function regression, and time series prediction, etc. Also it can automatically construct the structure of system model [2]. However, the accuracy is not very high, while the accuracy is very important for the control system. Even a little error may produce different results. Different training methods lead to different training accuracy, so we must find the methods

which produce high accuracy. As an interesting variant of the standard support vector machines, least squares support vector machines (LS-SVMs) have been proposed by Suykens and Vandewalle [3-4] for solving pattern recognition and nonlinear function estimation problems. Standard SVM formulation is modified in the sense of ridge regression. LS-SVM takes equality instead of inequality constraints of SVM in the problem formulation. As a result one solves a linear system instead of a QP problem, so LS-SVM is easy to training. There are several methods for training LS-SVM, but the computational cost are quite a big problem. Zeng proposed a SMO-Based Pruning Methods for Least Squares Support Vector Machines method for classification [5]. Sequential minimal optimization (SMO) are extended by Keerthi and Shevade to solve the linear equations in LS-SVMs. It is suitable for large scale problems and is convenient for timely data updating. This paper introduces this method in LS-SVM regression into internal model control to construct an inverse controller. The effectiveness of the proposed method in terms of computational cost and regression accuracy is demonstrated by numerical experiments. The paper is organized as follows: In section 2 LS-SVM regression model and its training algorithm are described; in sector 3 SMO and pruning algorithms for LS-SVM is introduced; in sector 4 least square support vector machine and internal model control is proposed; in sector 5 simulation and results are used to test the performance of algorithm; Finally, conclusions are drawn in Sector 6.

2 LS-SVM Regression Model and Its Training Algorithm

In the following, we briefly introduce LS-SVM regression. Consider a given training set of N data points $\{x_i, y_i\}_{i=1}^N$, with input data $x_i \in R$ and output $y_i \in R$. In feature space LS-SVM models take the form:

$$y(x) = w^T \psi(x) + b \quad (1)$$

where the nonlinear mapping $\psi(\cdot)$ maps the input data into a higher dimensional feature space. Note that the dimensional of w is not specified (it can be infinite dimensional). In LS-SVM for function estimation the following optimization problem is formulated

$$\min \frac{1}{2} w^T w + \frac{C}{2} \sum_{K=1}^N e_i^2 \quad (2)$$

subject to the equality constrains

$$y_i(w \cdot \varphi(x_i) + b) = 1 - e_i, \quad i = 1, \dots, N \quad (3)$$

where C is a regularization factor and e_i is the difference between the desired output and the actual output. For simplicity, we consider the problem without a bias term, as did in [5]. The Lagrangian for problem (2) is

$$R(w, e_i; \alpha_i) = \frac{1}{2} w^T w + \frac{1}{2} C \sum_i e_i^2 + \sum_i \alpha_i [y_i - w \cdot \varphi(x_i) - e_i] \quad (4)$$

where α_i are Lagrangian multiplier .

The nonlinear regression function (the output of LS-SVM) can be formulated by:

$$y(x) = \sum_{i=1}^l \alpha_i k(x, x_i) + b \tag{5}$$

$k(x_i, x_j)$ is a symmetric function which satisfies Mercer conditions. Some useful kernels are as following:

1) Polynomial kernel:

$$k(x, x_i) = [(x \cdot x_i) + 1]^q \tag{6}$$

2) RBF kernel:

$$k(x, x_i) = \exp(-\|x - x_i\|^2 / \sigma^2) \tag{7}$$

3) Sigmoid kernel:

$$k(x, x_i) = \tanh(v(x \cdot x_i + c)) \tag{8}$$

formula (6 ~ 8), parameter q, R, c are all real constant. In actual application, usually we must choice appropriate kernel function as well as the corresponding parameter according to the certain condition. The choice of the kernel function has several possibilities. In this work, the radial basis function (RBF) is used as the kernel function of the LS-SVM because RBF kernels tend to give good performance under general smoothness assumptions.

3 SMO-Based Pruning Algorithms for LS-SVM

The sparseness is very important for LS-SVM regression. The sparseness is imposed by subsequently omitting data that introduce the smallest training errors and retraining the remaining data. In the following, the SMO-Based pruning Algorithms are given in [5]. From (4), the Karush-Kuhn-Tucker (KKT) conditions for optimality are:

$$\begin{cases} \frac{\partial R}{\partial w} = 0 & w = \sum_i \alpha_i \varphi(x_i) \\ \frac{\partial R}{\partial e_i} = 0 & \alpha_i = Ce_i \\ \frac{\partial R}{\partial \alpha_i} = 0 & y_i - w \cdot \varphi(x_i) - e_i = 0 \end{cases} \tag{9}$$

by substituting the KKT conditions (9) into the Lagrangian (4), the dual problem is to maximize the flowing objective function:

$$\max(L(\alpha)) = -\frac{1}{2} \sum_j \sum_i \alpha_i \alpha_j Q(x_i, x_j) + \sum_i \alpha_i y_i \tag{10}$$

where $Q(x_i, x_j) = K(x_i, x_j) + \sigma_{ij}/C$, and if $i = j, \sigma_{ij} = 1$; otherwise $\sigma_{ij} = 0$.

The SMO algorithm works by optimizing only one α_i at a time keeping the others fixed, i.e., α is adjusted by a step t as follows:

$$\alpha_i^{new} = \alpha_i + t; \quad \alpha_j^{new} = \alpha_j, \quad \forall j \neq i \tag{11}$$

Define

$$f_i = -\frac{\partial L}{\partial \alpha_i} = -y_i + \sum_{j=1}^N \alpha_j Q(x_i, x_j) \tag{12}$$

the t can be suggested as in [5],

$$t = \frac{-f_i}{Q(x_i, x_i)} \tag{13}$$

The criterion for determination of pruning points is a crucial factor in pruning process. In this section, we detail a new criterion that is directly based on the dual objective function and easy to compute in SMO formulation. To derive the proper criterion for pruning, the dual objective function (10) is rewritten using the definition of f_i .

$$L(\alpha) = \frac{1}{2} \sum_i \alpha_i (y_i - f_i) \tag{14}$$

Along with the idea of SMO, we consider that the removal of a sample k does not directly affect the support values of other samples, but it introduces the update of all f_i , which leads to a difference in the objective function, it is suggested in [5], as :

$$d(L) = \frac{1}{2} \alpha_k^2 Q(x_k, x_k) - \alpha_k F_k \tag{15}$$

A summary of this training algorithm are as follows:

Step 1) Train the initial non-sparse LS-SVM using the SMO formulation as described in 4, using the training data set got in section 3.

Step 2) Repeat the following inner loop by time tt :

- remove a sample from the training set using criterion (15);
- update $f_i, \forall i \neq k$, of the remaining samples in the training set using $f'_i = f_i - \alpha_k Q(x_i, x_k)$, where k is the omitted data point.

Step 3) Retrain the LS-SVM using the SMO formulation based on the support values α and the updated f of the remaining data set, where $\alpha' = (\alpha_1, \dots, \alpha_{k-1}, \alpha_{k+1}, \dots, \alpha_N)$ and $f' = (f'_1, \dots, f'_{k-1}, f'_{k+1}, \dots, f'_N)$.

Step 4) Repeat Step 3) and Step 4) until the defined termination condition is satisfied.

Step 5) Get the LS-SVM model.

4 Least Square Support Vector Machine and Internal Model Control

The system structure of adaptive inverse control is shown in Fig 1. y_{sp} is reference input, u is output of LS-SVM based Controller, y is system output, y_m is predictive output of LS-SVM based internal model.

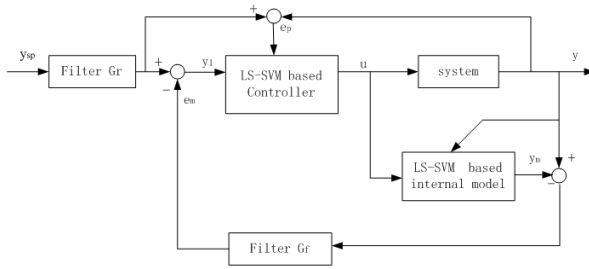


Fig. 1. LS-SVM based IMC system

4.1 Constructing of Internal Model

Assume that discrete SISO nonlinear system can be describe as below:

$$y(k+1) = f(y(k), \dots, y(k-n), u(k), \dots, u(k-m)), y \in R^n, u \in R^n, m \leq n \tag{16}$$

where f is a nonlinear function. Then we can define training samples set as [2]:

$$\begin{cases} D^1 = \{x_i^1, y_i^1\}, & i = 1, 2, \dots, l \quad Y_i^1 = y(k + 1) \\ X_i^1 = [y(k), \dots, y(k - n + 1), u(k), \dots, u(k - m + 1)] \end{cases} \tag{17}$$

then we can use the methods proposed in sector 3 to train the data sets , so we can get the internal model .

4.2 Designing of Inverse Model Controller

The controller of IMC is the inverse model of the controlled system. So the reversibility of a system must be first considered. For linear system, reversibility is a problem about controllability. If only a linear system is state controllable, then it is reversible [7].

Given $[y(k), \dots, y(k - n), u(k - 1), \dots, u(k - m)]$, when any $u'(k) \neq u(k)$, we can get: $f[y(k), \dots, y(k - n), u'(k), \dots, u(k - m)] \neq f[y(k), \dots, y(k - n), u(k), \dots, u(k - m)]$, so system is reversible at point $[y(k), \dots, y(k - n), u(k - 1), \dots, u(k - m)]^T$.

Theorem 1. If $f[y(k), \dots, y(k - n), u(k), \dots, u(k - m)]$ is strictly monotone function to $u(k)$, then system is reversible at point $[y(k), \dots, y(k - n), u(k - 1), \dots, u(k - m)]^T$. If above-mentioned conclusion is right at any time step k , the system (1) is reversible system.

According to (16), inverse model of system can be defined as:

$$u(k) = f[y(k + 1), y(k), \dots, y(k - n), u(k - 1), \dots, u(k - m)]^T$$

The training set of LS-SVM can be constructed as following [2]:

$$\begin{cases} D^2 = \{x_i^2, y_i^2\}, & i = 1, 2, \dots, l \quad Y_i^1 = u(k) \\ X_i^2 = [y(k + 1), y(k), \dots, y(k - n), u(k - 1), \dots, u(k - m)] \end{cases} \tag{18}$$

then we can use method described in sector 3 to train inverse model of system, which is also inverse model controller.

5 Simulation and Result

A benchmark problem [8] is used to illustrate the results of on-line SVM based adaptive internal model control. In the benchmark problem, the nonlinear plant is described as follows: $y(k + 1) = y(k)/(1 + y^2(k)) + u^3(k) + v$

Where y is the plant output, u is the input and v is a random disturb. It's easy to prove that it is monotone system, so according to theorem 1, it's reversible system. Assume that the structure of the system is unknown, according to (17) and (18) the training sample of internal model and internal model controller can be separately constructed as:

$$\begin{aligned} X_i^1 &= [y(k), y(k - 1), u(k), u(k - 1)] & Y_i^1 &= y(k + 1) \\ X_i^2 &= [y_1(k + 1), y_1(k), y(k - 1), u(k - 1)] & Y_i^2 &= u(k) \end{aligned}$$

Getting those data sets $\{X_i^1, Y_i^1\}$, $\{X_i^2, Y_i^2\}$ then we can train and test LS-SVM.

We use the SMO-Based on pruning methods to train the LS-SVM. The parameter are $C = 150, \sigma = 0.01$. Random disturb v has a positive step disturb with a amplitude of 0.1 in step 80 and a negative step disturb with a amplitude of 0.1 in step 120 and 160. In order to show the superiority of this method, we use LS-SVM and SVM to do internal model control. The differences between LS-SVM and SVM in computational cost and regression accuracy are shown in Table 1.

Table 1. Computational cost and Regression accuracy of LS-SVM and SVM

	LS-SVM	SVM
regression accuracy	0.0039	0.0302
computational cost	3.1560	45.312 s

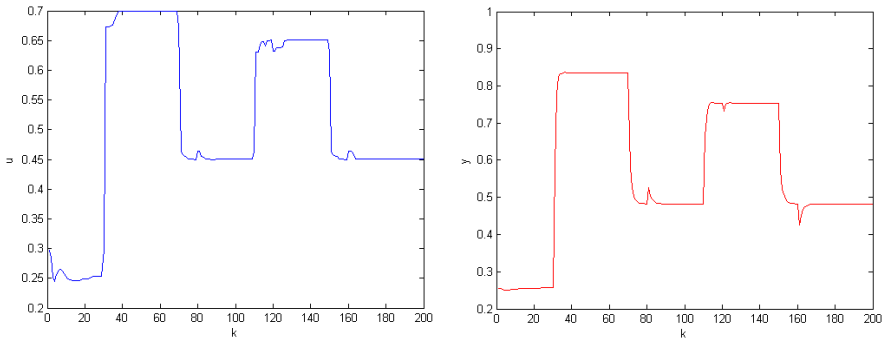


Fig. 2. Control input u and system output y

First 200 initial training samples constructed by random input are used to train a initial internal model and internal model controller. Simulation results are shown in Figure 2. From these results we can see that LS-SVM Regression model and its training algorithm can quickly approximate the system model with very high precision and the LS-SVM based predictive controller can control the system response very well. Also when there are random step disturbs, the controller can rapidly counteract influence from disturbs and come back system response in several time steps, which shows that the system has well robustness.

6 Conclusion

In this paper we introduce the use of SMO-Based pruning methods for least square support vector machines for solving internal model control problems. An introduction to LS-SVM is given at first, then gives its training algorithm, and uses it to build a internal model control framework, the numerical experiment has shown the efficiency of the LS-SVM based method. It can control nonlinear system with good performance and robustness.

Acknowledgement

The authors would like to thank the reviewers for their helpful comments and constructive suggestions, which have been very improving the presentation of this paper. This work was supported by the Natural Science Foundation of Jiangsu Province, China under Grant BK2006564 and China Postdoctoral Science Foundation under Grant 20060400274.

References

1. Hunt, K.J., Sburbuo, D.: Neural Networks for Nonlinear Internal Model Control. *IEEE Proc-D* **13** (1991) 431-438
2. Wang, H., Pi, D.Y., Sun, Y.X.: Fast Online SVR Algorithm Based Adaptive Internal Model Control. Springer-Berlin/Heidelberg **3972** (2006) 922 - 927
3. Suykens, J.A.K., Vandewalle, J.: Least Squares Support Vector Machine Classifiers. *Neural Processing Letter* **9** (1999) 293-300
4. Gritianini, N., Taylor, J.S.: An Introduction to Support Vector Machines. Cambridge University Press, 2000
5. Zeng, X.Y., Chen, X.W.: SMO-Based Pruning Methods for Sparse Least Squares Support Vector Machines. *IEEE transactions on Neural Networks* **16** (2005) 1541-1546
6. Radhakrishnan, T.K., Sundaram, S., Chidambaram, M.: Non-Linear Control of Continuous Bioreactors. Springer-Berlin/Heidelberg **20** (2005) 173-178
7. Wang, D.C., Fang, T.J.: An Internal Model Control Algorithm Based on Support Vector Machine. *Control Theory and Application* **21** (2004) 85-88 (in chinese)
8. Narendra, K.S., Parthasarathy, K.: Identification and Control of Dynamic Systems Using Neural Networks. *IEEE Transactions on Neural Networks* **1** (2000) 4-27

Regularization Paths for ν -SVM and ν -SVR

Gaëlle Loosli, Gilles Gasso, and Stéphane Canu

LITIS, EA 4051, Rouen, France
firstname.name@insa-rouen.fr

Abstract. This paper presents the ν -SVM and the ν -SVR full regularization paths along with a leave-one-out inspired stopping criterion and an efficient implementation. In the ν -SVR method, two parameters are provided by the user: the regularization parameter C and ν which settles the width of the ϵ -tube. In the classical ν -SVM method, parameter ν is an lower bound on the number of support vectors in the solution. Based on the previous works of [12], extensions of regularization paths for SVM and SVR are proposed and permit to automatically compute the solution path by varying ν or the regularization parameter.

1 Introduction

The utilization of SVM by neophyte users is still hampered by the need to supply values for control parameters in order to get the best attainable results. Mainly, SVM's users must make three choices: the kernel, its bandwidth and the regularization parameter. Within the usual formulation of the soft margins SVM, the regularization parameter takes its value between 0 (random) and ∞ (hard-margins). The ν -SVM [3] technique reformulates the SVM problem so that C is replaced by a ν parameter taking values in $[0, 1]$. This re-normalized parameter has a more intuitive meaning: it represents the minimal proportion of points in the solution and the maximal proportion of misclassified points. The SVR algorithm is a popular method for dealing with regression problems [4]. The algorithm minimizes the ϵ -insensitive cost while preserving the smoothness of the regression function. The trade-off is realized via a regularization parameter λ set by the user. The user also provides the width $\epsilon \in [0, \infty[$ of the tube. As the practical choice of ϵ is difficult, the ν -SVR method [4] was proposed and permits to automatically determine the value of ϵ . Given the importance of this problem for reaping all the potential benefits of the use of SVM and SVR, many research work have been dedicated to ways of helping the setting of the parameters. Most rely on either *outer* measures, such as cross-validation, to guide the selection, or to measures embedded in the learning method itself (see [5,6]). In place of these empirical approaches regularization paths have been widely studied recently [12,7] since they provide a smart and fast way to access all the optimal solutions of a problem according to all compromises between bias and regularity. However, having the whole regularization path is not enough. Indeed, the end user still needs to retrieve from it the best values for the regularization parameters. Instead of selecting these values by k -fold cross-validation or leave-one-out,

or other approximations [8], we propose to include the leave-one-out estimator inside the regularization path in order to have an idea of the generalization error at each step. We explain why it is less expansive than selecting the best parameter *a posteriori* and give a method to stop learning before attaining the end of the path to save useless efforts. This paper presents a low-cost (in term of computational time and memory) method for the auto-setting of the regularization parameter for the classification case. Contrarily to what is usually done for regularization path, our method does not start with all points as support vectors. Doing so we avoid the computation of the whole Gram matrix at the first step. Then, since the proposed method stops on the path, this extreme non-sparse solution is never attained and thus the whole Gram matrix never required. One of the main advantage of this is that it is possible to use this setting for large databases.

2 Regularization Path for ν -SVM

We consider a binary classification problem with training patterns $x_1 \dots x_m \in \mathcal{X}$ and associated classes $y_1 \dots y_m \in \{+1, -1\}$. A ν -SVM classifies a pattern x according to the sign of the decision function $f(\cdot) = \frac{1}{m} \sum_{i=1}^m \alpha_i y_i k(x_i, \cdot) + b$. A pattern x_i is called “support vector” when the corresponding coefficient $\alpha_i \neq 0$. The hyper-parameter ν can be scaled by the size of the training database. In this case, we have $\lambda = m\nu$ and λ is a lower bound on the number of support vectors. Since we have at least a point in the solution, we can set $1 \leq \lambda \leq m$. The primal ν -SVM problem is written as follows:

$$\begin{cases} \min_{f,b,\rho,\xi_i} \frac{m}{2} \|f\|^2 - \lambda\rho + \sum_{i=1}^m \xi_i \\ \text{s.t.} & y_i(f(x_i) + b) \geq \rho - \xi_i, \quad \xi_i \geq 0, \forall i \in [1, \dots, m] \quad \text{and} \quad \rho \geq 0 \end{cases}$$

with ρ being the margin to be optimized and the dual problem is:

$$\begin{cases} \max_{\alpha} -\frac{1}{2} \alpha^\top G \alpha \\ \text{s.t.} \quad \alpha^\top \mathbf{1} \geq \lambda, \quad \alpha^\top \mathbf{y} = 0 \quad \text{and} \quad 0 \leq \alpha_i \leq 1 \quad \forall i \in [1, \dots, m] \end{cases} \tag{1}$$

where $G(i, j) = \frac{1}{m} y_i y_j k(x_i, x_j)$. Our aim is to compute the ν -SVM solution for all values of ν . As shown in [1], the path is piecewise linear. It means that the support vectors set does not change between two values of ν . Hence we only need to identify when a change occurs in the sets. Similarly to what is done in active set methods solving the SVM [9], the first steps consists in identifying the best direction to follow and the second step is to determine how far to follow it. Let note $g(x_i) = y_i(f(x_i) + b) - \rho$. Then we have:

$$\begin{cases} \mathcal{L} : g(x_i) < 0 \quad \forall i \in \mathcal{L} & \alpha_i = 1 & \text{bounded points} \\ \mathcal{E} : g(x_i) = 0 \quad \forall i \in \mathcal{E} & 0 < \alpha_i < 1 & \text{margins points} \\ \mathcal{R} : g(x_i) > 0 \quad \forall i \in \mathcal{R} & \alpha_i = 0 & \text{useless points} \end{cases}$$

The idea of the path is to provide an iterative method that follows the path by pieces, stopping at each change among the groups. Each point of the path

reflects the optimal solution of the ν -SVM according to a particular value of λ . We begin with the smallest value of λ and let it grow up to its larger value. Since the provided solutions are equivalent as long as the groups do not change, we only need to identify for which values of λ a point is going to move from one group to another. We identify four possible movements: from \mathcal{L} to \mathcal{E} (from the wrong side of the margin to the margin), from \mathcal{R} to \mathcal{E} (the point becomes support vector), from \mathcal{E} to \mathcal{L} (a support vector becomes bounded) and from \mathcal{E} to \mathcal{R} (a support vector is no longer one). We will denote those steps respectively as $in(\ell)$, $in(r)$, $out(\ell)$ and $out(r)$. Events $in(\ell)$ and $in(r)$ happen when $\exists i \in \mathcal{L}$ or $i \in \mathcal{R}$ such that $g(x_i) = 0$. Events $out(\ell)$ and $out(r)$ occur respectively when $\exists i \in \mathcal{E}$ such that $\alpha_i = 1$ or $\alpha_i = 0$. Hence we need to express $g(x)$ and α depending on steps t and $t + 1$ as:

$$\begin{aligned}
 g^{t+1}(x_i) &= g^{t+1}(x_i) - g^t(x_i) + g^t(x_i) \\
 &= G(i, :) \alpha^{t+1} + b^{t+1} y_i - \rho^{t+1} - G(x_i, :) \alpha^t - b^t y_i + \rho^t + g^t(x_i) \quad (2) \\
 &= G(i, :) \delta_\alpha + \delta_b y_i - \delta_\rho + g^t(x_i)
 \end{aligned}$$

with $\delta_\alpha = \alpha^{t+1} - \alpha^t$, $\delta_b = b^{t+1} - b^t$ and $\delta_\rho = \rho^{t+1} - \rho^t$. $G(i, :)$ designates the i^{th} row of the matrix. From equation 2 and depending on the concerned event, it is possible to find which value of λ to choose next. We summarize in table 1 the events and the algorithm mechanic.

Table 1. Summary of the events. Each column stands for a particular event. In blue starred are noted the properties that are used to compute the corresponding λ^{t+1} .

Step	$in(r)$	$out(r)$	$out(\ell)$	$in(\ell)$
t	$i \in \mathcal{R}$ $g^t(x_i) > 0$ $\alpha_i = 0$	$i \in \mathcal{E}$ $*g^t(x_i) = 0$ $0 < \alpha_i < 1$	$i \in \mathcal{E}$ $*g^t(x_i) = 0$ $0 < \alpha_i < 1$	$i \in \mathcal{L}$ $g^t(x_i) < 0$ $\alpha_i = 1$
$t + 1$	$i \in \mathcal{E}$ $*g^{t+1}(x_i) = 0$ $0 < \alpha_i < 1$	$i \in \mathcal{R}$ $*g^{t+1}(x_i) \geq 0$ $*\alpha_i = 0$	$i \in \mathcal{L}$ $*g^{t+1}(x_i) \leq 0$ $*\alpha_i = 1$	$i \in \mathcal{E}$ $*g^{t+1}(x_i) = 0$ $0 < \alpha_i < 1$

Points in \mathcal{E} and Detection of $out(\ell)$ and $out(r)$. Events $out(\ell)$ and $out(r)$ are detected using their values of α . Indeed, one condition to remain in \mathcal{E} is to keep $0 < \alpha < 1$. Retrieving λ^{t+1} for which one of these conditions is violated for each point requires to write α_i depending on λ^{t+1} . Remark that in \mathcal{E} , $g^t(x) = 0$ and $g^{t+1}(x) = 0$. Equation 2 together with the constraints from 1 ($\alpha^\top \mathbf{1} \geq \lambda$, hence $\delta_\alpha^\top \mathbf{1} \geq \lambda^{t+1} - \lambda^t$ and $\alpha^\top \mathbf{y} = 0$, hence $\delta_\alpha^\top \mathbf{y} = 0$) leads to a system of linear equations $A\delta = (\lambda^{t+1} - \lambda)\mathbf{c}$, where

$$A = \begin{bmatrix} G & \mathbf{y} & -\mathbf{1} \\ \mathbf{y}^\top & \mathbf{0} & \mathbf{0} \\ \mathbf{1}^\top & \mathbf{0} & \mathbf{0} \end{bmatrix} \quad \delta = \begin{bmatrix} \delta_\alpha \\ \delta_b \\ \delta_\rho \end{bmatrix} \quad \mathbf{c} = \begin{bmatrix} \mathbf{0} \\ 0 \\ 1 \end{bmatrix}$$

This leads to $\delta = (\lambda^{t+1} - \lambda^t)A^{-1}\mathbf{c}$. So for points in the set \mathcal{E} there is:

$$\begin{cases} \alpha^{t+1} = \alpha^t + (\lambda^t - \lambda^{t+1})\eta_\alpha \\ b^{t+1} = b^t + (\lambda^t - \lambda^{t+1})\eta_b \\ \rho^{t+1} = \rho^t + (\lambda^t - \lambda^{t+1})\eta_\rho \end{cases}$$

where η denotes the vector $A^{-1}\mathbf{c}$. As mentioned earlier, a change in the groups will occur as soon as one of the α_i will meet one of the boundaries, *i.e.* when $\alpha_i = 0$ or $\alpha_i = 1$ for $i \in \mathcal{E}$. This gives two change conditions:

$$\lambda_{out(r)}^{t+1} = \frac{-\alpha_i^t}{\eta_i} + \lambda^t \quad \text{and} \quad \lambda_{out(\ell)}^{t+1} = \frac{1-\alpha_i^t}{\eta_i} + \lambda^t$$

Points in \mathcal{L} and \mathcal{R} and Detection of $in(\ell)$ and $in(r)$. Events $in(\ell)$ and $in(r)$ are detected using their values of $g(x_i)$. Indeed, one condition to remain in \mathcal{R} is to keep $g(x_i) > 0$ and $g(x_i) < 0$ to stay in \mathcal{L} . Retrieving λ^{t+1} for which one of these conditions is violated for each point requires to write $g^{t+1}(x_i)$ depending on λ^{t+1} . For a point moving inside \mathcal{E} , the particularity is that $g^{t+1}(x_i)$ becomes nul. Defining $h(x) = G(i, \cdot)\eta_\alpha + \eta_b - \eta_\rho$ leads to

$$\begin{aligned} g^{t+1}(x_i) &= G(i, \cdot)\delta_\alpha + \delta_b y_i - \delta_\rho + g^t(x_i) \\ &= (\lambda^{t+1} - \lambda^t)(G(i, \cdot)\eta_\alpha + \eta_b - \eta_\rho) + g^t(x_i) \\ &= (\lambda^{t+1} - \lambda^t)h^t(x_i) + g^t(x_i) = 0 \end{aligned}$$

and thus

$$\lambda_{in(\ell)}^{t+1} \frac{-g^t(x_i)}{h^t(x_i)} + \lambda^t \quad i \in \mathcal{L} \quad \text{and} \quad \lambda_{in(r)}^{t+1} \frac{-g^t(x_i)}{h^t(x_i)} + \lambda^t \quad i \in \mathcal{R}$$

Note that it may happen that several points reach \mathcal{E} at the same time. Even though this does not change equations, it is a relevant remark for implementation. Indeed, if a point is missed, the path is left and the missed point will not be selected afterward.

Regularization Path Algorithm. At each step we look for the smallest $\lambda^{t+1} > \lambda^t$ among $\{\lambda_{in(\ell)}^{t+1}, \lambda_{in(r)}^{t+1}, \lambda_{out(\ell)}^{t+1}, \lambda_{out(r)}^{t+1}\}$. We update the α^{t+1} according to the chosen λ^{t+1} and then the groups. The process is stopped when $\lambda = m$.

3 Regularization Path for the ν -SVR

Assuming m training points $\{(x_i, y_i) \in \mathcal{X} \times \mathbb{R}\}$, the ν -SVR algorithm optimizes an ϵ -insensitive cost ($L(y, f(x)) = \max(0, |y - f(x)| - \epsilon)$) and allows the automatic computation of the ϵ -tube [4]. Its primal formulation is:

$$\begin{cases} \min_{f, b, \epsilon, \xi, \xi^*} \frac{\lambda}{2} \|f\|^2 + \nu\epsilon + \sum_{i=1}^m \xi_i + \xi_i^* & s.t. \\ -\epsilon - \xi_i \leq y_i - f(x_i) \leq \epsilon + \xi_i^*, \quad \xi_i \geq 0, \quad \xi_i^* \geq 0, \quad \forall i \in [1, \dots, m] \text{ and } \epsilon \geq 0 \end{cases}$$

where λ is the regularization parameter. According to this formulation, the parameter ν varies in the interval $[0, m]$.

3.1 The Double Path

The regression function: $f(x) = \frac{1}{\lambda} \sum_{i=1}^m (\alpha_i^* - \alpha_i)k(x_i, x) + b$ is a solution of the previous problem where $k(\cdot, \cdot)$ is the kernel function and the Lagrange multipliers $\alpha_i^{(*)}$ are solutions of the dual problem [4]. Given fixed values of the regularization parameter λ and of ν , the KKT conditions permit the automatic determination of b and ϵ (see [4]). The quality of the regression depends on the chosen values. The aim of this section is to analyze the evolution of the regression function $f(x)$ according to the variations of λ for a fixed value ν : the λ -path. Conversely, keeping λ fixed at a specified value, the regression function can be studied with respect to ν : the ν -path. Following the original idea of [2] it can be shown that these paths are piecewise linear. The initial regularization path for SVR of Gunter and Zhu was extended to the double path (λ -path and ϵ -path) in [10]. In this paper, we give another formulation of the double path as we compute the ν -path instead of the ϵ -path. To do so, let define the following sets:

$$\begin{cases} \mathcal{L} : y_i - f(x_i) < -\epsilon, \forall i \in \mathcal{L}, \alpha_i = 1, \alpha_i^* = 0 & \text{bounded points} \\ \mathcal{R} : y_i - f(x_i) > \epsilon, \forall i \in \mathcal{R}, \alpha_i = 0, \alpha_i^* = 1 & \text{bounded points} \\ \mathcal{C} : |y_i - f(x_i)| < \epsilon, \forall i \in \mathcal{C}, \alpha_i = 0, \alpha_i^* = 0 & \text{useless points} \\ \mathcal{E}_{\mathcal{L}} : y_i - f(x_i) = -\epsilon, \forall i \in \mathcal{E}_{\mathcal{L}}, 0 \leq \alpha_i \leq 1, \alpha_i^* = 0 & \text{useful points} \\ \mathcal{E}_{\mathcal{R}} : y_i - f(x_i) = \epsilon, \forall i \in \mathcal{E}_{\mathcal{R}}, \alpha_i = 0, 0 \leq \alpha_i^* \leq 1 & \text{useful points} \end{cases}$$

The sets \mathcal{L} and \mathcal{R} contain respectively the points with errors belonging to the left part and right part of the ϵ -tube whereas the points of $\mathcal{E}_{\mathcal{L}}$ and $\mathcal{E}_{\mathcal{R}}$ lie on the left and right elbows (on the tube). The elements of \mathcal{C} are the points in the tube. Compared to the ν -SVM, we remark that there is more groups of points to tract.

3.2 Computation of the λ -Path

We suppose the value of ν is constant. The regression function can be written as: $f(x) = \frac{1}{\lambda} (\sum_{i=1}^m (\alpha_i^* - \alpha_i)k(x_i, x) + \beta_0)$ with $\beta_0 = \lambda b$. Let $f^t(x)$, the solution obtained for λ^t . The corresponding sets are $\mathcal{L}^t, \mathcal{R}^t, \mathcal{C}^t, \mathcal{E}_{\mathcal{L}}^t, \mathcal{E}_{\mathcal{R}}^t$. The solution is not modified as long as the sets are not modified. As previously, the key point is to determine the values of λ for which a point is moved from a set to another. The conditions for the occurring of these events are summarized in table 2. Let write $\lambda f(x) = \lambda f(x) - \lambda^t f^t(x) + \lambda^t f^t(x)$. Hence we have:

$$\lambda f(x) = \sum_{i \in \mathcal{E}_{\mathcal{L}}^t \cup \mathcal{E}_{\mathcal{R}}^t} (\delta \alpha_i^* - \delta \alpha_i) k(x_i, x) + \delta \beta_0 + \lambda^t f^t(x) \tag{3}$$

with $\delta \alpha_i = \alpha_i - \alpha_i^t, \delta \alpha_i^* = \alpha_i^* - \alpha_i^{*t}, \delta \beta_0 = \beta_0 - \beta_0^t$. In the latest relation, the sum is carried only over $\mathcal{E}_{\mathcal{L}}$ and $\mathcal{E}_{\mathcal{R}}$ as the Lagrange parameters corresponding to the other sets are fixed (equal to 0 or 1). For $j \in \mathcal{E}_{\mathcal{L}}^t$, we have: $y_j - f^t(x_j) = -\epsilon^t$. Therefore, for $\lambda^{t+1} < \lambda < \lambda^t$, the following equation holds: $\lambda(y_j + \epsilon) = \sum_{i \in \mathcal{E}_{\mathcal{R}}^t \cup \mathcal{E}_{\mathcal{L}}^t} (\delta \alpha_i^* - \delta \alpha_i) k(x_i, x_j) + \delta \beta_0 + \lambda^t(y_j + \epsilon^t)$. By defining $d = \lambda \epsilon$ and $\delta d = \lambda \epsilon - \lambda^t \epsilon^t$, we get:

$$(\lambda - \lambda^t)y_j = \sum_{i \in \mathcal{E}_{\mathcal{R}}^t} \delta \alpha_i^* k(x_i, x_j) - \sum_{i \in \mathcal{E}_{\mathcal{L}}^t} \delta \alpha_i k(x_i, x_j) + \delta \beta_0 - \delta d, \quad \forall j \in \mathcal{E}_{\mathcal{L}}^t$$

Table 2. Events in the doubly path of ν -SVR. The bold blue color denotes the conditions used to compute the parameters of the model and $r_i^k = y_i - f^k(x_i)$. Some elements of the last column are not specified as they are given in the previous columns.

Step	$in(\ell)$ \mathcal{L}^t to $\mathcal{E}_{\mathcal{L}}^t$	$in(r)$ \mathcal{R}^t to $\mathcal{E}_{\mathcal{R}}^t$	$in(c)$ \mathcal{C}^t to $\mathcal{E}_{\mathcal{L}}^t$ or \mathcal{C}^t to $\mathcal{E}_{\mathcal{R}}^t$	$out(\ell)$ $\mathcal{E}_{\mathcal{L}}^t$ to \mathcal{L}^t	$out(r)$ $\mathcal{E}_{\mathcal{R}}^t$ to \mathcal{R}^t	$out(c)$ $\mathcal{E}_{\mathcal{L}}^t$ to \mathcal{C}^t or $\mathcal{E}_{\mathcal{R}}^t$ to \mathcal{C}^t
t	$i \in \mathcal{L}$ $r_i^t < -\epsilon^t$ $\alpha_i = 1$ $\alpha_i^* = 0$	$i \in \mathcal{R}$ $r_i^t > \epsilon^t$ $\alpha_i = 0$ $\alpha_i^* = 1$	$i \in \mathcal{C}$ $ r_i^t < \epsilon^t$ $\alpha_i = 0$ $\alpha_i^* = 0$	$i \in \mathcal{E}_{\mathcal{L}}$ $\mathbf{r}_i^t = -\epsilon^t$ $0 \leq \alpha_i \leq 1$ $\alpha_i^* = 0$	$i \in \mathcal{E}_{\mathcal{R}}$ $\mathbf{r}_i^t = \epsilon^t$ $\alpha_i = 0$ $0 \leq \alpha_i^* \leq 1$	$i \in \mathcal{E}_{\mathcal{L}}$ or $\mathcal{E}_{\mathcal{R}}$
$t+1$	$i \in \mathcal{E}_{\mathcal{L}}$ $\mathbf{r}_i^{t+1} = -\epsilon^{t+1}$ $0 \leq \alpha_i \leq 1$ $\alpha_i^* = 0$	$i \in \mathcal{E}_{\mathcal{R}}$ $\mathbf{r}_i^{t+1} = \epsilon^{t+1}$ $\alpha_i = 0$ $0 \leq \alpha_i^* \leq 1$	$i \in \mathcal{E}_{\mathcal{L}}$ or $i \in \mathcal{E}_{\mathcal{R}}$ $\mathbf{r}_i^{t+1} = -\epsilon^{t+1}$ or $\mathbf{r}_i^{t+1} = \epsilon^{t+1}$ $0 \leq \alpha_i \leq 1, \alpha_i = 0$ $\alpha_i^* = 0, 0 \leq \alpha_i^* \leq 1$	$i \in \mathcal{L}$ $\mathbf{r}_i^{t+1} < -\epsilon^{t+1}$ $\alpha_i = 1$ $\alpha_i^* = 0$	$i \in \mathcal{R}$ $\mathbf{r}_i^{t+1} > \epsilon^{t+1}$ $\alpha_i = 0$ $\alpha_i^* = 1$	$i \in \mathcal{C}$ $ r_i^{t+1} < \epsilon^{t+1}$ $\alpha_i = 0$ $\alpha_i^* = 0$

Similarly, for $j \in \mathcal{E}_{\mathcal{R}}^t$, one can establish:

$$(\lambda - \lambda^t)y_j = \sum_{i \in \mathcal{E}_{\mathcal{R}}^t} \delta \alpha_i^* k(x_i, x_j) - \sum_{i \in \mathcal{E}_{\mathcal{L}}^t} \delta \alpha_i k(x_i, x_j) + \delta \beta_0 + \delta d, \quad \forall j \in \mathcal{E}_{\mathcal{R}}^t$$

Using the constraints $(\alpha^* - \alpha)^\top \mathbf{1} = 0$ (which leads to $(\delta \alpha^* - \delta \alpha)^\top \mathbf{1} = 0$) and $(\alpha^* + \alpha)^\top \mathbf{1} \leq \nu$ (hence $(\delta \alpha^* + \delta \alpha)^\top \mathbf{1} \leq 0$) of the dual problem, we obtain the linear system $A\delta = (\lambda - \lambda^t)\mathbf{c}$ where:

$$A = \begin{bmatrix} -K(\mathcal{E}_{\mathcal{L}}, \mathcal{E}_{\mathcal{L}}) & K(\mathcal{E}_{\mathcal{L}}, \mathcal{E}_{\mathcal{R}}) & 1 & -1 \\ -K(\mathcal{E}_{\mathcal{L}}, \mathcal{E}_{\mathcal{R}})^\top & K(\mathcal{E}_{\mathcal{R}}, \mathcal{E}_{\mathcal{R}}) & 1 & 1 \\ -\mathbf{1}^\top & \mathbf{1}^\top & 0 & 0 \\ \mathbf{1}^\top & \mathbf{1}^\top & 0 & 0 \end{bmatrix}, \delta = \begin{bmatrix} \delta \alpha \\ \delta \alpha^* \\ \delta \beta_0 \\ \delta d \end{bmatrix} \in \mathbb{R}^{|\mathcal{E}_{\mathcal{L}}| + |\mathcal{E}_{\mathcal{R}}| + 2}, \mathbf{c} = \begin{bmatrix} \mathbf{y}_{\mathcal{E}_{\mathcal{L}}} \\ \mathbf{y}_{\mathcal{E}_{\mathcal{R}}} \\ 0 \\ 0 \end{bmatrix}$$

and K the gram matrix. Let $\eta = A^{-1}\mathbf{c}$, the parameters are given by:

$$\alpha^{t+1} = \alpha^t + (\lambda - \lambda^t)\eta_\alpha, \quad \alpha^{*t+1} = \alpha^{*t} + (\lambda - \lambda^t)\eta_{\alpha^*} \tag{4}$$

$$\beta_0^{t+1} = \beta_0^t + (\lambda - \lambda^t)\eta_{\beta_0} \tag{5}$$

$$d^{t+1} = d^t + (\lambda - \lambda^t)\eta_d \tag{6}$$

Points in $\mathcal{E}_{\mathcal{L}}$ or $\mathcal{E}_{\mathcal{R}}$ and Detection of $out(\ell)$, $out(r)$ and $out(c)$. These events occur when the parameters α and α^* hints their boundaries 0 or 1 (see table 2). According to 4, we get:

$$\lambda_{out(\ell)}^{t+1} = \frac{1 - \alpha_i^t}{\eta_{\alpha_i}} + \lambda^t, \quad i \in \mathcal{E}_{\mathcal{L}}; \quad \lambda_{out(r)}^{t+1} = \frac{1 - \alpha_i^{*t}}{\eta_{\alpha_i^*}} + \lambda^t, \quad i \in \mathcal{E}_{\mathcal{R}}$$

$$\lambda_{out(c)}^{t+1} = \left\{ \frac{-\alpha_i^t}{\eta_{\alpha_i}} + \lambda^t, \quad i \in \mathcal{E}_{\mathcal{L}} \right\} \cup \left\{ \frac{-\alpha_i^{*t}}{\eta_{\alpha_i^*}} + \lambda^t, \quad i \in \mathcal{E}_{\mathcal{R}} \right\}$$

Points in \mathcal{L} , \mathcal{R} , \mathcal{C} and Detection of $in(\ell)$, $in(r)$ and $in(c)$. By substituting equations 4-5 in 3 and after some algebras, we get: $f(x) = \frac{\lambda^t}{\lambda} [f^t(x) - h^t(x)] +$

$h^t(x)$ with $h^t(x) = \sum_{i \in \mathcal{E}_{\mathcal{R}}^t} \delta\alpha_i^* k(x_i, x_j) - \sum_{i \in \mathcal{E}_{\mathcal{L}}^t} \delta\alpha_i k(x_i, x_j) + \delta\beta_0$. Using [6], the latest relation and table [2], the values of λ associated to these events are:

$$\lambda_{in(\ell)}^{t+1} = \frac{\lambda^t(f^t(x_i) - h^t(x_i) - \epsilon^t + \eta_d)}{y_i - h^t(x_i) + \eta_d}, \quad \lambda_{in(r)}^{t+1} = \frac{\lambda^t(f^t(x_i) - h^t(x_i) + \epsilon^t - \eta_d)}{y_i - h^t(x_i) - \eta_d}$$

$$\lambda_{in(c)}^{t+1} = \left\{ \lambda_{in(\ell)}^{t+1}, \text{ evaluated for } i \in \mathcal{C} \right\} \cup \left\{ \lambda_{in(r)}^{t+1}, \text{ evaluated for } i \in \mathcal{C} \right\}$$

λ -Path Algorithm. The next value λ^{t+1} is the largest value of λ less than λ^t . The difficult part of the method is to find an initial configuration of λ such as each elbow $\mathcal{E}_{\mathcal{R}}$ and $\mathcal{E}_{\mathcal{L}}$ contains at least one point. In [2], the authors suggest to choose $\lambda = \infty$ and to find the corresponding b . From this initial point, the value of λ is decreased in order to find two points in the elbows. Thus the algorithm is runned until one elbow becomes empty or the value of λ becomes small.

3.3 Computation of the ν -Path

In this case, the parameter λ is fixed and we examine the effect of ν on the regression solution. The proposed approach is closely similar to the derivation of the λ -path. Let the parameter ν^t corresponding to the solution $f^t(x)$. Let $\nu^{t+1} < \nu < \nu^t$ such as the sets obtained at the step t are not modified. As λ is constant, from [3], we obtain: $\lambda(f(x) - f^t(x)) = \sum_{i \in \mathcal{E}_{\mathcal{R}}^t \cup \mathcal{E}_{\mathcal{L}}^t} (\delta\alpha_i^* - \delta\alpha_i)k(x_i, x) + \delta\beta_0$. According to the conditions verified by the points belonging to $\mathcal{E}_{\mathcal{L}}$ and $\mathcal{E}_{\mathcal{R}}$ (respectively $y_i - f(x_i) = -\epsilon$ and $y_i - f(x_i) = \epsilon$) we obtain the set of equations:

$$\sum_{i \in \mathcal{E}_{\mathcal{R}}^t} \delta\alpha_i^* k(x_i, x_j) - \sum_{i \in \mathcal{E}_{\mathcal{L}}^t} \delta\alpha_i k(x_i, x_j) + \delta\beta_0 - \delta d = 0 \quad \forall j \in \mathcal{E}_{\mathcal{L}}^t \quad (7)$$

$$\sum_{i \in \mathcal{E}_{\mathcal{R}}^t} \delta\alpha_i^* k(x_i, x_j) - \sum_{i \in \mathcal{E}_{\mathcal{L}}^t} \delta\alpha_i k(x_i, x_j) + \delta\beta_0 + \delta d = 0 \quad \forall j \in \mathcal{E}_{\mathcal{R}}^t \quad (8)$$

with $\delta d = \lambda(\epsilon - \epsilon^t)$. Also here, the condition $(\alpha^* - \alpha)^T \mathbf{1} = 0$ (which leads to $(\delta\alpha^* - \delta\alpha)^T \mathbf{1} = 0$) holds whereas the inequality $(\alpha^* + \alpha)^T \mathbf{1} \leq \nu$ yields $(\delta\alpha^* + \delta\alpha)^T \mathbf{1} \leq \nu - \nu^t$. Grouping all these equations, we obtain a linear system: $A\delta = (\nu - \nu^t)\mathbf{c}$ with $\mathbf{c} = [\mathbf{0} \ \mathbf{0} \ \mathbf{0} \ \mathbf{1}]^T$. The values of ν corresponding to the events are computed by applying the same mechanism as previously. The events $in(\ell)$, $in(r)$ and $in(c)$ can be monitored by using the relation $f(x) = f^t(x) + \frac{\nu - \nu^t}{\lambda} h^t(x)$ derived from the updating equations of the parameters with respect to ν . The ν -path is similar to the λ -path. Here the initialization is very easy as we can choose $\nu \approx 0$. In this case, all the points are inside the tube or in the margin and the initial solution is very sparse. As the elbows are initialized, the algorithm proceeds from this point. We have presented a doubly path algorithm. The question arises how to switch from a path to the other. As at each step of a path all the parameters are available, the switching is easily carried.

4 Stopping on the Path Using Leave One Out

We want to find a stopping criteria along the path. To do so the idea is to compute together with the regularization path a sequence of estimates of the

generalization error to stop when this sequence reaches a minimum. Among all possible estimations of the generalization error, the leave-one-out seems to be the one advocated by practitioners. The major drawback of the leave-one-out estimate is the time required to compute it. Solutions have been proposed to overcome this deficiency. [8] propose to use an approximation easily available called the GCV (Generalized Cross Validation). Others [11] propose to take advantage of efficient implementation of the SVM with warm-start (starting from the current solution as an *a priori* on the next solution) to derive acceptable procedures [12]. Following this idea, we show next how to integrate those estimators in the algorithm and we point out that this method is much cheaper than an external LOO method. The leave-one-out error is defined as the mean error done for the removed points. We also compute a second leave-one-out estimation to have an idea of the variance of the solution:

$$LOO1_{error} = \frac{1}{n} \sum_i 1 - \text{sign}(\hat{y}_i y_i) \quad LOO2_{error} = \frac{1}{n} \sum_i \max(0, \rho - \hat{y}_i y_i)$$

This second formula is very helpful to detect over-fitting. Indeed, outliers will be very penalized. The leave-one-out error rates are estimated at each step. Since no point from \mathcal{R} participate to the solution, they would necessarily have a zero error if once removed from the training set. Hence we only need to compute the LOO errors of each point t of \mathcal{E} and \mathcal{L} . The solution S_{-t} is close to the current solution S given along the path. Thus we obtain S_{-t} from S with Simple- ν -SVM warm start. The cost of the computation of the LOO at each step is $(|\mathcal{E}| + |\mathcal{L}|)$ update steps. If the generalization error is significantly better for some range of parameters λ , we expect to see it through the LOO error rates. Hence we monitor this value to detect when it starts to increase significantly. Then we can stop learning a go back to S_λ which has given the lowest LOO errors. Doing so, we avoid to compute the part of the path for which most of the points are bounded support vectors. Indeed those solution contradict the SVM goal: sparseness.

For the ν -SVR algorithm, the generalization ability is evaluated using the following LOO error (computed by exploiting also a warm start procedure) $LOO = \frac{1}{m} \sum_{i=1, i \neq k}^m |y_i - f_k(x_i)|$ with $f_k(x)$, the solution obtained with the point k out of the training set. This implies $|\mathcal{E}_{\mathcal{L}}| + |\mathcal{E}_{\mathcal{R}}| + |\mathcal{L}| + |\mathcal{R}|$ updates at each step of the path.

4.1 Experimental Results

We have conducted experiments on artificial data-sets in order to illustrate how the LOO estimation can be a criteria to stop learning on the path before attaining non sparse solutions. Figure 1 give example of results on the mixture data-set, with an *rbf* kernel. Each LOO estimate provide useful information. The first one (based on the counting of the errors) gives a good approximation of the generalization error. The second one, based on the output value of the SVM represents the variance of the solution and we look for a low variance solution. From a practical point of view, determining the correct moment to stop

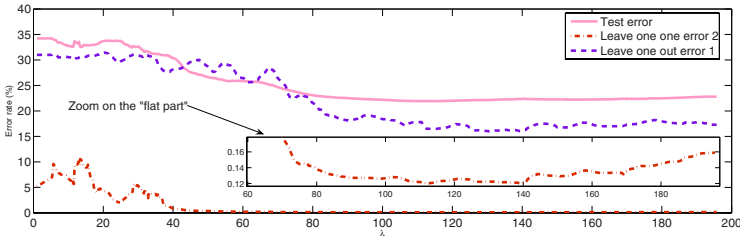


Fig. 1. Illustration on the mixture data-set of the LOO error rate evolution according to λ , reported with the test error

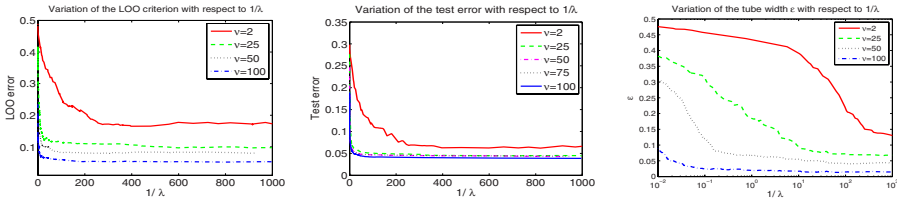


Fig. 2. Illustration of the λ -path for different values of ν . The plots show (from the left to the right) the LOO error, the test error and the width of the tube.

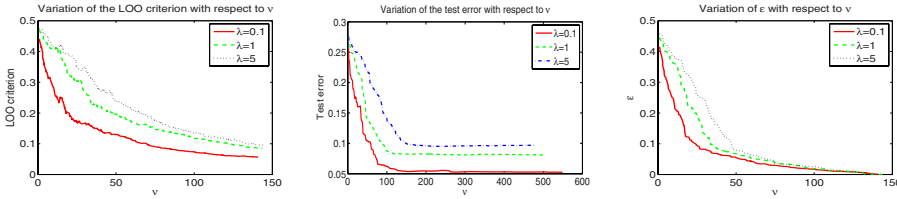


Fig. 3. Illustration of the ν -path for different values of λ

requires some heuristic and using a smoothed curved of the LOO error is useful. Our heuristic consists in choosing to stop when the smoothed LOO2 error has not been decreasing for a period lasting as long as it take for λ to grow of 10. Then we choose backward the λ corresponding to the minimum achieved by smoothed LOO1. The test of the ν -SVR algorithm is realized on a toy problem which consists to approximate the nonlinear function $y = \sin(\exp(3 * x))$. A gaussian kernel with bandwidth 0.1 was used. The results obtained for the λ -path are depicted on figure 2. When λ decreases, the LOO error decreases quickly so the algorithm can be stopped earlier. The same remark holds for the width of the tube. For the small values of ν , as the initial solution is sparse, the LOO computation is very fast and stopping earlier the algorithm yields a sparse solution. There is no need to explore the overall path. The illustration of the ν -path for different values of λ is displayed on figure 3. As ν decreases,

the tube vanishes and the LOO error decreases. However, it remains to find a suitable strategy to explore the hyper-parametric space (λ, ν) . It seems that an interesting methodology consists to run the λ -path for small values of ν . The "optimal" value of λ is plugged in the ν -path algorithm to find a final solution. This strategy has to be confirmed by intensive simulations.

5 Conclusion

This paper gathers the ν -SVM and the ν -SVR regularization paths. The motivation for using the ν derivations of the standard methods SVM and SVR is that they provide formulations with more intuitive and bounded hyper-parameters. We give details on the derivations of the regularization paths and we show how to include an estimation of the generalization error within the path so that learning can be stopped when the best solution is attained, without computing useless solutions. Applying this to the SVR is more tricky since we need to search on a surface instead of a line and we are currently developing this part.

Acknowledgments. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

References

1. Hastie, T., Rosset, S., Tibshirani, R., Zhu, J.: The entire regularization path for the support vector machine. *Journal of Machine Learning Research* **5** (2004) 1391–1415
2. Gunter, L., Zhu, J.: Computing the solution path for the regularized support vector regression. In: NIPS. (2005)
3. Chen, P.H., Lin, C.J., Schölkopf, B.: A tutorial on ν -support vector machines. *Applied Stochastic Models in Business and Industry* **21** (2005) 111–136
4. Schölkopf, B., Smola, A.: *Learning with Kernels*. MIT Press (2001)
5. Argyriou, A., Hauser, R., Micchelli, C.A., Ponti, M.: A dc-programming algorithm for kernel selection. *ICML* (2006)
6. Micchelli, C.A., Pontil, M.: Learning the kernel function via regularization. *Journal of Machine Learning Research* **6** (2005) 1099–1125
7. Bach, F., Heckerman, D., Horvitz, E.: On the path to an ideal ROC curve: Considering cost asymmetry in learning classifiers. In Cowell, R.G., Ghahramani, Z., eds.: *AISTATS, Society for Artificial Intelligence and Statistics* (2005) 9–16
8. Wahba, G. In: *Support Vector Machines, Reproducing Kernel Hilbert spaces and the randomized GACV*. B. Scholkopf and C. Burges and A. Smola edn. MIT Press (1999) 69–88
9. Vishwanathan, S.V.N., Smola, A.J., Murty, M.N.: SimpleSVM. *Proceedings of the Twentieth International Conference on Machine Learning* (2003)
10. Wang, G., Yeung, D.Y., Lochofsky, F.: Two-dimensional solution path for support vector regression. In: *Proc. of the 23rd International Conference on Machine Learning, ICML*. (2006)

11. Lee, J.H., Lin, C.J.: Automatic model selection for support vector machines. Technical report, Dept. of Computer Science and Information Engineering, National Taiwan University (2000)
12. Lee, M., Keerthi, S., Ong, C.J., DeCoste, D.: An efficient method for computing leave-one-out error in support vector machines with gaussian kernels. *Neural Networks, IEEE Transactions* **15** (2004) 750–757

A Fast and Accurate Progressive Algorithm for Training Transductive SVMs

Lei Wang¹, Huading Jia^{2,3}, and Shixin Sun¹

¹ School of Computer Science & Engineering, University of Electronic Science & Technology of China, Chengdu, 610054, P.R. China

² Institute of Image & Graphics, Sichuan University, Chengdu, 610064, P.R. China

³ School of Economics Information Engineering, Southwest University of Finance & Economics, Chengdu, 610074, P.R. China

{dr.wangl@163.com, jiahd@swfue.edu.cn, sxsun@uestc.edu.cn}

Abstract. This paper develops a fast and accurate algorithm for training transductive SVMs classifiers, which utilizes the classification information of unlabeled data in a progressive way. For improving the generalization accuracy further, we employ three important criteria to enhance the algorithm, i.e. confidence evaluation, suppression of labeled data, stopping with stabilization. Experimental results on several real world datasets confirm the effectiveness of these criteria and show that the new algorithm can reach to comparable accuracy as several state-of-the-art approaches for training transductive SVMs in much less training time.

1 Introduction

In the scenarios of labeled data is scarce and expensive while unlabeled data is easily available and cheap, transductive learning is more efficient than inductive one [4,9]. Transductive support vector machines (TSVMs) is a promising approach for improving the generalization accuracy of SVMs, which forces both labeled and unlabeled data far away from the decision boundary simultaneously with maximum margin.

First practical implementation of TSVMs appeared in [4], known as JT SVM, which used a label-switching-retraining procedure to generate the final classifier. However, one of deficiencies of JT SVM was that it often suffered from local minima [10]. To avoid this problem, several algorithms were proposed recently in global optimization framework. The promising LDS (low density separation) algorithm [2] adopted a density-sensitive “connectivity kernel” to force the decision boundary to cross low-density regions according to cluster assumption. The DA (deterministic annealing) algorithm [8] used the “deterministic annealing strategy” for the similar purpose. Unfortunately, both algorithms suffered from high computational costs.

Most recently, a cheaper progressive algorithm called PTSVM was given in [3], which had showed better accuracy than standard inductive SVMs. However, PTSVM is sensitive to the inconsistent samples and adopts a user-defined stopping condition, both aspects degrade the generalization ability of it.

In this paper, we develop a fast and accurate training algorithm for TSVMs based on the works in [3]. Three criteria are proposed to improve the generalization accuracy of the progress process, i.e. confidence evaluation, suppression of labeled data, stopping with stabilization. The experimental results on a wide range of datasets show that our algorithm can achieve approximately the same test accuracy as several state-of-the-art approaches such as LDS and DA, while runs at least one order of magnitude faster than them.

This remaining of this paper is arranged as follows: In Section 2 we describe main principle of TSVMs and progressive learning. In Section 3, we propose three important criteria for the benefit of generalization accuracy and give the proposed algorithm. The experimental comparison is done in Section 4 and conclusions are given in Section 5.

2 The Review of Transductive SVMs

We consider here the problem of binary classification. The training set consists of n labeled samples $\mathbf{L} = \{(\mathbf{x}_i, y_i)_{i=1}^n \mid \mathbf{x}_i \in \mathbb{R}^d, y = \pm 1\}$ and m unlabeled samples $\mathbf{U} = \{(\mathbf{x}_j)_{j=n+1}^{n+m}\}$. According to the principle of transductive learning [4,9], the TSVMs solves the following optimization problem (in nonlinear case).

$$\begin{aligned} \min_{\mathbf{w}, b, \xi, \xi^*} & \left\{ \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^n \xi_i + C^* \sum_{j=n+1}^{n+m} \xi_j^* \right\} \tag{1} \\ \text{s.t.} & \begin{cases} \forall_{i=1}^n : & y_i [\mathbf{w}^T \phi(\mathbf{x}_i) + b] \geq 1 - \xi_i, & \xi_i > 0 \\ \forall_{j=n+1}^{n+m} : & |\mathbf{w}^T \phi(\mathbf{x}_j) + b| \geq 1 - \xi_j^*, & \xi_j^* > 0 \end{cases} \end{aligned}$$

Where ξ_i and ξ_j^* are the slacks for labeled sample \mathbf{x}_i and unlabeled sample \mathbf{x}_j respectively; C and C^* are the corresponding penalty parameters.

With the Lagrange theory, we obtain the dual representation of (1):

$$\begin{aligned} \min_{\alpha, \alpha^*} & \left\{ \frac{\alpha^T \mathbf{K}^{ll} \alpha + 2\alpha^T \mathbf{K}^{lu} \alpha^* + \alpha^{*T} \mathbf{K}^{uu} \alpha^*}{2} - \alpha \cdot \mathbf{1} - \alpha^* \cdot \mathbf{1} \right\} \tag{2} \\ \text{s.t.} & \begin{cases} \forall_{i=1}^n : & 0 \leq \alpha_i \leq C \\ \forall_{j=n+1}^{n+m} : & 0 \leq \alpha_j^* \leq C^* \\ \sum_{i=1}^n y_i \alpha_i + \sum_{j=n+1}^{n+m} y_j^* \alpha_j^* = 0 \end{cases} \end{aligned}$$

Where α and α^* are Lagrange multipliers for labeled and unlabeled samples respectively. $\mathbf{K}^{ll}, \mathbf{K}^{uu}$ and \mathbf{K}^{lu} are corresponding kernel matrices for labeled samples, unlabeled samples and their joint.

Most TSVMs algorithms try to optimize (1) or (2) by utilizing all unlabeled samples simultaneously. However, a more efficient way is to only make use of the most informative ones. That is to say, in each iteration, only few unlabeled samples that potentially benefit the decision hyperplane most are utilized to train the TSVMs. We call this strategy as “progressive learning”.

The PTSVM algorithm proposed by Chi et al. [3] follows the “progressive learning” strategy. During the k -th iteration of it, only Ψ_k^+ and Ψ_k^- unlabeled samples closest to each margin hyperplanes are labeled (called “transductive samples”). Then the new training set $\mathbf{X}^{(k+1)}$ is updated as the combination of \mathbf{L} and all the transductive samples, as follows:

$$\mathbf{X}^{(k+1)} = \mathbf{L} \cup \sum_{i=1}^k (\Psi_i^+ \cup \Psi_i^-) \quad (3)$$

After that, the TSVMs classifier relearns on $\mathbf{X}^{(k+1)}$ by solving optimization (2) with an increased penalty parameter $C^{*(k+1)}$ for transductive samples.

Generally, two issues should be considered carefully in the progressive learning: (1) how to suppress the influence of inconsistent samples. (2) how to control the algorithm to converge with an optimal solution. Unfortunately, PTSVM in [3] doesn’t settle these issues very well.

For the first issue, PTSVM is sensitive to the inconsistent samples. The inconsistent samples refer to the transductive samples that have been wrongly labeled before according to predictions of current classifier. Since the learned classifiers in the early phase of progress process are unstable, the transductive samples is unavoidable. These samples are always more close to the decision hyperplane and cause more misclassification loss than the rest samples. Thus, there is a strong trend of preventing they being classified into the opposite class since the TSVMs try to minimize the objective in (1). So, the generalization accuracy is degraded for the disturbance of them.

For the second issue, PTSVM adopts a user-defined value to directly control the stopping condition. Consequently, the algorithm may stop with a non-optimal solution and more iterations are needed to reduce test errors further.

In the following section, we present three important criteria to address above issues more effectively.

3 The Proposed Progressive Algorithm

We first give three criteria for improving the generalization accuracy of progressive learning in this section, then present our new progressive algorithm.

3.1 Confidence Evaluation

Assigning small training weights for noise (or outlier) samples is an efficient strategy to reduce their negative influences and to enhance the robustness in inductive learning [6,7]. Similarly, we treat transductive samples discriminatively and set small weights for all potential inconsistent samples in the progressive learning in order to enhance the robustness. Here, we employ the confidence evaluation approach to assign weights for transductive samples. High confidence factor of one sample means high training weight and the sample is more likely to be not a inconsistent one.

Let c_j^* denote the confidence factor for transductive sample \mathbf{x}_j . Like the way in [7], we consider the effects of confidence factors by substituting $c_j^* \cdot \xi_j^*$ for the

slack ξ_j^* in the objective of (1). By the Lagrange theory, we deduce the following dual problem of (1) with the confidence factors.

$$\begin{aligned} \min_{\alpha, \alpha^*} & \left\{ \frac{\alpha^T \mathbf{K}^{ll} \alpha + 2\alpha^T \mathbf{K}^{lu} \alpha^* + \alpha^{*T} \mathbf{K}^{uu} \alpha^*}{2} - \alpha \cdot \mathbf{1} - \alpha^* \cdot \mathbf{1} \right\} \quad (4) \\ \text{s.t.} & \begin{cases} \forall_{i=1}^n : & 0 \leq \alpha_i \leq C \\ \forall_{j=n+1}^{n+p} : & 0 \leq \alpha_j^* \leq c_j^* C^* \\ \sum_{i=1}^n y_i \alpha_i + \sum_{j=n+1}^{n+p} y_j^* \alpha_j^* = 0 \end{cases} \end{aligned}$$

Different with dual problem in (2), we use $p(p < m)$ transductive samples here instead of m unlabeled samples since our algorithm is a progressive one. If the multipliers α, α^* are fixed, the final output function $f(\mathbf{x})$ of TSVMs are

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i y_i k(\mathbf{x}, \mathbf{x}_i) + \sum_{j=n+1}^{n+p} \alpha_j^* y_j^* k(\mathbf{x}, \mathbf{x}_j) + b \quad (5)$$

It's difficult to assign proper confidence factors for transductive samples since there is no further information we can rely on. Fortunately, under the assumption that the decision hyperplanes of TSVMs change smoothly within the iterations, we can utilize the outputs of previous classifier to make a good estimation of confidence factors, as follows.

$$c^{*(k)}(\mathbf{x}_j) = \begin{cases} \min \left\{ 1, |f^{(k)}(\mathbf{x}_j)|^q \right\} & \mathbf{x}_j \in TS^{(k)} \\ 0 & \mathbf{x}_j \notin TS^{(k)} \end{cases} \quad (6)$$

Where $f^{(k)}(\cdot)$ is the output function of the k -th iteration, $c^{*(k)}(\mathbf{x}_j)$ is the confidence factor for transductive sample \mathbf{x}_j , $TS^{(k)}$ denotes current set of transductive samples, and q is an user-defined constant, such as $q = 2$.

Obviously, transductive samples with $|f^{(k)}(\mathbf{x}_j)| < 1$ will have small confidence factors. Since inconsistent samples always lie in margin region in feature space, they have small confidence factors so that the misclassification loss caused by them is reduced significantly. Thus, the hyperplane of TSVMs needn't fit inconsistent samples very much and is prone to reach to the optimal one.

Hence, the confidence-based method in (4) is robust to inconsistent samples for training TSVMs so that better generalization ability can be obtained by suppression the disturbances of them. In this way, we call the method in this section as our first criterion for enhancing the performance of progressive learning, or "confidence evaluation" criterion.

3.2 Tuning the Penalty Parameters

The choice of penalty parameters C and C^* is very crucial for training TSVMs. In our progressive algorithm, we initially set a small value $C^{*(0)}$ for transductive samples and then increase it exponentially to C , as that in JTSVM [4].

$$C^{*(k)} = \begin{cases} \min \{ C^{* \max}, 2 \cdot C^{*(k-1)} \} & k \geq 1 \\ C^{*(0)} & k = 0 \end{cases} \quad (7)$$

Where $C^{*(0)} = 10^{-4} \cdot C$ and $C^{*\max} = C$. Moreover, since original labeled training samples can't well represent the underlying distribution of the unlabeled ones in many cases, the classifier can be fined by decreasing their influences in the latter iterations of the progressive learning when the decision hyperplane is relatively stable. Hence, we adopt a smooth way to reduce the penalty parameters of the labeled training samples in our progressive algorithm, as follows.

$$C^{(k)} = \left(1 - \exp\left(-\frac{C}{10 \cdot C^{*(k)}}\right) \right) \cdot C \tag{8}$$

Thus, the original labeled samples will have small impacts on the classifiers in the latter iterations, so that the accuracy can be fined by paying more attentions to the distribution of unlabeled ones. We call this method as ‘‘suppression of labeled samples’’, which is the second criterion proposed in this paper.

3.3 Stopping Conditions

We use an adaptive convergence strategy in our progressive TSVMs algorithm and stop it only when the following two convergence conditions are satisfied.

- (1) The algorithm is in a ‘‘pseudo-stabilization’’ status.
- (2) The solution is stable for successive two iterations.

The first condition means that the whole unlabeled data can be labeled and separated by current classifier with high possibility of correctness. We will discuss the ‘‘pseudo-stabilization’’ status in detail in Section 3.4.

For the second condition, we use the difference between the normal of current decision hyperplane and that of previous one to evaluate the stability of solution.

$$\mu^{(k)} = \|\mathbf{w}^{(k)} - \mathbf{w}^{(k-1)}\| \tag{9}$$

Where $\mathbf{w}^{(k)} = \sum_{i=1}^n \alpha_i^{(k)} \phi(\mathbf{x}_i) + \sum_{j=n+1}^{n+p^{(k)}} \alpha_j^{*(k)} \phi(\mathbf{x}_j)$ and $d^{(k)}$ is the total number of transductive samples in the k -th iteration.

We say that the solution of TSVMs is stable at the k -th iteration only when the difference $\mu^{(k)}$ is less than a user-defined threshold ϑ , such as $\vartheta = 10^{-3}$.

Obviously, the two convergence conditions will force the progressive TSVMs algorithm to stop at the optimal solution, and we call them the third criterion proposed in this paper, or ‘‘stopping with stabilization’’ criterion.

3.4 Selection of Transductive Samples

In our progressive TSVMs algorithm, we use a similar way for selecting transductive samples as that in [3]. Let N_{sv}^+ and N_{sv}^- denote the number of support vectors for the two classes and $y_j^{*(k)}$ be predicted label of unlabeled sample \mathbf{x}_j in the k -th iteration, we select $2A$ unlabeled samples with small values of $|y_j^{*(k)} f^{(k)}(\mathbf{x}_j) - 1|$ for each class to compose the transductive candidate set Ψ_k^+ and Ψ_k^- , where $A = \min\{N_{sv}^+, N_{sv}^-\}$. We ensure that at least one unlabeled sample in the margin

region is selected into Ψ_k^+ or Ψ_k^- . In the situation of none of such sample is selected, we judge that the algorithm is in a “pseudo-stabilization” status, since current TSVMs classifier seems to separate all the unlabeled samples correctly.

Then, we trim Ψ_k^+ and Ψ_k^- by only keeping the samples with smaller absolute outputs than the averages so that the most informative samples are kept.

$$\Psi_k^+ = \left\{ \mathbf{x}_i \mid \mathbf{x}_i \in \Psi_k^+, |f(\mathbf{x}_i)| \leq |D_k^+|, \text{ where } D_k^+ = \frac{1}{2A} \sum_{s=1}^{2A} f(\mathbf{x}_s), \forall \mathbf{x}_s \in \Psi_k^+ \right\} \quad (10)$$

$$\Psi_k^- = \left\{ \mathbf{x}_j \mid \mathbf{x}_j \in \Psi_k^-, |f(\mathbf{x}_j)| \leq |D_k^-|, \text{ where } D_k^- = \frac{1}{2A} \sum_{t=1}^{2A} f(\mathbf{x}_t), \forall \mathbf{x}_t \in \Psi_k^- \right\} \quad (11)$$

3.5 The Algorithm

The proposed confidence-based progressive TSVMs algorithm is as follows.

Algorithm 1. *cPTSVM*

Input: $\mathbf{L} = \{(\mathbf{x}_i, y_i)_{i=1}^n\}$ and $\mathbf{U} = \{(\mathbf{x}_j)_{j=n+1}^{n+m}\}$.

(a): Learn initial classifier f^0 with inductive SVMs algorithm on \mathbf{L} . Let $k = 0$.

(b): $k = k + 1$. Repeat steps (b1-b5) until the stopping conditions in Section 3.3 are satisfied.

(b1): Select and trim positive and negative transductive candidate sets Ψ_k^+ and Ψ_k^- according to (10) and (11).

(b2): Prepare training set $\mathbf{X}^{(k)}$ according to (3).

(b3): Evaluate confidence factors for transductive samples by (6).

(b4): Calculate $C^{(k)}$ and $C^{*(k)}$ according to (7) and (8).

(b5): Train TSVMs classifier $f^{(k)}$ on $\mathbf{X}^{(k)}$ by solving (4).

(c): Label all the rest unlabeled samples.

So, the cPTSVM algorithm yields a TSVMs classifier and the predicted labels for all unlabeled samples. Moreover, the algorithm can be extended to multiclass problem by the “One-Against-All” strategy.

4 Experiments and Results

We present experimental studies for the performance of cPTSVM algorithm on a collection of real word datasets listed in Table 1. The *diabetes*, *kr-vs-kp*, *DNA*, *waveform*, *adult* datasets are from UCI repository¹ while the *coil20*, *text*, *uspst* datasets are classical semi-supervised learning tasks from [2]. All datasets are normalized before model selection, learning and classification.

The first experiment tries to show the superior accuracy of cPTSVM algorithm when training TSVMs classifiers, and the test errors are compared with

¹ Available at <http://www.ics.uci.edu/~mllearn/MLRepository.html>

JTSVM[4] and PTSVM[3]. For each dataset, we draw a training set from the whole data with bootstrap technique [1] and use the rest as test set. In such way, total 10 training sets and 10 test sets are obtained for each dataset. To simulate the semi-supervised learning, in each training set, we select 100 samples as the labeled ones and the rest as the unlabeled ones.

Table 1. Datasets used in our experiments

Datasets	#classes	#attributes	#samples
diabetes	2	8	768
kr-vs-kp	2	36	3,196
DNA	3	180	3,186
waveform	3	21	5,000
adult20*	2	14	6,032
coil20	20	1,024	1,440
text	2	7,511	1,946
uspst	10	256	2,007

* the *adult20* dataset use 20% of the original *adult*.

The model selection is done by finite grid searching method on a small separate validation set. The values considered for the RBF kernel parameter σ are $2^{-3}, 2^{-2}, 2^{-1}, 2^0, 2^1, 2^2, 2^3, 2^4$ while those for the penalty parameter C are $10^{-1}, 10^0, 10^1, 10^2, 10^3$. Then, the optimal values of σ and C are selected by 5-fold cross-validation based on above combination. For fairness, we also set the same values of σ and C for JTSVM and PTSVM. Table 2 reports the average test errors and the standard deviation of these algorithms on above datasets.

Table 2. Comparing test errors ($\times 100\%$) of JTSVM, PTSVM and cPTSVM algorithms

Datasets	JTSVM	PTSVM	cPTSVM
diabetes	28.93 \pm 1.60	25.74 \pm 1.03	24.83 \pm 0.91
kr-vs-kp	11.52 \pm 1.05	9.15 \pm 0.71	8.06 \pm 0.54
DNA	9.16 \pm 0.76	7.69 \pm 0.58	7.41 \pm 0.49
waveform	13.92 \pm 1.37	11.40 \pm 0.90	10.76 \pm 0.74
adult20	19.85 \pm 2.14	18.84 \pm 1.21	17.38 \pm 1.20
coil20	18.70 \pm 0.89	7.45 \pm 0.65	4.77 \pm 0.44
text	7.44 \pm 0.82	6.32 \pm 0.62	5.51 \pm 0.57
uspst	23.21 \pm 1.43	18.47 \pm 0.85	16.08 \pm 0.69

* the **bold** represent the **best** test errors

Obviously, cPTSVM algorithm performs the best on all datasets. It achieves much lower test errors than the PTSVM and JTSVM. The best two cases are the *coil20* and *uspst*, where PTSVM achieves the average errors of 7.45% and 18.47% while cPTSVM decreases them significantly to 4.77% and 16.08%, respectively. Experimental results also indicate that the cPTSVM is more robust than the other two algorithms since it obtains smaller standard deviations.

The proposed three criteria of cPTSVM is responsible for its superior accuracy. To see clearly how they take effects, we investigate test errors of cPTSVM during iterations on the *uspst* dataset. The cases of 100 and 500 labeled training samples are considered. Training parameters are set with $\sigma = 4, C = 100$ by 5-fold cross-validation. Fig.1 reports the main experimental results.

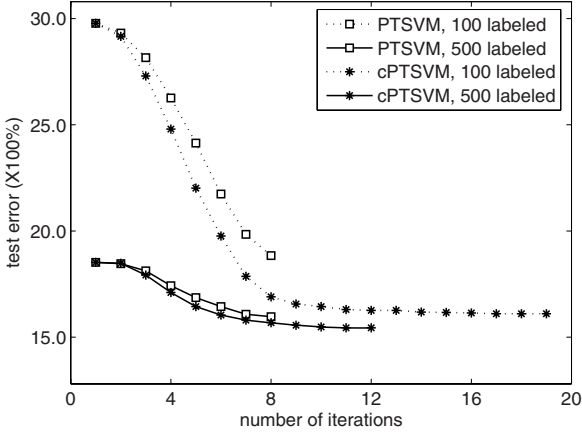


Fig. 1. Comparison of convergence curves of test error on the *uspst* dataset

The plots show that the cPTSVM seems to reduce the test error more effectively than PTSVM, especially when the number of labeled training samples is few. Obviously, we can attribute the superior ability of cPTSVM for reducing the error in the early iterations to its robustness to inconsistent samples (see the first criterion). Besides, the extra iterations (than PTSVM) in Fig.1 are very meaningful for reducing the test errors further. This is because the cPTSVM utilizes better stopping conditions (see the third criterion).

We also compare the test errors and the training time of cPTSVM with those of two state-of-the-art semi-supervised methods(LDS and DA) on *text*, *coil20* and *uspst* datasets. We use the same method to obtain 10 training sets and 10 test sets for each classification tasks as the first experiment. The training parameters are set as follows: $\sigma = 8, 32, 4$ for *coil20*, *text*, *uspst* respectively; $C = 100$ for all of them; the softening parameter of LDS is set with $\rho = 8, 8, 4$ respectively. Table 3 reports the comparison results.

Table 3. Comparison of DA, LDS, cPTSVM in terms of test errors and training time

	test errors ($\times 100\%$)			training time (sec.)		
	DA	LDS	cPTSVM	DA	LDS	cPTSVM
coil20	4.14	4.86	4.77	673.4	227.4	19.7
text	6.86	5.13	6.51	2,607.6	1,291.0	51.7
uspst	14.92	15.79	16.08	859.2	352.5	16.6

* the **bold** represent the **best** test errors

The experimental results show that the cPTSVM achieves approximately the same error rates on all the three datasets as the LDS algorithm and the DA algorithm, but runs at least one order of magnitude faster than them. Two reasons can account for the superior speed of it. First, the main costs of cPTSVM is the minimization of (4), which can be solved by simply modifying popular SVMs solvers, such as SVM^{light} [5]. So, the actual runtime for solving cPTSVM scales quadratically with the total training points. In contrast, the times for solving DA and LDS are in a cubic relation [2,8]. Second, in each iteration of cPTSVM algorithm, only a portion of unlabeled samples are used to train the TSVMs classifiers rather than the total of them.

5 Conclusions

We have developed a fast and accurate algorithm for training TSVMs classifiers in semi-supervised learning context. It employs a similar process of utilizing the unlabeled data progressively [3] and uses three important criteria to improve the generalization accuracy further. Experimental results confirm the effectiveness of the proposed criteria and show that the new algorithm has approximately the same test accuracy as state-of-the-art approaches such as LDS and DA while the needed training time is much less.

References

1. Breiman, L.: Bagging Predictors. *Mach. Learn.*, **24** (1996) 123–140
2. Chapelle, O., Zien, A.: Semi-supervised Classification by Low Density Separation. *Proc. 10th Int. Workshop on AI&Statistics*, (2005) 57–64
3. Chi, M.M., Bruzzone, L.: A Novel Transductive SVM for Semi-supervised Classification of Remote Sensing Images. *11th SPIE Int. Symposium on Remote Sensing*, (2005) 153–164
4. Joachims, T.: Transductive Inference for Text Classification using Support Vector Machine. *16th Int. Conf. on Mach. Learn.*, (1999) 200–209
5. Joachims T.: Making Large-scale SVM Learning Practical. In: Scholkopf B. et al.(ed.), *Advances in Kernel Methods: Support Vector Learning* (MIT Press, Cambridge, 1999) 169–184
6. Li, M.K., Sethi, I.K.: Confidence-based Classifier Design. *Pattern Recognition*, **39** (2006) 1230–1240
7. Liu, H., Huang, S.T.: Fuzzy Transductive Support Vector Machines for Hypertext Classification. *Int. J. of Uncertainty, Fuzziness and Knowledge-based Systems*, **12** (2004) 21–36
8. Sindhwani, V., Keerthi S., Chapelle O.: Deterministic Annealing for Semi-supervised Kernel Machines. *Proc. 23rd Int. Conf. on Mach. Learn.*, (2006) 841–848
9. Vapnik, V.: *The Nature of Statistical Learning Theory*. 2nd edn. Springer-Verlag, Berlin (1995)
10. Wang, Y., Huang, S.T.: Training TSVM with the Proper Number of Positive Samples. *Pattern Recognition Letters*, **26** (2005) 2187–2194

Fast Support Vector Data Description Using K-Means Clustering

Pyo Jae Kim, Hyung Jin Chang, Dong Sung Song, and Jin Young Choi

School of Electrical Engineering and Computer Science,
Automation and Systems Research Institute, Seoul Nat'l University,
San 56-1 Shillim-dong, Kwanak-ku Seoul 151-744, Korea
{[pjkim](mailto:pykim@iccl.snu.ac.kr),[hjchang](mailto:hjchang@iccl.snu.ac.kr),[dssong](mailto:dssong@iccl.snu.ac.kr),[jychoi](mailto:jychoi@iccl.snu.ac.kr)}@neuro.snu.ac.kr
<http://iccl.snu.ac.kr>

Abstract. Support Vector Data Description (SVDD) has a limitation for dealing with a large data set in which computational load drastically increases as training data size becomes large. To handle this problem, we propose a new fast SVDD method using K-means clustering method. Our method uses *divide-and-conquer* strategy; trains each decomposed sub-problems to get support vectors and retrains with the support vectors to find a global data description of a whole target class. The proposed method has a similar result to the original SVDD and reduces computational cost. Through experiments, we show efficiency of our method.

1 Introduction

Classification tasks, such as detecting a fault in the machine, image retrieval and surveillance system, need different approaches from general pattern classification or regression methods. These problems are called *one-class problems*, and the solutions have to aim for describing boundaries of each classes. Based on these boundaries, the classifier makes a decision whether the testing object is in a target class or not.

SVDD [1] is a well-known one class classification algorithm. This method finds a boundary of a target class in data space by assuming a hypersphere which has minimum volume enclosing almost all target class objects in feature space. SVDD uses *support vectors* to describe the boundary of target class as Support Vector Machine(SVM) [2] does. *Support vectors* are found by solving convex quadratic programming (QP).

Although QP has an advantage of avoiding a local minimum problem, it requires severe computation as the number of training objects increase. This problem has been known as ‘Scale problem’. There is much literature for reducing computational cost. Those suggested studies can be roughly categorized into two groups.

One group is mainly focused on solving QP quickly. Chunking [3], decomposition [4] algorithms and Sequential Minimal Optimization (SMO) [5] are in this group. The chunking and decomposition methods break down a large problem into a series of small problems. By discarding non *support vector* data in small

problems, the algorithms are able to train fast and reduce memory space. SMO is a more advanced version of decomposition method.

The other group is based on the *divide-and-conquer* strategy dealing with large scale data problems [6], [7], [8]. The large data set is decomposed into several simple sub-problems similar to decomposition methods, but each sub-problem is solved by its own local expert. Thus such approaches need additional decision rules to combine each local expert's decisions or to decide one of local experts to new test object. This decision rules cause additional computational cost. Parallel mixture of SVMs [6] and Bayesian Committee Support Vector Machine (BC-SVM) [7] can be the examples of this group.

In this paper, we propose a new method for a scale problem of SVDD. Our method is not aimed to solve QP quickly, but based on the *divide-and-conquer* strategy. We decompose a large data set into a series of small sub-data groups using K-means clustering algorithm [9]. Each small sub-problem is solved by its own local expert with SVDD, but it needs not an additional decision rule unlike the existing approaches. Instead of the decision rule, our method retrains using only *support vectors* which are found by each local experts. Through training decomposed data and retraining *support vectors* only, a global solution can be found faster than the original SVDD. Because our approach does not rely on the performance of a QP solver, it has a chance to be improved when faster QP solver (such as SMO) is used.

The proposed method is applied to data sets which are various in size and shape. Comparing with the original SVDD, we show our method has similar data description results with less computational load.

2 KMSVDD: Support Vector Data Description Using K-Means Clustering

In this section, we give detail explanation of the proposed method. We present the basic theory of SVDD and then introduce a model of KMSVDD.

2.1 Basic Theory of Support Vector Data Description

SVDD is similar to the hyperplane approach of Schölkopf et al. [10] which estimates decision plane to separate the target objects with maximal margin. However, instead of using a hyperplane, SVDD uses a minimum hypersphere to find an enclosed boundary containing almost all target objects [1]. Assume a hypersphere with center \mathbf{a} and radius R . So the cost function is defined as below:

$$F(R, \mathbf{a}) = R^2 + C \sum_i \xi_i, \quad (1)$$

where ξ_i are slack variables $\xi_i \geq 0$, and the parameter C controls the trade-off between the volume and the errors [1]. Also (1) should be minimized under the following constraints:

$$\|\mathbf{x}_i - \mathbf{a}\|^2 \leq R^2 + \xi_i, \quad \xi_i \geq 0, \quad \forall i. \quad (2)$$

Constraints can be incorporated into the cost function by using Lagrange multipliers:

$$\begin{aligned}
 L(R, \mathbf{a}, \alpha_i, \gamma_i, \xi_i) &= R^2 + C \sum_i \xi_i & (3) \\
 &- \sum_i \alpha_i \{R^2 + \xi_i - (\|\mathbf{x}_i\|^2 - 2\mathbf{a} \cdot \mathbf{x}_i + \|\mathbf{a}\|^2)\} - \sum_i \gamma_i \xi_i, \\
 &\alpha_i \geq 0, \quad \gamma_i \geq 0.
 \end{aligned}$$

L should be minimized with respect to R , \mathbf{a} , ξ_i and maximized with respect to α_i, γ_i . After setting partial derivatives of L to zero, solve each equation with a QP solver. Those objects \mathbf{x}_i with $0 < \alpha_i \leq C$ are called *support vectors*, and used to describe a boundary description. Especially, objects \mathbf{x}_i with $\alpha_i = C$ are called *outliers*.

To test whether an object \mathbf{z} is within the hypersphere, the distance to the center of the hypersphere is used:

$$\|\mathbf{z} - \mathbf{a}\|^2 = \mathbf{K}(\mathbf{z}, \mathbf{z}) - 2 \sum_i \alpha_i \mathbf{K}(\mathbf{z}, \mathbf{x}_i) + \sum_{i,j} \alpha_i \alpha_j \mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) \leq R^2, \quad (4)$$

where \mathbf{K} is a kernel function to solve non-separable problems. When the distance is equal to or smaller than the radius R , the test object \mathbf{z} is accepted.

2.2 KMSVDD

KMSVDD, SVDD algorithm using K-means clustering, is described in Fig. 1. It can be summarized as the following three steps.

Step 1. Decompose a data set using K-means clustering

Assume k center points and partition a training data set into k sub-problems using K-means clustering algorithm.

Because a retraining process exists after this step, clustering quality is not a key factor of the proposed method’s performance. Other clustering methods such as LVQ can be used, but we select K-means clustering algorithm because it has good performance with less computational load.

Step 2. Get local data descriptions using SVDD

Execute individual training on each k sub-problems.

As a result of training, k sub-problems’ local descriptions and *support vectors* representing each local descriptions are obtained. We newly define a *working set* composed of these *support vectors* and will use it for finding a global data description in the next **Step 3**. Because many *inliers* which are within the hypersphere do not affect making a final data description [2], so they can be ignored and exclude from *working set*.

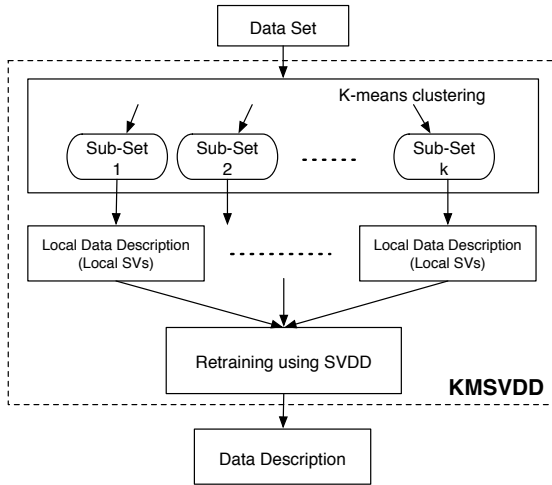


Fig. 1. The scheme of KMSVDD algorithm

Step 3. Retrain with working set using SVDD

To get a global description of a training data set, retrain only with *support vectors* which are obtained by **Step 2**.

In general, k sub-problems' local descriptions have lots of confronting sides with neighbors. It means that some *support vectors* are overlapped and generated uselessly so the description results are far from the result of using only the original SVDD. Through retraining process, useless *support vectors* (truly *inliers* of a target class) will be removed. Therefore the retrained result can be similar to that of the original SVDD.

The proposed algorithm adds two additional steps to the original SVDD: clustering and retraining step. So two learning parameters which user has to determine are added. One is the cluster number k and the other is the kernel parameter used in SVDD learning of **Step 2, 3**.

Learning process of the proposed algorithm can be visualized as Fig. 2. Although, the number of training data is same, training k sub-problems is much faster than using whole data at a time.

2.3 Computational Complexity of KMSVDD

A general QP solver runs in $O(N^3)$ time, where N is the size of training data. We are not improving the performance of a QP solver, but cutting down the size of training data for each QP solver. So the complexity of KMSVDD with k clusters is

$$O(kN) + kO((N/k)^3) + O((\alpha k)^3) \approx kO((N/k)^3), \quad (5)$$

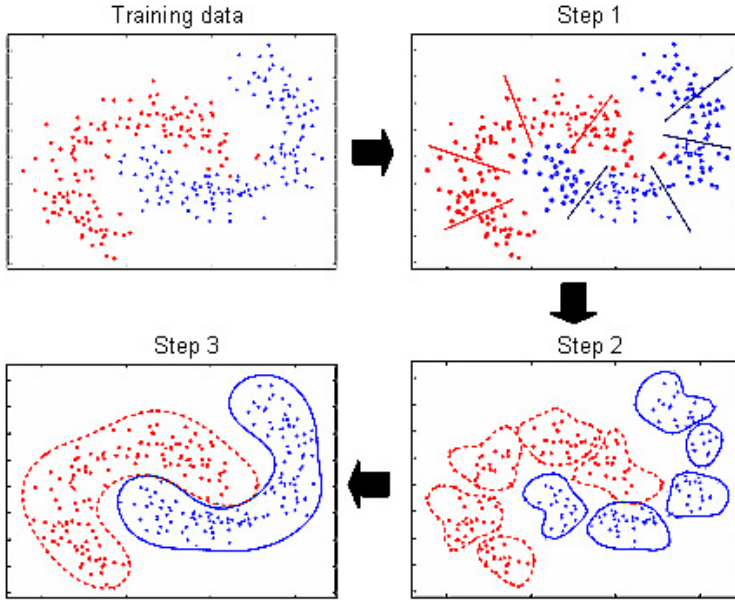


Fig. 2. Visualization of KMSVDD learning process. KMSVDD has three steps and gets a global data description finally.

where k is the cluster number and α is the average number of *support vector* for each sub-problems' description, so the size of *working set* is αk . The first term $O(kN)$ is for the complexity of K-means clustering algorithm. The second is for the complexity of local SVDD QP solver, provided the training data N is equally partitioned into k sub-problems. The third term $O((\alpha k)^3)$ is for retraining process. Generally under the sparse solution conditions, the size of *working set* αk is smaller than the initial size of training data N . So the influence of the new complexity increments caused by K-means clustering and retraining is negligibly small in contrast to the effect of decomposition. Therefore KMSVDD assures the reduction of computational cost for large N .

3 Experimental Results

Following three types of simulations are performed for comparison in training time and accuracy between KMSVDD and SVDD. Tax's data description toolbox [11] is used and simulations are run on a PC with a 3.0 GHz Intel Pentium IV processor and 1G RAM. We use quadratic program solver (*quadprog*) provided by MATLAB in all experiments so the training time difference caused by the QP solver doesn't occur. Optimal parameter values of Gaussian kernel are founded using 10-fold cross validation method.

First, we observe the effect of cluster number k on the training time of KMSVDD. We measure the training time by changing the cluster number using two artificial data sets (Banana shape, Donut shape [1]).

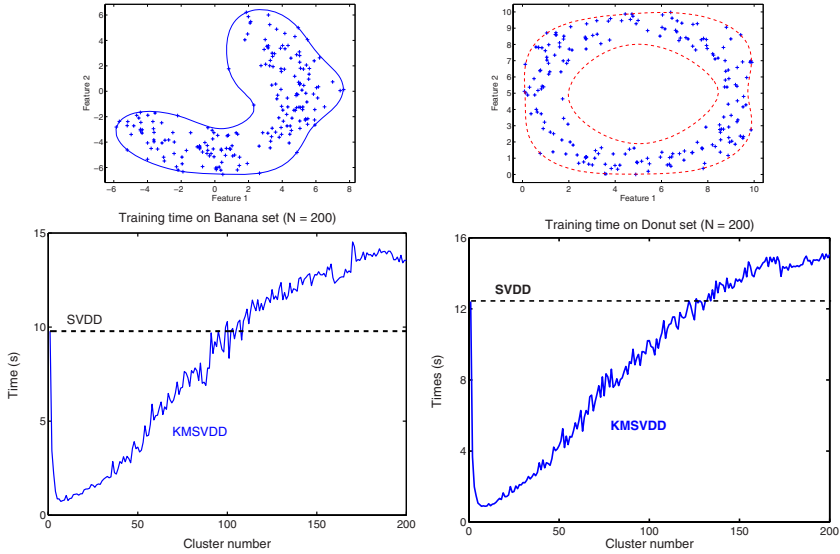


Fig. 3. Training time of KMSVDD on various cluster numbers using one class Banana and Donut shape set (data size $N = 200$). Upper figures show data distributions used in experiments. The training time varies convexly as k increases and the optimal cluster number k is different depending on the data distribution.

As shown in Fig. 3, the training time of KMSVDD varies depending on the cluster number k . If k increases, which means diminishing size of each sub-problem, the computational load of local descriptions would be reduced, but the total number of *support vectors*, which are used in retraining, would increase. The increment of *working set*'s size makes the retraining process's computational cost expensive. To some extent, the training time is reduced as k increases, but when k is over a certain limit, the training time increases on the contrary. Therefore the training time varies convexly as k increases and an adequate selection of k is necessary.

We also perform similar experiment on the Banana shape data set having various data size. The change of an optimal cluster number k , which has minimum training time, for each data size is observed. The result is depicted in Fig. 4. From the results, we can find that the optimal cluster number k is different depending on the data distribution and training data size.

Second, we compare the training time of KMSVDD with the original SVDD on various size Banana shape data set. As in Table 1, the training time of SVDD grows

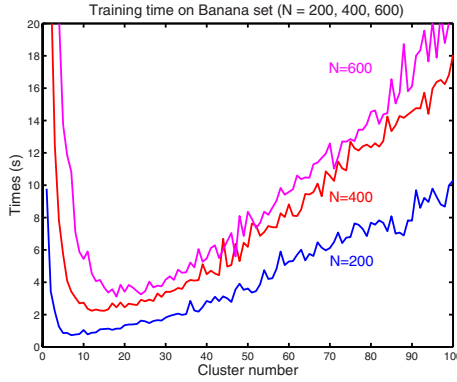


Fig. 4. Training time of KMSVDD using various size Banana shape data sets ($N = 200, 400, 600$ for each case). The optimal cluster number k varies depending on the data size.

drastically, whereas KMSVDD’s training time grows relatively slowly. KMSVDD can reduce the training time even though the cluster number k is not optimal.

Table 1. Training time on two classes Banana shape data sets using the original SVDD and KMSVDD with various cluster numbers k . As the data size increases, the training time of SVDD grows drastically whereas the proposed method’s training time grows relatively slowly.

	k	Data Size				
		200	600	1000	1200	1400
SVDD	1	16.8	488.4	3311.4	6527.4	11626.0
KMSVDD	2	4.4	167.7	1298.1	2157.4	4132.3
	10	0.8	12.0	112.7	327.9	458.3
	20	0.9	4.1	12.7	26.6	102.4
	30	1.3	3.1	6.2	13.9	11.9
	40	1.7	3.7	6.5	8.7	14.6
	50	2.0	4.8	6.9	10.9	13.5

Finally, we test training accuracy of both methods using real data set. UCI Iris and Wine data sets [12] are used as real data. Each data set has three classes respectively. For each class, one-against-all method [13] is used and test objects’ false positive and false negative numbers are measured as the training accuracy. As we can see in Table 2, KMSVDD can result in similar accuracy with less computational load.

Also two methods results similar data descriptions as shown in Fig. 5(left), (right) though KMSVDD obtains k local data descriptions for each class before the retraining process as in Fig. 5(center) if proper learning parameters are used.

Table 2. Classification error of one-against-all classifiers for each class using UCI data set (Iris, Wine). The number of false positive and false negative objects are measured with 10-fold cross-validation method (values in brackets are standard deviations). KMSVDD shows comparable results with the original SVDD.

Class index	Iris ($N = 150$)			Wine ($N = 178$)		
	1	2	3	1	2	3
SVDD	2.3(0.48)	7.5(0.97)	6.1(1.28)	5.5(1.78)	11.7(2.11)	7.0(1.56)
KMSVDD	0.8(0.94)	8.2(1.57)	5.6(2.17)	4.7(2.86)	10.8(1.39)	6.1(2.57)

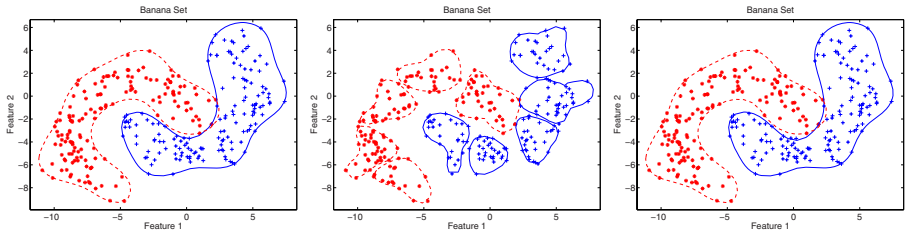


Fig. 5. Data descriptions using two classes Banana shape set $N = 200$. Data description by the original SVDD (*left*). Local (*center*) and Global (*right*) data descriptions using KMSVDD. Both methods have similar data description results.

4 Conclusion

In this paper, we present a fast one-class classifier, called KMSVDD, by combining the K-means clustering method with SVDD. SVDD has a problem that as the number of training data increases, training time also increases drastically. Most of training time is consumed calculating QP problems. To reduce the training time, we decompose a large training data set into small and compact sub-problems using K-means clustering method. Training each small k sub-problem to get *support vectors* is much faster than using a whole data at a time. Sub-group training makes many local descriptions, so an additional decision rule is needed to classify (or to get a global description). To solve this problem we re-train data only with *support vectors* of every local descriptions. The simulation results of KMSVDD show remarkable training time reduction comparing with SVDD, and the training performance is comparable.

As for further research, we will deal with as following issue.

Through the mathematical analysis on the relationship between the cluster number and training time of the proposed algorithm, we will give the guideline to determine proper (or optimal) cluster number k .

The effectiveness of the proposed algorithm may be further investigated by using large scale real data sets including the UCI Forest database [12] or MNIST handwritten digit database [13].

Acknowledgments. This work was supported by the Brain Korea 21 Project and ASRI (Automation and Systems Research Institute).

References

1. Tax, D.M.J.: Support Vector Data Description. *Machine Learning* **54** (2004) 45–46
2. Vapnik, V.N.: *Statistical Learning Theory*. Wiley, New York (1998)
3. Joachims, T.: Making Large-scale SVM Learning Practical. *Advances in Kernel Methods - Support Vector Learning*. MIT Press, Cambridge, MA (1999) 169–184
4. Osuna, E., Freund, R., Girosi, F.: Training Support Vector Machines. *Conf. Computer Vision and Rattern Recognition* (1997) 130–136
5. Platt, J.C.: Fast Training of Support Vector Machines using Sequential Minimal Optimization. *Advances in Kernel Methods - Support Vector Learning*. MIT Press, Cambridge, MA (1999) 41–65
6. Collobert, R., Bengio, S., Bengio, Y.: A Parallel Mixture of SVMs for Very Large Scale Problems. *Neural Computation* **14** (5) (2002) 143–160
7. Schwaighofer, A., Tresp, A.: The Bayesian Committee Support Vector Machine. *Proc. Int. Conf. Artificial Neural Networks* (2001) 411–417
8. Rida, A., Labbi, A., Pellegrini, C.: Local Experts Combination through Density Decomposition. *Proc. Seventh Int. Workshop AI and Statistics* (1999)
9. Duda, R.O., Hart, P.E., Stork, D.G.: *Pattern Classification*. Wiley-Interscience (2001)
10. Schölkopf, B., Burges, C., Vapnik, V.N.: Extracting Support Data for a Given Task. *Proc. of First Int. Conf. Knowledge Discovery and Data Mining* (1995) 252–257
11. Tax, D.M.J.: DDtools, the Data Description Toolbox for Matlab. http://www-ict.ewi.tudelft.nl/~ddavidt/dd_tools.html (2006)
12. Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J.: UCI Repository of machine learning databases. <http://www.ics.uci.edu/~mllearn/MLRepository.html>, Univ. of California, Irvine, Dep. of Information and Computer Science (1998)
13. Bottou, L., Cortes, C., Denker, J., Drucker, H., Guyou, I., Jackel, L., LeCun, Y., Muller, U., Sackinger, E., Simard, P., Vapnik, V.: Comparison of Classifier Methods: A Case of Study in Handwriting Digit Recognition. *Proc. Int. Conf. Pattern Recognition* (1994) 77–87

A Kernel-Based Two-Stage One-Class Support Vector Machines Algorithm

Chi-Yuan Yeh and Shie-Jue Lee

Department of Electrical Engineering, National Sun Yat-Sen University,
Kaohsiung 804, Taiwan
d943010011@student.nsysu.edu.tw, leesj@mail.ee.nsysu.edu.tw

Abstract. One-class SVM is a kernel-based method that utilizes the kernel trick for data clustering. However it is only able to detect one cluster of non-convex shape in the input space. In this study, we propose an iterative two-stage one-class SVM to cluster data into several groups. In the first stage, one-class SVM is used to find an optimal weight vector for each cluster in the feature space, while in the second stage the weight vector is used to refine the clustering result. A mechanism is provided to control the optimal hyperplane to work against outliers. Experimental results have shown that our method compares favorably with other kernel based clustering algorithms, such as KKM and KFCM on several synthetic data sets and UCI real data sets.

1 Introduction

Partitioning-based clustering algorithms that attempt to find a user-specified number of clusters are well-known in unsupervised data clustering. However, they are not suitable to discover clusters with non-convex or overlapped shapes. Recently, a number of kernel-based learning methods have been proposed for data clustering. They utilize the kernel trick of doing all calculations in the low-dimensional input space to perform inner products in the new high-dimensional feature space where the data are expected to be more separable.

Schölkopf *et al.* integrated kernel-based methods with K-means [1]. The kernel K-means clustering algorithm (KKM) transforms implicitly the input data into a high-dimensional feature space via a nonlinear mapping function, and then performs K-means iterative procedure in the feature space. The results indicate that KKM outperforms conventional K-means due to the nonlinear mapping. Girolami [2] and Dhillon *et al.* [3], [4] also proposed alternate versions of the KKM to improve the quality and speed of KKM. Zhang and Chen integrated kernel-based methods with fuzzy C-means. The experimental results indicate that Kernel Fuzzy C-means (KFCM) has better performance in the ring dataset due to the RBF kernel function and is robust to noise and outliers due to the fuzzy membership degrees [5], [6]. Mizutani and Miyamoto integrated possibilistic approaches with KFCM (KPCM), and used the entropy-based objective function instead of the standard objective function. The experimental results

indicate that KPCM outperforms KFCM [7]. Kim *et al.* indicated that kernel-based methods were about 15% more accurate than conventional methods [8]. Du *et al.* indicated that kernel-based clustering algorithms can extract arbitrarily shaped clusters but are not robust to noise data, while nonlinear distance methods are robust to noise data but cannot extract arbitrarily shaped clusters [9].

Tax and Duin [10] and Ben-Hur *et al.* [11] introduced a kernel-based method, called support vector data description (SVDD), or Support Vector Clustering (SVC); it computes the smallest sphere in feature space enclosing the image of the input data. This method not only can extract arbitrary geometrical shapes of clusters, but also are robust against noise and outliers. Müller *et al.* [12] and Schölkopf *et al.* [1], [13] proposed another method called one-class Support Vector Machines (one-class SVM) to find a hyperplane which has maximum margin to separate the data set from the origin in the feature space. However, they are only able to detect one cluster in the feature space. We are interested in applying one-class SVM to solving multiple clusters problem. In this study, an iterative strategy integrating two-stage one-class SVM (TSOCS) is proposed to segment multiple clusters with non-convex shapes. In the first stage, one-class SVM is used to find an optimal weight vector for each cluster in the feature space, and in the second stage, the weight vector is used to refine the clustering result. The proposed method is compared with KKM and KFCM which are also partitioning-based clustering algorithms on a few synthetic and real data sets. Experimental results have shown that this method outperforms the other two methods.

2 Background

2.1 One-Class Support Vector Machines

One-class SVM is a kernel-based method; it constructs a classifier only using a set of positive training patterns. If the data set does not contain outliers, one-class SVM find a hyperplane which has maximum margin ($\mathbf{w} \cdot \phi(\mathbf{x}_i) = \rho$) to separate the data set from the origin. The distance between the hyperplane and the origin is $\rho/\|\mathbf{w}\|$. Therefore, the objective function of the optimal hyperplane and inequality constraints are:

$$\begin{aligned} \min \quad & \frac{1}{2} \|\mathbf{w}\|^2 - \rho \\ \text{s.t.} \quad & \mathbf{w} \cdot \phi(\mathbf{x}_i) \geq \rho, \forall \mathbf{x}_i \in X \end{aligned} \quad (1)$$

where $\phi : X \mapsto F$ is a nonlinear mapping from the input space X to the feature space F , \mathbf{w} is a weigh vector in the feature space and, ρ is an offset of the hyperplane.

To allow for the possibility of outliers in the data set, and to make the method more robust, the projection value from an image to the \mathbf{w} needs not be strictly larger than ρ , but the small projection value should be penalized. Therefore,

slack variables ξ , $\xi_i \geq 0, \forall \xi_i$, are introduced to account for the small projection value and the new objective function, and constraints become:

$$\begin{aligned} \min \quad & \frac{1}{2} \|\mathbf{w}\|^2 - \rho + \frac{1}{\nu n} \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \mathbf{w} \cdot \phi(\mathbf{x}_i) \geq \rho - \xi_i, \xi_i \geq 0, \forall \mathbf{x}_i \in X \end{aligned} \tag{2}$$

where $\nu \in (0, 1]$ is a parameter which gives a trade-off between the maximum margin and the errors. With a small ν , penalty on small projection value becomes substantial, thus few outliers should exist and the margin is small. On the other hand, when ν is large, many outliers with small projection value may exist to take advantage of the small penalty and the margin is generally large. In practice, the size of ν determines the number of outliers and support vectors.

In order to solve the constrained optimization problem, a Lagrangian is introduced as follows:

$$L = \frac{1}{2} \|\mathbf{w}\|^2 - \rho + \frac{1}{\nu n} \sum_{i=1}^n \xi_i - \sum_{i=1}^n \alpha_i [(\mathbf{w} \cdot \phi(\mathbf{x}_i)) - \rho + \xi_i] - \sum_{i=1}^n \beta_i \xi_i, \tag{3}$$

where $\alpha_i \geq 0$ and $\beta_i \geq 0$ are Lagrange multipliers. L has to be minimized with respect to \mathbf{w} , ξ and ρ given α and β , and then maximized with respect to α and β . Setting partial derivatives equal to 0 yields the following Karush-Kuhn-Tucker (KKT) conditions:

$$\mathbf{w} = \sum_{i=1}^n \alpha_i \cdot \phi(\mathbf{x}_i) \tag{4}$$

$$\alpha_i = \frac{1}{\nu n} - \beta_i \leq \frac{1}{\nu n} \tag{5}$$

$$\sum_{i=1}^n \alpha_i = 1 \tag{6}$$

In addition, the KKT complementary conditions are:

$$\alpha_i [(\mathbf{w} \cdot \phi(\mathbf{x}_i)) - \rho + \xi_i] = 0 \tag{7}$$

$$\beta_i \xi_i = 0 \tag{8}$$

For a data point \mathbf{x}_i with $\beta_i = 0$, we have $\xi_i > 0$ and $\alpha_i = 1/(\nu n)$. Equation (4) implies $(\mathbf{w} \cdot \phi(\mathbf{x}_i)) < \rho$. In other words, the image $\phi(\mathbf{x}_i)$ lies between the hyperplane and the origin of the unit ball. This is called a bounded support vector (BSV), or is sometimes called an outlier. For a data point \mathbf{x}_i with $\beta_i > 0$ if $\beta_i \neq 1/(\nu n)$, then $\xi_i = 0$ and $0 < \alpha_i < 1/(\nu n)$. Equation (4) implies $(\mathbf{w} \cdot \phi(\mathbf{x}_i)) = \rho$, so $\phi(\mathbf{x}_i)$ lies on the hyperplane in the feature space. Such a point will be referred to as a support vector (SV). If $\beta_i = 1/(\nu n)$, then $\xi_i = 0$ and $\alpha_i = 0$. Equation (4) implies $(\mathbf{w} \cdot \phi(\mathbf{x}_i)) > \rho$, so $\phi(\mathbf{x}_i)$ lies outside the hyperplane in the feature space. Such a point will be referred to as a non-support vector

(non-SV). In addition, the constraint of Equation (6) implies that when $\nu \leq 1/n$ no outliers exist.

Substituting Equations (4), (5) and (6) into Equation (3), we can eliminate the variables \mathbf{w} , ξ_i , ρ , and β_i , turning the Lagrangian into the Wolfe dual form which is a quadratic function in α_i 's. In order to avoid working in the high-dimensional feature space F , one picks a feature space where the dot product can be calculated directly using a kernel function K in the input space, and the Wolfe dual form becomes:

$$\begin{aligned} \min \quad & \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq \frac{1}{\nu n}, \quad \sum_{i=1}^n \alpha_i = 1 \end{aligned} \quad (9)$$

This constrained optimization problem can be solved using a standard QP solver. The decision function of one-class SVM can be written as:

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \phi(\mathbf{x}) - \rho) = \text{sgn}\left(\sum_{i=1}^n \alpha_i K(\mathbf{x}_i, \mathbf{x}) - \sum_{i=1}^n \alpha_i K(\mathbf{x}_i, \mathbf{x}_j)\right) \quad (10)$$

where \mathbf{x}_j is a support vector.

Three commonly used kernel functions are Polynomial kernel, Sigmoid kernel, and RBF kernel. Throughout this paper, the RBF kernel is adopted. That is:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-q\|\mathbf{x}_i - \mathbf{x}_j\|^2) \quad (11)$$

where q is a width parameter of the RBF kernel. This equation implies that each input point is mapped on the octant surface of the unit ball in the high-dimensional feature space.

2.2 Related Work

Since SVC or one-class SVM is only able to detect one cluster in the feature space, different mechanisms were introduced to solve multiple clusters problem. Ben-Hur *et al.* proposed a clustering labelling method to solve this problem [11]. The hypersphere is mapped back to the input space, where it can separate data set into several clusters, after finding support vectors. The advantage of this method is that it does not require the number of clusters as an input. However, it may generate hundreds of clusters with many clusters just containing one data point. In addition, it is very time-consuming in practice. Chiang and Hao proposed a multiple-sphere support vector clustering algorithm based on an online cluster cell growing method [14]. When a pattern belongs to one cluster and lies outside the sphere, it must retrain SVC for this cluster. The disadvantage of this method is that it is very time-consuming. Camastra and Verri proposed a clustering method inspired by the conventional K-means algorithm where each cluster is iteratively refined using a SVC [15]. This method is also called Kernel Grower (KG) because a threshold ρ is introduced to control the number of

cluster members that tend to grow after each iteration. Furthermore, they set the parameter C , which is equivalent to $1/(\nu n)$, equal to 1. Therefore no cluster can contain outliers (BSV). Because of this, the KG method lacks the ability to handle possible outliers. In this study, we use TSOCS based on iterative strategy to cluster data. We modulate the parameter ν to provide a trade-off between the optimal hyperplane and the outliers. We set ν large and use TSOCS to discount the influence of noise or outliers on the hyperplane. In other words, the mechanism can get an optimal weight vector in the first stage. In the second stage, the optimal weight vector is used to refine the clustering result.

3 The Proposed Method

The value of parameter ν provides a trade-off between the distance from the origin to the hyperplane and the slack distances resulted from outliers. Consider the 96 patterns shown in Fig. 1 where four groups of data are marked with different notations as follows: $G_1 : \circ$, $G_2 : *$, $G_3 : \square$, $G_4 : +$

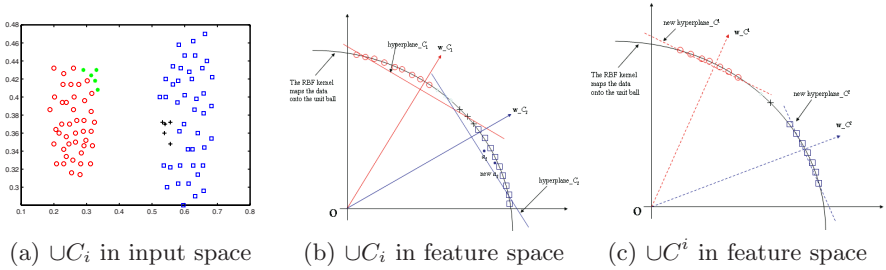


Fig. 1. An example for the analysis of parameter ν

There are two real clusters: $C_1 = \{G_1, G_2\}$, $C_2 = \{G_3, G_4\}$. At some stage, a clustering algorithm partitions this data set as $C_1 = \{G_1, G_4\}$ and $C_2 = \{G_2, G_3\}$. Therefore, data points from G_2 and G_4 are identified in the wrong cluster. Now, suppose a one-class SVM is being trained for C_1 . As mentioned above, the parameter ν determines the number of outliers and support vectors. It is an upper bound on the fraction of outliers and a lower bound on the fraction of SVs. Large values of ν allow for many outliers and SVs to exist as shown in Table 1. In addition, when one-class SVM for C_i is accomplished, the non-SVs of cluster C_i are called C^i . Setting ν from $2/10$ to $9/10$, the C^1 only contains the elements of G_1 and C^2 only contains the elements of G_3 . When we train one-class SVM for C^1 and C^2 , the position of the new hyperplane is located on G_1 and G_3 , respectively. Based on the RBF kernel and Equation (10), we can see that the projection value between the image of a pattern from G_1 and the weight vector of C^1 is generally larger than the projection value between the image of a pattern from G_1 and the weight vector of C^2 , thus this pattern will be assigned to C_1 . The projection value between the image of a pattern from G_4

Table 1. Numbers of non-SVs, SVs or BSVs with various values of ν

ν		9/10	5/10	2/10	1/10	1/48	0.1/48
non-SVs	G_1	4	22	38	40	43	43
	G_4	0	0	0	2	1	1
SVs	G_1	2	3	2	1	2	2
	G_4	0	0	0	1	2	2
BSVs	G_1	37	18	3	2	0	0
	G_4	5	5	5	2	0	0

and the weight vector of C^2 is generally larger than the projection value between the image of a pattern from G_4 and the weight vector of C^1 , thus this pattern will be assigned to C_2 . While setting $\nu \leq 1/48$, C^1 contains the elements from G_1 and G_4 as shown in Table 1. C^2 also contains the elements from G_2 and G_3 . In this case, a pattern from G_1 may be assigned to C_1 or C_2 . As a result, low accuracy is obtained.

When the initial cluster members are randomly drawn from the data set, we may have some patterns belonging to the wrong clusters as G_4 above. In this case, large ν and second one-class SVM will generally set a tighter control for the hyperplane and the problem.

The proposed algorithm uses one-class SVM to compute the optimal weight vector for each cluster. Like partitioning-based clustering algorithms, we assume that the number of clusters, k , is known in advance. We set $\nu > 1/n$ to discount the influence of outliers on the optimal hyperplane in the feature space. When one-class SVM for each cluster C^i is accomplished, each pattern is assigned to the maximum projection value by the following equation:

$$C_i = \{\mathbf{x}_j \in X, \phi(\mathbf{x}_j) \in F \mid i = \arg \max_{i=1, \dots, k} \sum_{\mathbf{x}_l \in C^i} \alpha_l K(\mathbf{x}_j, \mathbf{x}_l)\} \quad (12)$$

where $X = [\mathbf{x}_1, \dots, \mathbf{x}_n]^T \in \mathbb{R}^{n \times d}$ is an unlabelled data set, \mathbf{x}_j represents a d -dimensional pattern, $j = 1, \dots, n$, and C_i are exclusive clusters, $i = 1, \dots, k$. The proposed algorithm consists of the following steps:

Initiation: Given cluster number k and the width of RBF kernel q , initial cluster members are prepared using the input variable with the largest variance. The data set X is divided into k disjoint subsets $XD_i (XD_i \subset X)$ according to the selected input variable, then initial cluster members are drawn randomly from the partition ($C_{i,t} \subset XD_i$). Note that $C_{1,t} \cup C_{2,t} \cup \dots \cup C_{k,t} \subset X, t = 1$.

First-stage: Train a one-class SVM for each cluster $C_{i,t}$ to obtain non-SVs ($C^{i,t}$).

Second-stage: Train a new one-class SVM for each cluster $C^{i,t}$ to obtain new $\alpha_{ij,t}, j = 1, 2, \dots, |C^{i,t}|$.

Assignment: Assign each pattern to its closet cluster according to Equation (12). Note that $C_{1,t} \cup \dots \cup C_{2,t} \cup C_{k,t} = X$. If no cluster changes, exit; otherwise, go to First-stage step with $t = t + 1$.

The whole process of the proposed algorithm is shown in Fig. 2.

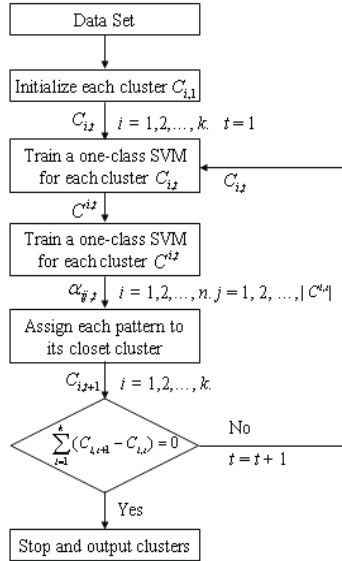


Fig. 2. The flowchart of the TSOCS algorithm

4 Experimental Results

In order to test and compare the performance of various kernel partitioning-based clustering algorithms, two synthetic data sets (Ring Set and Delta Set) and six UCI real data sets [16], i.e., the Wine recognition data set, the Fisher IRIS data set, the Wisconsin’s breast cancer data set, Heart Disease data set, Pima Indians Diabetes data set, and the Thyroid Disease data set. Note that the Wisconsin’s breast cancer data set contains some missing values. We have removed 16 patterns with missing values from the data set; therefore the data set considered has 683 patterns. The parameters used in KFCM are a termination criterion of $r = 0.001$ and a weighting exponent of $m = 2.0$. The parameter used in one-class SVM is $\nu > 1/n$. Each clustering method is applied to each data set for 20 times with different initial cluster members. For the 2-D Ring Data Set, three kernel partitioning-based clustering algorithms achieve 100.00% accuracy due to the RBF kernel function. Fig. 3 (a), (b) and (c) show the results of applying three kernel partitioning-based clustering algorithms on the 2-D Delta Data Set, which also appeared in [15]. Similar to the conventional K-means method, KKM cannot separate this data set well. KFCM outperforms KKM;

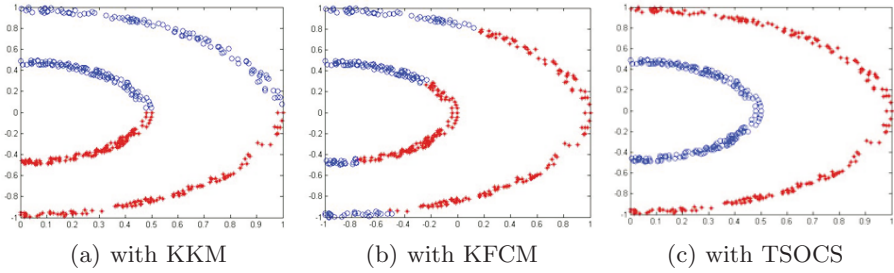


Fig. 3. Results obtained on Delta Data Set

sometimes KFCM can achieve 100.00% accuracy, but the probability is very low (5%). Note that TSOCS can obtain 100.00% accuracy in all 20 trials.

From Table 2, we can see that all three kernel partitioning-based clustering algorithms have the same results in 20 trials on the Wisconsin's breast cancer data set. However, TSOCS slightly outperforms KKM and KFCM. In the best case, TSOCS misclassified 14 patterns in class1 and 7 patterns in class 2. For the Heart Disease data set, TSOCS outperforms the other two methods, and KFCM outperforms KKM. For the Pima Indians Diabetes data set, TSOCS also outperforms the other two methods, and KFCM outperforms KKM. From these results, we can see that these two data sets are hard to separate in the high dimension feature space. For the IRIS data set, TSOCS outperforms KFCM and KKM. In the best case, TSOCS can totally identify class 1 and achieve 100.0% of accuracy for this class, misclassify 3 patterns in class 2, and misclassify 3 patterns in class 3. KKM is affected by initial centers of the clusters, and in the worst case it just obtains 50.67% of accuracy, though most trials can achieve 96.0% of accuracy. The results indicate that the performance of KKM clustering depends on the initial guess of the centers of the clusters. For the Wine recognition data set, TSOCS outperforms the other two methods. KKM outperforms KFCM on the Wine recognition data set. In the best case, TSOCS can separate class 1 and class 3 completely and misclassify 8 patterns in class 2. The best case of KFCM can achieve 96.63% of accuracy, but for most trials, the accuracies are between 56.74% and 80.0%. In addition, KKM and KFCM can identify class 1 completely, but cannot separate class 2 and class 3 in most trials. For the Thyroid Disease data set, TSOCS outperforms the other two methods, and KKM outperforms KFCM. KKM and KFCM can separate class 1 completely, but cannot separate class 2 and class 3 in most trials. In the best case, TSOCS can separate class 2 completely, misclassify 4 patterns in class 1 and 4 patterns in class 3.

Running the independent samples t -test for the 20 trials from TSOCS and the 20 trials from the KKM method shows that except the Wine data set, accuracy data are significantly different with a p -value < 0.05 on the other five UCI data sets. As far as the TSOCS and the KFCM method are concerned, similar statistical test shows that the accuracy data are significantly different with a p -value < 0.01 for all six UCI data sets. Therefore, we conclude that the advantages of

Table 2. Average accuracy for six real data sets from 20 trials

Method Data Set	KKM			KFCM			TSOCS		
	worst	best	average	worst	best	average	worst	best	average
Wisconsin	96.19	96.19	96.19	94.14	94.14	94.14	96.78	96.93	96.85
Heart	59.26	80.00	77.69	77.78	81.48	79.80	81.48	82.22	81.85
Pima	66.67	66.80	66.78	67.84	67.97	67.96	72.53	76.95	74.26
IRIS	50.67	96.00	89.33	94.67	94.67	94.67	96.00	96.00	96.00
Wine	55.62	96.63	93.37	56.74	96.07	77.28	94.94	95.51	95.23
Thyroid	88.84	88.84	88.84	80.00	80.47	80.05	96.28	96.28	96.28

TSOCS over the other two kernel-based methods are not incidental consequence of random initial partitions.

5 Conclusions

We have described an algorithm of using iterative TSOCS to detect multiple clusters of non-convex shape. Experimental results allow us to make the following remarks regarding the unsupervised clustering problem:

- Kernel partitioning-based clustering algorithms outperform conventional partitioning-based clustering algorithms because they can better solve non-linearly separable clusters.
- TSOCS outperforms other kernel partitioning-based clustering algorithms in all cases. Our idea is to set a large ν in the secondary-stage one-class SVM to control the position of the hyperplane. This has resulted in a positive effect on the correct assignment of the cluster members.
- Regarding the time performance, TSOCS is slower than the other two methods because one-class SVM has to be solved frequently. KFCM can be substantially faster than the other two methods because it does not have to update cluster members in each iteration.
- Regarding the initialization of the algorithms, KKM and KFCM strongly depend on the guess of the initial values. In this study, we use a heuristic method to solve this problem, and this issue is not so important for the proposed method.

References

- [1] Schölkopf, B., Smola, A., Müller, K.R.: Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation* **10** (1998) 1299–1319
- [2] Girolami, M.: Mercer kernel-based clustering in feature space. *IEEE Transactions on Neural Networks* **13** (2002) 780–784
- [3] Dhillon, I., Guan, Y., Kulis, B.: Kernel k-means, spectral clustering, and normalized cuts. In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. (2004) 551–556

- [4] Dhillon, I., Guan, Y., Kulis, B.: A unified view of kernel k-means, spectral clustering and graph cuts. Technical Report TR-04-25, UTCS (2004)
- [5] Chen, S., Zhang, D.: Robust image segmentation using FCM with spatial constraints based on new kernel-induced distance measure. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* **34** (2004) 1907–1916
- [6] Zhang, D., Chen, S.: Clustering incomplete data using kernel-based fuzzy c-means algorithm. *Neural Processing Letters* **18** (2003) 155–162
- [7] Mizutani, K., Miyamoto, S.: Possibilistic approach to kernel-based fuzzy c-means clustering with entropy regularization. In: *Modeling Decisions for Artificial Intelligence*. Volume 3558 of *Lecture Notes in Computer Science.*, Springer (2005) 144–155
- [8] Kim, D., Lee, K., , Lee, D., Lee, K.: Evaluation of the performance of clustering algorithms in kernel-induced feature space. *Pattern Recognition* **38** (2005) 607–611
- [9] Du, W., Inoue, K., Urahama, K.: Robust kernel fuzzy clustering. In: *Fuzzy Systems and Knowledge Discovery*. Volume 3613 of *Lecture Notes in Computer Science.*, Springer (2005) 454–461
- [10] Tax, D., Duin, R.: Support vector domain description. *Pattern Recognition Letters* **20** (1999)
- [11] Ben-Hur, A., Horn, D., Siegelmann, H., Vapnik, V.: Support vector clustering. *Journal of Machine Learning Research* **2** (2002) 125–137
- [12] Müller, K., Mika, S., Ratsch, G., Tsuda, K., Schölkopf, B.: An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks* **12** (2001) 181–201
- [13] Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A., Williamson, R.: Estimating the support of a high-dimensional distribution. *Neural Computation* **13** (2001) 1443–1472
- [14] Chiang, J., Hao, P.: A new kernel-based fuzzy clustering approach: Support vector clustering with cell growing. *IEEE Transactions on Fuzzy Systems* **11** (2003) 518–527
- [15] F. Camastra, A.V.: A novel kernel method for clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27** (2005) 801–805
- [16] Newman, D., Hettich, S., Blake, C., Merz, C.: *UCI repository of machine learning databases* (1998)

A Confident Majority Voting Strategy for Parallel and Modular Support Vector Machines

Yi-Min Wen^{1,2} and Bao-Liang Lu^{1,*}

¹ Department of Computer Science and Engineering, Shanghai Jiao Tong University,
800 Dong Chuan Road, Shanghai 200240, China
{wenyimin; bllu}@sjtu.edu.cn

² Hunan Industry Polytechnic, Changsha 410208, China

Abstract. Support vector machines (SVMs) have been accepted as a fashionable method in machine learning community, but they cannot be easily scaled to handle large scale problems because their time and space complexities are around quadratic in the number of training samples. To overcome this drawback of conventional SVMs, we propose a new *confident* majority voting (CMV) strategy for SVMs in this paper. We call the SVMs using the CMV strategy CMV-SVMs. In CMV-SVMs, a large-scale problem is divided into many smaller and simpler sub-problems in training phase and some confident component classifiers are chosen to vote for the final outcome in test phase. We compare CMV-SVMs with the standard SVMs and parallel SVMs using majority voting (MV-SVMs) on several benchmark problems. The experiments show that the proposed method can significantly reduce the overall time consumed in both training and test. More importantly, it can produce classification accuracy, which is almost the same as that of standard SVMs and better than that of MV-SVMs.

1 Introduction

In recent years, there are many very large-scale data sets like public-health data, gene expression data, national economics data, and geographic information data. Using these very large data sets, researchers can get higher accuracy, discover infrequent special cases, and avoid over-fitting. However, most of existing machine learning methods are hard to be used to deal with these very large data sets because a very long training time and huge space are required. Therefore, one of the most challenging problems in machine learning community is to develop new learning model to efficiently handle these large data sets.

Today, support vector machine (SVM) [1] has been widely used in the field of pattern recognition for its strong theoretical foundations and good generalization

* To whom correspondence should be addressed. This work was supported in part by the National Natural Science Foundation of China under the grants NSFC 60375022 and NSFC 60473040, and the Microsoft Laboratory for Intelligent Computing and Intelligent Systems of Shanghai Jiao Tong University.

Table 1. The contingency table

	label $y = 0$	label $y = 1$
prediction $h(x) = 0$	Tp	Fp
prediction $h(x) = 1$	Fn	Tn

performance. However, both its training time complexity and space complexity are $O(N^2)$, where N denotes training set size. The reason is that training SVMs is to solve a quadratic programme problem in essence. Many efforts are made to scale SVMs, such as choosing representative samples by preprocessing training data [2] [3] [4] [5], avoiding to solve the quadratic programme problem [6] [7] [8], and using geometric algorithms [9] [10].

The divide-and-conquer principle has been applied to scale SVMs. The SVMs using the divide-and-conquer principle in a serial way include the standard SVMs training method SMO [11], SVM^{light} , and libSVM, as well as using boosting to scale SVMs [12]. The SVMs using the divide-and-conquer principle in a parallel way, which will be named as parallel SVMs later on, include support vector mixtures [13], bayesian committee support vector machine (BC-SVM) [14], min-max modular SVMs (M^3 -SVM) [15], and parallel mixture of SVMs [16]. Between sequential and parallel implementation, there are hierarchical and parallel methods [17], [18], [19], which filter non-support vectors in a cascade way.

From the point of view of parallel learning, parallel SVMs have many merits over monolithic SVMs. The first is that parallel SVMs can be benefited from cheap clustering systems by MPI, PVM, and the current grid computing [20]. The second is their reliability that parallel SVMs will still work even though some of their components fail. The third is their speedup, which can bring convenience to parameter selection.

In this paper, a confident majority voting (CMV) strategy is proposed to scale SVMs, which is inspired by an ensemble learning approach [21]. We call the SVMs using the CMV strategy CMV-SVMs. In CMV-SVMs, a large-scale task is divided into many smaller and simpler sub-problems in training phase and some confident component classifiers are chosen to vote for the final outcome in test phase. The experiments show that the proposed method can significantly reduce the overall time consumed in both training and test. More importantly, it produces classification accuracy which is almost the same as that of standard SVMs and better than that of MV-SVMs.

This paper is organized as follows. Section 2 introduces the model of CMV-SVMs, the definition of classification confidence, and the training and test algorithms for CMV-SVMs. In section 3, some experiments and analysis are presented for giving evidence of the advantages of CMV-SVMs. In section 4, the bias-variance decomposition strategy is employed to explore the reason why CMV-SVMs generalize better than MV-SVMs do. Finally, Section 5 is conclusions.

2 Confident Majority Voting

2.1 Definition of Classification Confidence

Given a problem with a training data set $S_{tr} = \{(x_1, y_1), \dots, (x_N, y_N)\}$, where $x_i \in \mathcal{X} \subseteq \mathcal{R}^n$ is an instance, $y_i \in \{0, 1\}$ is its class label, and N denotes the training data set size. After training, a classifier $h : \mathcal{X} \rightarrow \{0, 1\}$ will be obtained. Given a test sample x , $h(x)$ will output the class label of x . In order to evaluate the performance of $h(x)$, a contingency matrix is defined as in Table 1.

In real-world applications, the class label y of a test sample x is not known. A good classifier should output a class label for x with high classification confidence. Otherwise its output cannot be believed and used to handle real-world problems. For example, in the field of medical diagnostics, the classification confidence is very paramount. In this paper, the classification confidence for a test sample x is used to choose the classifiers which can vote for the final classification. The classification confidence for x that is classified as class ω is defined as follows:

$$T(x) = P(y = \omega | h(x) = \omega) = \frac{P(h(x) = \omega | y = \omega) * P(y = \omega)}{P(h(x) = \omega)}, \tag{1}$$

where $T(x)$ denotes the classification confidence for x .

Many work has been made to compute classification confidence [22]. As in Proposition 1, after setting an appropriate neighbor size for a test sample x in a validation data set, the performance of a classifier in the neighborhood of a test sample x is used to evaluate the classification confidence of x . In addition, it should be noted that the classification confidence has been defined as local class accuracy in the work of Woods [21].

Proposition 1. *Subscribing the size of neighborhood of a test sample x in a validation data set. The performance of a classifier in the neighborhood of x is evaluated according to Table. 1. If x is classified as class 0, then its classification confidence can be computed as: $T(x) = \frac{T_p}{T_p + F_p}$. If x is classified as class 1, then its classification confidence can be computed as: $T(x) = \frac{T_n}{T_n + F_n}$.*

2.2 Training and Test Algorithms

CMV-SVMs can be regarded as a parallel implementation of the divide-and-conquer principle and a mixture of ensemble and modular learning.

The training algorithm for CMV-SVMs can be described as follows:

1. Initiation: constant M , i.e., the number of the repeat of training data set partitioning, the value of partition K , and the appropriate parameters for SVMs.
2. For $n = 1, 2, \dots, M$

Partitioning: the training data set S_{tr} is randomly partitioned into K subsets with almost the same size, i.e. $\cup_{j=(n-1)*K+1}^{j=n*K} S_{tr}^j = S_{tr}$ and $\cap_{j=(n-1)*K+1}^{j=n*K} S_{tr}^j = \Phi$, where Φ denotes an empty set. The aim of making equal sizes of subsets is intended to keep load balance, although the training time of SVMs does not only depend on the training data size.

3. Training: $M * K$ support vector machines as component classifiers, $h_j, 1 \leq j \leq M * K$, are trained on the corresponding subsets $S_{tr}^j, 1 \leq j \leq M * K$. Because no communication is required in the training phase among the component classifiers, they can be trained in a parallel way.
4. Validating: Use training data set S_{tr} as a validation set to evaluate each component classifier $h_j, 1 \leq j \leq M * K$ and save the examination results. The validation results are used to evaluate the classification confidence for a test sample.

The test algorithm for CMV-SVMs can be described as follows:

1. Initiation: given a appropriate neighborhood size q , a classification confidence threshold $\varepsilon, 0 \leq \varepsilon \leq 1$, and a test sample x .
2. Classifying: Compute $h_j(x), 1 \leq j \leq M * K$ in parallel. If all the $h_j(x), 1 \leq j \leq M * K$ are the same, any $h_j(x)$ can be used as the final class label of x , then return. If not, goto next step.
3. Calculate all the classification confidence of x , i.e. $T_j(x), 1 \leq j \leq M * K$. Find the largest classification confidence: $imax = \operatorname{argmax}_{j=1}^{M * K} T_j(x)$.
4. Find the set $\nabla = \{j | (T_{imax}(x) - T_j(x) \leq \varepsilon, 1 \leq j \leq M * K)\}$.
5. Confident combining: If $|\nabla| = 1$, where $|\nabla|$ denotes the size of the set ∇ , then use h_{imax} to classify, else choose classifiers $h_j, j \in \nabla$ to vote.

3 Experiments and Results

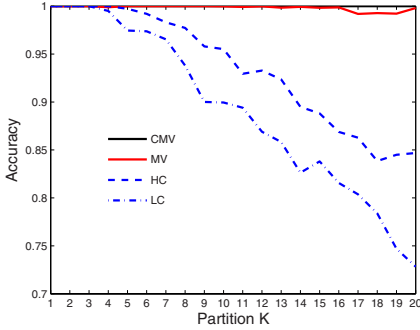
In order to evaluate the performance of the proposed method, some experiments are performed to compare our CMV-SVMs with standard SVMs and MV-SVMs. The experimental platform is PC with 1G RAM and 3G CPU. The training algorithm used is libSVM with cache of 40M and kernel function of RBF. Three data sets are used in the experiments, the first two are artificial data and the last is a real data set. The statistics of all the classification problems and the parameters used for SVMs are shown in Table. [2](#).

The artificial data sets include two-spirals data and checkboard data. The data of the two-spirals are uniformly chosen from two curves of $\rho = \theta$ and $\rho = -\theta$, where $(\rho \theta)$ means polar coordinates. The data of checkboard problem are chosen from a 2D checkboard that divides a 200×200 square into four quadrants in which the points are uniformly distributed [\[18\]](#). Forest coverType data set comes from UCI [\[23\]](#), and only the samples of its second and sixth classes are chosen, in which one half of the data are used as test data and the rest data are used for training.

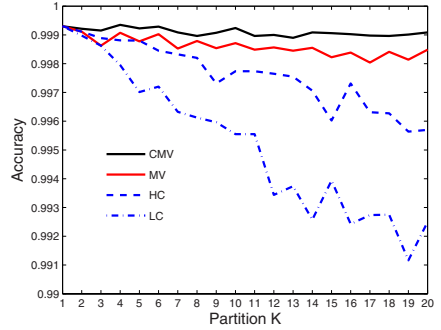
In order to get reliable experimental results, 100 training sets and a common test set are randomly generated for the two-spirals and the checkboard problems, respectively. For the Forest coverType classification task, 100 training sets are randomly generated and each of them contains two-thirds of the whole training data. As a result, the experiments are performed in 100 times and the average results are presented. In order to systematically evaluate the proposed method,

Table 2. Problem description and the parameters used in SVMs

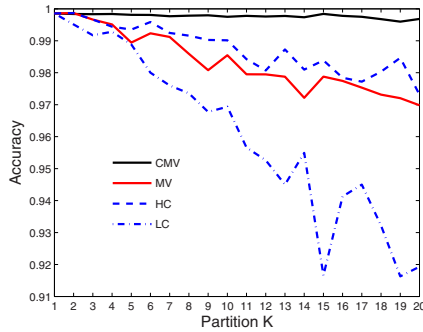
Problems	#attributes	#training data	#test data	c	σ	neighbor size q
Two spirals	2	3000	20000	128	2	5
Checkboard	2	32000	80000	1000	31.62	90
Forest coverType	54	28132	28132	128	0.25	90



(a)



(b)



(c)

Fig. 1. Classification accuracy comparison with $M = 1$. (a) Two-spirals, (b) Checkboard, and (c) Forest coverType. Here HC means the component SVM classifier with the highest classification accuracy, and LC means the component SVM classifier with the lowest classification accuracy

the value of K is set to 2, 3, ..., 20 in the experiments. $K = 1$ means that the classifier is trained by the entire training data, i.e. standard SVM is used.

From Fig. 1, we can see firstly that the generalization ability of CMV-SVMs is almost the same as standard SVMs in case of different partitions and sometimes better than standard SVMs. Secondly the generalization accuracy of CMV-SVMs is higher than all its component SVM classifiers. This demonstrates that the confident combining can efficiently make the component SVM classifiers work cooperatively. In addition, considering Table. 3, it seems that CMV-SVMs can

Table 3. Comparing the accuracy of CMV-SVMs with the accuracy of *Knn*

Problems	<i>Knn</i>		CMV-SVMs		
	neighbor size	accuracy	$K = 1$	$K = 2, 3, 4, \dots, 20$	
				mean	variance
Twospirals	5	1.000	1.000	1.000	0.00007
Checkboard	90	0.995	0.999	0.999	0.00012
Forest coverType	90	0.989	0.999	0.998	0.00061

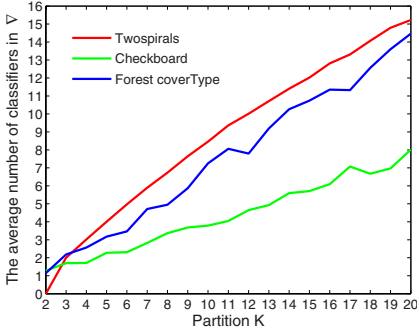


Fig. 2. The average number of confident SVMs for one test instance

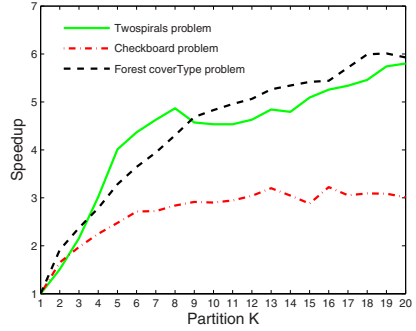


Fig. 3. The speedup, here the CPU time includes both training and test time

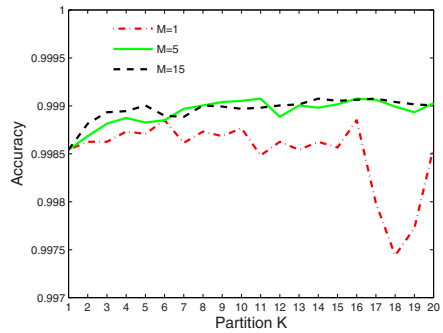
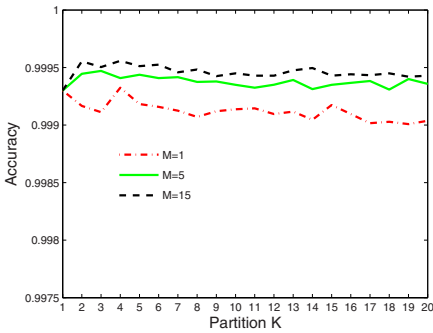


Fig. 4. Large values of M can improve classification accuracy. The left is on checkboard data set, and the right is on Forest coverType data set.

get higher accuracy than k -NN does. Therefore, k -NN cannot substitute CMV-SVMs even CMV-SVMs use the information of the nearest neighbor of a test instance. Thirdly the generalization accuracy of CMV-SVMs is higher than that of MV-SVMs.

Fig. 2 illustrates the average number of classifiers for one test sample when all $h_j(x), 1 \leq j \leq M * K$ are not the same. From Fig. 2 we can see that classification

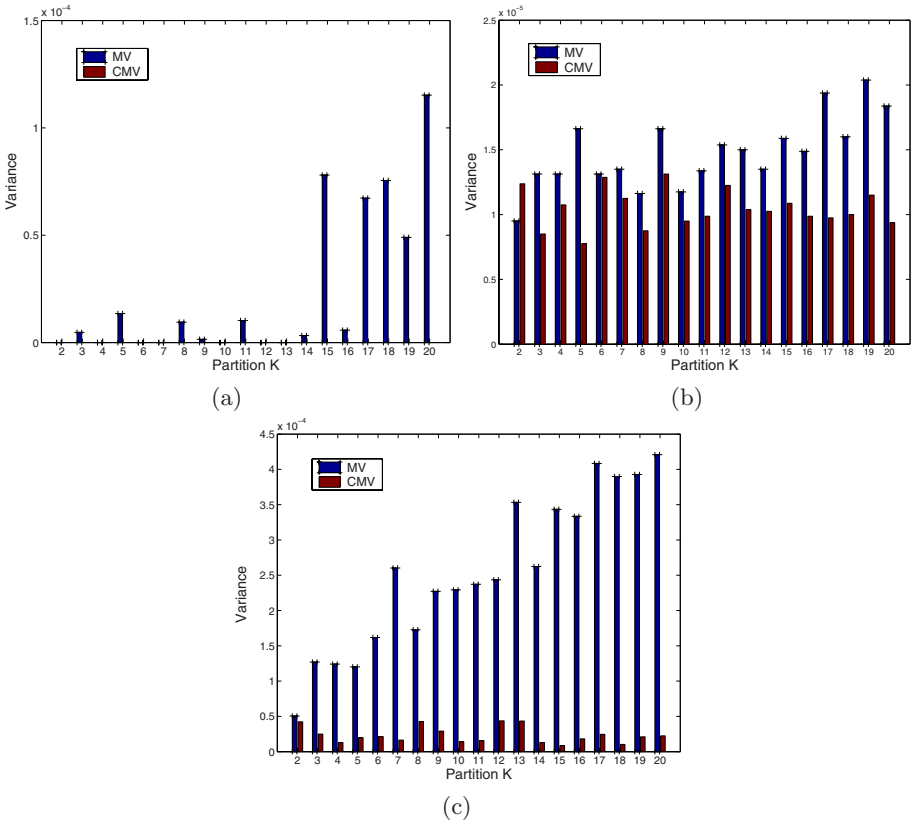


Fig. 5. Comparison of the variance of MV and CMV in case of different partitions: (a) Two-spirals, (b) Checkboard, and (c) Forest coverType

confidence requirement filters some component SVMs. It is like that for a given question only experts with richer experience can be selected to take part in decision-making. Therefore, the confident combining strategy can improve the generalization accuracy. The deeper reason is explored again by bias-variance decomposition [24] strategy in the next section.

In Fig. 3, the CPU time considered includes both training and test time. It can be seen that CMV-SVMs can significantly reduce the overall time. Fig. 4 shows that the larger the value of M the higher the accuracy. It seems because the larger M will lead to more diverse SVM classifiers and so more confident classifiers can be found to combine for classifying a test sample.

4 Bias-Variance Decomposition

Zhou *et al.* proposed an approach GASEN, which selects some neural networks based on the evolved weights to make up the ensemble, to show many could be

better than all in neural networks ensembling [25]. By the bias-variance decomposition, it was explored that GASEN can significantly reduce both the variance and the bias simultaneously. Liking GASEN, CMV-SVMs selects some confident classifiers to make up the ensemble. In order to further explore the reason why CMV-SVMs generalize better than MV-SVMs do, the bias-variance decomposition is also employed in this paper.

4.1 Bias and Variance

Bias-variance analysis provides a powerful tool to study learning algorithms. By it, one can get insight into the error production of an algorithm and find the ways to improve the algorithms. According to Dietterich [26], the statistics bias of a learning algorithm is the persistent or systematic error that the learning algorithm is expected to make when trained on training sets of size N . Given a set S of training examples, algorithm A outputs a hypothesis $A(S) = \hat{h}_S$. It is convenient to define $\hat{p}_S(x)$ to be the probability that \hat{h}_S misclassifies test point x . This probability is 1 if \hat{h}_S misclassifies x , and 0 otherwise.

$$\hat{p}_S(x) = \begin{cases} 1, & \text{if } \hat{h}_S(x) \neq y, \\ 0, & \text{if } \hat{h}_S(x) = y. \end{cases} \tag{2}$$

Based on the above definition, given a sequence of training sets S_1, S_2, \dots, S_l , each of size N , and a common test set S_{ts} , applying learning algorithm A to construct hypotheses $\hat{h}_{S_1}, \hat{h}_{S_2}, \dots, \hat{h}_{S_l}$, the averaged probability of error can be defined to be the average of these \hat{p}_S 's, where the average is taken over all possible training sets:

$$\bar{\hat{p}}(A, N, x) = \lim_{l \rightarrow \infty} \frac{1}{l} \sum_{i=1}^l \hat{p}_{S_i}(x). \tag{3}$$

The expect error rate of A for a test point x is

$$E(A, N, x) = \bar{\hat{p}}(A, N, x). \tag{4}$$

The definition of Bias and Variance is like below:

$$B(A, N, x) = \begin{cases} 0, & \text{if } \bar{\hat{p}}(A, N, x) \leq 0.5, \\ 1, & \text{if } \bar{\hat{p}}(A, N, x) > 0.5. \end{cases} \tag{5}$$

$$V(A, N, x) = \begin{cases} \bar{\hat{p}}(A, N, x), & \text{if } \bar{\hat{p}}(A, N, x) \leq 0.5, \\ \bar{\hat{p}}(A, N, x) - 1, & \text{if } \bar{\hat{p}}(A, N, x) > 0.5. \end{cases} \tag{6}$$

So, the variance is the increase in the error rate at x relative to the bias.

4.2 Result Analysis

With the experimental methodology illustrated in Section 3, the bias and variance of CMV and MV are computed according to Dietterich's method. Fig. 5 shows that the variances of CMV are smaller than the variances of MV in case of different partitions in all the classification tasks, the only exception lies in the case of Checkboard data classification when $K = 2$. The biases of CMV and MV are zero in all cases and are not displayed. These evidences can explain why CMV-SVMs can generalize better than MV-SVMs do.

From Fig. 5, we can see that the variance of CMV keeps stable while the variance of MV gets bigger with increasing the number of partitions. These evidences can explain why MV-SVMs generalize worse and worse with increasing the number of partitions, while CMV-SVMs maintain their generalization accuracy.

5 Conclusion

In this paper, we have proposed a novel support vector machine called CMV-SVM to scale SVMs. Comparison with other parallel SVMs, CMV-SVMs are more easily to be implemented. Several experimental results indicate that the proposed confident majority voting strategy can get higher accuracy than majority voting does and the proposed CMV-SVMs can not only significantly reduce the overall time consumed in training and test, but also produces classification accuracy that is almost the same as standard SVMs do.

The limitation of the proposed CMV-SVMs lies in the necessity of storing all the training samples to evaluate the classification confidence for novel inputs. However, choosing the confident components can ensure better performance for modular learning system. The future work includes to modify the method of computing classification confidence and compare CMV-SVMs with other parallel SVMs on large-scale problems systematically.

References

1. Vapnik, V.N.: Statistical Learning Theory. Wiley Interscience (1998)
2. Boley, D., Cao, D.W.: Training Support Vector Machine using Adaptive Clustering. In: Proceedings of SDM'04. Lake Buena Vista, USA (2004)
3. Evgeniou, T., Pontil, M.: Support Vector Machines with Clustering for Training with Very Large Datasets. In: Lectures Notes in Artificial Intelligence **2308** (2002) 346-354
4. Yu, H., Yang, J., Han, J.W.: Classifying Large Data Sets using SVM with Hierarchical Cluster. In: SIGKDD'03. Washington, DC, USA (2003)
5. Pavlov, D., Chudova, D., Smyth, P.: Towards Scable Support Vector Machines using Squashing. In: Proceedings of the sixth ACM SIGDD conference on knowledge discovery and data mining (2000)
6. Fung, G., Mangasarian, O.L.: Proximal Support Vector Machine Classifiers. In: Proceedings of the seventh ACM SIGDD conference on knowledge discovery and data mining. San Francisco, CA (2001)

7. Mangasarian, O.L., Musicant, D.R.: Active Set Support Vector Machine Classification. In: *Advances in Neural Information Processing Systems*. MIT Press **13** (2001) 577-583
8. Mangasarian, O.L., Musicant, D.R.: Lagrangian Support Vector Machines. *Journal of Machine Learning Research* **1** (2001) 161-177
9. Tsing, I.W., Kwok, J.T., Cheung, P.M.: Core Vector Machines: Fast Svm Training on Very Large Data Sets. *Journal of Machine Learning Research* **6** (2005) 363-392
10. Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., Murthy, K.R.K.: A Fast Iterative Nearest Point Algorithm for Support Vector Machine Classifier Design. *IEEE Trans. Neural Networks* **11** (2000) 124-136
11. Platt, J.: Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. Technical Report MSR-TR-98-14, Microsoft Research (1998)
12. Pavlov, D., Mao, J.C., Dom, B.: Scaling-up Support Vector Machines using Boosting Algorithm. In: *Proceedings of International Conference on Pattern Recognition* (2000)
13. Kwok, J.T.: Support Vector Mixture for Classification and Regression Problems. In: *Proceedings of the International Conference on Pattern Recognition*. Brisbane, Queensland, Australia (1998) 255-258
14. Schwaighofer, A., Tresp, V.: The Bayesian Committee Support Vector Machine. In: *Lectures notes in computer science* **2130** (2001)
15. Lu, B.L., Wang, K.A., Utiyama, M., Isahara, H.: A Part-versus-part Method for Massively Parallel Training of Support Vector Machines. In: *Proceedings of IJCNN'04*. Budapest, Hungary (2004) 735-740
16. Collobert, R., Bengio, Y., Bengio, S.: A Parallel Mixture of Svms for very Large Scale Problems. In: *Neural Information Processing Systems*. MIT Press **17** (2004)
17. Graf, H.P., Cosatto, E., Bottou, L., Durdanovic, I., Vapnik, V.: Parallel Support Vector Machines: The Cascade Svm. In: *Neural Information Processing Systems*. MIT Press **17** (2005)
18. Wen, Y.M., Lu, B.L.: A Cascade Method for Reducing Training Time and the Number of Support Vectors. In: *Lecture Notes in Computer Science* **3173** (2004) 480-486
19. Wen, Y.M., Lu, B.L.: A Hierarchical and Parallel Method for Training Support Vector Machines. In: *Lecture Notes in Computer Science* **3496** (2005) 881-886
20. Foster, I.: The Grid: A New Infrastructure for 21st Century Science. *Physics Today* **55** (2002) 42-47
21. Woods, K., Kegelmeyer, W.P.J., Bowyer, K.: Combination of Multiple Classifiers using Local Accuracy Estimates. *IEEE Trans. Pattern Analysis and Machine Intelligence* **19** (1997) 405-410
22. Zaragoza, H., Alché-Buc, F.: Confidence Measures for Neural Network Classifiers. In: *IPMU'98* (1998)
23. Blake, C.L., Merz, C.J.: UCI Repository of Machine Learning Databases. Univ. California, Dept. Inform. Comput. Sci., Irvine, CA. [online] (1998) Available: <ftp://ftp.ics.uci.edu/pub/machine-learning-databases>
24. German, S., Bienenstock, E., Doursat, R.: Neural Networks and the Bias/Variance Dilemma. *Neural Computation* **4** (1992) 1-58
25. Zhou, Z.H., Wu, J.X., Tang, W.: Ensembling Neural Networks: Many Could be Better than All. *Artificial Intelligence* **137** (2002) 239-263
26. Dietterich, T.G. Kong, E.B.: Machine Learning Bias, Statistical Bias, and Statistical Variance of Decision Tree Algorithms. Technical report, Department of Computer Science, Oregon State University (1995)

Soft-Sensor Method Based on Least Square Support Vector Machines Within Bayesian Evidence Framework

Wei Wang¹, Tianmiao Wang¹, Hongxing Wei¹, and Hongming Zhao²

¹ Beijing University of Aeronautics and Astronautics, Beijing 100083, P.R. China
wxw@me.buaa.edu.cn

² Institute of High Energy Physics, Chinese Academy of Sciences, Beijing 100039, P.R. China

Abstract. Based on the character and requirement of the dynamic weighing of loader, the soft sensor technique was adapted as the weighing method, and the least square support vector machine (LS-SVM) as its modelling method. Also the Bayesian evidence framework was used in LS-SVM for selecting and tuning its parameter. And then, after the nonlinear regression algorithms of LS-SVM and the principle of Bayesian evidence framework were introduced, the soft sensor model based on LS-SVM was given. In the end, emulation analysis results indicate that soft-sensor method based on LS-SVM within Bayesian evidence framework is a valid means for solving dynamic weighing of loader.

1 Introduction

Dynamic weighing of loader is a complex engineering problem, and many researchers have been emphasizing on it for quite some time. Because not only the forces of the working equipment are complex, but also the construction environment is usually cruel, therefore using conventional measuring method based on sensor and other hardware, we can't get an approving measure precision. In this paper, we introduce the soft sensor technology [1] to dynamic weighing of loader, and set up the corresponding soft sensor model architecture.

There have been many soft sensor modelling methods. Among them, ANNs modelling method has proven to be a powerful method in soft sensor modelling for its simple structure and easy realization. Despite many advances, there still remains a number of shortcomings, for example, the network architecture difficult to confirm, the existence of local minima problem for the training course, excessive dependence on quantity and quality of training data set, and the generalization ability is not very well, etc.. In essential, all the before-mentioned are brought because the ANNs is based on the gradual theory, i.e., machine learning course requires infinite samples in theory, but there are not so many samples in fact. In order to solve this problem, Vapnik bring forward a creative small-sample learning theory, i.e., statistical learning theory [2], based on the theory, a novel powerful machine learning method called Support Vector Machines(SVM)

was developed. The SVM optimization solutions are based on structural risk minimization which ensures its powerful machine learning performance and synchronously excellent generalization ability, based on limit samples. However it has the limitation of speed in training large data set. Least square support vector machines [3,4] (LS-SVM) is the improved SVM, it transfers the convex optimization problem to a problem solving a set of linear equations, therefore it simplify the operation and has higher efficiency. Hereby we propose LS-SVM to be the soft sensor modelling method [5] for dynamic weighing of loader.

Synchronously, to render the LS-SVM to exhibit best performance, we employ the three levels of inference within Bayesian evidence framework to optimize its parameters. In the end, in this paper we adopt the LS-SVM within Bayesian evidence framework as soft sensor modelling method for dynamic weighing of loader.

This paper is organized as following: after this introduction section, the theory of LS-SVM and the parameters inference algorithm within Bayesian evidence framework are introduced respectively in section 1 and section 2. Section 3 is the application study, in this section, the soft sensor modelling method was applied to the dynamic weighing of loader ,based on the characteristic of dynamic weighing, the soft sensor model structure has been given, then the simulation results was presented. And in the last section we give some concluding remarks.

2 Least Square Support Vector Machines

Consider a given training set of l data points $(x_1, y_1), (x_2, y_2), \dots, (x_i, y_i), \dots, (x_l, y_l)$, with input data $x_i \in \mathbf{R}^n$ and output data $y_i \in \mathbf{R}$. In feature space LS-SVM model taken as the form

$$\hat{y} = \omega^T \varphi(x) + b. \quad (1)$$

After using the above training set, we can get a nonlinear model which can not only compromise between the model complexity and training error, but also compromise the precise and the generalization ability. Then toward an arbitrary sampled point input x_d outside of the above training set, by the model we can get a corresponding target value $y_d = f(x_d)$, which has the least error between y_d and y . Compared with standard SVM, LS-SVM involves equality instead of the inequality constraints. Furthermore, LS-SVM uses the least squares loss function instead of the ε -insensitive loss function [4,6]. In this way, the solution follows from a linear KKT system instead of a computationally hard QP problem. Therefore it is easier to optimize and the computing time is short. For LS-SVM, the optimization problem is formulated as

$$\min_{\omega, b, e} J_1(\omega, e) = \frac{\mu}{2} \omega^T \omega + \frac{\xi}{2} \sum_{i=1}^l e_i^2 = \mu E_W + \xi E_D. \quad (2)$$

Subject to the equality constraints $y_i = \omega^T \varphi(x_i) + b + e_i$ ($i = 1, \dots, l$), where $e_i \in \mathbf{R}$ is the difference between y_d and y , and ξ is a regularization constant.

Smaller ξ can avoid overfitting in case of noisy data. And in this formula, the regularization and error term are defined as E_W and E_D respectively. Then the Lagrangian for (1) and (2) is

$$L(\omega, b, e, a) = J(\omega, e) - \sum_{i=1}^l a_i [\omega^T \varphi(x_i) + b + e_i - y_i], \quad (3)$$

with Lagrange multipliers $a_i \in \mathbf{R}$, the conditions for optimality are given by

$$\begin{cases} \frac{\partial L}{\partial \omega} = 0 \rightarrow \omega = \sum_{i=1}^l a_i \varphi(x_i) \\ \frac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^l a_i y_i = 0 \\ \frac{\partial L}{\partial e_i} = 0 \rightarrow a_i = \xi e_i, \quad i = 1, \dots, l \\ \frac{\partial L}{\partial a_i} = 0 \rightarrow b = y_i - \omega^T \varphi(x_i) - e_i, \quad i = 1, \dots, l \end{cases}, \quad (4)$$

By eliminating e_i and ξ , the optimization problem is to solve linear equations

$$\begin{bmatrix} 0 & Y^T \\ Y & \Omega + \gamma^{-1} I_l \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} 0 \\ 1_v \end{bmatrix}, \quad (5)$$

where $Y = [y_1, \dots, y_l]^T$, $a = [a_1, \dots, a_l]^T$, $1_v = [1, \dots, 1]^T$. According to the Mercer condition, one takes a kernel.

$$\Omega_{ij} = \varphi(x_i)^T \varphi(x_j) = K(x_i, x_j), \quad (6)$$

then the resulting LS-SVM recurrent model becomes

$$y(x) = \sum_{i=1}^l a_i K(x, x_i) + b. \quad (7)$$

Based on the feature of the object to be studied, we chose the RBF kernels

$$K(x_i, x_j) = \exp\left\{-\frac{\|x_j - x_i\|^2}{2\sigma^2}\right\}$$

as the LS-SVM kernel function, and in the case the remaining unknowns are the regularization parameter ξ and the kernel width parameter σ .

3 Parameters Optimizing of LS-SVM Within Bayesian Evidence Framework

Compared with the traditional approach, the Bayesian method [7] provides a rigorous framework for the automatic adjustment of the regularization parameters to their near-optimal values, without the need to set data aside in a validation set [8,9]. Moreover, the Bayesian framework [10] allows objective comparison among solutions using different kernel function or different kernel parameters.

The basic idea of the Bayesian evidence framework is a maximum a posterior solution to infer problems with Gaussian priors and an appropriate likelihood function based probabilistic interpretation, i.e., the optimal parameters or model are got according as its maximal posterior probability. The evidence framework is divided into three levels of inference. The model parameters ω and b , the regularization parameter γ , and the kernel width parameter σ are inferred respectively by applying Bayesian formula on the three different levels.

3.1 Level 1 Inference

Given the data points $D = \{(x_i, y_i)\}_{i=1}^l$, it is assumed that the hyperparameters μ, ξ and the LS-SVM model H are based on the kernel function K , we obtain the model parameters ω and b by maximizing the posterior $p(\omega, b/D, \log \mu, \log \xi, H)$. Application of Bayes' rule [8] at the first level of inference gives:

$$p(\omega, b/D, \log \mu, \log \xi, H) = \frac{p(D/\omega, b, \log \mu, \log \xi, H)p(\omega, b/\log \mu, \log \xi, H)}{p(D/\log \mu, \log \xi, H)}, \tag{8}$$

where H is the model corresponding to the kernel function $K(x_i, x_j)$ with different parameters, the $p(D/\log \mu, \log \xi, H)$ is a normalizing constant, that is the evidence to be used in next level of inference, the $p(\omega, b/\log \mu, \log \xi, H)$ is the prior, and the $p(D/\omega, b, \log \mu, \log \xi, H)$ is the likelihood. Assuming a Gaussian prior over ω with variance $1/\mu$, and a separate uniform distribution over b , then

$$p(D/\log \mu, \log \xi, H) \propto \exp(-\mu E_W), \tag{9}$$

and assuming a Gaussian noise on the target variable, then

$$p(\omega, b \log \mu, \log \xi, H) \propto \exp(-\xi E_D), \tag{10}$$

hence

$$p(\omega, b/D, \log \mu, \log \xi, H) \propto \exp(-\mu E_W) \exp(-\xi E_D) \propto \exp(-J_1(\omega, e)), \tag{11}$$

i.e.

$$-\log [p(\omega, b/D, \log \mu, \log \xi, H)] \propto \frac{\mu}{2}(\omega^T \omega) + \frac{\xi}{2}e_i^2 = \mu E_W + \xi E_D. \tag{12}$$

We can see that the course of maximum a posteriori estimate ω_{MP} and b_{MP} is to by optimizing (2).

3.2 Level 2 Inference

The performance of the LS-SVM also depends upon the choice of the regularization parameter ξ and the kernel parameter σ of the RBF kernel. In this section,

Bayes' rule is applied on the second level of inference to infer the hyperparameters μ and ξ . On the given data points D , $p(\log \mu, \log \xi/H)$ is assumed to be a flat prior, one can get

$$p(\log \mu, \log \xi/D, H) = \frac{p(D/\log \mu, \log \xi, H)p(\log \mu, \log \xi/H)}{p(D/H)} \propto p(D/\log \mu, \log \xi, H), \tag{13}$$

where the likelihood $p(D/\log \mu, \log \xi, H)$ is the normalizing constant of the level 1 inference, which can be got by substituting (9), (10) and (11) into (8).

In order to find a trade-off between the model fit and a complexity term to avoid overfitting (10), the evidence is optimized on level 2 inference, i.e. to get optimized parameters μ_{MP} and ξ_{MP} for the parameter μ and ξ respectively. Then the optimization prople (11) is transferred into

$$J_2(\mu, \xi) = \mu J_\omega(\omega_{MP}) + \xi J_e(\omega_{MP} + b_{MP}) + \sum_{i=1}^m \frac{\log(\mu + \xi \lambda_i)}{2} - \frac{m}{2} \log \mu - \frac{m-1}{2} \log \xi, \tag{14}$$

to define effective number of parameter (11) is

$$d_{eff} = 1 + \sum_{i=1}^m \frac{\xi \lambda_i}{\mu + \xi \lambda_i} = 1 + \sum_{i=1}^m \frac{\gamma \lambda_i}{\mu + \gamma \lambda_i}, \tag{15}$$

where $\gamma = \xi/\mu$, $m \leq l-1$. Based on the optimal conditions of (14), one can get

$$\partial J_2/\partial \mu = 0 \rightarrow 2\mu_{MP} J_\omega(\omega_{MP}; \mu_{MP}, \xi_{MP}) = d_{eff}(\mu_{MP}, \xi_{MP}) - 1, \tag{16}$$

$$\partial J_2/\partial \xi = 0 \rightarrow 2\xi_{MP} J_e(\omega_{MP}; b_{MP}; \mu_{MP}, \xi_{MP}) = l - d_{eff}, \tag{17}$$

then (17) is corresponding to noise level(variance) $1/\xi$, i.e.

$$1/\xi = \frac{1}{2} \sum_{i=1}^l e_i^2 / (l - d_{eff}). \tag{18}$$

Then one can reformulate (14) into the optimization problem

$$\min_{\gamma} \sum_{i=1}^{l-1} \log(\lambda_i + \frac{1}{\gamma}) + (l-1) \log(J_\omega(\omega_{MP}) + \gamma J_e(\omega_{MP}, b_{MP})), \tag{19}$$

where,

$$\begin{cases} J_e(\omega_{MP}, b_{MP}) = \frac{1}{2\lambda^2} y^T N_c V (\Lambda + I_l/\gamma)^{-2} V^T N_c y \\ J_\omega(\omega_{MP}) = \frac{1}{2} y^T N_c V \Lambda (\Lambda + I_l/\gamma)^{-2} V^T N_c y \\ J_\omega(\omega_{MP}) + \gamma J_e(\omega_{MP}, b_{MP}) = \frac{1}{2} y^T N_c V (\Lambda + I_l/\gamma)^{-1} V^T N_c y \end{cases}, \tag{20}$$

with eigenvalue decomposition $N_c \Omega N_c = V^T \Lambda V$, then we can get the optimal γ_{MP} , and by (16) and (17), we get

$$\mu_{MP} = (d_{eff} - 1) / (2J_\omega(\omega_{MP})), \tag{21}$$

$$\xi_{MP} = (l - d_{eff}) / (2J_e(\omega_{MP}, b_{MP})). \tag{22}$$

3.3 Level 3 Inference

Different models can be obtained by choosing different kernel function or different kernel parameters. For an RBF kernel with tuning parameters σ , the corresponding models H_σ is calculated by the posterior

$$p(H_\sigma / D) = \frac{p(D / H_\sigma) p(H_\sigma)}{p(D)} \propto (D / H_\sigma) p(H_\sigma) \propto p(D / H_\sigma), \tag{23}$$

then giving a series of possible kernel parameter values $\sigma_1, \dots, \sigma_m$, by ranking the evidences of different models $p(D / H_1), \dots, p(D / H_m)$, and selecting the tuning parameters with the greatest model evidence, we can get the optimal kernel parameter σ_{MP} .

4 Application Study

4.1 Soft Sensor Modeling Framework

For the dynamic weighing of loaders, because the hydraulic pressure signal can be easily acquired correspondingly. Also during the course of lifting, by putting a hydraulic pressure sensor into the lifting cylinder to carry out dynamic weighing, not only make the loader work naturally but also improve the applicability of the measurement method. During the course of lifting, the weight W to be measured not only have relation to the hydraulic pressure signal p of the lifting cylinder, the position ΔL of lifting cylinder and the lift crane's lifting velocity v , but also to the random noises n of industry field, etc. Hence, we introduce a soft sensor modelling method derived from conventional soft sensor technology, so we take the measurable output variable and some parts of

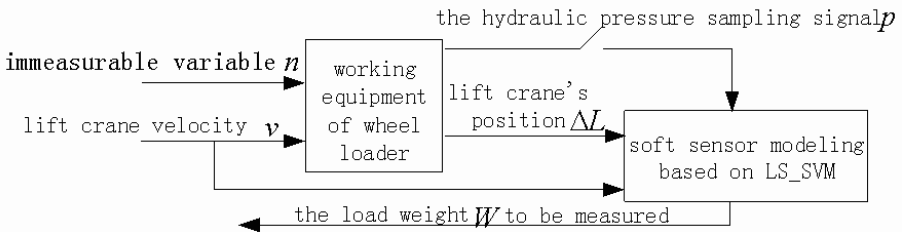


Fig. 1. Soft sensor model framework of dynamic weighing

input variable of industrial object as the input variables of soft sensor modelling, and the input to be measured as the output variable. Therefore we adopt above-mentioned factors except the random noises n as the secondary variables, and the weight to be measured as the primary variable, then we set up the soft sensor modelling framework (Fig. 1), utilize batch data 031 as the training data set to train the least square machines in order to establish the soft sensor modelling, and the batch data 131 as verifying data set in order to obtain the weight W .

4.2 Simulation Analysis

Parameters tuning. While the type of kernel function was confirmed, the performance of the LS-SVM would only be determined by the selection of the kernel parameters. There are two parameters to be confirmed, that is regularization parameter ξ and the kernel parameter σ . Within Bayesian evidence framework, we employ level 2 inference to optimize the parameter ξ , and level 3 inference to optimize the parameter σ , the deducing course is illustrated in Fig. 2 and Fig. 3.

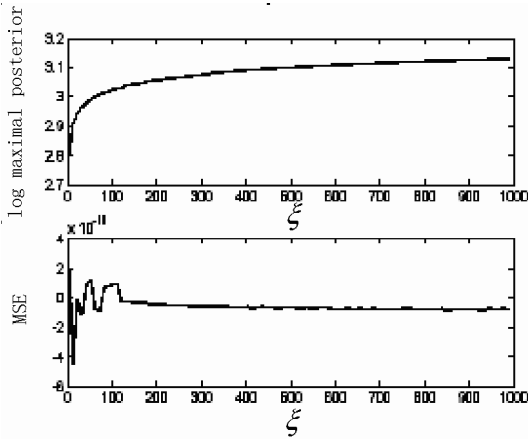


Fig. 2. The posterior and its testing error using level 2 inference

Fig. 2 shows that, while the regularization parameter ξ becomes more larger, the corresponding training error gets smaller, but when ξ increases to some extent, the change of the training error becomes smooth. Also considering the verifying error is independent of the parameter ξ , therefore we choose $\xi = 900$.

Synchronously, if the kernel parameter is small, which indicates that the relationship among SVs is tight, but if it is too small, it would make the model complex excessively, and will result in overfitting, i.e., the generalization ability getting worse. On the contrary, if the kernel parameter σ is large, which indicates

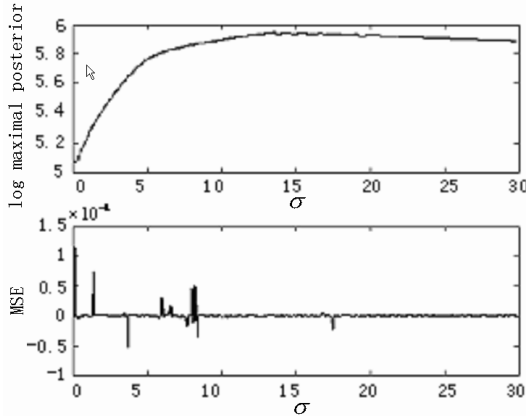


Fig. 3. The posterior and its testing error using level 3 inference

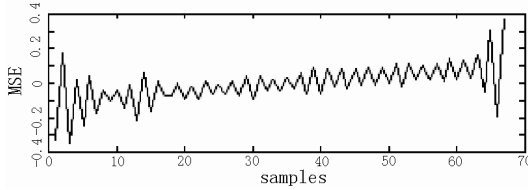


Fig. 4. The training error performance of LS-SVM

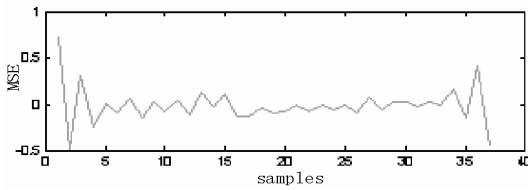


Fig. 5. The verification error performance of LS-SVM

that the relationship among SVs is incompact, but if it is too large, it will result in defective fitting. Fig. 3 reflect the kernel parameter σ inference course, and based on the inference result, we choose the kernel parameter $\sigma = 14.5$.

Simulation results. Then we apply the LS-SVM confirmed to carry out soft sensor modeling for dynamic weighing of loader, Fig. 4 and Fig. 5 show the training and verifying error curve respectively, the curves reflect the Mean Squared Error(MSE) analysis results between the estimated value with LS-SVM soft sensor model and the true value. We can find that, because there is concussion caused by scooping force at the beginning process of lift crane’s raising course, while at the ending course there is the structure design influence of working equipment,

the corresponding training errors are on the high side, yet the global training error of the lift crane's raising course is good, only reaching to 8.4680e-007; at the same time, although the validation error is larger than the corresponding training error, yet it exhibits better generalization ability, the error just attain to 6.8638e-006. In a word, using the LS-SVM as the modelling method we can achieve satisfactory performance, hence it is an effective soft sensor modeling method for the dynamic weighing of loaders.

5 Conclusion

Based on the characteristic of dynamic weighing of loader, the sensor modeling method is proposed to estimate the immeasurable variables during the weighing course. And we adopt LS-SVM as the soft sensor modeling method. Also to render the LS-SVM to exhibit best performance, we employ level 2 inference to optimize its regularization parameter , and the level 3 inference to optimize its kernel parameter σ .

After the theory of LS-SVM and the parameters optimizing course within Bayesian evidence framework have been introduced, we set up the soft sensor modelling architecture. Then we apply the LS-SVM to carry out soft sensor modelling for dynamic weighing of loader. The simulating results indicates that soft-sensor method based on LS-SVM within Bayesian evidence framework is a valid means for solving dynamic weighing of loader.

References

1. Huang, F.L.: Soft-Sensing Idea and Soft-Sensing Techniques. *Acta Metrologica Sinca* **25** (2004) 284-286
2. Vapnik, V. N.: *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin Heidelberg New York (1995)
3. Suykens, J.A.K., Vandewalle, J.: Least Square Support Vector Machines for Classification and Nonlinear Modelling. *Neural Network World* **10** (2000) 29-48
4. Suykens, J.A.K., Vandewalle, J.: The Least Square Support Vector Machines Classifiers. *Neural Network Letters* **19** (1999) 293-300
5. Yan, W.W., Shao, H.H., *et al.*: Soft Sensor Modelling Based on Support Vector Machine and Bayesian Model Selection, *Computing and Chemical Engineering* **28** (2004) 1489-1498
6. Gestel, T.V., Suykens, J.A.K., *et al.*: Financial Time Series Prediction using Least Squares Support Vector Machines within the Evidence Framework. *IEEE Transactions on Neural Network* **12** (2001) 809-820
7. Kowk, J.T.: The Evidence Framework Applied to Support Vector Machines. *IEEE Transactions on Neural Network* **11** (2000) 1162-1173
8. Law, M. H., Kowk, J. T.: Applying the Bayesian Evidence Framework to ν -support Vector Regression. *Proceedings of the 12th European Conference on Machine Learning*, Freiburg, Germany (2001) 312-323

9. Müller, K.R., Smola, A.J., et al.: Predicting Time Series with Support Vector Machines. In Proceedings of International Conference on Artificial Neural Networks. Berlin: Springer LNCS 1327 (1997) 999-1004
10. Mackay, D.J.C.: Probable Networks and Plausible Predictions-A Review of Practical Bayesian Methods for Supervised Neural Networks. *Network computation in Neural Systems* **6** (1995) 469-505
11. Gestel, T.V., Espinoza, M., *et al.*: A Bayesian Nonlinear Support Vector Machine Error Correction Model. *Journal of forecasting* **25** (2006) 77-100

Simulation of Time Series Prediction Based on Smooth Support Vector Regression

Chao Zhang, Pu Han, Guiji Tang, and Guori Ji

Department of Mechanical Engineering and The Key Laboratory of Condition Monitoring & Control for Power Plant Equipments of Education, North China Electricity Power University, 071003 Baoding, Hebei, China;
Department of Automation, North China Electricity Power University, 071003 Baoding, Hebei, China

Abstract. Time series analysis and prediction is an important means of dynamic system modelling, but traditional methods of time series prediction such as statistics and artificial neural network (ANN) are not fit for complicated non-linear system. Hence, a new method of support vector regression (SVR) was introduced to solve the prediction problem of complicated time series. For the purpose of reducing complexity of calculation, smooth arithmetic based on SVR was imported to forecast the time series of vibration data collected from turbine system. The result of simulation indicated that smooth support vector regression (SSVR) is obviously superior to ANN method on performance of prediction. Compared with SVR, SSVR has faster speed of convergence and higher fitting precision, which effectively extends the application of support vector machine.

Keywords: time series prediction, support vector machine, regression, smooth method, turbine.

1 Introduction

Time series prediction is a problem of forecasting the future trend based on historical data, which is an important way of dynamic data analysis and processing, and is already widely applied in many fields such as weather, finance, physic, electric power, control, and so on. Traditional technologies of time series prediction adopt methods of statistics and artificial neural network [1]. However, statistics method is not fit for complicated time series; artificial neural network has better ability of non-linear approximation, but insufficient or excessive training easily appears. Moreover, artificial neural network is sensitive to the initial value of connecting weights, and has poor ability of generalization. Therefore, it's necessary to search a better method to solve the problem of complicated system modeling.

Support Vector Machine (SVM) [2] is a new type of learning machine which is based on Statistical Learning Theory and Structural Risk Minimization principle. According to limited amount of training samples, SVM suggests a best

tradeoff between complexity of model (the learning precision for given samples) and learning ability (the ability to recognize any sample without error) to obtain best generalization. The application of SVM for problem of regression is named Support Vector Regression (SVR), which has good effect on time series prediction of complicated dynamic system.

Based on the research of SVR arithmetic [3-8], a smooth method of SVR was imported, which transforms the constrained quadratic optimization problem to an unconstrained convex quadratic optimization problem, and effectively reduces complexity of SVM's training. The simulation experiment was done to try to find a new way of turbine state prediction and fault diagnosis.

2 Time Series Prediction Based on SSVR

The essence of time series prediction is to seek a mapping $f : R^m \rightarrow R^n$ to approach the latent non-linear mechanism F for given data set, which can be used as predictor. According to the given samples based on probability $P(x, y)$, the basic problem of regression is to find out a function $f \in F$ (F is the set of functions) to minimize the desired risk function as follows:

$$R[f] = \int l(y - f(x))dP(x)$$

where $l(\cdot)$ is a loss function which denotes deviation between y and $f(x)$. Because $P(x, y)$ is usually unknown, we can not get $R[f]$ via upper equation.

According to Structural Risk Minimization principle:

$$R[f] \leq R_{emp} + R_{gen}$$

where R_{emp} is empirical risk, R_{gen} is a measurement of complexity of f . Therefore, the limit of $R[f]$ can be ascertained.

For given s groups of samples: $\{x_i, y_i\}, i = 1, 2, \dots, s, x_i \in R^m, y_i \in R$, a non-linear mapping Φ is used to map data x into a higher dimensional feature space G . In this feature space the mapping f is found out to linearly approach the given data. According to Statistical Learning Theory, function f has the form as follows:

$$\begin{aligned} f(x) &= (\omega, \phi(x)) + b \\ \phi : R^m &\rightarrow R, \omega \in G \end{aligned} \quad (1)$$

where (\cdot, \cdot) is inner product operation, the problem of function approximation equals to minimizing the functional as follows:

$$R_{reg} = R_{emp} + \lambda \|\omega\|^2 = \sum_{i=1}^s l(y_i - f(x)) + \frac{1}{2} \|\omega\|^2 \quad (2)$$

Selecting function of ε -insensitive as loss function:

$$|y - f(x)|_\varepsilon = \max\{0, |y - f(x)| - \varepsilon\} \quad (3)$$

then the empirical risk is:

$$R_{emp}^\varepsilon[f] = \frac{1}{s} \sum_{i=1}^s |y_i - f(x)|_\varepsilon \quad (4)$$

where ε is used to control the error of regression, presenting not to penalize the term whose deviation is less than ε . Doing so is helpful to prevent over-fitting and increase the robustness of regression. $R_{emp}^\varepsilon[f]$ minimizing Eq. (1) embodies the kernel of Statistical Learning Theory, which controls not only the training error, but also the complexity of model for the purpose of obtaining smaller desired risk to enhance the model's ability of generalization.

Now, the regression problem changes to optimal problem as follows:

$$\begin{aligned} \min J &= \frac{1}{2} \|\omega\|^2 + C \sum_{i=1}^s (\xi_i^* + \xi_i) \\ \text{s.t.} \quad &\begin{cases} y_i - (\omega, \Phi(x_i)) - b \leq \varepsilon + \xi_i^* \\ (\omega, \Phi(x_i)) + b - y_i \leq \varepsilon + \xi_i \\ \xi_i^*, \xi_i \geq 0 \end{cases} \end{aligned} \quad (5)$$

where x_i^*, x_i are slack terms to make the solution of Eq. (5) existing; C is regular factor which is a constant and is used to get tradeoff between complexity of model and learning ability. Normally, the bigger is the value of C , the higher is the degree of fitting.

SVR needs to resolve a constrained Quadratic Programming (QP) problem. Because standard arithmetic of SVR is complicated and may expend more training time when dealing with large numbers of samples, its application is limited in practice. Hence, many new arithmetic of SVR are put forward, such as Osuna's theorem [9], SVM^{light} arithmetic by Joachims [10], Sequential Minimal Optimization (SMO) arithmetic by Platt [11], Successive Over-Relaxation (SOR) arithmetic by Mangasarian [12], and so on. Recently, Lee Y J and Mangasarian imported a kind of Smooth Support Vector Machine (SSVM) arithmetic [13]. This arithmetic introduces smooth method to transform the constrained quadratic optimization problem to an unconstrained convex quadratic optimization problem, which effectively simplifies the arithmetic of SVR.

SSVR first transforms loss function to quadratic ε -insensitive loss function, then Eq. (5) can be expressed as:

$$\begin{aligned} \min J &= \frac{1}{2} \|\omega\|^2 + C \sum_{i=1}^s (\xi_i^*)^2 + C \sum_{i=1}^s \xi_i^2 \\ \text{s.t.} \quad &\begin{cases} y_i - (\omega, \Phi(x_i)) - b \leq \varepsilon + \xi_i^* \\ (\omega, \Phi(x_i)) + b - y_i \leq \varepsilon + \xi_i \\ \xi_i^*, \xi_i \geq 0 \end{cases} \end{aligned} \quad (6)$$

then uses $C/2$ to replace C and adds a term of $b^2/2$. The objective function of Eq. (6) becomes:

$$\min J = \frac{1}{2} (\|\omega\|^2 + b^2) + \frac{C}{2} \sum_{i=1}^s (\xi_i^*)^2 + \frac{C}{2} \sum_{i=1}^s (\xi_i)^2 \quad (7)$$

Introducing transformation:

$$\begin{aligned} z_i^* &= y_i - (\omega, \Phi(x)) - b - \varepsilon \\ z_i &= (\omega, \Phi(x)) + b - y_i - \varepsilon \end{aligned} \tag{8}$$

to define function $(u)_+ = \max\{u, 0\}$, then makes $\xi_i^* = (z_i^*)_+$, $\xi_i = (z_i)_+$. Eq. (7) changes to an unconstrained quadratic optimization problem as follows:

$$\min J = \frac{1}{2}(\|\omega\|^2 + b^2) + \frac{C}{2} \sum_{i=1}^s (z_i^*)_+^2 + \frac{C}{2} \sum_{i=1}^s (z_i)_+^2 \tag{9}$$

Eq. (9) is an unconstrained convex quadratic optimization problem whose solution is exclusive, but its objective function is not twice differentiable. Therefore, a strictly convex and infinitely differentiable smooth function $p(u, \alpha) = \frac{1}{\alpha} \ln(1 + e^{\alpha u})$, $\alpha > 0$ is defined. Using $p(u, \alpha)$ to replace $(u)_+$, we can get the objective function of smooth support vector regression:

$$\min J = \frac{1}{2}(\|\omega\|^2 + b^2) + \frac{C}{2} \sum_{i=1}^s p(z_i^*, \alpha)^2 + \frac{C}{2} \sum_{i=1}^s p(z_i, \alpha)^2 \tag{10}$$

It can be proved that $p(u, \alpha)$ is able to approach $(u)_+$ when $\alpha = 10$; and the solution of Eq. (10) converge to the solution of original problem when $\alpha \rightarrow \infty$.

3 Simulation Experiment

3.1 Comparison of Data Fitting Performance

First we compared SSVR with ANN on performances of data fitting through a simple example. RBF artificial neural network regression and SSVR were used to approach a group of data. The results are showed in Fig. 1.

In a) of Fig. 1, the fitting curve of RBF ANN regression reflects the time trend of the given data set on the whole, but in outside data point, the fitting curve is not smooth and is unable to reflect the latent mechanism of original time series. The main reason is that RBF ANN has poor ability of generalization. ANN must have appropriate network structure, and unilaterally pursuing least training errors may easily lead to excessive fitting and bad ability of generalization. At present, research on ANN structure has no perfect achievements, and some parameters of network are selected by experience, which restricts its ability of generalization.

b) of Fig. 1 indicates that SSVR arithmetic has good effect on data fitting for given time series. Compared with a) of Fig. 1, SSVR's fitting curve in outside data point is improved effectively. On the whole, the fitting curve is very smooth and reflects the intrinsic mechanism of original time series, because SSVR arithmetic compromisingly chooses training errors and complexity of regression through controlling support vectors and other parameters, which avoids the emergence of excessive fitting and gets better ability of generalization.

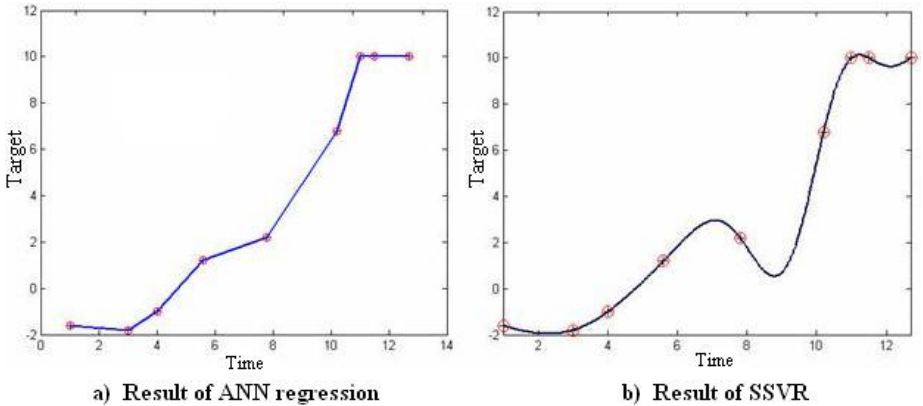


Fig. 1. Comparison of fitting ability between ANN and SSVR

3.2 Comparison of Training Speed

In the test of training speed, Housing data set was used (506 samples in total). And RBF kernel function $K(x_i, x_j) = \exp(-|x_i - x_j|/\sigma^2)$ was selected, and other parameters were set according to reference [14]: $C = 1000, \varepsilon = 30, \sigma = 1.5$. The computer has 1.5GHz CPU and 256M memory, and the operation system is Windows 2000 Professional SP4. The results of test are showed in Table 1.

Table 1. Training Time of different regression arithmetic for Boston Housing data set

(Training samples, Test samples)	Time of regression (s)		
	SSVR	SVR	ANN
(160, 346)	0.10	0.11	0.10
(240, 266)	0.16	0.20	0.14
(320, 186)	0.24	0.35	0.23
(400, 106)	0.31	0.55	0.30

From Table 1 we can conclude that the three methods of regression have nearly same training time on condition of small quantity of samples, and SSVR arithmetic has no obvious advantage. But ANN arithmetic has bigger errors and lower fitting precision. On condition of large quantity of samples, the operating time of SSVR is nearly linear to the quantity of samples. The convergence speed of SVR is slower and the memory needed is square to the quantity of samples.

3.3 Simulation of Prediction

In the simulation experiment of prediction, the historical data of turbine was used (130 samples in total). First we deal with the original data by calculating the percentage of basic frequency amplitude in pass band to construct the time

series. 60 samples were used as training samples, and the last 70 samples were used as prediction samples. The training samples were delayed 1 to 5 time units to constitute the input vector as follow:

$$X = \{u(k - 1), u(k - 2), u(k - 3), u(k - 4), u(k - 5)\}$$

SSVR arithmetic used RBF kernel function, and other parameters were set as: $C = 2000, \varepsilon = 10^{-6}, \sigma = 1.3$. The results of simulation are showed in Fig. 2 and Fig. 3.

a) of Fig. 2 and Fig. 3 reflect that the fitting curve of RBF ANN is not smooth and can not embody the latent mechanism of the time series. Besides, RBF ANN has bigger errors in prediction than SSVR and influences the effect of final prediction. b) of Fig.2 and Fig. 3 shows that SSVR arithmetic has better

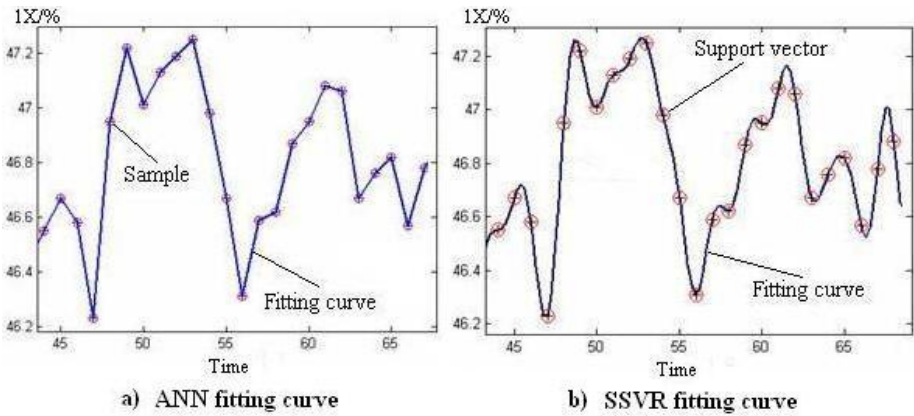


Fig. 2. Fitting curve of SSVR and ANN

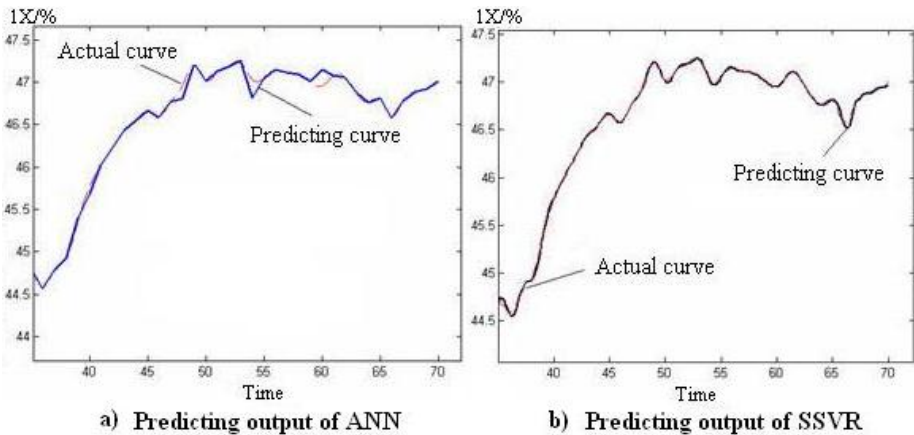


Fig. 3. Predicting curve of SSVR and ANN

stability. Its output curves are very smooth, and the support vectors usually occur on rate of 100%, which ensures the prediction curve to approach the actual output. Therefore, SSVR arithmetic is very fit for the situation of complicated time series prediction.

4 Conclusion

The theoretical and experimental analysis suggests that SSVR arithmetic improves the standard arithmetic of SVR. SSVR arithmetic not only inherits the good ability of generalization of SVR, but also avoids the disadvantage of ANN, and has higher precision of prediction. The reason is that SSVR arithmetic is based on VC theory and Structural Risk Minimization principle, and the design of the machine randomly extracts subset of training set as support vectors, which represent the steady characteristic of whole data set. SSVR arithmetic enhances the speed of regression by reducing computing complexity and lowers the need of memory. Therefore, SSVR arithmetic is very fit for the problem of large samples. But SSVR arithmetic still needs to resolve quadratic programming problem, and has no theoretical proof for the selection of regular factor and kernel function, which is still an open problem for research.

References

1. Gencay, R., Liu, T.: Nonlinear Modelling and Prediction with Feedforward and Recurrent Networks. *Physica D* **108** (1997) 119-134
2. Vapnik, V.: *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin (1995)
3. Xu, D., Yang, J., Zhai, Y., Han, P.: An Online Identification Algorithm Based on SVR's Limited Memory and Its Application. *Dynamical Engineering* **25** (5) (2005) 680-684
4. Yang, J., Zhai, Y., Wang, D., Xu, D.: The Series Prediction Based on Support Vector Regression. *Proceedings of the CSEE* **25** (17) (2005) 110-114
5. Sun, D., Wu, J., Xiao, J.: The Application of SVR to Prediction of Chaotic Time Series. *Journal of System Simulation*, **16** (3) (2004) 519-521
6. Du, S., Wu, T.: Support Vector Machines for Regression. *Journal of System Simulation* **15** (11) (2003) 1580-1585
7. Qu, W., Fan, G., Yang, B.: Research on Complicated Time Series Prediction Based on Support Vector Machines. *Computer Engineering*, **31** (23) (2005) 1-3
8. Tu, J., Cai, J.: Smooth Support Vector Regression. *Journal of Hubei University (Natural Science)* **28** (1)(2006) 28-31
9. Osuna, E., Freund, R., Girosi, F.: An Improved Training Algorithm for Support Vector Machines. *Proceedings of the 1997 IEEE Workshop on Neural Networks and Signal Processing*. New York: IEEE Press (1997) 276-285
10. Joachims, T.: *Making large-scale SVM learning practical. Advances in Kernel Methods - Support Vector Learning*.: MIT Press (1990) 169-184

11. Platt, J.: Fast training of support vector machines using sequential minimal optimization. *Advances in Kernel Methods - Support Vector Learning*. Cambridge: MIT Press (1999) 185-208
12. Mangasarian, O.L., Musicant D R: Successive overrelaxation for support vector machines. *IEEE Transactions on Neural Networks*, **10 (5)** (1999) 1032-1037
13. Lee, Y.J., Mangasarian, O.L.: SSVM: a Smooth Support Vector Machine for Classification. *Computational Optimization and Applications* **20** (2001) 5-22
14. Quan, Y., Yang, J., Yao, L., Ye, C.: Successive Overrelaxation for Support Vector Regression. *Journal of Software* **15 (2)** (2004) 200-206

Extension Neural Network Based on Immune Algorithm for Fault Diagnosis

Changcheng Xiang^{1,2}, Xiyue Huang², Gang Zhao^{2,3}, and Zuyuan Yang²

¹ Department of mathematics, Hubei Institute for Nationalities, Hubei, China, 445000

² Automation College, Chongqing University, Chongqing, China, 400030

³ Chongqing Agent of missile force, PLA

xcc7426681@126.com, xyhuang@cqu.edu.cn

zhaogang11@163.com, zy7704@163.com

Abstract. In this paper, the extension neural network (ENN) is proposed. To tune the weights of the ENN for achieving good clustering performance, the immune algorithm (IA) is applied to learning the ENN's weights, which is replaced the BP algorithm. The affinity degree between the antibody and the antigen is measured by extension distance (ED), which is modified to the conjunction function (CF) in Extensions. The learning speed of the proposed ENN is shown to be faster than the traditional neural networks and other fuzzy classification methods. Moreover, the immune learning algorithm has been proved to have high accuracy and less memory consumption. Experimental results from two different examples verify the effectiveness and applicability of the proposed work.

1 Introduction

Neural networks model brain-style information processing at various abstraction levels. Neural network technologies are parallel systems used for solving regression and classification problems, it has been reported superior in numerous application areas, such as pattern recognition, full-text and image analysis, financial data analysis, process analysis and modeling as well as monitoring and control, and fault diagnosis [1,2,3,4]. They can estimate a relation function between the inputs and outputs from a learning process, and also can discover the mapping form feature space into space of classes. Classification or cluster analysis is one of the most important applications of neural networks.

With the development of many new knowledge and the demand of many application, many hybrid neural networks are proposed, such as Wavelet Neural Network (WNN)[5], Chaos Neural Network (CNN)[6], Fuzzy Neural Network (FNN)[7], Immune Neural Network (INN)[8] and etc. These hybrid neural network is combination of the neural network and the corresponding theory.

In our world, there are some classification problems whose features are defined in a range. For example, water can be defined as a cluster of temperature from 0 degree to 100 degree and the permitted operation voltages of a specified motor may be between 100 and 120 V. For these problems, it is not easy to implement an appropriate classification method using current neural networks. M.H. Wang

proposed the ENN[9] which is a combination of the neural network and the extension theory[10] to solve these problems. The extension theory proves a novel distance measurement for classification processes, and the neural network can embed the salient features of parallel computation power and learning capability. In other words, the ENN permits classification of problems, which have range features, supervised learning, or continuous input and discrete output. But the learning algorithm is need more learning times.

In this paper we have proposed a immune algorithm-based learning mechanism for adjusting the connections range between the input node and the output node of ENN. This method differs from the existing approaches. In the second part, the ENN is proposed, and in the third part, the learning mechanism based on immune algorithm is given. Finally, the experiments results is proved the algorithm is effective.

2 Extension Neural Network

Extension theory[10] was originally invented by Cai to solve contradictions and incompatibility problems in 1983, the following is introduced the basic of extension.

2.1 Basic of Extension Theory

Definition 1 (Matter-element). *Defining the name of a matter by N ; one of the characteristics of the matter by c ; and the value of c by v ; we can use an ordered ternary $R = [N, c, v]$ as the fundamental element to state a matter and call it a matter-element in extension theory.*

If the value of the characteristic has a classical domain or a range, we define the matter-element for the classical domain as follows: $R = [N, c, <w^L, w^U >]$, where w^L and w^U are the lower bound and upper bound of classical domains, respectively.

If $R = [N, C, V]$ is a multi-dimensional matter-element, $C = \{c_1, c_2, \dots, c_n\}$ is a set of n characteristic vector and $V = \{v_1, v_2, \dots, v_n\}$ a value vector of characteristic.

Definition 2 (Extension Set). *If U is a space of objects and x a generic element of U ; k is a relational function from U to $(-\infty, \infty)$; T is a matter transform to the element of U ; then an extension set A in U is defined as a set of ternary pairs:*

$$\tilde{A}(T) = \{(u, y, y') | u \in U, y = k(u) \in (-\infty, \infty), y' = k(Tu) \in (-\infty, \infty)\}, \quad (1)$$

$y = k(u)$ is the conjunction function of extension set $\tilde{A}(T)$

Definition 3 (Conjunction Function). *Let $X_0 = \langle a, b \rangle, X = \langle c, d \rangle$ and $x_0 \in X_0$, The conjunction function is defined as follows:*

$$K(x) = \frac{\rho(x, X_0)}{D(x, X_0, X)}, \quad (2)$$

where

$$\rho(x, X_0) = \left| x - \frac{a+b}{2} \right| - \frac{b-a}{2} \tag{3}$$

and

$$D(x, X_0, X) = \begin{cases} -1 & x \in X_0 \\ \rho(x, X) - \rho(x, X_0) & \text{others} \end{cases} \tag{4}$$

The conjunction function can be used to calculate the membership degree between x and X_0 . When $K(x) \geq 0$, it indicates and describes the degrees to which x belongs to X_0 . When it $K(x) < 0$ describes the degree to which x does not belong to X_0 .

2.2 Extension Neural Network

The proposed ENN is a combination of the neural network and the extension theory. The extension theory proves a novel distance measurement for classification processes, and the neural network can embed the salient features of parallel computation power and learning capability.

The structure of the ENN is depicted in Fig. 1.

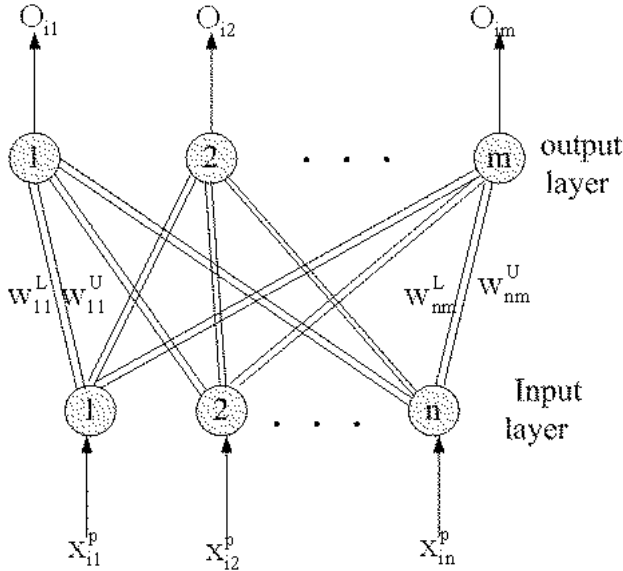


Fig. 1. The structure of extension neural network(ENN)

It comprises both the input layer and the output layer. The nodes in the input layer receive an input feature pattern and use a set of weighted parameters to generate an image of the input pattern. In this network, there are two connection values (weights) between input nodes and output nodes; one connection represents the lower bound for this classical domain of the features, and the other

connection represents the upper bound. The connections between the j -th input node and the k -th output node are w_{kj}^L and w_{kj}^U . This image is further enhanced in the process characterized by the output layer. Only one output node in the output layer remains active to indicate a classification of the input pattern. The operation mode of the proposed ENN can be separated into the learning phase and the operation phase. The learning phase is used by immune algorithm and discussed in the next section.

3 Immune Algorithm-Based Learning Mechanism

Recently, artificial immune systems (AIS) have captured the attention of various researchers due to their ability to perform tasks such as learning and memory acquisition. This approach is suitable for solving multi-modal and combinatorial optimization problems. AIS have been applied in problem solving. Therefore, AIS are not only related to the creation of abstraction or metaphorical models of the biological immune system, they also include theoretical immunology models being applied to tasks such as optimization, control, and autonomous robot navigation.

In this paper, the immune algorithm is applied to train the weight of ENN.

Antibody: The weight of ENN is described by the matter-element as following.

$$R_k = \begin{bmatrix} N_k & c_1 & (w_{k1}^L, w_{k1}^U) \\ & c_2 & (w_{k2}^L, w_{k2}^U) \\ & \dots & \dots \\ & c_n & (w_{kn}^L, w_{kn}^U) \end{bmatrix} \tag{5}$$

$$antibody = \begin{bmatrix} w_{11}^L & w_{12}^L & \dots & w_{k1}^L & w_{k2}^L & \dots & w_{n1}^L & w_{n2}^L & \dots & w_{mn}^L \\ w_{11}^U & w_{12}^U & \dots & w_{k1}^U & w_{k2}^U & \dots & w_{n1}^U & w_{n2}^U & \dots & w_{nm}^U \end{bmatrix} \tag{6}$$

where w_{ki}^L and w_{ki}^U is the connection weight lower bound and upper bound between the i -th input node and the k -th output node. $i=1,2,\dots,n, k=1,2,\dots,m$.

Antigen: The input samples is the antigen. the samples matter-element is described

$$RI_i = \begin{bmatrix} M_i & c_1 & v_{i1} \\ & c_2 & v_{i2} \\ & \dots & \dots \\ & c_n & v_{in} \end{bmatrix} \tag{7}$$

$$antigen_i = [v_{i1} \ v_{i2} \ , \dots \ , v_{in}] \tag{8}$$

Fitness function: The fitness function have overall influence to convergence of the learning algorithm, and is the key factor to obtain the optimization fitness value. An important characteristics of ENN is that the smaller the covariance between the output value and the input, the better the performance of the ENN.

Let the samples be $\{X_1, T_1\}, \{X_2, T_2\}, \dots, \{X_Q, T_Q\}$, Q is the total number of the pattern where X_i is input samples, T_i is output of the sample.

The fitness is $f = \frac{n_{success}}{n_{total}}$, where $n_{success}$ is correct numbers of output n_{total} is the total numbers of all samples. If the sum function of error square

$$Error = \frac{1}{2} \sum_{j=1}^Q \sum_{i=1}^m (t_{ij} - ED_{ij})^2 \tag{9}$$

is achieved the require, the learning is stop. where t_{ij} is the j -th expectation output of the i -th pattern ED_{ij} is the j -th really output of the i -th pattern. Our learning algorithm of ENN is based on the fundamental procedures of IA and can be summarized as follows:

Step 1: The j -th center of the i -th pattern is computed by Eq.(10).

$$z_{kj} = \frac{\sum_{i=1}^{N_k} v_{kj}^j}{N_k}, \tag{10}$$

where N_k is the i -th pattern samples numbers.

Step 2: Creating initial antibody population randomly, i.e. a population of weights of ENN which are randomly specified by Eq.(11)

Step 3: Evaluating affinity vector between each antibody and each antigen using Eq.(4).

$$Aff(Ag_i, Ab) = \{Aff_i(1), Aff_i(2), \dots, Aff_i(n)\}, \tag{11}$$

where

$$Aff_i(k) = ED(Ag_i, Ab(k)) = \sum_{j=1}^m b_j \left[\frac{|v_{ij} - z_{ij}| - \frac{1}{2}(w_{kj}^U - w_{kj}^L)}{|\frac{1}{2}(w_{kj}^U - w_{kj}^L)|} \right] \tag{12}$$

$k = 1, 2, \dots, n$

Step 4: ranking the affinity vector. if the output of $Max\{Aff_i(k), k = 1, 2, \dots, n\}$ is the same as the exsection output of the input samples, the output of the corresponding antigen is reserved and the counter $N_{success}$ is added one time. Until all samples is trained completely, The fitness of the antibody $f = \frac{n_{success}}{n_{total}}$ is computed.

Step 5: Ranking the fitness value of all antibody and cloning the antibody according the value.

$$N_c(P) = INT[N - \frac{f_p}{\sum f_p} \bullet N], \tag{13}$$

where $p = 1, 2, \dots$, is the antibody symbols $INT[]$ is integer.

Step 6: All antibodies using super mating operator as follows.

$$w_{kj}^L = w_{kj}^L + \xi \left(\frac{1}{N_k} \sum_{i=1}^{N_k} v_{ij} - w_{kj}^L \right), w_{kj}^U = w_{kj}^U + \xi \left(\frac{1}{N_k} \sum_{i=1}^{N_k} v_{ij} - w_{kj}^U \right) \tag{14}$$

where $\xi \in [0, 1]$ is a random number, N_k is the k -th pattern samples numbers, $k = 1, 2, \dots, m; j = 1, 2, \dots, n$.

Step 7: Selecting antibodies of the fitness value to reserve and Generating new antibodies radomes. If termination conditions are met go to step Step 8, otherwise, go to step 3.

Step 8: Stop, return the best antibody and translate it into the weights of ENN.

Table 1. Classic input and out samples

G.N	$< 0.4f$	$0.4 - 0.5f$	$1f$	$2f$	$3f$	$> 3f$	F.T
1	3.35	46.6	12.15	1.94	2.3	1.67	F1
2	4.43	51	11.02	3.02	1.3	2.43	F1
3	3.29	50	11.61	1.24	0.9	1.3	F1
4	5.72	46.3	12.31	3.62	1.5	0.59	F1
5	6.32	45.8	15.23	3.56	2.3	3.19	F1
6	1.51	3.29	2.92	6.59	2.5	2.54	F2
7	2.43	1.19	54.49	4.64	0.8	1.78	F2
8	0.54	2.92	48.82	6.64	3.9	1.51	F2
9	0.81	1.73	52	6.43	3.6	1.89	F2
10	1.24	1.35	49.79	4.64	1.0	2.27	F2
11	1.78	1.46	22.46	23.8	19	8.59	F3
12	0.92	1.24	30.38	22	16	5.67	F3
13	0.65	2.11	21.98	26.2	18	11.1	F3
14	1.13	0.92	24.46	22.3	15	15.8	F3
15	0.92	1.40	26.08	26	20	11.4	F3

G.T, generator number; F.T, Fault types; F1, Oil-resonance fault; F2, imbalance fault; F3, misalignment fault

4 Experiments

To demonstrate the effectiveness of the proposed IA-ENN in fault diagnosis method, 15 sets of field-test data from steam-turbine generator sets in China [11][12] were tested (the data are shown in Table 1). The input data include the six amplitude values of the vibrational spectrum, where f is the frequency of the generator rotor. It is clear that vibration diagnosis in steam-turbine generator sets is a most complicated and nonlinear classification problem. let $population = 50$, $iter = 500$, $Error = 0.001$, $N = 10, \xi = 0.05$, let IA-ENN train the weight be

$$Ab(F1) = \begin{bmatrix} 3.34 & 45.73 & , & 11.01, & 1.25 & 0.88 & 0.60 \\ 6.33 & 51.02, & 15.24, & 3.63 & 2.29 & 3.15 \end{bmatrix}$$

$$Ab(F2) = \begin{bmatrix} 0.60 & 1.19 & , & 48.82 & 4.61 & 0.83 & 1.48 \\ 2.41 & 3.26 & , & 53.00 & 6.53 & 3.79 & 2.56 \end{bmatrix}$$

$$Ab(F3) = \begin{bmatrix} 0.70 & 1.89 & 21.88 & 22.05 & 14.97 & 5.76 \\ 1.75 & 2.05 & 30.29 & 26.13 & 19.02 & 14.75 \end{bmatrix}$$

Table 2. Classic input and out samples

G.N	< 0.4 <i>f</i>	0.4 – 0.5 <i>f</i>	1 <i>f</i>	2 <i>f</i>	3 <i>f</i>	> 3 <i>f</i>	EDF1	EDF2	EDF3	F.T
1	3.01	40.6	11.15	2.94	2.3	1.07	0.756	2.536	16.708	F1
2	0.50	2.22	48.82	6.54	2.9	1.61	1.888	0.694	1.641	F2
3	3.29	47	11.05	1.44	0.8	1.01	0.713	2.825	19.272	F1
4	2.43	1.18	53.40	4.02	0.35	1.68	1.863	0.774	1.584	F2
5	1.03	0.82	24.56	21.33	14.08	14.8	2.731	2.509	0.932	F3
6	0.89	1.04	29.38	22	15.8	5.47	2.668	1.953	0.851	F3
7	0.79	1.09	28.38	21.98	15.6	5.07	2.635	1.942	0.836	F3

Table 3. Different learning methods of the same sample

Classifiers	AWN[11]	FNN[13]	BPNN[12]	I-BPNN[12]	IA-ENN
Structure	6-13-3	6-16-3	6-16-3	6-16-3	6-3
epochs(iter)	900	500	2561	979	127
time(s)	*	*	482	182	89

The precision is 0.03

then the weight of ENN trained the samples by immune algorithm is $Ab = [Ab(F1), Ab(F1), Ab(F1)]$, then the test samples is applied by ENN, the results is showed Tab.2. To compare the diagnosis performance, the epochs(iter) and the time consumption of several different neural network are shown in Tab.3. From the table, the iter and the time consumption is less than others methods.

5 Conclusion

A method for fault diagnosis of steam-turbine generator sets using wavelet neural networks and particle swarm optimization has been proposed. Compared with traditional BP neural networks and other fuzzy classification methods, it permits an adaptive process for significant and new information, and gives shorter learning times. From the tested examples, the proposed IA-WNN has been proved to have the advantage of less learning time, and less time consumption.

Acknowledgement

This work was supported by the National Natural Science Foundation of China under grant no. 60443006 and is partially supported by CSTC Grant, 2006BA6016.

References

1. Raivio, K., Simula, O. and Henriksson, J.: Improving Decision Feedback Equalizer Performance using Neural Networks. *Electronics Letters* **27**(23) (1991) 2151 - 2153
2. Simula, O. and Kangas, J.: Process Monitoring and Visualization using Self-organizing Maps. *Neural Networks for Chemical Engineers*, volume 6 of *Computer Aided Chemical Engineering*, chapter 14, Elsevier, Amsterdam (1995)

3. Haitao Tang and Simula, O.: The Optimal Utilization of Multi-service Scp. In Intelligent Networks and New Technologies (1996) 175 - 188
4. Tryba, V. and Goser, K.: Self-Organizing Feature Maps for Process Control in Chemistry. In Artificial Neural Networks, Amsterdam, Netherlands, North-Holland, (1991) 847 - 852
5. Wong, K., Leung, A.C.: On-line Successive Synthesis of Wavelet Networks. Neural Process. Lett. **7** (1998) 91-100
6. Inoue, M. and Nishi, Y.: Dynamical Behavior of A Chaos Neural Network of An Associative Schema Model . Progress of Theoretical Physics **5** (1995) 837-850
7. Oh, S.K., Pedrycz, W., Park, H.S.: Hybrid Identification in Fuzzy-neural Networks. Fuzzy Set. Syst. **138**(2) (2003) 399-426
8. Hou, T.H., Su, C.H., Zhi, H.: Chang using Neural Networks and Immune Algorithms to Find The Optimal Parameters for An IC Wire Bonding Process. Expert Systems with Applications (2006) ARTICLE IN PRESS
9. Wang, M. H.: Extension Neural Network and Its Applications. Neural Network **16** (2003) 779-784
10. Cai, W.: The Extension Set and Non-compatible Problems (in Chinese). J. Scient. Explore **1** (1983) 83-97
11. Li, H., Sun, C.X., Liao, R.J., Chen, W.G., Hu, X.S.: Improved BP Algorithm in Vibration Fault Diagnosis of Steam Turbine-generator Set. Journal of Chongqing University, **22**(1999) 47-52
12. Li, H., Sun, C.X., Hu, X.S., Yue, G., Tang, N.F., Wang, K.: Study of Method on Adaptive wavelets Network for Vibration Fault Diagnosis of Steam Turbine-generation Set. Transactions of China Electrotechnical Society **15** (2000) 52-57
13. Zhan B.D., Sun C.X., Jian, O., Zhou, Y.C.: A Fuzzy Clustering Method for Turbo Generator Vibration Faulty Diagnosis. Turbine Technology **44**(5) (2002) 289-291

A New Fault Detection and Diagnosis Method for Oil Pipeline Based on Rough Set and Neural Network

Liu Jinhai, Zhang Huaguang, Feng Jian, and Yue Heng

School of Information Science and Engineering, Northeastern University,
Shenyang, 110004, P.R. China
jh_lau@126.com, hg_zhang@21cn.com, yueheng1968@163.com

Abstract. This paper proposed a new fault-detection method based on the combination of Rough Set (RS) and Artificial Neural Network (ANN), called hybrid fault-detection method based on RS and ANN (HFDMSNN), which uses RS to reduce parameters of a pipeline system and then uses ANN (three-layer neural network) to form a detection model. This method could detect fault of pipeline not only in stationary status but also in non-stationary status. The efficiency of the HFDMSNN in detecting fault in real pipeline system is evaluated by an experiment in a long product oil pipeline in Shandong China. From the results, it is observed that the proposed HFDMSNN is able to identify the status of complex pipeline effectively.

1 Introduction

It is one of important problems to detect leaks on oil transmission pipelines [1]. The usual way to detect leaks real-timely is based on stationary status calculations, such as mass balance method and negative wave method. Traditionally, the first and also the most important problem for negative wave diagnosis and mass balance is that they are effectively only when the pipeline running in steady status. It is difficulty to detect fault of oil transmission pipeline in non-stationary status that only through collecting pressure, flow, and temperature signals. However, the status of a pipeline, for example a product oil pipeline, is affected by not only pressure, instantaneous flow, temperature, but also density, status of pumps, turndown ratio, diameter of pipeline, etc., and there also exist interactional relations among parameters. These methods of stationary status play a little role for fault diagnose and often occur missing alarms and false alarms especially on intricate oil pipelines which have some intermediate stations, intermediate pumps and other factors.

To detect fault on complicated oil duct, a new fault-detection method is proposed in this paper, which based on the combination of Rough Set and Artificial Neural Network, called HFDMSNN. In HFDMSNN, besides pressure, flow, and temperature signal, more signals are processed, like status of pumps, turndown ratio of valve, diameter of pipeline and so on.

In the paper, after introduction, the HFDMSNN is described in Section 2 which be explained in two phase, followed by an experiment of the HFDMSNN in Section 3, the experimental results will be given with the conclusions in the last section.

2 Description of HFDMRSNN

The common advantage of RS and ANN is that they do not need any additional information about data like probability in statistics or grade of membership in fuzzy-set theory. RS has proved to be very effective in many practical applications. However, in RS theory, the deterministic mechanism for the description of error is very simple. Therefore, the rules generated by RS are often unstable and have low classification accuracies. So RS cannot predict loads with high accuracy. ANN is considered the most powerful classifier for low classification-error rates and robustness to noise. But ANN has two obvious shortcomings when applied to large data-problems. The knowledge of ANN is buried in their structures and weights. It is often difficult to extract rules from a trained ANN. So HFDMRSNN is presented. The combination of RS and ANN is very natural for their complementary features [2]. The RS approach as a pre-processing tool for the ANN. RS theory provides useful techniques to reduce irrelevant and redundant attributes from a large database with a lot of attributes. ANN has the ability to approach any complex functions and possess as a good robustness to noise. The HFDMRSNN has two main phases: the training phase and the defect-detection phase. The flow diagram of HMOPRSANN is shown in Fig. 1. Using RS reduce parameters of a pipeline system and using ANN produce a detection model, the training phase completes the foundation of detection model. In defect-detection phase, real-time data is inputted into detection model that founded by the training phase, the status of the pipeline system is obtained.

2.1 Training Phase

2.1.1 Attributes Reduction

The objective of this stage is to find a minimal subset of related attributes that preserves the classification of the original attributes of the decision table. This can be a useful method for identification of the most significant and important variables in a given neural network. Consequently, we can succeed in managing the architecture of the neural network, and provide a reasonable solution for the explanation problem. The rough set theory provides the tool to deal with this issue. The concept of reduct can be successfully used to achieve this objective (Pawlak, 1991). There are many reducts that can be discovered at the analysis phase of the decision table. We are mostly interested in the best reduct. The general problem of finding all the reducts is NP-hard, but in most cases it is not necessary to find all the reducts. The criteria taken to solve this problem is that the best reduct is the one with the minimum number of attributes, and that if there are two or more reducts with the same number of attributes, then the reduct with the least number of combination of its attributes is selected (Hu & Cercone, 1995). The core is another concept of the rough set theory, and represents the intersection of all the generated reducts from the decision table. Hence the algorithm of deriving the best reduct was built, based on computing the core from the decision table (Yahia et al., 1998).

The process of attributes reduction is divided into three steps, as is shown in Fig. 2.

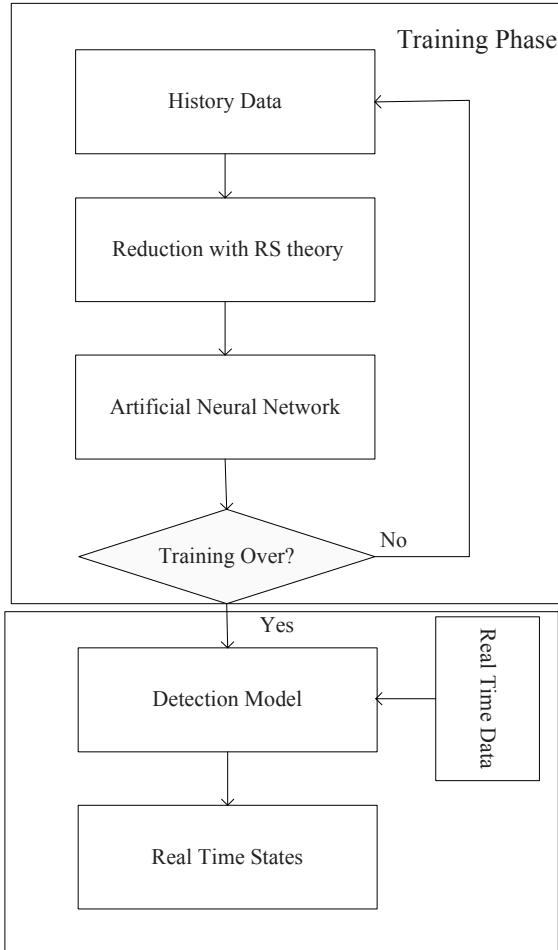


Fig. 1. The flow diagram of HMOPRSANN

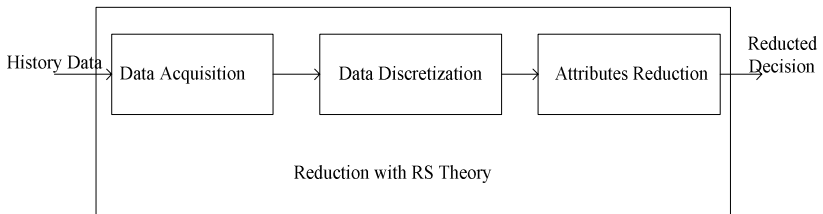


Fig. 2. The process of attributes reduction

2.1.1.1 Acquisition of Data. The first step is to receive the history data of the pipeline as a set of training example cases in the form of a matrix, in which columns are represented by parameters that acquired from sensors and rows are represented by values of parameters.

2.1.1.2 Discretization of Data. The input history data is continuous, so the first step is to convert the continuous data to discrete data, so that they can be dealt with by RS. We can compute the set of candidate cuts to disperse the history data [4]. There are no such data that the input is same, but the output is far away from each other. Therefore, the decision table will be consistent. If it is inconsistent, the reason may be that the partition is not proper.

2.1.1.3 Reduction of Attributes. In this paper, we use reduct algorithm based on tree expression, whose complexity is linear to the number of instances [7]. This algorithm is the better method to reduct information system (IS) having large records.

For an IS $\{U, C \cup D\}$, where U is the universe, C is the set of condition attributes and D is the set of decision attributes, and an attributes sequence $S(C) = a_1 \prec a_2 \prec \dots \prec a_{m-1} \prec a_m$. Let $B \subseteq C$, $S(B)$ is attributes sequence on B , a_{i+1} is the next element of a_i on $S(B)$ and a_{k-1} is the previous element of a_k on $S(B)$, the reduct algorithm based on tree expression is as follows:

Let $B=C, S(B)=S(C), RED=\emptyset$.

1. Build completely attribute-value tree $T(a_i, S(B), U)$, a_i is the first element of $S(B)$;
2. Delete all dead children tree on $T(a_i, S(B), U)$, then get the close attribute-value tree $T_c(a_i, S(B), U)$, on which has attribute a_k having the most subscript value;
3. Let $RED=RED \cup \{a_k\}$. After delete all elements behind a_k (that is the right of a_k), move a_k to the head of $S(B)$ (the most left of $S(B)$), then we have $B=\{a_i, a_{i+1}, \dots, a_k\}, S(B) = a_k \prec a_i \prec a_{i+1} \prec \dots \prec a_{k-1}$;
4. Repeat (1)-(3) before $RED=B$.
 RED is the reduct of $\{U, C \cup D\}$ on $S(C)$.

2.1.2 Artificial Neural Network

An important network design issue is the selection and implementation of the network configuration. In this paper, as is shown in Fig. 3, a three-layer multilayer perception (MLP) made up of an input layer, a hidden layer, and an output layer including four status is chosen, which is the smallest achievable structure is applied to reduce the developing expense of the fault detector neural network, while maintaining the desired level of accuracy and robustness of the fault detector.

The procedures of the back-propagation training algorithm can be described as follows. First, let the weights and threshold levels are randomly drawn from a uniform distribution inside a range [8].

$$\left[-\frac{2.4}{F_i}, +\frac{2.4}{F_i}\right], \tag{1}$$

where F_i denotes the total number of inputs of the i th neuron in the network. Then, first calculate the actual outputs of the neurons in the hidden layer by the equation:

$$y_j(n) = \varphi[\sum_{i=1}^p x_i(n) \times w_{ij}(n) - \theta_j], \tag{2}$$

where $\varphi(\bullet)$ denotes the activation function, x_i denotes the i th input, w_{ij} denotes the weight between the i th input and the j th hidden neuron, θ_j is the j th threshold, and p is the number of input of j th neuron in the hidden layer.

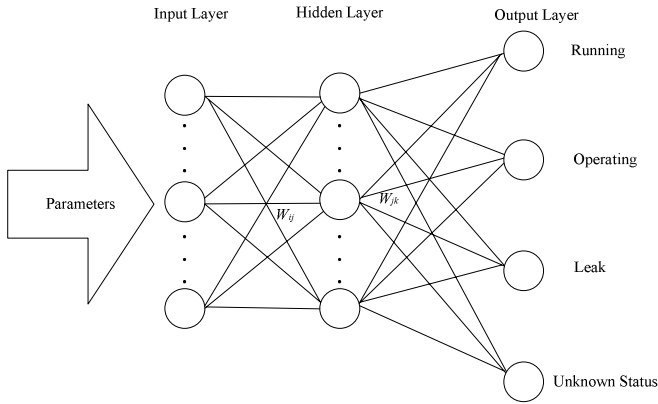


Fig. 3. Three-layer neural network

Second, calculate the actual output of the neurons in the output layer using the equation:

$$y_k(n) = \varphi[\sum_{j=1}^q x_{jk}(n) \times w_{jk}(n) - \theta_k], \tag{3}$$

where w_{jk} denotes the weight between the j th hidden layer and the k th output, and q is the number of inputs of the k th neuron in the output layer.

Next, update the weights in the output layer using the equation:

$$w_{jk}(n+1) = w_{jk}(n) + \eta \cdot \delta_k(n) \cdot y_j(n), \tag{4}$$

where η denotes the learning rate,

$$\delta_k(n) = y_k(n) \cdot [1 - y_k(n)] \cdot e_k(n) \tag{5}$$

$$e_k(n) = d_k(n) - y_k(n), \tag{6}$$

where $d_k(n)$ denotes the actual output, and update the error gradient for the neurons in the hidden layer using the equation:

$$w_{ij}(n+1) = w_{ij}(n) + \eta \cdot \delta_j(n) \cdot x_i(n), \tag{7}$$

where

$$\delta_j(n) = y_j(n) \cdot [1 - y_j(n)] \cdot \left[\sum_{k=1}^r \delta_k(n) \cdot w_{jk}(n) \right]. \tag{8}$$

The above procedures will stop until the error criterion or the maximum iterations are satisfied.

2.2 Detect-Detection Phase

In training phase, all weights of three-layer neural network have been determined, and the model of defect-detection has been founded. In detect-detection period, real-time data is inputted into the model, the status then be given.

3 Experiments

3.1 Description of Experiments

This HFDMSNN has been tested using history data of a product oil transport pipeline in Shandong China. The tested pipeline is 780km long, which is divided into 9 segments via 10 intermediate stations and a dispatching center, as is shown in Fig. 4. Parameters including pressure, temperature, flow, turndown ratio of valve, status of pump, etc., is acquired from the sensors in ten intermediate stations and in dispatching center. All of the parameters is inputted into server computer in dispatching center, then the server computer must computes the data and gives system status immediately (less than 1 second normally).

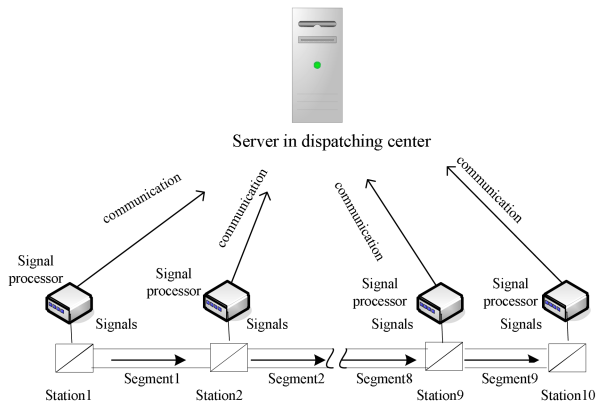


Fig. 4. The schematic of pipeline measurement instrumentation

There are 77 parameters that need to be disposed in the set of training example cases, which is shown in Table. 1.

Table 1. The collecting parameters of pipeline measurement instrumentation

\ddot{I}_{11}	\ddot{I}_{12}	\ddot{I}_{13}	\ddot{I}_{14}	\ddot{I}_{15}	\ddot{I}_{16}	\ddot{I}_{17}	\ddot{I}_{18}	\ddot{I}_{19}	\ddot{I}_{110}
\ddot{I}_{21}	\ddot{I}_{22}	\ddot{I}_{23}	\ddot{I}_{24}	\ddot{I}_{25}	\ddot{I}_{26}	\ddot{I}_{27}	\ddot{I}_{28}	\ddot{I}_{29}	\ddot{I}_{210}
\ddot{I}_{31}	\ddot{I}_{32}	\ddot{I}_{33}	\ddot{I}_{34}	\ddot{I}_{35}	\ddot{I}_{36}	\ddot{I}_{37}	\ddot{I}_{38}	\ddot{I}_{39}	\ddot{I}_{310}
\ddot{I}_{41}	\ddot{I}_{42}	\ddot{I}_{43}	\ddot{I}_{44}	\ddot{I}_{45}	\ddot{I}_{46}	\ddot{I}_{47}	\ddot{I}_{48}	\ddot{I}_{49}	\ddot{I}_{410}
\ddot{I}_{51}	\ddot{I}_{52}	\ddot{I}_{53}	\ddot{I}_{54}	\ddot{I}_{55}	\ddot{I}_{56}	\ddot{I}_{57}	\ddot{I}_{58}	\ddot{I}_{59}	
\ddot{I}_{61}	\ddot{I}_{62}	\ddot{I}_{63}	\ddot{I}_{64}	\ddot{I}_{65}	\ddot{I}_{66}	\ddot{I}_{67}	\ddot{I}_{68}	\ddot{I}_{69}	\ddot{I}_{610}
\ddot{I}_{71}	\ddot{I}_{72}	\ddot{I}_{73}	\ddot{I}_{74}	\ddot{I}_{75}	\ddot{I}_{76}	\ddot{I}_{77}	\ddot{I}_{78}	\ddot{I}_{79}	\ddot{I}_{710}
\ddot{I}_{81}	\ddot{I}_{82}	\ddot{I}_{83}	\ddot{I}_{84}	\ddot{I}_{85}	\ddot{I}_{86}	\ddot{I}_{87}	\ddot{I}_{88}	\ddot{I}_{89}	\ddot{I}_{810}
\ddot{I}_{91}	\ddot{I}_{92}	\ddot{I}_{93}	\ddot{I}_{94}	\ddot{I}_{95}	\ddot{I}_{96}	\ddot{I}_{97}	\ddot{I}_{98}	\ddot{I}_{99}	\ddot{I}_{910}

Note: S_{P1} to S_{P10} : Pressure signal of station 1 to station 10; S_{T1} to S_{T10} : Temperature of product oil of station 1 to station 10; S_{F1} to S_{F10} : Flow signal of station 1 to station 10; S_{D1} to S_{D10} : Density signal of station 1 to station 10; S_{SP1} to S_{SP10} : Status of pump of station 1 to station 10; S_{TV1} to S_{TV10} : Turndown ratio of valve of station 1 to station 10; S_{Dia1} to S_{Dia9} : Diameter of segment 1 to segment 9; S_{V1} to S_{V9} : Negative pressure velocity in segment 1 to segment 9.

Because the status of the pipeline should be calculated rapidly, we can consider the status of each segment (altogether 9 segments) as a calculating cell which can reduce the computing time of server computer. After attributes reduction, for the i th segment, there are 14 correlative parameters which is the inputs of MLP are held, as is shown in table. 2. According to the inputs of MLP, we select 30 neurons made up of hidden layer.

Table 2. Relative parameters of i th segment

\ddot{I}_{11}	\ddot{I}_{12}	\ddot{I}_{13}	\ddot{I}_{14}
\ddot{I}_{21}	\ddot{I}_{22}	\ddot{I}_{23}	\ddot{I}_{24}
\ddot{I}_{31}	\ddot{I}_{32}	\ddot{I}_{33}	\ddot{I}_{34}
\ddot{I}_{41}	\ddot{I}_{42}	\ddot{I}_{43}	\ddot{I}_{44}

Note: $0 < i < 10$; when $i=1$, $SP(i-1)=0$; when $i=8$, $SP(i+2)=0$; when $i=9$, $SP(i+1)=0$, $SP(i+2)=0$.

3.2 Results

Leak experiments have been taken 30 times. Work status change experiments have been taken 155 times. In all 30 leaks, there are 29 leaks could have been classified into “Leak”, and 1 leak has been classified into “Unknown”. In other operating condition, this method can classify the statuses preferably. The results are shown in table 3.

Table 3. Experiment Results

Experiment Items	Experiment Times	Experiment Result			
		Running	Operating	Leak	Unknown
Increasing Flow	50	1	49		
Decreasing Flow	50		48		2
Altering Filter	50		49	1	
Altering Oil Product	5		5		
Leak	30			29	1

4 Conclusion

This paper proposes a leak fault detection method by the use of Rough Set and Artificial Neural Network. Together with Rough Set, reluctant parameters are reduced; and together with Artificial Neural Network, a decision model is developed which can classify real-time data into four different statuses. The method can distinguish leak fault from normal running and adjusting pump in pipeline. Experiment results show that the proposed method can effectively detect and diagnose leak fault.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under grant no. 60572070 to Zhang Huaguang. This work is also supported in part by Open Project Foundation of Key Laboratory of Process Industry Automation, Ministry of Education China, and the China Postdoctoral Science Foundation to Feng Jian.

References

1. Feng, J., Zhang, H.G.: Oil Pipeline Leak Detection and Location using Double Sensors Pressure Gradient Method. Proceedings of the 5th World Congress on Intelligent Control and Automation (2004) 3134-3137
2. Hou, Z.J., Lian, Z.W., Yao, Y., Yuan, X.J.: Cooling-Load Prediction by the Combination of Rough Set Theory and an Artificial Neural-Network based on Data-Fusion Technique. Applied Energy **83** (2006) 1033-1046
3. Pawlak: Rough Sets. Int. J. Comput. Sci. **11** (1982) 341-356
4. Zhao, J., Wang, G.Y., Wu, Z.F., Tang, H., Li, H.: Method of Data Discretization based on Rough Set Theory. Mini-Microsystems **25** (2004) 60-64
5. Li, R.P., Wang, Z.O.: Mining Classification Rules using Rough Sets and Neural Networks. European Journal of Operational Research **157** (2004) 439-448
6. Li, C.J., Jansuwan, C.: Dynamic Projection Network for Supervised Pattern Classification. Int. J. Approximate Reasoning **40** (2005) 243-261

7. Zhao, M.: Data Description based on Reduct Theory. Institute of Automation, Chinese Academy of Sciences (2004)
8. Yahia, M.E., Mahmud, R., Sulaiman, N., Ahmad, F.: Rough Neural expert systems. *Expert Systems with Applications* **18** (2000) 87–99
9. Gorzalczany, M.B., Piasta, Z.: Neuro-Fuzzy Approach Versus Rough-Set Inspired Methodology for Intelligent Decision Support. *Information Sciences* **120** (1999) 45–68
10. Lingras, P.: Comparison of Neo-fuzzy and Rough Neural Networks. *Information Sciences* **110** (1998) 207-215
11. Yahia, M.E., Mahmud, R.: Hybrid Expert System of Rough Set and Neural Network. *Malaysian Journal of Computer Science* **12** (1999) 1–8

Probabilistic Neural Network Based Method for Fault Diagnosis of Analog Circuits

Yanghong Tan, Yigang He, and Meirong Liu

College of electric & information engineering, Hunan University, Changsha 410082, China
yghe@hnu.cn, tanyho@126.com

Abstract. A novel method for fault diagnosis of analog circuits with tolerance based on wavelet packet decomposition (WP) and probabilistic neural networks (PNN) is proposed in the paper. The fault feature vectors are extracted after feasible domains on the basis of WP decomposition of responses of a circuit is solved. Then by fusing various uncertain factors into probabilistic operations, parameters and structures of PNNs for diagnose faults are obtained based on genetic optimization method leading to best detection of faults. Finally, simulations indicated that PNN classifiers can correctly 7% more than BPNN of the test data associated with our sample circuits.

1 Introduction

There is a growing interest in the last two decades in the development of automatic tools for testing circuits and the well-consolidated techniques in digital implementations have been mature and cost effective^[1], while the testing of analog and mixed analog-digital systems is more complicated and less understood and yet relevant to applications. In fact, the automation fault diagnosis(FD) for analog circuits is subject to many items, such as the poor fault models, the presence of noise, the nonlinearities and the tolerance issues presented. So it is mainly based both on the maintainers' experience and on specifications of its functionality [1-2].

Some methodologies of general validity have been proposed and developed in recent years [1-7]. Neural networks (NNs) have been applied to a variety of problems in the area of pattern recognition, signal processing etc. The classifiers in terms of NNs trained on the fault dictionary examples have been successfully applied to FD providing satisfactory results while minimizing computation costs [3-7, 18-21].

Wavelet-based approaches to detect abrupt faults in dynamic systems by decomposing output signals into elementary building blocks of wavelet transforms, the local reforms of the signals caused by the faults can be identified by analyzing the non-stationary of the output signals. But little literature is available in the FD of analog circuits. Mammalian and F.Aminian (2000) [8] proposed a neural FD method using wavelet as a preprocessor to reduce the number of input features to a manageable size. Unfortunately, the decomposition based on wavelet transform [5-8] is limited to the approximate signals [9-12].

The research presented here is an attempt to exploit wavelet packets to extract appropriate feature vectors from signals sampled from the CUT under various faults. The processing contains details as well as approximate coefficients. Optimal sets are available for training neural networks after normalization and PCA processing and the output of neural networks reveals the fault patterns. And PNNs are adopted to classify the faults to make up for the shortcomings of BP algorithms.

2 Feasible Domains of Circuits with Tolerance Based on WP Method

The feasible domains of responses of a circuit would be intervals rather than points. WP [5-8, 18] has been successfully supplied to many fields such as signal compression. It can be expressed as

$$2^{\frac{p-1}{2}} W_{2^n} (2^{p-1} x - l) = \sum_m h_{m-2l} 2^{\frac{p}{2}} W_n (2^p x - m), \tag{1}$$

$$2^{\frac{p-1}{2}} W_{2^{n+1}} (2^{p-1} x - l) = \sum_m g_{m-2l} 2^{\frac{p}{2}} W_n (2^p x - m), \tag{2}$$

where p and l are scale factors, $W_0(x) = \phi(x)$, $W_1(x) = \varphi(x)$, $\phi(x)$ is scale shift function and $\varphi(x)$ is mother wavelet. The projection of $f(x) \in L^2(\mathbb{R})$ is $\langle f(x), W_n(2^p x - k) \rangle$.

Let $S_{n,k}^p = 2^{\frac{p}{2}} \int_{-\infty}^{\infty} f(x) W_n(2^p x - k) dx$ and assumed that the response of CUT is out whose j th sample be out_j , then the mean and variance value can be expressed as

$$\mu_{out} = \frac{1}{N} \sum_{j=1}^N out_j, \quad \sigma_{out} = \frac{1}{N} \sum_{j=1}^N (out_j - out_0)(out_j - \mu_{out}), \tag{3}$$

where out_0 the response of the circuit without tolerance. The WP coefficient of output out can be written [5-8] as $S_{n,k}^p = \langle out, W_n(2^p x - k) \rangle$.

Let $f(out) = S_{n,k}^p$, its Taylor expansion of at the neighbor of μ_{out} is expressed as

$$S_{n,k}^p = f(\mu_{out}) + a(out - \mu_{out}) + b(out - \mu_{out})^2, \tag{4}$$

$$a = \left. \frac{d}{dout} f(\mu_{out}) \right|_{out=\mu_{out}}, \quad b = \left. \frac{1}{2} \cdot \frac{d^2}{dout^2} f(\mu_{out}) \right|_{out=\mu_{out}}. \tag{5}$$

The coefficients a and b are approximated as

$$a \approx \frac{\Delta f}{\Delta out} = \frac{f(out_+) - f(\mu_{out})}{out_+ - \mu_{out}} = \frac{f(\mu_{out}) - f(out_-)}{\mu_{out} - out_-}, \tag{6}$$

$$b \approx \frac{\left(\frac{\Delta f}{\Delta out}\right)_{out_+} - \left(\frac{\Delta f}{\Delta out}\right)_{out_-}}{2\Delta out} = \frac{f(out_+) + f(out_-) - 2f(\mu_{out})}{2\Delta out}, \tag{7}$$

Then the mean and variance of WP decomposition are

$$\mu_{S_{n,k}^p} = f(\mu_{out}) + \frac{1}{2}b\sigma_{out}^2, \quad \sigma_{S_{n,k}^p}^2 = \overline{(S_{n,k}^p)^2} - 2\overline{S_{n,k}^p}\mu_{S_{n,k}^p} + \mu_{S_{n,k}^p}^2. \tag{8}$$

Formula (7) can be simplified as

$$\mu_{S_{n,k}^p} = f(\mu_{out}), \quad \sigma_{S_{n,k}^p}^2 = a^2\sigma_{out}^2. \tag{9}$$

So WP coefficients of *out* can be regard as noises centering at $\mu_{S_{n,k}^p}$ with $\sigma_{S_{n,k}^p}$.

3 Extraction of Fault Features Based on WP Method

Let μ_{ufi}, μ_{ufj} and $\sigma_{ufi}, \sigma_{ufj}$ be the mean and variance values of WP decomposition coefficients of response of test nodes under the *i*th and *j*th faults. Then the fault features (FFs) can be expressed as

$$\max J = \sum_f \frac{\left| \mu_{ufi} - \mu_{ufj} \right|}{\sigma_{ufi}^2 + \sigma_{ufj}^2}, \tag{10}$$

$$\text{s.t. } v_{kmin} \leq v_k \leq v_{kmax}; k = 1, 2, \dots, K, \quad x_{jmin} \leq x_j \leq x_{jmax}; j = 1, 2, \dots, J,$$

where x_j is the parameter of the *j*th element in the circuit with its tolerance range of $[x_{jmin}, x_{jmax}]$ ($j = 1, 2, \dots, J$) and *J* is the total number of elements in the circuit. And v_k is the parameter of the *k*th point of inflexion of the piece-wise source with its range can be varied from v_{kmin} to v_{kmax} and *k* is the total number of inflexion points. w_{k1} and w_{k2} are the weight value to measure the significance of output voltages and currents through stimulus terminals. The optimization can be realized conveniently based on genetic method.

In a question of *L* faults^{[[5-8]}, the mean and variance values of the *k*th samples feature vector are $\mu_{ui,k}, \mu_{uj,k}, \sigma_{ui,k}, \sigma_{uj,k}$ and $\mu_{ii,k}, \mu_{ij,k}, \sigma_{ii,k}, \sigma_{ij,k}$ of output voltages and currents through stimulus terminals that are decomposed based on wavelet packet coefficients under the *i*th and *j*th faults. Let

$$J(i, j, k) = w_{uk} \frac{\left| \mu_{ui,k} - \mu_{uj,k} \right|}{\sigma_{ui,k}^2 + \sigma_{uj,k}^2} + w_{ik} \frac{\left| \mu_{ii,k} - \mu_{ij,k} \right|}{\sigma_{ii,k}^2 + \sigma_{ij,k}^2}, \tag{11}$$

then $J_k = \sum_{i=1}^{L-1} \sum_{j=i+1}^L J(i, j, k)$. So the best decomposition level of wavelet packet (LoWP) is defined as

$$\begin{aligned} & \max \sum_k |\mathbf{J}_k|, \\ & \text{s.t. } v_{k\min} \leq v_k \leq v_{k\max}; k = 1, 2, \dots, K, \quad x_{j\min} \leq x_j \leq x_{j\max}; j = 1, 2, \dots, J. \end{aligned} \tag{12}$$

That is, if $\sum_k |\mathbf{J}_k|$ is increasing with future WP decomposition then $\text{LoWP} = \text{LoWP} + 1$,

otherwise $\text{LoWP} = \text{LoWP}$. After the satisfactory LoWP is evaluated, the obtained J_k are sorted in order of $J_{f_1} \geq J_{f_2} \geq \dots J_{f_k}$ and $F_{i,j} = \{f_d | J_{fd} \geq \lambda, \lambda > 0\}$, $d \in [f_1, f_k]$ are obtained. So the final feature vector is

$$F = \left\{ \bigcup_{i=1}^L F_{i,j} \right\}. \tag{13}$$

4 GPNN Methods for FD of Analog Circuits

The diagram of PNN [15-18] method for fault diagnosis of analog circuits is shown in Fig.1. After obtained the WP decomposition coefficients of the response of the sampled circuits with tolerance, the feasible domains are got. And the fault feature vectors (FFVs) are available according to the principles proposed. Thus the input patterns of probabilistic neural networks are available.

The architecture of PNN [13-18] is shown in Fig.1. It contains radial base layer and competition layer. Let the input vector be $x = [x_1, x_2, \dots, x_n]$. The hidden layer accepts the input vector and maps it into probabilistic density functions.

Let the fault models be normal distributions, the probabilistic density functions are

$$\phi_{ij}(x) = \frac{1}{(2\pi)^{\frac{n}{2}} \sigma_{ij}^n} e^{-\frac{(x-\mu_{ij})(x-\mu_{ij})^T}{2\sigma_{ij}^2}}, \tag{14}$$

where μ_{ij} is the initial expectation value of the i th feature vectors of the j th faults and σ_{ij} is the smoothness factor. So the probabilities of the pattern x belongs to cluster $C_i (i = 1, 2, \dots, M)$ is

$$p_j(x) = \sum_{i=1}^{P_j} w_{ij} \phi_{ij}(x), j = 1, 2, \dots, M, \tag{15}$$

where w_{ij} is a weight of hidden layer, and $\sum_{i=1}^{P_j} w_{ij} = 1, w_{ij} \geq 0$. Then

$$p_j(x) = \frac{1}{(2\pi)^{\frac{n}{2}} \sigma_{ij}^n} \frac{1}{P_j} \sum_{i=1}^{P_j} w_{ij} e^{-\frac{(x-\mu_{ij})(\mu-x_{ij})^T}{2\sigma_{ij}^2}}, \tag{16}$$

where P_j is the total number of the j th faults. According to Bayes rulers, $C(x)$ is

$$C(x) = \arg \max \{p_j(x)\}, j=1, 2, \dots, M. \tag{17}$$

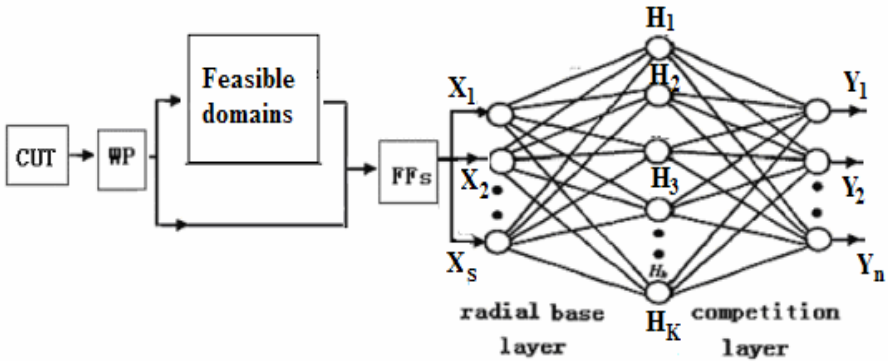


Fig. 1. GPNN method for FD of circuits

The classification is $\Lambda^* = \arg \max \prod_{i=1}^K \prod_{j=1}^{P_j} p_j(x)$. To simplify the calculation, let

$$\Lambda^* = \arg \max \sum_{i=1}^K \sum_{j=1}^{P_j} \log [p_j(x)]. \tag{18}$$

Assumed that P_j is independent of $C_i, i=1, 2, \dots, M, i \neq j$, we get

$$E = \sum_{i=1}^K \sum_{j=1}^{P_j} \log [p_j(x)]. \tag{19}$$

The learning would be time consuming and memory costing if gradient descent is adopted to optimize the parameters of wavelets PNN. So optimization is realized based on genetic algorithms. Let the nodes of hidden layer of WPNN is N , the chromosomes of initial generation are $\Lambda = [a_k, b_k, N, w_{ij}, \mu_{ij}, \sigma_{ij}^{-1}]$, where μ_{ij} is the initial expectation value of the i th feature vectors of the j th faults and σ_{ij} is the smoothness factor. Thus the optimization aims at finding the best N for given train error goal by choosing best chromosomes $\Lambda^* = [a_k^*, b_k^*, N^*, w_{ij}^*, \mu_{ij}^*, \sigma_{ij}^{-1*}]$.

5 Diagnosis Examples

Example 1. An amplifier with element tolerance of 10% is shown in Fig.2. The circuit is stimulated by a pulse source whose amplitude is 5v, and the duration, rising and falling time are 10us, 0.5us and 0.5us respectively. Consider the following 27 single faults and 3 multiple faults.

- 27 single faults:
 BES,CES,BO and BCS for Q_1 and Q_2 ;
 $R_2=10\Omega, 47\Omega, 4.7K$ and OC ;
 $R_1=10\Omega, SC, 1.1K, 5K$ and OC;
 $R_6=10\Omega$ and 5K;
 $R_4=10\Omega, 200, 2K,$ and OC;
 $R_5=100\Omega, 5K, SC$ and OC .
- 3 Multiple faults:
 $R_5=50\Omega$ and $R_1=22k$;
 $R_5=500\Omega$ and $R_1=22k$;
 $R_5=500\Omega, R_1=22k, Q_2BCS$ and R_2OC .

Define the correct rate of FD (CRFD) be

```

y = sum(sqr(a3 - t))
u=size(y, 2); p=0;
for i=1:u
    if y(i) <= pr
        p=p+1;
    end
end
present = p/u
    
```

where t and y are the target and real output of a NN, pr is a predefined constant, then p is the CRFD. The fault features are extracted from the DC points, samples of previous five harmonics and transient response. The number of output neurons is the same as the number of elements.

The target output is the probability $p_i = 1 (i = 1, 2, \dots, n)$ if the i th element is faulty, otherwise $p_i = 0$. In the process of FD, the faults are classified by “Principle of maximum probability”: $C(x) = \text{argmax}\{p_i(x)\}, i = 1, 2, \dots, M, 0 < p_i < 1 (i = 1, 2, \dots, n)$. The hidden layer nodes N and CRFD are returned when the network is steady. And the same BPNN method is employed using the returned N in order to compare the CRFD of GPNN to Adaptive BP. The results are shown in table 1 and Fig.3. The CRFDs are improved more than 7% by adopting GPNN, but it takes more time to convergent.

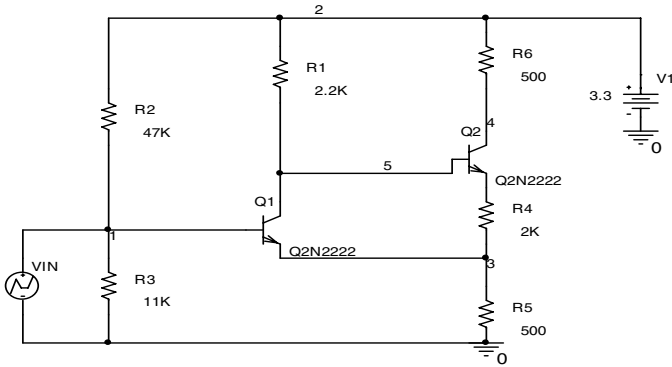


Fig. 2. An amplifier

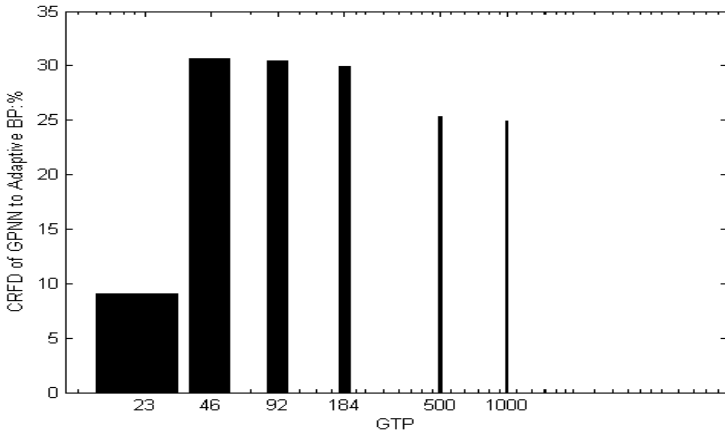


Fig. 3. CRFD gap between GPNN and BPNN method

Example 2. The tolerance of element is 5% in the large-scale circuit [6] shown in Fig.4. The identical faults are considered and the same fault classes are considered as literature [6]. The circuit is torn to 8 sub-circuits. The diagnosis result is denoted in table 2 and the CRFD gap between GPNN and BPNN is shown in Fig.5.

Table 1. Comparison of BPNN and GPNN (Note : GTP: Groups of train patterns)

GTP	23	46	92	184	500	1000
BP	62.93%	63.61%	65.10%	65.78%	70.49%	70.65%
GPNN	71.98%	94.23%	95.52%	95.71%	95.80%	95.59%

Table 2. Table 2 Results of FD for the circuit shown in Fig.4.

Sub-circuits	1	2	3	4	5	6	7	8
N	69	33	46	33	48	22	30	56
epochs	1067	1115	945	917	715	1244	1354	1362
CRFD	87.98	90.89	94.18	90.60	92.91	92.60	86.96	87.19

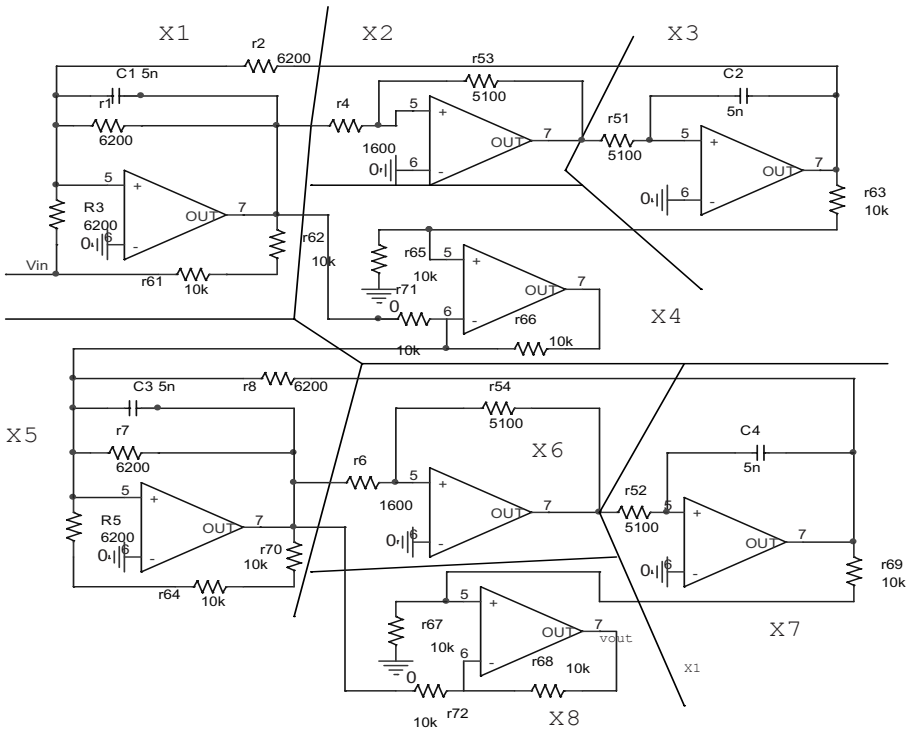


Fig. 4. A large-scale circuit

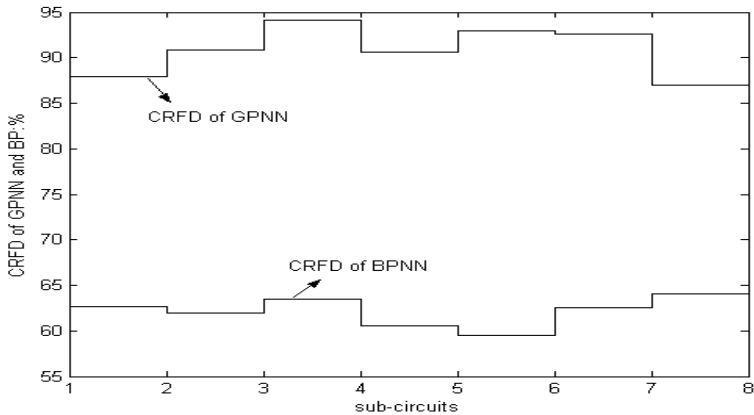


Fig. 5. CRFD of GPNN and BPNN

6 Conclusions

A novel GPNN method for fault diagnosis of analog circuits with tolerance is proposed after FDs resolved and fault feature vectors extracted based on WP

decomposition. And by using various uncertain factors into probabilistic operations, GPNN method to diagnose faults are proposed whose parameters and structure obtained from genetic optimizations resulting in best detection of faults. Finally, simulations indicated that GPNN classifiers can correctly identify at least 7% more than BPNN of the test data associated with our sample circuits.

Acknowledgement

The authors thank the Natural Science Foundation Council of China under Grant No. 50677014, Doctoral Special Fund of Ministry of Education, No.20060532002, the Program for New Century Excellent Talents in University Grant NCET-04-0767, Foundation of Hunan Province Science and Technology Grant, 06JJ2024, 03GKY3115, 04FJ2003, 05GK2005 for financial support.

References

1. Yang, S.Y.: Fault Diagnosis and Reliability Design of Analog Circuits. Tsinghua University, P.R.China, Beijing (1993)
2. Navid, N., Willson, A.A.: A theory and an Algorithm for Analog Circuit FD. IEEE Trans.on CAS **26** (7) (1979) 440-457
3. Bandler, J.W.: Recent Advances in Fault Location of Analog Networks. In: Proc. IEEE Int. Symp CAS. (1984) 660-663
4. Lee, J.H., Bedrosian, S.D.: Fault Isolation Algorithm for Analog Electronic Systems using the Fuzzy Concept. IEEE Trans. on CAS **26** (7) (1979) 518-522
5. Sheu, H.T., Chang, Y.H.: Robust FD For Large-Scale Analog Circuit With Measurement Noises, IEEE Trans. on CAS **44** (3) (1997) 198-209
6. Salama, A.E., et al.: A Unified Decomposition Approach for Fault Location in Large Analog Networks. IEEE Trans. on CAS **31** (7) (1984) 609-621
7. Aminian, F., Aminian, M., Collin, H.W.: Analog Fault Diagnosis of Actual Circuits Using Neural Networks. IEEE Trans. Instrum.Meas **51** (3) (2002) 544-550
8. Ozawa, T., Yamada, M.: Diagnosability in the Decomposition Approach for Fault Location in Large Analog Networks. IEEE Trans. on CAS **32** (4) (1985) 415-426
9. Timo, Sorsa, Koivo, H.N., Koivisto, H.: Neural Network in Process Fault Diagnosis. IEEE Trans. Systems, Man, Cybernetics **21** (4) (1991) 815-825.
10. Bernieri, A., D'Apuzzo, M., Somson, L., Savastano, M.: A Neural Network Approach for Identification and Fault Diagnosis on Dynamic Systems. IEEE Trans. Instru. Measu. **43** (6) (1994) 867-873.
11. Catelani, M., Fort, A.: Soft Fault Detection and Isolation in Analog Circuit: Some Result and a Comparison between a Fuzzy Approach and Radial Basis Function Network. IEEE Trans.on Instru.Measu. **51** (2) (2002) 196-202
12. Zhang, B.L., Coggins, R., Jabri, M.A., Dersch, D., Flower, B.: Multiresolution Forecasting for Futures Trading Using Wavelet Decompositions. IEEE Trans. Neural Networks **12** (4) (2001) 765- 775
13. Yen, G.G., Lin, K.C.: Wavelet Packet Feature Extraction for Vibration Monitoring. IEEE Trans. Industrial Electronics **47** (3) (2000) 650-667

14. Wang, Z.H., Gielen, G., Sansen, W.: Probabilistic Fault Detection and the Selection of Measurements for Analog Integrated Circuits. *IEEE Trans. Computer -aided Design of Integrated CAS* **17** (9) (1998) 862-871
15. Mao, K.Z., Tan, K.C., Ser, W.: Probabilistic neural-network Structure Determination for Pattern Classification. *IEEE Trans. Neural Networks* **11** (4) (2000) 1009-1016
16. Wantanabe, S., Fukumizu, K.: Probabilistic Design of Layered Neural Networks Based on Their Unified Framework. *IEEE Trans on Neural Networks* **6** (3) (1995) 691- 702
17. Ney, N.: On the Probabilistic Interpretation of Neural Network Classifiers and Discriminative Training Criteria. *IEEE Trans. Pattern Analysis and Machine Intelligence* **17** (2) (1995)107-119
18. He, Y.G., Tan, Y.T., Sun, Y.C.: Wavelet Neural Network Approach for Fault Diagnosis of Analog Circuits. *IEE Proc.-Circuits Devices and Systems* **151** (2004) 379-384
19. He, Y.G., Tan, Y.T., Sun, Y.C.: Class-based Neural Network Method for Fault Location of Large Scale Analog Circuits. *Proceedings of IEEE Int. Symp Circuits and Systems* (2003) 733-736
20. He, Y.G., Sun, Y.C.: Neural Networks-based Nonlinear L_1 -norm Optimization Approach for Fault Diagnosis of Nonlinear Circuits with to Tolerance, *IEE proceedings: Circuits, Devices and Systems* **148** (4) (2001) 223-228
21. He, Y.G., Sun, Y.C.: Fault Isolation in Nonlinear Analog Circuits with Tolerance Using the Neural Network based L_1 -norm. *Proc. IEEE Int. Symp. Circuits & Systems* (2001) 854-857

Gear Fault Diagnosis by Using Wavelet Neural Networks

Y. Kang¹, C.C. Wang², and Y.P. Chang¹

¹ Department of Mechanical Engineering, Chung Yuan Christian University

² R&D Center for Membrane Technology, Chung Yuan Christian University
200, Chung Pei Rd., Chung Li 32023, Taiwan
yk@cycu.edu.tw

Abstract. Fault diagnosis in gear train system is important in order to transmitting power effectively. The artificial intelligent such as neural network is widely used in fault diagnosis and already substituted for traditional methods such as kurtosis method, time analysis and so on. The symptoms of vibration signals in frequency domains have been used as inputs to the neural network and diagnosis results are obtained by network computation. This study presents gear fault diagnosis by using wavelet neural networks (WNN) and Morlet wavelet is used as the activation function in hidden layer of back-propagation neural networks (BPNN). Furthermore, the diagnosis results are compared within both methods of WNN and BPNN in four gear cases.

1 Introduction

Gearboxes are widely used in the rotary machinery in order to transmitting power. The definition of the gear diagnosis is using vibration signals to classify faults by expert knowledge or experiments when the gear system broke down. However, it is hard to distinguish effective information from the vibration signals. Randall [1] used spectrum and cepstrum of vibration signals researches for gear fault diagnosis. Wang and Wong [2] are used the autoregressive model to detect gear faults and the diagnosis results are obtained accurately than traditional residual kurtosis method.

The advantages of neural network are not only which the weightings of neural network are obtained from neural computation, but also the diagnosis results are objective than traditional expert's experiences. Recently, back-propagation neural network (BPNN) is widely used to solve fault diagnosis problems. Fong and Hui [3] applied an intelligent data mining technique that combines neural network and rule-based reasoning to extract information from the knowledge database for online machine fault diagnosis. Kang et al. [4] extracted frequency symptoms of vibration signals to detect faults by using BPNN for motor bearing system.

Although BPNN has better identification ability for fault detection than other traditional inference method, there are still some shortcomings, such as large learning epochs and existence of local minima. Zhang and Benveniste [5] combine wavelet theory and BPNN to form wavelet neural network (WNN) and use Grossmann Morlet function, I. Daubechies function, and many others to replace sigmoid function in

BPNN and improve the shortcomings of BPNN. Base on the superior convergence of WNN, this study applied WNN to gear fault diagnosis and compare with BPNN in four different gear train cases.

2 Wavelet Neural Networks

WNN is based on wavelet principals to combine wavelet theory within BPNN. Because the Morlet function is a derivational continuous function and has an express equation than other wavelet function, in this study, the WNN used wavelet activation function to replace the sigmoid activation function in hidden layer and form a new kind of approximate approach to fitting nonlinear models.

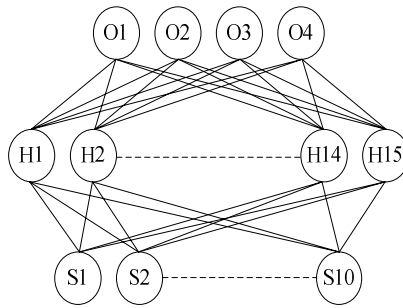


Fig. 1. The structure of WNN

Table 1. Relational matrix

Symptoms	S ₁	S ₂	S ₃	S ₄	S ₅	S ₆	S ₇	S ₈	S ₉	S ₁₀
Gear skew (O ₁)	0.3	0.05	0.05	0.05	0.4	0	0	1	0	0
Shaft not parallel (O ₂)	0.05	0.05	0	0	0.2	0.3	0.1	0	0.7	1
Tooth breakage (O ₃)	1	0.2	0.3	0.1	1	0.6	0	0	0	0
Wear (O ₄)	0.9	0.3	0	0	1	0	0	0	0	0

In this study, the structure of the WNN is showed in Fig. 1, the input layer has ten neurons (S = [s₁, s₂, ..., s₁₀]) corresponding to the ten frequency symptoms of vibration signal in the gearbox. The output layer has four neurons (O = [o₁, o₂, ..., o₄]) associated with the four kind of typical faults in the gearbox. The second layer is wavelet (hidden) layer that is utilized Morlet wavelet as the activation function in hidden layer of BPNN. Table 1 show the relational matrix of training samples with four gear faults and the symbols explanation is listed in Appendix. In this study, the wavelet activation function is used the Morlet function and its differentiation equation can be written as

$$h(x) = \cos(1.75x) \cdot e^{-\frac{x^2}{2}}, \quad h'(x) = -1.75 \cdot \sin(1.75x) \cdot e^{-\frac{x^2}{2}} - \cos(1.75x) \cdot e^{-\frac{x^2}{2}} \cdot x \quad (1)$$

The activation function in output layer is used by sigmoid function and its differentiation equation can be written as

$$\sigma(x) = 1/(1 + e^{-x}), \quad \sigma'(x) = \sigma(x)(1 - \sigma(x)) \quad (2)$$

On the basis of the gradient descent algorithm for the minimization of error, the correction increments of weights coefficients Δw_{ji} , Δw_{kj} and wavelet dilation and translation coefficients Δa_j , Δb_j are adjusted to minimize the least-square error. T_i^p is the i th ideal output of the p th input sample and o_i^p is the i th actual output of the p th input sample. The error of the network is defined by

$$E = \frac{1}{2} \sum_{p=1}^4 \sum_{i=1}^4 (T_i^p - o_i^p)^2 \quad (3)$$

Forward computing of WNN can be written sequentially as follows

$$\text{net}_j = \sum_{k=0}^L w_{kj} x_k, \quad \psi_{a,b}(\text{net}_j) = \psi\left(\frac{\text{net}_j - b_j}{a_j}\right) \quad (4)$$

$$\text{net}_i = \sum_{j=0}^M w_{ji} \psi_{a,b}(\text{net}_j), \quad o_i(\text{net}_i) = 1/(1 + e^{-\text{net}_i}) \quad (5)$$

Backward computing of WNN can be written sequentially as follows

$$\begin{aligned} \Delta w_{ji} &= -\eta \frac{\partial E}{\partial w_{ji}} = -\eta \frac{\partial E}{\partial o_i(\text{net}_i)} \frac{\partial o_i(\text{net}_i)}{\partial \text{net}_i} \frac{\partial \text{net}_i}{\partial w_{ji}} \\ &= -\eta (T_i^p - o_i^p) (-o_i^p) (1 - o_i^p) (\psi_{a,b}(\text{net}_j)) \end{aligned} \quad (6)$$

$$\begin{aligned} \Delta w_{kj} &= -\eta \frac{\partial E}{\partial w_{kj}} = -\eta \frac{\partial E}{\partial o_i(\text{net}_i)} \frac{\partial o_i(\text{net}_i)}{\partial \text{net}_i} \frac{\partial \text{net}_i}{\partial \psi_{a,b}(\text{net}_j)} \frac{\partial \psi_{a,b}(\text{net}_j)}{\partial \text{net}_j} \frac{\partial \text{net}_j}{\partial w_{kj}} \\ &= -\eta (T_i^p - o_i^p) (-o_i^p) (1 - o_i^p) w_{ji} (\psi'_{a,b}(\text{net}_j)) \frac{1}{a_j} x_k \end{aligned} \quad (7)$$

$$\begin{aligned} \Delta a_j &= -\eta \frac{\partial E}{\partial a_j} = -\eta \frac{\partial E}{\partial o_i(\text{net}_i)} \frac{\partial o_i(\text{net}_i)}{\partial \text{net}_i} \frac{\partial \text{net}_i}{\partial \psi_{a,b}(\text{net}_j)} \frac{\partial \psi_{a,b}(\text{net}_j)}{\partial a_j} \\ &= -\eta (T_i^p - o_i^p) (-o_i^p) (1 - o_i^p) w_{ji} (\psi'_{a,b}(\text{net}_j)) \frac{1}{a_j^2} (b_j - \text{net}_j) \end{aligned} \quad (8)$$

$$\begin{aligned} \Delta b_j &= -\eta \frac{\partial E}{\partial b_j} = -\eta \frac{\partial E}{\partial o_i(\text{net}_i)} \frac{\partial o_i(\text{net}_i)}{\partial \text{net}_i} \frac{\partial \text{net}_i}{\partial \psi_{a,b}(\text{net}_j)} \frac{\partial \psi_{a,b}(\text{net}_j)}{\partial b_j} \\ &= -\eta (T_i^p - o_i^p) (-o_i^p) (1 - o_i^p) w_{ji} (\psi'_{a,b}(\text{net}_j)) \frac{1}{a_j} (-1) \end{aligned} \quad (9)$$

The WNN is trained using the error between the actual output and the ideal output, to modify the weights w_{ji} , w_{kj} and wavelet parameters Δa_j , Δb_j until the output of the neural network is close to the ideal output, with an acceptable accuracy. The weights coefficients and wavelet parameters are updated as follows

$$w_{ji}(t+1) = w_{ji}(t) + \Delta w_{ji}, \quad w_{kj}(t+1) = w_{kj}(t) + \Delta w_{kj} \quad (10)$$

$$a_j(t+1) = a_j(t) + \Delta a_j, \quad b_j(t+1) = b_j(t) + \Delta b_j \quad (11)$$

Using the measured vibration signals computed with the trained weights coefficients w_{ji} , w_{kj} and wavelet parameters Δa_j , Δb_j when detecting faults. Each of the output neuron of WNN represents a kind of gear fault and the output value represents the degree of certainty for corresponding fault. If the value is closed to 1, which represent the possibility of the fault is high.

3 Case Studies

The experiment of gear train system is shown in Fig. 2. It consists of a motor, a converter and a pair of spur gears in which the transmitting gear has 46 teeth and the passive gear has 30 teeth. The vibration signals are measured from two accelerometers that mounted on the bearing housing of gear system. In this paper, four kinds of

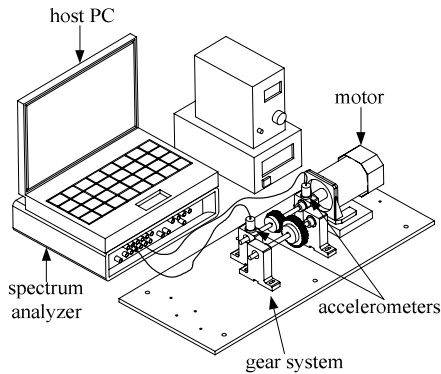


Fig. 2. The experimental setup

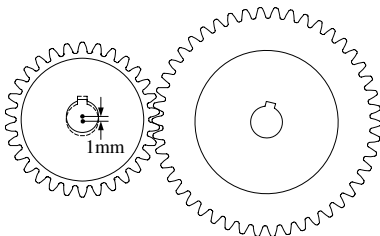


Fig. 3. The sketch of the gear skew



Fig. 4. Shaft not parallel of gear system



Fig. 5. Tooth breakage gear



Fig. 6. Wear gear

typical gear faults (a) gear skew and the sketch map is shown in Fig.3, (b) shaft not parallel and the actual figure of shaft not parallel fault in gear system is shown in Fig. 4, (c) tooth breakage and the actual figure of tooth breakage fault in gear system is shown in Fig. 5, (d) wear and the actual figure of wear fault in gear system is shown in Fig. 6.

3.1 Gear Skew

The centerline of spur gear above the transmission shaft shifts 1 mm to center position. The rotary speed of motor was set as 1500rpm (25Hz). The frequency spectrum distributed over rotary speed of transmission shaft and rotary speed of passiveness shaft, resonance frequency and mesh frequency. The frequency spectrum diagram of the gear skew is shown in Fig. 7.

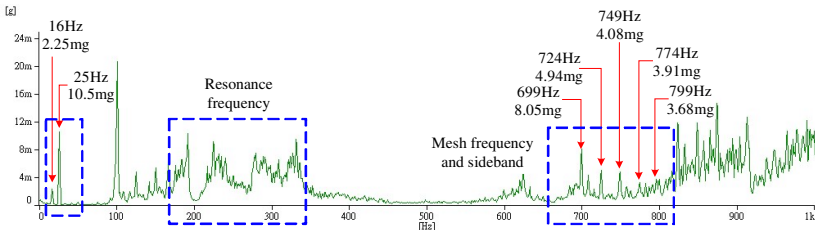


Fig. 7. The frequency spectrum diagram of the gear skew

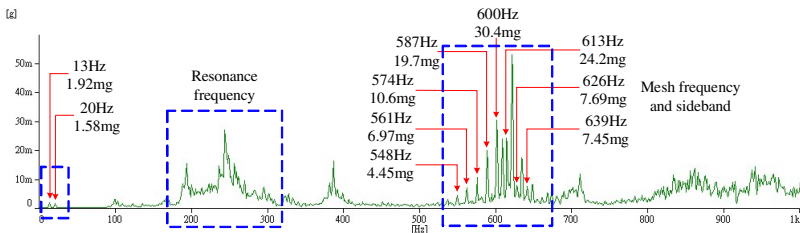


Fig. 8. The frequency spectrum diagram of the shaft not Parallel

3.2 Shaft Not Parallel

The passiveness shaft slanted seven degree angle with transmission shaft. In this case, the rotary speed of motor is set as 1200rpm (20Hz). The frequency spectrum diagram of shaft not parallel is shown in Fig. 8. The mesh frequency and sideband are distributed from 550Hz to 650Hz.

3.3 Tooth Breakage

Two of spur gear teeth above the transmission shaft are cut off. In this case, the rotary speed of motor is set as 2100rpm (35Hz). The frequency spectrum diagram of tooth breakage fault is shown in Fig. 9. The rotary speed and resonance frequency are existing under 1k Hz, the mesh frequency is over 1k Hz and the amplitude of vibration is larger than others.

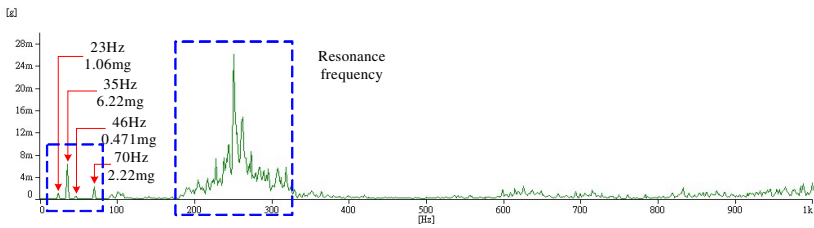


Fig. 9. The frequency spectrum diagram of the tooth breakage

3.4 Wear

There are three spur gear teeth above the transmission shaft are worn. In this case, the rotary speed of motor is set as 1500rpm (25Hz). The frequency spectrum diagram of wear fault is shown in Fig. 10.

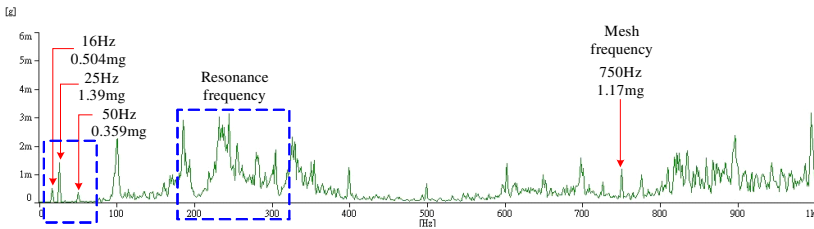


Fig. 10. The frequency spectrum diagram of the wear

3.5 Discussion

The frequency symptoms extracted from vibration signals that measured by accelerometers that mounted on the bearing housing in gear system. In Table 2, the diagnosis results are obtained by using BPNN and WNN with the trained weighting coefficients w_{ji} and w_{kj} , respectively. From case 1 to 4 belong to gear skew, shaft not parallel, tooth

breakage and wear of gear system, respectively. There are three test diagnostic samples for each case. The graphical comparisons of four cases are shown from Fig. 11 to Fig. 14. The deep and light color of the column indicates the certainty of fault by using WNN and BPNN, respectively. For each case, the column with no-grid means the Test No. 1; the column with oblique line means the Test No. 2; the column with meshed line means the Test No. 3. The diagnosis results by using WNN conform to BPNN in case 1~3, the certainties of both methods are near 1 that indicates the gear faults are satisfied with human’s diagnosis. Although the diagnosis results are satisfied with some tests of case 4, the degree of certainty (0.796) of BPNN is less than WNN (0.926) in Test No. 1 and the diagnose ability by using WNN is better than BPNN.

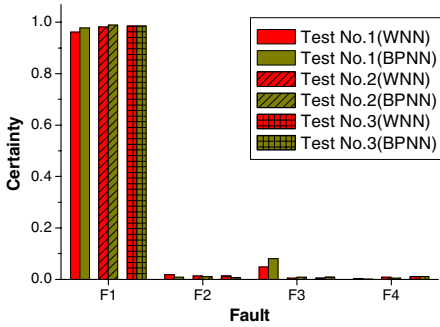


Fig. 11. The Graphical comparison for gear skew (case 1)

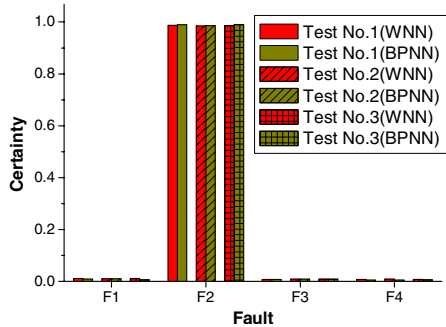


Fig. 12. The Graphical comparison for shaft not Parallel (case 2)

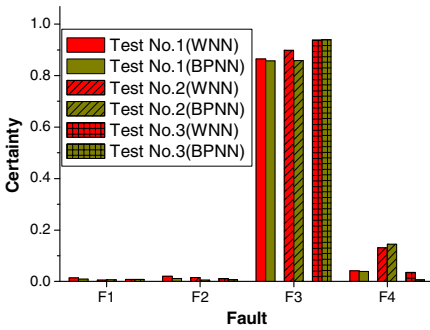


Fig. 13. The Graphical comparison for tooth breakage (case 3)

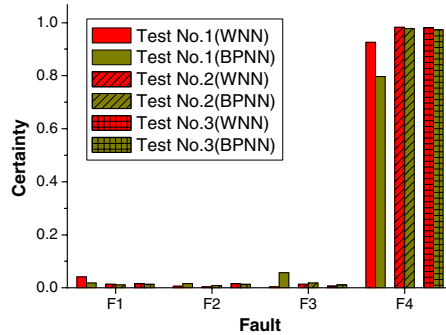


Fig. 14. The Graphical comparison for gear wear (case 4)

Furthermore, the WNN method not only has high accuracy for fault diagnosis but also has fewer weightings update epochs. In this study, there are five hidden node numbers used to train with BPNN and WNN, respectively. The network error is set 0.001, the relation between training epochs and hidden neurons numbers are listed in Table 3. When the hidden neurons numbers increase, the difference of update epochs between WNN and BPNN will increase.

Table 2. Diagnosis result (the degree of certainty)

Case No.	Faults Test No.	WNN				BPNN			
		O ₁	O ₂	O ₃	O ₄	O ₁	O ₂	O ₃	O ₄
1	1	0.962	0.018	0.048	0.003	0.978	0.008	0.080	0.001
	2	0.982	0.013	0.005	0.009	0.989	0.011	0.008	0.005
	3	0.985	0.013	0.005	0.011	0.985	0.007	0.009	0.011
2	1	0.011	0.987	0.008	0.008	0.009	0.990	0.008	0.005
	2	0.010	0.985	0.009	0.009	0.010	0.986	0.009	0.005
	3	0.011	0.986	0.009	0.008	0.006	0.990	0.009	0.006
3	1	0.014	0.020	0.865	0.041	0.009	0.012	0.857	0.039
	2	0.005	0.015	0.899	0.131	0.007	0.005	0.858	0.145
	3	0.008	0.011	0.938	0.035	0.008	0.007	0.940	0.039
4	1	0.041	0.007	0.004	0.926	0.018	0.016	0.057	0.796
	2	0.014	0.004	0.013	0.983	0.011	0.009	0.019	0.978
	3	0.016	0.016	0.007	0.981	0.013	0.013	0.011	0.973

Table 3. The relation between training epochs and hidden neurons numbers

Method \ Hidden Number	10	15	20	30	50
BPNN	2566	2079	1941	1729	1578
WNN	1804	1087	856	567	409

4 Conclusions

This study has developed WNN for fault diagnosis of gear train systems. Four cases are analyzed and diagnosis results are obtained by using WNN. Also, the results are compared to BPNN, which can be inferred as

1. The accuracy of WNN is superior to BPNN for diagnosis of gear train systems.
2. The samples training epochs by using WNN is less than BPNN.

Acknowledgements

This study was supported by grant number NSC92-2623-7-033-003-ET from National Science Council and by the Center-of-Excellence Program on Membrane Technology from Ministry of Education, Taiwan, R.O.C.

References

1. Randall, R.B.: A New Method of Modeling Gear Faults. *ASME J. Mech. Des.* **104** (1982) 259-267
2. Wang, W., Wong, A.K.: Autoregressive Model-Based Gear Fault Diagnosis. *ASME J. Vib. Acoust.* **124** (2002) 172-179

3. Fong, A.C.M., Hui, S.C.: An Intelligent Online Machine Fault Diagnosis System. *Computing and Control Engineering Journal* **12** (2001) 217-223
4. Kang, Y., Wang, C.C., Chang, Y.P., Hsueh, C.C., Chang, M.C.: Certainty Improvement in Diagnosis of Multiple Faults by Using Versatile Membership Functions for Fuzzy Neural Networks. *Lecture Notes in Computer Science* **3973** (2006) 370-375
5. Zhang, Q., Benveniste, A.: Wavelet Network. *IEEE Trans. ON Neural Networks* **3** (1992) 889-898

Appendix 1: Symbols for Frequency Symptoms

S_1, S_2 : The rotary frequency of the transmission and passiveness shaft.

S_3, S_4 : The twice rotary frequency of the transmission and passiveness shaft.

$S_5 \sim S_7$: The 1x, 2x, 3x toothmeshing frequency, respectively.

S_8 : The 1x toothmeshing frequency \pm rotary frequency of the transmission shaft.

S_9 : The 1x toothmeshing frequency \pm rotary frequency of the passiveness shaft.

S_{10} : The 2x toothmeshing frequency \pm rotary frequency of the passiveness shaft.

An Adaptive Threshold Neural-Network Scheme for Rotorcraft UAV Sensor Failure Diagnosis

Juntong Qi^{1,2}, Xingang Zhao^{1,2}, Zhe Jiang^{1,2}, and Jianda Han¹

¹ Robotics Laboratory, Shenyang Institute of Automation,
Chinese Academy of Sciences (CAS),
114# Nanta Street, Shenyang, P.R. China

² Graduate School, Chinese Academy of Sciences
Beijing, P.R. China
{Qijt,Zhaoxingang,Zhjiang,Jdhan}@sia.cn

Abstract. This paper presents an adaptive threshold neural-network scheme for Rotorcraft Unmanned Aerial Vehicle (RUAV) sensor failure diagnosis. The approach based on adaptive threshold has the advantages of better detection and identification ability compared with traditional neural-network-based scheme. In this paper, the proposed scheme is demonstrated using the model of a RUAV and the results show that the adaptive threshold neural-network method is an effective tool for sensor fault detection of a RUAV.

1 Introduction

Unmanned aerial vehicles (UAV) are useful for many applications where human intervention is considered difficult or dangerous. Additionally, Rotorcraft UAV (RUAV) can operate in many different flight modes which the fixed-wing one is unable to achieve, such as vertical take-off/landing, hovering, lateral flight, pirouette, and bank-to-turn. However, the RUAV do not have the graceful degradation properties of fixed wing aircrafts or airships in case of failures. Therefore, a failure in any part of the RUAV can be catastrophic. If the failure is not detected, identified and accommodated, the RUAV may crash.

State estimation or observation Sensor Fault Detection and Identification (SFDI) techniques have been widely used based on Kalman [1] [2] or other filters [3]. However, it is well known that these schemes might perform poorly in the presence of significant nonlinearities and uncertainties [4]. As an alternative method, neural-network (NN) based approach for SFDI (NNSFDI) have been proposed and developed in recent decade [5] [6] [7]. Because of the online and offline learning, the NNSFDI does not require modeling, is robust and has a high potential to handle nonlinearities.

In this paper, the detection and identification of sensor failure in RUAV is investigated and the design of a sensor fault diagnosis scheme is presented. A NN based scheme, with an adaptive threshold algorithm, is applied. Simulations with the adaptive threshold NNSFDI have been conducted.

2 NNSFDI Scheme

Neural networks with a three-layer BP network structure are used to approximate the nonlinear continuous functions. The three-layer neural network shown in Fig 1 with the input layer, the hidden layer and the output layer is used in NNSFDI scheme. In this study, we adopted the following sigmoidal activation function:

$$\sigma(z) = 1/(1 + e^{-z}) \tag{1}$$

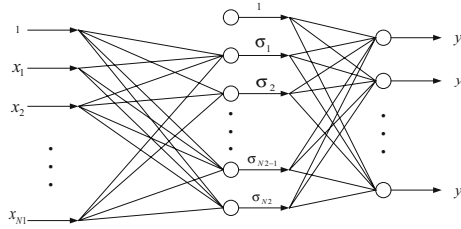


Fig. 1. Three-layer neural networks structure

The application of NNSFDI can be separate in three steps:

- Fault Detection (FD) — when a sensor failure occurs, it can be detected by the fault diagnosis scheme;
- Fault Type Identification (FTI) — when the scheme detect a failure, the FTI distinguish the type of the fault such as angular velocity, acceleration and so on;
- Fault Identification (FI) — the detected failure which supported by FD algorithm is identified to the specific sensor channel.

2.1 NNSFDI Relevant Parameters

The SFDI scheme is based on the observability of a rotorcraft UAV system. Using online learning NN estimators, the SFDI problem can be approached by introducing a main set of ‘m’ NNs (MNNs) and a set of ‘n’ decentralized NNs (DNNs), where ‘m’ is the number of the types of the sensors in the flight control system and ‘n’ is the number of the specified sensors, which are the same kind, for which a SFDIA is desired. The MNNs and DNNs are classified as Table 1 with the rotorcraft UAV.

Let us assume without loss of generality that MNN-i is considered: we denote the MNN-i outputs as $\widehat{d1}_{MNN-i}(k)$, $\widehat{d2}_{MNN-i}(k)$ and $\widehat{d3}_{MNN-i}(k)$. They are computed at time ‘k’, using measurements form time instant ‘k-t1’ to ‘k-t2’. The inputs to MNN-i are consisted of the measurements form the onboard sensors. The DNNs’ outputs are $\widehat{d1}_{MNN-i,DNN}(k)$, $\widehat{d2}_{MNN-i,DNN}(k)$ and $\widehat{d3}_{MNN-i,DNN}(k)$ which are the same state variables to the MNNs’, but the input of each DNN does not include measurements from the respective sensor.

Table 1. The MNNs and DNNs variables

Parameter	MNN-1	MNN-2	MNN-3	MNN-4
catalog	angular velocity	Euler angle	acceleration	position
DNN-1 ($d1$)	p	u	ax	gx
DNN-2 ($d2$)	q	v	ay	gy
DNN-3 ($d3$)	r	w	az	gz

In the rotorcraft UAV system, the SFDI scheme specifies 4 MNNs and 3 DNNs for each MNN catalog. As is shown in the Tabl 1, the MNN-1 detects the angular velocities' failure around 3-axis which are presented by DNN-1 to DNN-3 separately. The Euler angel, the acceleration and the position of the helicopter are diagnosed by MNN-2, MNN-3 and MNN-4 as a result of a similar set of sensor variables present the same standard deviation, sensor noise and system noise. Fig. 2 shows a block diagram of the NNSFDI process while Fig. 3 shows a block diagram of the NNSFDI structure.

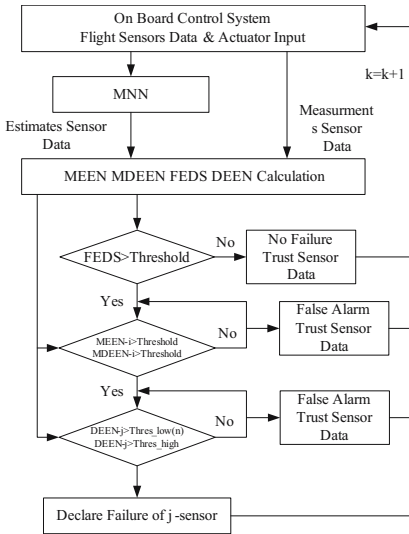


Fig. 2. Block diagram of the NNSFDI scheme

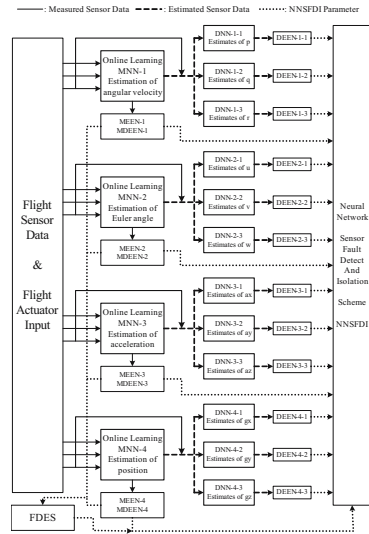


Fig. 3. Block diagram of the NNSFDI structure

It has been studied that the incidence of a sensor failure will present a perturbation of the normal dynamic response of the aircraft which translates into large discrepancies between the outputs of the MNNs and DNNs, on the one hand, and the actual measurements (i.e. $d1(k)$, $d2(k)$ and $d3(k)$) on the other. Now we define four different failure detection parameters below. One is the MNN

Estimation Error Norm (MEEN) parameter which is get by the MNN estimates and the actual measurements:

$$MEEN - i(k) = \sum_{g=1}^n \left\| dg_{MNN-i}(k) - \widehat{dg}_{MNN-i}(k) \right\|_2. \tag{2}$$

where n is the number of *DNNs* in *MNN-i*.

The second parameter is MNN and DNN Estimation Error Norm (MDEEN) which is computed by MNN and DNN estimations:

$$MDEEN - i(k) = \sum_{g=1}^n \left\| \widehat{dg}_{MNN-i,DNN}(k) - \widehat{dg}_{MNN-i}(k) \right\|_2. \tag{3}$$

where n is the number of *DNNs* in *MNN - i*.

The third parameter is DNN Estimation Error Norm (DEEN) which is computed MNN estimates and the actual measurements:

$$DEEN_{MNN-i,dg}(k) = \left\| dg(k) - \widehat{dg}_{MNN-i,DNN}(k) \right\|_2. \tag{4}$$

where $g = 1, 2, \dots, n$, n is the number of *DNNs* in *MNN - i*.

The last parameter is Fault Detection Error Summation (FDES) which is computed by MEEN and MDEEN, the μ_m and v_m are weight coefficients for fault detection.

$$FDES(k) = \sum_{m=1}^n \mu_m MEEN - m(k) + \sum_{m=1}^n v_m MDEEN - m(k). \tag{5}$$

where n is the number of *MEEN* in *MDEEN* and $\sum \mu_m = 1, \sum v_m = 1$.

2.2 Fault Detection

The occurrence of the sensor failure leads to large value of FDES between the measurements and the MNN estimates, in particular large values of MEEN or MDEEN. Then, the condition for a sensor (sensors) failure(s) to be declared is:

$$FDES(k) \geq \min(MEEN - i_{threshold}, MDEEN - i_{threshold}). \tag{6}$$

The thresholds $MEEN - i_{threshold}$ and $MDEEN - i_{threshold}$ are based on the adaptive threshold algorithm described in the Sections 3.

2.3 Fault Type Identification

When a sensor failure declared, the fault type identification phase consists of differentiating among the angular rate, the acceleration, the Euler angle and the position. For different types of sensor, different error norms associated with it

are computed for identification which hierarchical level can decrease the computational difficulties in the fault detection phase. The criterion for generic failure type identification is:

$$\begin{aligned}
 &MEEN - i(k) \geq MEEN - i_{Threshold} \\
 \text{or } &MDEEN - i(k) \geq MDEEN - i_{Threshold}.
 \end{aligned}
 \tag{7}$$

The MEEN provides better performance for step-type sensor failures whereas the MDEEN performs better for ramp-type of sensor failures. When the criterion is fulfilled, the corresponding type of sensor fault is identified.

2.4 Fault Identification

In FI phase, we propose a two threshold approach to reduce the error fault declaration while avoid failing to report a failure. A lower and a higher threshold level are selected for the DNNs. If a lower threshold for the *i*-th DNN is exceeded once, the status of the corresponding *i*-th sensor is declared suspect and the numerical architecture of the *i*-th DNN is not updated. Should this status persist for a certain number of time instants and/or should the estimation error in successive time instants exceed the higher threshold, the sensor is then declared failed. The principle of the FI is shown below:

$$\begin{aligned}
 &DEEN_{MNN-i,dg}(k) \geq DEEN_{Threshold}^{low} \quad (n \text{ times}) \\
 \text{or } &DEEN_{MNN-i,dg}(k) \geq DEEN_{Threshold}^{high}.
 \end{aligned}
 \tag{8}$$

3 Adaptive Threshold Algorithm

Previous studies on the sensor fault detection based on NN schemes always propose constant threshold that the SFDI can not appropriate to the multiple flight situations. The variation of the thresholds for SFDI has the advantage of the goal to achieve maximum “correct detection”/“false alarm” ratios.

The thresholds which present in the SFDI scheme are imposed on “Thresh” is given by the following relationship:

$$\begin{aligned}
 &Thresh_{Adaptive} = aver(Thresh) + \alpha \cdot dev(Thresh) \\
 &\quad + \beta \cdot roc(Thresh) + bias(Thresh).
 \end{aligned}
 \tag{9}$$

Where *aver(Thresh)* is the average value of *Thresh*, *dev(Thresh)* is the standard deviation of *Thresh*, *roc(Thresh)* is the rate of change of *Thresh*, *bias(Thresh)* is the bias of *Thresh* which is computed based on the above equations, α is a deviation bound factor and β is a rate of change bound factor.

The thresholds must be estimated within the SFDI scheme and the parameters used in the equation are listed in Table 2.

Table 2. The MNNs and DNNs variables

Parameter	FEDS	MEEN-1	MEEN-2	MEEN-3	MEEN-4
α	2.4	3.2	5.7	1.9	5.3
β	0.14	0.06	0.11	0.09	0.02
bias	0.7	0.4	0.6	0.4	0.3

Parameter	MDEEN-1	MDEEN-2	MDEEN-3	MDEEN-4
α	3.1	1.8	0.9	2.5
β	0.01	0.02	0.05	0.01
bias	0.3	0.5	0.5	0.4

The averages of the thresholds are computed online based on the last 20 flight data using the following average filter:

$$\bar{X} = \frac{1}{20} \sum_{i=1}^{20} X(n - i) \tag{10}$$

Where n is the current flight data sequence. The logical algorithm of computing the SFDI adaptive threshold is shown in Fig. 4

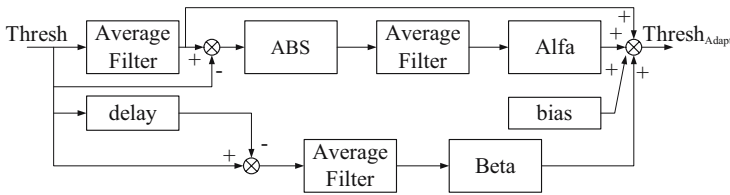


Fig. 4. Block diagram of the adaptive threshold scheme

4 Simulation

The flight data used for the NN scheme for offline and online learn is the data acquired from the SIA-Heli-90 RUAV platform. With the purpose of illustrating the functionality of the SFDI scheme with NN scheme based on adaptive thresholds and the minimize the false detections as compare with the equivalent fixed thresholds scheme, two cases of failure have been simulated using both algorithms.

Failure # 1: A heading failure of compass at t = 30s.

Failure # 2: A pitch rate failure of IMU at t = 30s.

In the first case of failure, we simulate a compass sensor fault that the heading channel involving drifting bias of 10° with fast transient period of 0.1s. As is

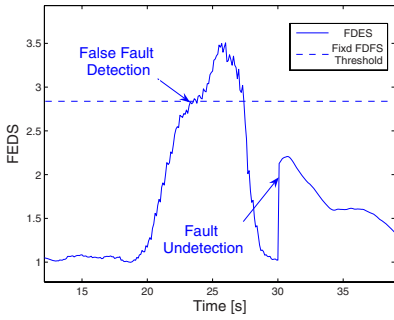


Fig. 5. Fixed FDES threshold based SFDI scheme in *Failure # 1*

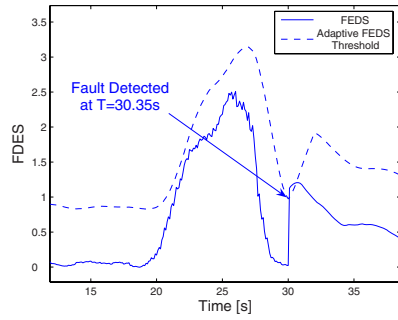


Fig. 6. Adaptive FDES threshold based SFDI scheme in *Failure # 1*

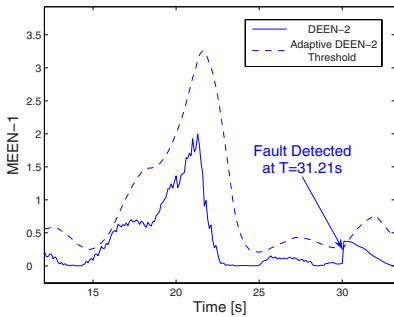


Fig. 7. Fixed MEEN-1 threshold based SFDI scheme in *Failure # 2*

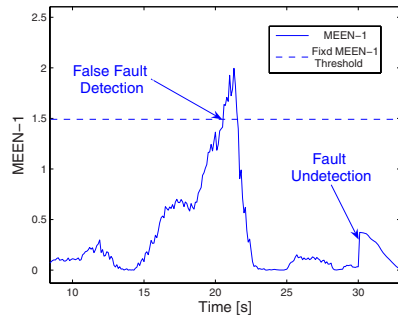


Fig. 8. Adaptive MEEN-1 threshold based SFDI scheme in *Failure # 2*

shown in Fig. 5, the fixed FDES threshold is exceeded at nominal conditions that without any occurrence of a failure, but undetected in the real failure instance. The fixed threshold SFDI scheme declares a false detection at $t = 22.9s$ while the heading channel of compass failure occurring at $t = 30s$ remains undetected. With the change of the scheme to the adaptive thresholds, however, the adaptive algorithm does not produce false failure detection and detects the real system failure immediately at $t = 30.35s$ after its happening which is shown in the Fig. 6. The following MEEN-2 and DNN-1 phases can work well with the adaptive threshold algorithm and detect the compass fault in heading channel at $t = 30.91s$.

In the second case of failure, a pitch rate failure of IMU is simulated involving a drifting bias of $10^\circ/s$ with fast transient period of $0.1s$. The FDES scheme can detect the yaw rate failure not only with the fixed threshold algorithm but also the adaptive threshold algorithm which are not shown the figure in this paper. But to the MEEN-1 which is the compass FTI parameter, the fix MEEN-1 threshold is detect a fake failure and undetected in the real failure which is presented in the Fig. 7. The fixed MEEN-1 threshold is exceeded at nominal conditions that without any occurrence of a failure at $t = 20.3s$ but undetected

in the real failure at $t = 30$ s. In the Fig. 8, we change to adaptive MEEN-1 thresholds. The algorithm detects the real IMU failure immediately at $t = 31.21$ s after the fault occurring at $t = 30$ s and does not declare a false detection. The following DNN-1 phase can work well with the adaptive threshold algorithm and detect the compass fault in heading channel at $t = 31.82$ s.

5 Conclusion

In this paper, we propose a neural-network fault detection and identification scheme based on adaptive threshold. The adaptive threshold algorithm can improve the performance of the SFDI scheme and reduce the probability of false fault declaration. In addition, the adaptive threshold approach eliminates the need for thresholds changing with flight condition varying.

The proposed SFDI scheme has been tested with the flight data acquired from our SIA-Heli-90 RUAV testbed [8], and shown significantly improved performance than fixed threshold algorithm.

Acknowledgments. This work is partially supported by the National High Technology Research and Development Program of China (863 Program) (No. 2003AA421020).

References

1. Song, Q., Qi, J.T. and Han, J.D.: An adaptive UKF algorithm and its application in mobile robot control, Proceedings of the 2006 IEEE International Conference on Robotics and Biomimetics (2006) 1117-1122.
2. Kameyama, T. and Ochi, Y.: Design of a failure tolerant flight control system with unscented kalman filter, AIAA Guidance, Navigation, and Control Conference (2005) 6345.
3. Shin, J.Y., Wu, N.E. and Belcastro, C.: Linear parameter varying control synthesis for actuator failure: based on estimated parameter, AIAA Guidance, Navigation, and Control Conference (2002) 4546.
4. Napolitano, R., An, Y. and Seanor, B.: Kalman filters and neural-network schemes for sensor validation in flight control systems, IEEE Transaction on Control System Technology **6** (1998) 596-611.
5. Napolitano, R., An, Y. and Seanor, B.: A fault tolerant flight control system for sensor and actuator failures using neural networks, Aircraft Design **3** (2000) 103-128.
6. Jakubek, S. and Strasser, T.: Fault diagnosis using neural networks with ellipsoidal basis functions, Proceeding of American Control Conference (2002) 4513-4518.
7. Perhinschi, M.G., Napolitano, R., Campa, G. and Fravolini, M.L.: An adaptive threshold approach for the design of an actuator failure detection and identification scheme, IEEE Transaction on Control System Technology **14** (2006) 519-525.
8. Qi, J.T., Zhao, X.G., Jiang, Z. and Han, J.D.: Design and implement of a rotorcraft UAV test bed, Proceedings of the 2006 IEEE International Conference on Robotics and Biomimetics (2006) 109-144.

KPCA Plus FDA for Fault Detection

Peiling Cui and Jiancheng Fang

School of Instrumentation Science & Opto-electronics Engineering,
Beijing University of Aeronautics & Astronautics, Beijing 100083, China
PeilingCui@buaa.edu.cn

Abstract. Kernel principal component analysis (KPCA) is widely used for fault detection. In this paper, a KPCA plus Fisher discriminant analysis (FDA) scheme is adopted to improve the fault detection performance of KPCA. Simulation results are given to show the effectiveness of the improvements for fault detection performance in terms of high fault detection rate.

1 Introduction

The demands for product quality and operation safety in the process industry have spurred the recent development of many fault diagnosis methods. Multivariate statistical methods such as principal component analysis (PCA), independent component analysis (ICA) and partial least square (PLS) have been widely applied in chemical industry for fault diagnosis [1–5]. Among these methods, PCA is the most popular one. PCA represents high-dimensional process data in a reduced dimension, which brings convenience for process monitoring. However, linear correlation among the variables is assumed in PCA, which degrades the performance of PCA in monitoring the nonlinear systems.

To cope with this problem, extended versions of PCA suitable for handling nonlinear systems have been developed. The existing approaches include methods based on neural network [6–10], methods based on principal curve [11], methods based on kernel function [12–17]. For methods based on neural network, auto-associative neural network (AANN) [6], input-training neural network (ITNN) [7–9] et al are used. For methods based on kernel function, kernel principal component analysis (KPCA) first put forward by Schölkopf et al [12] is a new nonlinear extension of PCA.

KPCA computes the principal components in a high-dimensional feature space F , which is nonlinearly related to the input space. Since a PCA in F can be formulated in terms of the dot products in F , this same formulation can also be performed using kernel functions (the dot product of two data in F) without explicitly computing in F .

KPCA has already shown better performance than PCA in several fields [12–14], but a few of applications in the field of fault diagnosis is reported [15–17]. In Ref. [15–17], nonlinear process monitoring method based on KPCA and dynamic KPCA are investigated. However, KPCA performs a PCA in feature space F . In this sense, KPCA aims at reconstruction but not classification, so it is not fit for the task of fault detection very well.

In this paper, a KPCA plus Fisher discriminant analysis scheme for fault detection is provided. Fisher discriminant analysis (FDA) [18] is a well-known method for feature extraction and dimension reduction. It aims to find an optimal transformation by minimizing the within-class distance and maximizing the between-class distance simultaneously, which makes up for the above mentioned shortcoming of KPCA method.

2 KPCA

The basic idea of KPCA is to map the input data \mathbf{x} into a feature space F first via a nonlinear mapping ϕ , and then perform a linear PCA in F . However, it is difficult to do so directly because the dimension h of the feature space F can be arbitrarily large or even infinite. In implementation, the implicit feature vector in F does not need to be computed explicitly, while it is just done by computing the inner product of two vectors in F with a kernel function.

Let $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n \in R^m$ be the n training samples (column vectors) for KPCA learning. By the nonlinear mapping ϕ , the measured inputs are extended into the hyper-dimensional feature space as follows

$$\phi: \mathbf{x} \in R^m \rightarrow \phi(\mathbf{x}) \in F^h \tag{1}$$

The mapping of \mathbf{x}_i is simply noted as $\phi(\mathbf{x}_i) = \phi_i$. The sample covariance in the feature space can be constructed by

$$\mathbf{C}_\phi = \frac{1}{n} \sum_{i=1}^n (\phi_i - \mathbf{m}_\phi)(\phi_i - \mathbf{m}_\phi)^T, \tag{2}$$

here, column vector $\mathbf{m}_\phi = \sum_{i=1}^n \phi_i / n$. Nonzero eigenvalues of covariance matrix \mathbf{C}_ϕ are positive, and Schölkopf [12] et al. has suggested the following way to find these positive eigenvalues.

It is easy to see that every eigenvector \mathbf{v} of \mathbf{C}_ϕ can be linearly expanded by

$$\mathbf{v} = \sum_{i=1}^n \alpha_i \phi_i. \tag{3}$$

To obtain the coefficients $\alpha_i (i=1, 2, \dots, n)$, a kernel matrix \mathbf{K} with size $n \times n$ is defined, and its elements are determined by virtue of kernel tricks.

$$K_{ij} = \phi_i^T \phi_j = (\phi_i \bullet \phi_j) = k(\mathbf{x}_i, \mathbf{x}_j) \tag{4}$$

here, $k(\mathbf{x}_i, \mathbf{x}_j)$ is the calculation of the inner product of two vectors in F with a kernel function. Centralize \mathbf{K} by

$$\bar{\mathbf{K}} = \mathbf{K} - \mathbf{1}_n \mathbf{K} - \mathbf{K} \mathbf{1}_n + \mathbf{1}_n \mathbf{K} \mathbf{1}_n, \tag{5}$$

here, matrix $\mathbf{1}_n = (1/n)_{n \times n}$.

Let column vectors $\boldsymbol{\gamma}_i (i=1,2,\dots,p; 0 < p \leq n)$ be the orthonormal eigenvectors of $\bar{\mathbf{K}}$ corresponding to the p largest positive eigenvalues, $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_p$. The orthonormal eigenvectors $\boldsymbol{\beta}_i (i=1,2,\dots,p)$ of \mathbf{C}_ϕ corresponding to the p largest positive eigenvalues, $\lambda_1, \lambda_2, \dots, \lambda_p$, are

$$\boldsymbol{\beta}_i = \frac{1}{\sqrt{\lambda_i}} (\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \dots, \boldsymbol{\phi}_n) \boldsymbol{\gamma}_i. \tag{6}$$

To a new column-vector sample \mathbf{x}_{new} , the mapping to the feature space is $\phi(\mathbf{x}_{\text{new}})$. The projection of \mathbf{x}_{new} onto the eigenvectors $\boldsymbol{\beta}_i (i=1,2,\dots,p)$ is $\mathbf{t} = (t_1, t_2, \dots, t_p)^T$, and it is also called KPCA-transformed feature vector.

$$\mathbf{t} = (\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \dots, \boldsymbol{\beta}_p)^T \phi(\mathbf{x}_{\text{new}}). \tag{7}$$

The i th KPCA-transformed feature t_i can be obtained by

$$\begin{aligned} t_i &= \boldsymbol{\beta}_i^T \phi(\mathbf{x}_{\text{new}}) = \frac{1}{\sqrt{\lambda_i}} \boldsymbol{\gamma}_i^T (\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \dots, \boldsymbol{\phi}_n)^T \phi(\mathbf{x}_{\text{new}}) \\ &= \frac{1}{\sqrt{\lambda_i}} \boldsymbol{\gamma}_i^T [k(\mathbf{x}_1, \mathbf{x}_{\text{new}}), k(\mathbf{x}_2, \mathbf{x}_{\text{new}}), \dots, k(\mathbf{x}_n, \mathbf{x}_{\text{new}})]^T, \tag{8} \end{aligned}$$

$i = 1, 2, \dots, p.$

By using Eq. (8), the KPCA-transformed feature vector of a new sample vector can be obtained.

3 KPCA Plus Fisher Discriminant Analysis

3.1 Fisher Discriminant Analysis

Fisher discriminant analysis (FDA) [18] is a well-known method for feature extraction and dimension reduction. It aims to find an optimal transformation by minimizing the within-class distance and maximizing the between-class distance simultaneously, and thus the FDA-transformed vectors are more discriminative than the original input sample vector.

Let $\mathbf{X} \in R^{m \times n}$ be the input set for FDA training with n sample vectors, and \mathbf{x} be a column vector of the input set \mathbf{X} with m elements. Suppose there are c classes of sample types, and $X_i (i=1,2,\dots,c)$ be the subsets of \mathbf{X} with $n_i (i=1,2,\dots,c)$ sample vectors, here $n = \sum_{i=1}^c n_i$. The within-class scatter matrix of class l is given by

$$S_l = \sum_{\mathbf{x}_i \in X_l} (\mathbf{x}_i - \bar{\mathbf{x}}_l)(\mathbf{x}_i - \bar{\mathbf{x}}_l)^T, \tag{9}$$

here $\bar{\mathbf{x}}_l = \frac{1}{n_l} \sum_{\mathbf{x}_i \in X_l} \mathbf{x}_i$ is the sample mean of class $l(l=1,2,\dots,c)$. The within-class-scatter matrix of total samples is

$$S_w = \sum_{i=1}^c S_i. \tag{10}$$

The between-class scatter matrix S_b is given by

$$S_b = \sum_{i=1}^c n_i (\bar{\mathbf{x}}_i - \bar{\mathbf{x}})(\bar{\mathbf{x}}_i - \bar{\mathbf{x}})^T, \tag{11}$$

here $\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i$ is the total sample mean vector, whose elements correspond to the means of the columns of \mathbf{X} .

Performing FDA in the sample space means maximizing the between-class scatter S_b and minimizing the within-class scatter S_w . This is equivalent to maximizing the following function

$$J(\mathbf{w}) = \arg \max_{\mathbf{w} \neq \mathbf{0}} \frac{|\mathbf{w}^T S_b \mathbf{w}|}{|\mathbf{w}^T S_w \mathbf{w}|}. \tag{12}$$

The FDA vectors are equal to the generalized eigenvectors of the eigenvalue problem

$$S_b \mathbf{w}_i = \lambda_i S_w \mathbf{w}_i, \tag{13}$$

where the eigenvalues λ_i indicate the degree of overall separability among the classes, and the norm of \mathbf{w}_i is usually chosen to be $\|\mathbf{w}_i\|=1$. The FDA transform matrix can be written as follows

$$\mathbf{W}_{FDA} = [\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_a], \tag{14}$$

where $\mathbf{W}_{FDA} \in R^{m \times a}$ is a matrix with a FDA vectors as columns. Note that there are $c-1$ nonzero generalized eigenvalues at most, and so the upper bound on a is $c-1$.

For a new sample \mathbf{x}_{new} , the FDA-transformed feature \mathbf{y}_{FDA} can be calculated as follows

$$\mathbf{y}_{FDA} = \mathbf{W}_{FDA}^T \mathbf{x}_{new}. \tag{15}$$

3.2 KPCA Plus FDA

In section 2, we have learned that KPCA performs a PCA in the feature space. In this sense, KPCA aims at reconstruction but not classification, so it is not fit for the task of fault detection very well.

According to Cover’s theorem [22], the nonlinear data structure in the input space is more likely to be linear after high-dimensional nonlinear mapping. Hence, FDA as a linear discriminative analysis method can be applied. After using KPCA, the input sample is projected onto feature space (see Eq. (7)) to get the KPCA-transformed feature vector. In fault detection problems, normal samples and faulty samples for training are all labeled as different classes, it makes sense to use this information to build a more reliable method to classify faulty sample from normal data. FDA is an example of a linear, class specific method, in the sense that it tries to “shape” the scatter in order to make it more reliable for fault detection. In this paper, after performing KPCA, FDA is performed in the projection space to get the features more discriminative for fault detection. Fig.1 gives the flow chart of KPCA plus FDA method.

For a new sample \mathbf{x}_{new} , we can rewrite Eq. (8) as

$$\mathbf{y}_{KPCA} = \mathbf{W}_{KPCA}(\mathbf{x}_{new}), \tag{16}$$

where \mathbf{W}_{KPCA} is the KPCA transform function, and \mathbf{y}_{KPCA} is the KPCA-transformed feature vector with length p . Then KPCA-FDA-transformed feature of \mathbf{x}_{new} can be derived as follows

$$\mathbf{y}_{KPCA+FDA} = \mathbf{W}_{FDA}^T \mathbf{W}_{KPCA}(\mathbf{x}_{new}), \tag{17}$$

where $\mathbf{y}_{KPCA+FDA}$ can be used for fault detection after KPCA-FDA transform.

3.3 KPCA Plus FDA for Fault Detection

The monitoring method based on KPCA plus FDA is similar to that based on FDA method [23], and then T^2 statistic and Q -statistic in the feature space can be interpreted in the same way.

$$T^2 = \mathbf{y}_{KPCA+FDA}^T \mathbf{S}^{-1} \mathbf{y}_{KPCA+FDA}, \tag{18}$$

where $\mathbf{y}_{KPCA+FDA}$ is the KPCA-FDA-transformed feature obtained from Eq. (17), and \mathbf{S} is the covariance matrix of KPCA-FDA-transformed feature vectors of the training samples.

The $100\alpha\%$ control limit for T^2 statistic is obtained using the F -distribution

$$T_\alpha^2 = \frac{(n^2 - 1)a}{n(n - a)} F_\alpha(a, n - a), \tag{19}$$

where $F_\alpha(a, n - a)$ is the upper $100\alpha\%$ critical point of the F -distribution with a and $n - a$ degrees of freedom [20-21].

The residual vector of FDA can be defined as [23].

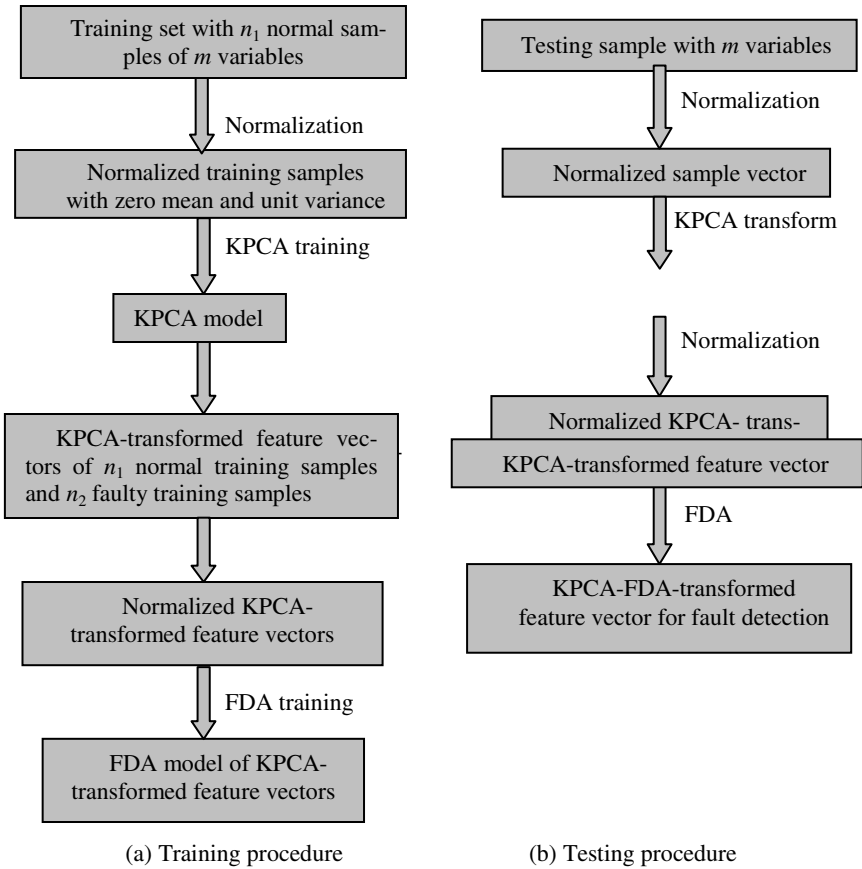


Fig. 1. Flow chart of KPCA plus FDA method

$$\begin{aligned}
 \mathbf{r} &= (\mathbf{I} - \mathbf{W}_{FDA} \mathbf{W}_{FDA}^T) \mathbf{y}_{KPCA} \\
 &= (\mathbf{I} - \mathbf{W}_{FDA} \mathbf{W}_{FDA}^T) \mathbf{W}_{KPCA} (\mathbf{x}_{new}),
 \end{aligned} \tag{20}$$

where $\mathbf{W}_{FDA} \in R^{p \times a}$, $\mathbf{W}_{KPCA}(\mathbf{x}) \in R^{p \times 1}$ is the KPCA-transformed feature vector of sample \mathbf{x} , and the resulting residual vector $\mathbf{r} \in R^{p \times 1}$.

The Q statistic is [24]

$$Q = \mathbf{r}^T \mathbf{r}. \tag{21}$$

The $100(1-\alpha)\%$ control limit for Q statistic can be computed from its approximate distribution [24].

$$Q_\alpha = \theta_1 \left[\frac{c_\alpha (2\theta_2 h_0^2)^{1/2}}{\theta_1} + 1 + \frac{\theta_2 h_0 (h_0 - 1)}{\theta_1^2} \right]^{1/h_0}, \tag{22}$$

where $\theta_j = \sum_{j=a+1}^n \sigma_j^{2i}$, $h_0 = 1 - (2\theta_1\theta_3)/(3\theta_2^2)$, and c_α is the normal deviate corresponding to the upper $100(1 - \alpha)$ percentile. $\sigma_j (j = a + 1, \dots, n)$ is obtained via singular value decomposition as follows

$$\frac{1}{\sqrt{n-1}} \mathbf{Y}_{KPCA}^T = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T, \tag{23}$$

here $\mathbf{Y}_{KPCA} \in R^{p \times n}$ is the KPCA-transformed feature vectors of the normal training data, and $\mathbf{\Sigma} \in R^{n \times p}$ contains the nonnegative real singular values of decreasing magnitude ($\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_p$).

4 Simulations

4.1 System Description

To validate the fault detection performance of KPCA plus FDA scheme, the following system is simulated and analyzed. This system was used in [15-16] and was originally used by Dong and McAvoy [11]. It consists of three variables nonlinearly related with t , which are generated by

$$\mathbf{x} = \begin{bmatrix} t \\ t^2 - 3t \\ -t^3 + 3t^2 \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix}, \tag{24}$$

where white Gaussian noise $e_i \sim N(0, 0.01)$ and $t \in [0.01, 1]$. Letting $\alpha = 0.01$, 100-run Monte Carlo simulations are performed.

Gaussian kernel function (Eq. (25)) is used

$$k(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{\sigma}\right), \tag{25}$$

where $\|\cdot\|$ is l_2 -norm and σ is the width of a Gaussian function.

Training samples and testing samples used in KPCA method are:

1) Training data consists of 100 samples (samples 1–100) obtained using Eq. (24).

2) The testing data set consists of two parts: the first 100 samples (samples 101–200) are collected in the same condition as the normal condition in training data, and the second 100 testing samples (samples 201–300) are obtained using the system but with x_3 expressed as $x_3 = -1.1t^3 + 3.2t^2 + e_3$.

In the training procedure of KPCA plus FDA method, normal training samples and faulty training samples are all needed. In the following, training samples and testing samples used in KPCA plus FDA method are given:

1) Normal training data is the same as the training data used in KPCA method. Another testing data set is generated by the same method as used in KPCA method, and then samples 101-200 are used as faulty training data.

2) The testing data set is the same as that used in KPCA method.

4.2 Fault Detection Results

Fault detection results of KPCA method (data from Ref. [16]) are compared with that of KPCA plus FDA method proposed in this paper in terms of type I error and type II error, as shown in Table 1. Type I error corresponds to false alarm rate, and type II error corresponds to missing alarm rate. Because there are two classes of samples, the number of non-zero eigenvalue of KPCA plus FDA method is one. The result of KPCA plus FDA method in Table 1 is obtained using the eigenvector corresponding to the non-zero eigenvalue.

Table 1 shows that, kernel parameter σ plays an important role in fault detection. For $\sigma = 0.02 - 0.10$, type I error and type II error of KPCA plus FDA method are all smaller than KPCA method. For $\sigma = 0.01$, although type II error of KPCA plus FDA method (12.6%) is a little higher than that of KPCA method (9.2%), type I error of KPCA plus FDA method (4.2%) is significantly smaller than that of KPCA method (76.7%). It can be seen that, KPCA plus FDA method has shown superior performance of fault detection in terms of type I error and type II error compared to KPCA method.

The selection of kernel parameters is critical to KPCA calculation since the degree to which the nonlinear characteristic of a system is captured depends on kernel function. For KPCA plus FDA method, a wider kernel parameter range is reasonable, and then KPCA plus FDA scheme has stronger adaptive ability to kernel parameters.

Table 1. Fault detection results (%) of KPCA method and that of KPCA plus FDA method

σ	Method	Type I error			Type II error		
		min	max	average	min	max	average
0.01	KPCA	68.0	84.0	76.7	2.0	18.0	9.2
	KPCA plus FDA	2.0	8.0	4.2	5.0	22.0	12.6
0.02	KPCA	34.0	62.0	20.6	10.0	33.0	20.6
	KPCA plus FDA	1.0	7.0	3.9	4.0	25.0	9.8
0.03	KPCA	18.0	45.0	31.0	23.0	40.0	31.2
	KPCA plus FDA	2.0	7.0	3.5	4.0	23.0	10.8
0.04	KPCA	10.0	30.0	19.8	31.0	51.0	40.5
	KPCA plus FDA	2.0	7.0	4.6	3.0	26.0	15.5
0.05	KPCA	30.0	23.0	12.5	37.0	59.0	49.9
	KPCA plus FDA	2.0	7.0	4.3	11.0	32.0	21.4
0.10	KPCA	0.0	5.0	2.5	70.0	90.0	77.6
	KPCA plus FDA	2.0	5.0	3.8	32.0	57.0	42.4

5 Conclusions

In recent years, KPCA has been utilized directly for nonlinear process monitoring, and it has been proven to outperform conventional PCA method. This paper focuses on the improvement of KPCA for fault detection. FDA is performed for fault detection after KPCA calculation, and is named as KPCA plus FDA method. KPCA performs a PCA in feature space. In this sense, KPCA aims at reconstruction but not classification, so it is not fit for the task of fault detection very well. FDA aims to find an optimal transformation by minimizing the within-class distance and maximizing the between-class distance simultaneously, and this makes up for the shortcoming of KPCA method. Simulations conducted on a nonlinear process have demonstrated that, KPCA plus FDA method is more efficient for fault detection than KPCA method in terms of low false alarm rate and low missing alarm rate.

References

1. Chen, J.H., Chen, H.-H.: On-line Batch Process Monitoring using MHTM-based MPCA. *Chemical Engineering Science* **61** (2006) 3223-3239
2. Lieftucht, D., Kruger, U., Irwin, G.W.: Improved Reliability in Diagnosing Faults using Multivariate Statistics. *Computers and Chemical Engineering* **30** (2006) 901-912
3. Detroja, K.P., Gudi, R.D., Patwardhan, S.C., Roy, K.: Fault Detection and Isolation Using Correspondence Analysis. *Industrial & Engineering Chemistry Research* **45** (2006) 223-235
4. Zuo, M.J., Lin, J., Fan, X.F.: Feature Separation using ICA for a One-dimensional Time Series and its Application in Fault Detection. *Journal of Sound and Vibration* **287** (2005) 614-624
5. Lee, G., Han, C.H., Yoon, E.S.: Multiple-fault Diagnosis of the Tennessee Eastman Process Based on System Decomposition and Dynamic PLS. *Industrial & Engineering Chemistry Research* **43** (2004) 8037-8048
6. Kramer, M.A.: Non-linear Principal Component Analysis using Autoassociative Neural Networks. *AIChE Journal* **37** (1991) 233-243
7. Tan, S., Mavrouniotis, M.L.: Reducing Data Dimensionality through Optimizing Neural Networks Inputs. *AIChE Journal* **41** (1995) 1471-1480
8. Jia, F., Martin, E.B., Morris, A.J.: Non-linear Principal Component Analysis for Process Fault Detection. *Computers and Chemical Engineering* **22** (1998) S851-S854
9. Geng, Z.Q., Zhu, Q.X.: Multiscale Nonlinear Principal Component Analysis (NLPCA) and its Application for Chemical Process Monitoring. *Industrial & Engineering Chemistry Research* **44** (2005) 3585-3593
10. Chen, J.H., Liao, C.-M.: Dynamic Process Fault Monitoring Based on Neural Network and PCA. *Journal of Process Control* **12** (2002) 277-289
11. Dong, D., McAvoy, T.J.: Nonlinear Principal Component Analysis—Based on Principal Curves and Neural Networks. *Computers and Chemical Engineering* **20** (1996) 65-78
12. Schölkopf, B., Smola, A.J., Müller, K.: Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Computation* **10** (1998) 1299-1399
13. Mika, S., Schölkopf, B., Smola, A.J., Müller, K.-R., Scholz, M., Ratsch, G.: KPCA and De-noising in Feature Spaces. *Advances in Neural Information Processing Systems* **11** (1999) 536-542

14. Schölkopf, B., Mika, S., Burges, C.J.C., Knirsch, P., Müller, K.-R., Rätsch, G., Smola, A.J.: Input Space Versus Feature Space in Kernel-based Methods. *IEEE Transactions on Neural Networks* **10** (1999) 1000-1016
15. Lee, J.-M., Yoo, C.K., Choi, S.W., Vanrolleghem, P.A., Lee, I.-B.: Nonlinear Process Monitoring using Kernel Principal Component Analysis. *Chemical Engineering Science* **59** (2004) 223-234
16. Choi, S.W., Lee, C., Lee, J.-M., Park, J.H., Lee, I.-B.: Fault Detection and Identification of Nonlinear Processes Based on KPCA. *Chemometrics and Intelligent Laboratory Systems* **75** (2005) 55-67
17. Cho, J.-H., Lee, J.-M., Choi, S.W., Lee, D., Lee, I.-B.: Fault Identification for Process Monitoring using Kernel Principal Component Analysis. *Chemical Engineering Science* **60** (2005) 279-288
18. Fukunaga, K.: *Introduction to Statistical Pattern Classification*. San Diego, Calif.: Academic Press (1990)
19. Wu, Y., Huang, T.S., Toyama, K.: Self-Supervised Learning for Object Based on Kernel Discriminant-EM Algorithm. *Proceedings of International Conference on Computer Vision, Kauai, HI* **1** (2001) 275-280
20. MacGregor, J.F., Kourti, T.: Statistical Process Control of Multivariate Processes. *Control Engineering Practice* **3** (1995) 403-414
21. Tracy, N.D., Young, J.C., Mason, R.L.: Multivariate Control Charts for Individual Observations. *Journal of Quality Technology* **24** (1992) 88-95
22. Haykin, S.: *Neural Networks*. Prentice-Hall, Englewood Cliffs, NJ. (1999)
23. Chiang, L. H., Russell, F. L., Braatz, R. D.: *Fault Detection and Diagnosis in Industrial Systems*. Springer, London (2001)
24. Jackson, J.E., Mudholkar, G.S.: Control Procedures for Residuals Associated with Principal Component Analysis. *Technometrics* **21** (1979) 341-349

Distribution System Fault Diagnosis Based on Improved Rough Sets with Uncertainty

Jing Dai¹ and Qiuye Sun²

¹ Educational Technology Institute of Tsinghua University, Beijing 100084
dai-j05@mails.tsinghua.edu.cn

² School of Information Science and Engineering
Northeastern University, Shenyang, Liaoning 110004, China
sunqiuye@126.com

Abstract. The volume of data with a few uncertainties overwhelms classic information systems in the distribution control center and exacerbates the existing knowledge acquisition process of expert systems. The paper describes a systematic approach for detecting superfluous data. It is considered as a "white box" rather than a "black box" like in the case of neural network. The approach therefore could offer user both the opportunity to learn about the data and to validate the extracted knowledge. To deal with the uncertainty and deferent structures of the system, rough sets and fuzzy sets are introduced. The reduction algorithm based on uncertainty rough sets is improved. The rule reliability is deduced using fuzzy sets and probability. The simulation result of a power distribution system shows the effectiveness and usefulness of the approach.

1 Introduction

In recent years, there is an increasing urgent demand for useful knowledge rather than mountains of data during the emergency condition by the operators. Quick and correct fault diagnosis has realism meaning to reduce time of electric energy's interruption and improve the reliability of power supply. It is easier with complete and exact information than without. However, when distribution system occurs fault, the certainty and integrity of information will be damaged by a good many causations [1]. In order to improve the accuracy and rapidity of fault diagnosis, it is necessary to discovery a new method that has high fault tolerant and can compress data space and filtrate error data [2]-[4]. To deal with indiscernibility, rough set theory was first proposed by Pawlak (1982) [5]. And a lot of methods are put forward based on rough sets by scholars. It is a pity that most of them make attributes reduction for discrete data and ignore the voltage and current. But for the distribution, especially in the country, the discrete signals, such as the switched and protection devices, are not exact in some cases, where the current and voltage are more important attributes for the fault diagnosis. But the rough set cannot deal with the continuous attribute directly and the hard discretization method may damage the useful information [6], [7]. And in a distribution system, for the tens of possible structures, the imaginable fault grids will be more than 1000 and the rules several times to them. The chain fault grids are too much to add up. So the classical method of establishing expert rules considering all kinds of possible faults is not enough now.

Facing to the problems, fuzzy sets are employed to handle linguistic input information (by a soft discretization to voltage and current) and ambiguity in output decision, while rough set extracts the relevant domain knowledge to the fault diagnosis from history data, and then the chain faults are deduced by the probability-rough methods. A worked examples is applied to explain the procedure of the diagnosis and a simulation on the actual power distribution proves the effectiveness and usefulness of the approach.

The rest of this paper is structured as follows. Section 2 and section 3 introduces the preliminaries and deduces the reduction and diagnosis procedure. Section 4 discusses the uncertainty of input and output varieties in the distribution. Section 5 describes the experimentation carried out on the real problem and presents the results. Section 6 concludes the paper.

2 Preliminaries

2.1 Rough Set

An uncertainty-rough classification problem can be described as follows. The universe $U = \{x_i | i = 1, \dots, n\}$ is described by the discretization attributes $\{P_1, P_2, \dots, P_p\}$. Each attribute measures some important feature of and is limited to linguistic terms $A(P_i) = \{F_{ik} | k = 1, \dots, C_i\}$. Each object $x_i \in U$ is classified by a set of classes $A(Q) = \{F_l | l = 1, \dots, C_Q\}$. Each $F_l \in A(Q)$ may be a crisp or membership function and Q is decision attribute. The set $U/P = \{F_{ik} | i = 1, \dots, p; k = 1, \dots, C_i\}$ can be regarded as a kind of partitions of U by a set of attributes P using uncertainty model.

Definition 1. [8]-[11] Given U is limited universe, the set function $P : 2^U \rightarrow [0, 1]$ is probability estimation. And the lower and upper probability of probability-rough set are defined by

$$\begin{aligned} P_{\underline{A}\alpha}(X) &= \{x \in U | P(x[x]) \geq \alpha\} \\ P_{\overline{A}\beta}(X) &= \{x \in U | P(x[x]) > \beta\} \end{aligned} \tag{1}$$

where $\leq \beta < \alpha \leq 1$.

Definition 2. Given arbitrary fuzzy set $\mu_A(x) : U \rightarrow [0, 1]; \forall x \in U$ and $F_{ik} \in U/P$. The lower and upper membership function of fuzzy-rough set are defined by

$$\begin{aligned} \mu_{\underline{A}}(F_{ik}) &= \inf_{x \in U} \max\{1 - \mu_{F_{ik}}(x), \mu_A(x)\} \\ \mu_{\overline{A}}(F_{ik}) &= \sup_{x \in U} \min\{\mu_{F_{ik}}(x), \mu_A(x)\} \end{aligned} \tag{2}$$

3 Rough Sets with Uncertainty

Similar to the Definition 1 and Definition 2, the lower and upper approximation functions of uncertainty-rough model can be obtained.

Definition 3. The information entropy can be defined as

$$\begin{aligned} \mu_{\underline{A}}(F_{ik}) &= \inf_{x \in U} \max\{1 - \mu_{F_{ik}}(x), \mu_A(x), \alpha\} \\ \mu_{\overline{A}}(F_{ik}) &= \sup_{x \in U} \max\{\min\{\mu_{F_{ik}}(x), \mu_A(x)\}, \beta\} \end{aligned} \tag{3}$$

The positive region of a rough set is the maximum membership degree classified by $F_{ik}; \forall k = 1, 2, \dots, C_i$, i.e.,

$$\mu_{POS}(F_{ik}) = \sup_{F_l \in A(Q)} \{\mu_{F_l}(F_{ik})\} \tag{4}$$

Membership of $x \in U$ to the positive region can be calculated by

$$\mu_{POS}(x) = \sup_{F_{ik} \in A(P_i)} \min\{\mu_{F_{ik}}(x), \mu_{POS}(F_{ik})\} \tag{5}$$

Dependency degree $\gamma_S(Q)$ ($0 \leq \gamma_S(Q) \leq 1$) of Q on the set of attributes P is defined by

$$\gamma_P(Q) = \frac{\sum_{x \in U} \mu_{POS}(x)}{n} \tag{6}$$

From the above-mentioned methods, the system after reduction is obtained and a fuzzy logic consequence method is applied to it. With the fuzzy sets $\tilde{A}(x)$ and $\tilde{B}(x)$, fuzzy set is

$$(\tilde{A} \rightarrow \tilde{B})(x, y) \stackrel{def}{\longleftrightarrow} [\tilde{A}(x) \wedge \tilde{B}(y)] \vee [1 - \tilde{A}(x)] \tag{7}$$

From (23) a fuzzy relationship matrix R can be calculated by different input-output relationships. Then when a fault occurred, a decision with membership function can be obtained from fuzzy relationship matrix as following.

$$\tilde{B}(x) = \tilde{A}(x) \circ R \tag{8}$$

where mark “ \circ ” denotes fuzzy relation synthesized.

Considering multi-fault condition, a probability-rough consequence method is introduced.

Supposing $a = P(x|[x]), b = P(Y|[x])$, the compound probability can be defined as, $a_{11} = P(X \cap Y|[x]), a_{12} = P(X \cap (\sim Y)|[x]), a_{21} = P((\sim X) \cap Y|[x]), a_{22} = P((\sim X) \cap (\sim Y)|[x])$. Obviously the compound probability satisfies the following restriction condition.

$$\begin{aligned} a_{11} + a_{12} &= a, a_{21} + a_{22} = 1 - a \\ a_{11} + a_{21} &= b, a_{12} + a_{22} = 1 - b \end{aligned} \tag{9}$$

The entropy function of compound probability is

$$H(P) = - \sum_{i,j=1}^2 a_{ij} \log a_{ij} \tag{10}$$

When a $a_{ij} = 0$, $H(P)$ can be calculated as $0 \times \log 0 = 0$.

From (9) we can see that only one free variety. We can choose a_{11} as free variety, then

$$\begin{cases} a_{12} = a - a_{11} \\ a_{21} = b - a_{11} \\ a_{22} = 1 - (a + b) + a_{11} \end{cases} \tag{11}$$

Take (23) to (9) can obtain

$$\begin{aligned}
 H(P) &= -a_{11} \log a_{11} - (a - a_{11}) \log(a - a_{11}) \\
 &\quad - (b - a_{11}) \log(ba_{11}) \\
 &\quad - [1 - (a + b) + a_{11}] \log[1 - (a + b) + a_{11}]
 \end{aligned} \tag{12}$$

From definition 3 and probability we can know that

$$\begin{cases} \mu_{X \cap Y}(x) \geq \max\{0, \mu_X(x) + \mu_Y(x) - 1\} \\ \mu_{X \cap Y}(x) \leq \max\{\mu_X(x), \mu_Y(x)\} \end{cases} \tag{13}$$

We can get $\max\{0, a + b - 1\} \leq a_{11} \leq \min\{a, b\}$. Get the derivative of a_{11} from (12), and make the derivative is zero.

$$\frac{dH(P)}{da_{11}} = \log \frac{(a - a_{11})(b - a_{11})}{a_{11}[1 - (a + b) + a_{11}]} = 0 \tag{14}$$

From (14), we can get $a_{11} = ab$, and it is easy to see that $H(P)$ is the most when $a_{11} = ab$ and $H(P)$ is the least when $a_{11} = \max\{0, a + b - 1\}$ or $a_{11} = \min\{a, b\}$. With (9) we can get

$$\begin{cases} \mu_{X \cap Y}(x) = \mu_X(x)\mu_Y(x) \\ \mu_{X \cup Y}(x) = \mu_X(x) + \mu_Y(x) - \mu_X(x)\mu_Y(x) \end{cases} \tag{15}$$

It is with the most entropy. Or get

$$\begin{cases} \mu_{X \cap Y}(x) = \max\{0, \mu_X(x) + \mu_Y(x) - 1\} \\ \mu_{X \cup Y}(x) = \min\{1, \mu_X(x) + \mu_Y(x) - 1\} \\ \text{when } a_{11} = \max\{0, a + b - 1\} \end{cases} \tag{16}$$

and

$$\begin{cases} \mu_{X \cap Y}(x) = \min\{\mu_X(x)\mu_Y(x)\} \\ \mu_{X \cup Y}(x) = \max\{\mu_X(x)\mu_Y(x)\} \\ \text{when } a_{11} = \min\{a, b\} \end{cases} \tag{17}$$

They are with the least entropy.

4 Distribution System Data Modelling

The error of current value can be defined as [12]

$$\Delta I\% = (KI_2 - I_1)/I_1 \times 100\% = \varepsilon\% + \lambda_{i1}\Phi + \lambda_{i2}I_d \tag{18}$$

where Φ is the magnetic flux of instrument current transformer, I_d is the short circuit current value, K is the transformation ratio, I_1 and I_2 are primary and secondary rated current, λ_{i1} and λ_{i2} are the coefficient and $\varepsilon\%$ is the level error of the fluxmeter of current mutual inductance which can be defined as following,

$$\varepsilon\% = \frac{100}{I_1} \times \sqrt{\frac{1}{T} \int_0^T (K_I i_2 - i_1)^2 dt} \tag{19}$$

where I_1 is primary virtual value, i_1 and i_2 are primary and secondary short circuit current, T is the period of short circuit current.

The error of voltage value is

$$\begin{aligned} \Delta U\% &= (KU_2 - U_1)/U_1 \times 100\% \\ &= \varepsilon\% + \lambda_{u1} \cos \varphi + \lambda_{u2} I_0 \end{aligned} \tag{20}$$

where $\cos \varphi$ is power factor, I_0 is no-load current value, λ_{u1} and λ_{u2} are the coefficient, K is the transformation ratio, U_1 and U_2 are primary and secondary rated voltage, and $\varepsilon\%$ is the level error of the instrument potential transformer which can be defined as the level error of the fluxmeter of current mutual inductance.

For $\varepsilon\%$ can be described with different level (four levels are defined in China), it can be regarded as a fuzziness variety. And the other varieties are decided by short circuit situation data and user as a random occurrence. Then the error of current value and voltage value can be defined as a kind of fuzziness randomness variety as following.

$$\xi(\Delta I\%, \Delta U\%) = \begin{cases} \mu_1 + \sigma_1; \Delta I\%, \Delta U\% = \omega_1 \\ \mu_2 + \sigma_2; \Delta I\%, \Delta U\% = \omega_2 \\ \mu_3 + \sigma_3; \Delta I\%, \Delta U\% = \omega_3 \\ \mu_4 + \sigma_4; \Delta I\%, \Delta U\% = \omega_4 \end{cases} \tag{21}$$

where μ_i is the accuracy level of mutual inductance described by experts, σ_i is the random error and ω_i is the probability of measure error.

As for the fault voltage and fault current in the certainty system, the following equation can be obtained.

$$\begin{bmatrix} \dot{V}_a \\ \dot{V}_b \\ \dot{V}_c \end{bmatrix} = l \begin{bmatrix} \dot{Z}_{aa} & \dot{Z}_{ab} & \dot{Z}_{ac} \\ \dot{Z}_{ba} & \dot{Z}_{bb} & \dot{Z}_{bc} \\ \dot{Z}_{ca} & \dot{Z}_{cb} & \dot{Z}_{cc} \end{bmatrix} \begin{bmatrix} \dot{I}_a \\ \dot{I}_b \\ \dot{I}_c \end{bmatrix} + R \begin{bmatrix} \dot{I}_{fa} \\ \dot{I}_{fb} \\ \dot{I}_{fc} \end{bmatrix} \tag{22}$$

where $\dot{I}_f = \dot{I}_a - \dot{I}_{La}$; \dot{V}_a : a-phase voltage; \dot{I}_a : a-phase current; \dot{I}_{fa} : a-phase fault current; \dot{I}_{La} : a-phase load current; l : fault distance; R : fault resistance; \dot{Z} : line impedance matrix. Note that the voltage equation contains two unknown variables-fault distance (l) and fault resistance (R). The fault voltage and current equation with uncertainty can be defined.

$$\dot{V}_a = (l\dot{Z} + R)\dot{I}_a - R\dot{I}_{La} \tag{23}$$

where \dot{V}_a , \dot{I}_a , \dot{I}_{La} and \dot{Z} are known and l , R are random. For $\dot{I}_a \gg \dot{I}_{La}$, the $R\dot{I}_{La}$ can be deemed as σ_i in (21) and $(l\dot{Z} + R)\dot{I}_a$ can be disassembled with a certain variety and a fuzzy variety.

Considering the 6 classes fault as following. [13]

(a) Single line-to-ground fault: Let us assume that phase a is shorted to ground. We see that $V_{af} = 0$; $I_b = I_c = 0$; $I_a = I_f$. The three sequence networks are connected at the fault point.

(b) Line-to-line fault: $V_{bf} = V_{cf}$; $I_a = 0$; $I_b + I_c = 0$. Since the fault does not involve ground, the zero-sequence network is absent.

(c) **Double line-to-ground fault:** $V_{bf} = V_{cf} = 0; I_a = 0; I_f = I_b + I_c$.

(d) **Three -phase fault:** There are no zero or negative-sequence current. $V_{af} = V_{bf} = V_{cf} = 0; I_a + I_b + I_c = 0$.

(e) **One open conductor:** Open-conductor conditions may be caused by broken conductors or a deliberate single-phase switching operation. Such faults may involve the opening of one phase or two phases of a three-phase circuit.

A section of a three-phase system with phase *a*. Let v_a, v_b and v_c be series voltage drops in phases, respectively; I_a, I_b and I_c are the line currents.

$$v_b = v_c = 0; I_a = 0$$

(f) **Tow open conductors:** Consider the condition with phases *b* and *c* open. $v_a = 0; I_b = I_c = 0$.

5 Simulation

A typical 3 partitions and 3 connections country power distribution system of Shenyang, the Northeastern city of China, is shown in Fig.1. It contains 71 nodes and 72 circuits, 23 identified fault grids and 19 the operable switches (numbered 0, 5, 10, 14, 18, 19, 22, 27, 30, 36, 41, 44, 47, 51, 54, 58, 64, 69, 72). There are 105 kinds of network structures including over-loud and insecure ones in the case of no fault. Maybe some of structures are impossible, but most of them are acceptable and potential.

For any of the structures, such as 5, 22, 27, 36, 41, 69 switches off and the others on, the system structure can be reconfigured as Fig.2 shown. 7 electric isolated islands on,

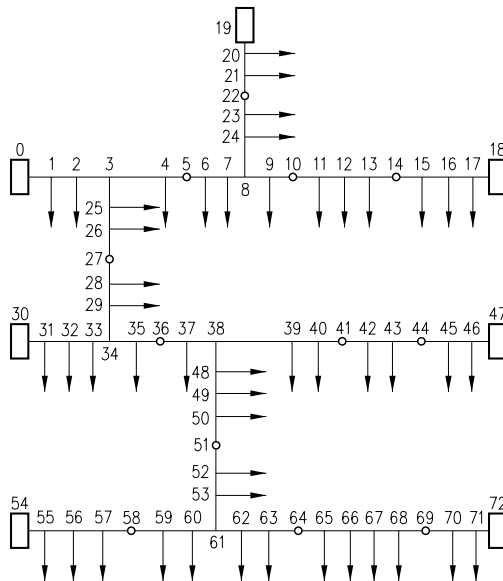


Fig. 1. A typical 3 partitions and 3 connections countryside power distribution system

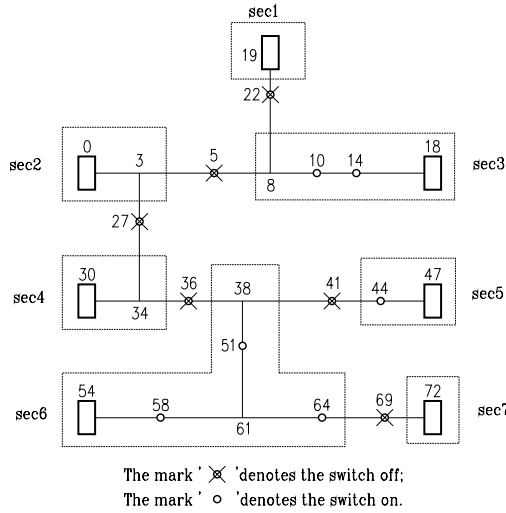


Fig. 2. A kind of structure of distribution when there is no fault

are divided. There are 1 to 4 fault grids differently in each electric isolated island. The whole fault grids reach 13. The similar count is in any other structures. As a whole, the imaginable fault grids will be more than 1000 and the rules several times to them. The chain fault grids are too much to add up.

In the system, the current values of the nodes with operable switches are measurable, named by node number (such as I_0, I_5, \dots, I_{72}). And the measurable voltage values are only on the outside power supply nodes, also named by node number ($U_0, U_{18}, \dots, U_{72}$). The count of condition attributes is 26, and the count of decision attributes is only 2 (fault grid and fault type). Fault type is composed of a: single line-to-ground, b: line-to-line, c: double line-to-ground, d: three-phase, e: one open conductor and f: tow open conductors. With 1000 history fault records, the origin decision table can be established.

For the error of the data in the requirement and transport process, the origin rules are not authoritative completely. So the reduction end condition is $\gamma_{P_{k+1}}(Q) - \gamma_{P_k}(Q) < \varepsilon; \forall \varepsilon > 0$. In the simulation we take $\varepsilon = 0.2$. The fuzzy membership function type is shown in Fig.1.

The count of attributes after reduction is 18, except $I_{14}, I_{22}, I_{47}, I_{54}, I_{64}, I_{72}, U_{30}, U_{47}$. The effectual reduction rate is $Card(U - \{P\}) / Card(U) = 0.31$.

The fault diagnosis process based on probability is stopped by $\mu_{\cap X}(x) \leq 0.3$ or $Card(\cup X) \geq 4$, which can save the time of reasoning time. And the fault type of chain faults needed to be deduced.

For the reduction rules, we take 200 groups of fault data of different system structures to examine. For instance, by the structure as shown in Fig.1 some fault results are shown in Table.1. Some of the rules with the lower confidence are leaved out in the table for the space.

Table 1. Some Examples of Diagnosis by Examination

Test	Diagnosis				
f_{10-14} d	f_{10-14} , 0.92 d, 0.69	f_{14-18} , 0.11 b, 0.33		$f_{10-14} \cap_{14-18}$ 0.10	$f_{10-14} \cup_{14-18}$ 0.93
f_{38-51} and f_{51-58} c	f_{38-51} , 0.71 d, 0.55	f_{51-58} , 0.82 c, 0.51	f_{54-58} 0.62	$f_{38-51} \cap_{51-58}$ 0.58 $f_{38-51} \cap_{54-58}$ 0.44 $f_{54-58} \cap_{51-58}$ 0.51	$f_{38-51} \cup_{51-58}$ 0.95 $f_{38-51} \cup_{54-58}$ 0.89 $f_{54-58} \cup_{51-58}$ 0.93
f_{58-64} e	f_{58-64} 0.95 e, 1	f_{51-58} 0.33 e, 1		$f_{58-64} \cap_{51-58}$ 0.31	$f_{58-64} \cup_{51-58}$ 0.97

Table 2. Rules of Worked Example

λ	Fault types					
	Total	Symmetrical fault	Other fault	short	Open con- ductor	Multi-fault
0.8	85.7%	91.1%	86.4%		92.0%	67.9%
0.5	86.3%	87.7%	84.2%		89.9%	79.5%
0.3	83.1%	84.4%	84.8%		88.6%	71.0%

The whole test conclusions are shown in Table.2. The exactness rate is defined as $\frac{\sum_{\mu_i \geq \lambda, f_i = f_{test}} Card(x_i) \cdot \mu_i}{\sum_{\mu_i \geq \lambda} Card(x_i) \cdot \mu_i}$, where $Card(x_i)$ is the count of the rules; μ_i is the weight of each rule; λ is the membership function threshold value.

From the table we can see that the symmetrical fault and open conductor fault are easy to diagnosis, but the multi-fault is hard. And a better result can be obtained by $\lambda = 0.5$.

6 Conclusion

An overabundance of data may cause serious inconvenience to those engineers and dispatchers who have to analyze and respond to an emergency. By improving how we retrieve information, our information anxiety will be reduced and our confidence in making the correct decision increases. The pursuing aim of system is: the right information to the right people at the right place and the right time.

The proposed method in this paper can help reduce the amount of data stored at the source and increase the quality of information presented to the end users. A computational intelligence approach using uncertainty rough sets is described in the paper. The knowledge is represented by a group of fuzzy rules. In order to reduce abundant fuzzy rules and attributes or inconsistent information, rough set theory is employed to find a minimal reduction and form a group of final fault diagnostic rules. The rule reliability is computed using fuzzy sets and probability. It not only generates useful and easily readable rules with voltage and current but also allows faster knowledge acquisition and information transfer to streamline the decision making process by probability fuzzy method. The technique optimizes the way we handle masses of information by

reducing the amount of data presented to the operator, particularly during and after an emergency condition. Effectively, it conveys a clearer picture of the system condition.

References

1. Peng, J. T., Chien, C. F., Tseng, L. B.: Rough Set Theory for Data Mining for Fault Diagnosis on Distribution Feeder. *IEEE Proceedings-Generation, Transmission and Distribution* **151(6)** (2004) 689-697
2. Hor, C. L., Crossley, P. A.: Extracting Knowledge from Substations for Decision Support. *IEEE Transactions on Power Delivery* **20(2)** (2005) 595-602
3. Yu, Y. H., Bai, Y. C., Xi, G. F., Xu, S. M., Luo, J. B.: Fault Analysis Expert System for Power System. *Power System Technology* **2** (2004) 1822-1826
4. Xu, X. P., Peters, J. F.: Rough Set Methods in Power System Fault Classification. *Electrical and Computer Engineering, Canadian* (2002) 100-105
5. Pawlak, Z.: Rough sets. *Int. J. Comp. Inform. Science.* **11** (1982) 341-356
6. Yeung, D. S., Chen, D. G., Tsang, E. C. C., Lee, J. W. T., Wang, X. Z.: On the Generalization of Fuzzy Rough Sets. *IEEE Transactions on Fuzzy Systems* **3(13)** (2005) 343-361
7. Pawlak, Z., Skowron, A.: Rough Membership Functions. In: Yager, R., et al., Eds., *Advances in Dempster Shafer Theory of Evidence*. Wiley, New York (1994) 251-271
8. Jensen, R., Shen, Q.: Fuzzy-rough Attribute Reduction with Application to Web Categorization. *Fuzzy Sets and System* **141(3)** (2004) 469-485
9. Bhatthatt, R. B., Gopalm: On fuzzy-rough Sets Approach to Feature Selection. *Pattern Recognition Letters* **26 (7)** (2005) 965-975
10. Lambert-Torres, G.: Application of Rough Sets in Power System Control Center Data Mining. *Power Engineering Society Winter Meeting* **1** (2002) 627-631
11. Jensen, R., Shen, Q.: Semantics-preserving Dimensionality Reduction: Rough and Fuzzy-rough-based Approaches. *IEEE Transactions on Knowledge and Data Engineering* **16(12)** (2004) 1457-1471
12. China Electric Power Encyclopedia Committee. *China Electric Power Encyclopedia: Power System Volume*, China Power Publication (2001) 106-125
13. Prabha, K.: *Power System Stability and Control*. McGraw-Hill (1994) 877-884

A Design Method of Associative Memory Model with Expecting Fault-Tolerant Field*

Guowei Yang^{1,2}, Shoujue Wang², and Qingxu Yan³

¹ School of Automation Engineering, Qingdao University, Qingdao 266071, China
ygw_ustb@163.com

² Laboratory of Artificial Neural Network, Institute of Semiconductors Chinese Academy of Sciences, Beijing 100083, China
wsjue@red.semi.ac.cn

³ School of Information Engineering, China University of Geosciences, Beijing 100083, China

Abstract. A design method of associative memory model with expecting fault-tolerant field is proposed. The benefit of this method is to make the designed associative memory model memory sample fault-tolerant field which implements the hoped situation. For any different P samples in n dimensional binary information space $D^n = [1, -1]^n$ and any the p compartmentalization C_1, C_2, \dots, C_p of D^n , an associative memory model with expecting fault-tolerant field C_1, C_2, \dots, C_p can be designed by the method. The method better solves the difficult synthesis problems of associative memory models.

1 Introduction

There is a very long history in the research of the associative memory [1-2]. In 1961, Steinbuch put forward the concept of studying matrix. In 1972, Kohonen set up related matrix memory device, and then, Nakano proposed the associating machine. In 1988, Kosko provided the bi-directional associative memory model. In 1990, Jeng proposed the index bi-directional associative memory model. In 1998, Lee provided and realized a kind of multi-valued mode associative memory [3]. In 2000, Yingquan Wu etc. gave three layers feed-forward bidirectional associative memory [4-5]. In 2001, Yingquan Wu etc. improved the above work by taking out some constraint conditions [6]. In 2003, Jyh-Yeong Chang, Chien-Wen Cho constructed a second-order asymmetric bidirectional associative neural network design with a maximum field of attraction [7]. In 2003, Kamio, T., Morisue, M. constructed an associative memory network by using cell nerve network with copying stencil steadiness space [8], Costantini, G., Casali, D., Perfetti, R. proposed associative memory neural network method of storing gray-coded gray-scale images [9]. Mehmet Kerem Mezzinoglu etc. respectively proposed a new design method of complex value multi-state Hopfield associative memory and binary system associative memory design in 2003 and 2004

* Supported by the National Natural Foundation of China No.60673101 and 863 Project of China (No.2006AA04Z110, 2006AA01Z123).

[10-11]. For nearly ten years in China scholars such as Zhang Ling, Zhang Bo, Liang Xuebin, Chen Songcan, Tao Qing, Wei Hui, Yang Guowei etc. have made the further research work [12-17], and developed the research of the associative memory. Base on the associative memory research results, we can establish the artificial brain and the artificial life with associative memory function of which is similar to that of humanbeings [1,18].

Associative memory divides into heteroassociate memory and autoassociate memory. Heteroassociate memory is a kind of mapping mode feed-forward, which make the input directly mapping to the needed output by constructing some mapping (transform), For instance the multi-layer network, perceiving device ,etc. (remarksome heteroassociate associative memory can be realized by the network motive evolvement). The process of autoassociate memory is generally a kind of evolution course systematic in dynamics which makes the mode to be discerned as the initial state of the network, and the network evolves according to certain dynamics law, the final state is the associate result. As the network export, the Model Hopfield network belongs to this cluster. But to form this kind of associative memory network need take longer time, and sometimes this kind of associative memory network can not form even in standing time (the evolvement can not converge), which means this kind of network model associative memory is slow.

The associative memory gets a great achievement. But up till now the size and position of the existing associative memory model fault-tolerant fields (attractive field) can not be shifted by the designer’s desire. None of them can achieve the purpose that what kind of the sample fault-tolerant field I want and then I can finish it. Some associative memory model fault-tolerant fields even have no general recognition, just exiting in theory [1-16]. This deformity of associative memory model has a strong impact on its range of application and effect.

Example 1. By using models in [4, 5, 6, 12] to associate memory the following two mode of Nine Square

-1 ^o	1 ^o	1 ^o
1 ^o	1 ^o	1 ^o
1 ^o	1 ^o	1 ^o

1 ^o	1 ^o	1 ^o
1 ^o	1 ^o	1 ^o
1 ^o	1 ^o	-1 ^o

Fig. 1. Nine Square two big difference mode

Therefore the Hamming distance of the two mode is 2. Therefore, both of the fault-tolerant radius are 1, and the two fault-tolerant fields only have one sample point. So once there is noise, the two mode can not associate correctly (classification or recognition), that is, we can not guarantee the results if this model apply to two kind mode classify recognition. However, we can find the big gap and difference between the two mode, both of their fault-tolerant should be a little larger.

We know that the research of associative memory model includes analyze and synthesize the associative memory model system. The purpose of analysis is to find the functions of the system, while synthesis is to build a system with destined functions. The reason why the above deficiency exists mainly because ① The traditional method of the associative memory model is to "design the associative memory model first, then find the sample fault-tolerant field or prove the presence of the sample fault-tolerant field". Such design concept is "system analyze method", it can foreknow functions the system possess, but can not guarantee system must possess those functions, for example, it can not guarantee a sample possess the fault-tolerant field; ② It is an unresolved system synthetic intractable problem that "design the sample fault-tolerant field first, then construct the associative memory model with such fault-tolerant field".

In the article, A method that design the sample expecting fault-tolerant field first, base on the expecting sample fault-tolerant field(set of learning point), we can construct an ahead masking associative memory with the expecting sample fault-tolerant field model by learning. The associative memory model designed by this method not only can associate the given sample, but also can guarantee the size and position of sample fault-tolerant field as designed. Consequently, it can achieve the purpose that what kind of the sample fault-tolerant field I want and then I can finish it. The operation of the model design method is easy and its algorithms can be finished in limited step.

2 Design of Expecting Sample Fault-Tolerant Field (Attractive Field)

Consider n dimensional information spaces $D^n = \{(x_1, \dots, x_i, \dots, x_n)^T | x_i = 1 \text{ or } -1\}$. Suppose that $(X^1, Y^1), (X^2, Y^2), \dots, (X^p, Y^p)$ are different P sample pairs from each other in D^n , $X^1 = (x_{11}, x_{21}, \dots, x_{n1})^T, \dots, Y^1 = (y_{11}, y_{21}, \dots, y_{n1})^T, P \leq 2^n$.

Let P nonempty sets $C_j \subseteq D^n, j = 1, \dots, p$, if $C_j \cap C_i = \emptyset, i \neq j, i, j = 1, 2, \dots, p, \bigcup_{j=1}^p C_j = D^n$, we call C_1, C_2, \dots, C_p is P compartmentalization of D^n . By permutation and combination, there are many ways to compartmentalize D^n in P blocks. And we also can figure out the compartmentalization of $X^1 \in C_1, X^2 \in C_2, \dots, X^p \in C_p$. Let $\left\{ \begin{matrix} m \\ r \end{matrix} \right\}$ represent the number of r nonempty nonintersectant subsets compartmentalization of the set with m elements. Then $\left\{ \begin{matrix} m \\ r \end{matrix} \right\} = r \left\{ \begin{matrix} m-1 \\ r \end{matrix} \right\} + \left\{ \begin{matrix} m-1 \\ r-1 \end{matrix} \right\}$. And the number of the P compartmentalization of D^n is $\left\{ \begin{matrix} 2^n \\ p \end{matrix} \right\}$. The number of the P compartmentalization of D^n with $X^1 \in C_1, X^2 \in C_2, \dots, X^p \in C_p$ is $p^{2^n - p}$.

We can prove the following theorem by mathematics knowledge.

Theorem 1. Let $d_i = \min\{d_H(X^i, X^j) | i \neq j\}, D(X^i) = \{X | d_H(X^i, X) < \frac{d_i}{2}, X \in D^n\}, i = 1, \dots, p-1$, and $D(X^p) = \{X | d_H(X^i, X) \geq \frac{d_i}{2}, X \in D^n, 1 \leq$

$i \leq p - 1$ }, where d_H is Hamming distance. Then $D(X^1), D(X^2), \dots, D(X^p)$ is P compartmentalization of D^n . The compartmentalization feature is that X^i is center of $D(X^i)$, and in average significance lets $D(X^i)$ at a maximum radius.

Now suppose $X^1 \in C_1, \dots, X^i \in C_i, i = 1, \dots, p$ and C_1, C_2, \dots, C_p is a compartmentalization of D^n . Mapping

$$f : D^n \longrightarrow D^n$$

$$(f_1(X), \dots, f_n(X)) = f(X) = \begin{cases} Y^1, & \text{while } X \in C_1, \\ \vdots & \vdots \\ Y^i, & \text{while } X \in C_i, \\ \vdots & \vdots \\ Y^p, & \text{while } X \in C_p \end{cases}$$

While there is a neural network NN precisely implement f , this network NN is a associative memory model of $(X^1, Y^1), (X^2, Y^2), \dots, (X^p, Y^p)$, and C_i is the fault-tolerant field of C^i . Especially, while $X^1 = Y^1, X^2 = Y^2, \dots, X^p = Y^p$, this network NN is an autoassociate memory model of X^1, X^2, \dots, X^p , and C_i is the fault-tolerant field(attractive field) of X^i . There are many design methods of $C_i, i = 1, \dots, p$ from the above illustration, to different needs, we can design C_i in particular, for example, the design in Theorem 1.

Now we design an ahead masking associative memory model with expecting fault-tolerant field to precisely implement mapping f .

3 An Ahead Masking Associative Memory Model Topology

The topology of an ahead masking associative memory model is made up of a general feed-forward network^[19] with n input nodes, and circumscribe with double layers feed-forward neural network each layer contains n neurons, and neurons on the same layer do not connect with each other.

General Feed-forward network (no closed loop) is the neural network that all the neurons have serial number except input nodes (seen as neurons) input nodes are connected with all the neurons, neurons which are lined in front are connected with all the neurons behind them, neurons are not connected with themselves, and neurons which are lined in latter are not connected with neurons before them.

To feed-forward network, suppose nerve network contains N input notes (number 1 to N) and M neurons (number $N + 1$ to $N + M$), $W_{ij} (i = 1, 2, \dots, N, j = N + 1, N + 2, \dots, N + M)$ is the connecting weighting value between neurons. If $W_{ij} = 0$ denotes there is no connection between i to j , then the topology of the neurons can be described by i to j condition of $W_{ij} = 0$.

Therefore, the sufficient and necessary condition of General Feed-forward network GFFN is

$$W_{ij} = 0, \quad \text{if } j \leq N \quad \text{or} \quad i \geq j.$$

The sufficient and necessary condition of single layer perceiving device is :

$$W_{ij} = 0, \quad \text{if } i > N \quad \text{or} \quad j \leq N.$$

The sufficient and necessary condition of single hidden layer lamination feed-forward network(number of the hidden neurons is a , $a < M$) is:

$$W_{ij} = 0, \quad \text{while} \begin{cases} j \leq N, \\ \text{or } i \leq N, j > N + a, \\ \text{or } i > N, j \leq N + a. \end{cases}$$

The sufficient and necessary condition of three layers perceiving device (i.e. two hidden layers feed-forward network, which the number of neurons on the second hidden layer respectively is a and b , $a + b < M$)

$$W_{ij} = 0, \quad \text{while} \begin{cases} j \leq N, \\ \text{or } i \leq N, j > N + a, \\ \text{or } i > N, j \leq N + a, \\ \text{or } N + a \geq i \geq N, j > N + a + b, \\ \text{or } i > N + a, j \leq N + a + b, \\ \text{or } i > N + a + b. \end{cases}$$

It is clear that General Feed-forward network (GFFN) is the most widespread feed-forward network, and any lamination feed-forward network is one exceptional case of it. While the above GFFN condition, that is formula (2), is the feed-forward network basic condition.

Determination of the Ahead Masking Associative Memory Model with Expecting Fault-tolerant Field: Determination of the ahead masking associative memory model with expecting fault-tolerant field includes three parts of work. First part is general feed-forward network (GFFN) model determination (sequential learning algorithm) in the model, second part is determination of the connecting weighting value(matrix) from general feed-forward network (GFFN) output neuron to the first layer of double layers feed-forward network, third part is the determination of double layers feed-forward network model.

Determination of General feed-forward network (GFFN) model (sequential learning algorithm): Supposes C_1, C_2, \dots, C_p is P compartmentalization of $D^n = [1, -1]^n$, and $X^1 \in C_1, X^2 \in C_2, \dots, X^p \in C_p$.

Suppose that the input space is the n dimension space $D^n = \{(x_i, \dots, x_i, \dots, x_n)^T | x_i = 1 \text{ or } -1\}$, neuron uses the linear output during learning, that is output $= \sum WI$, neuron uses hard limited step activation function during recognition

$$f(Z) = \begin{cases} 1, & Z \geq 0, \\ 0, & Z < 0. \end{cases}$$

Activation threshold value of the neuron is W_0 , input biggest coupling number of the neuron is $N + a$ ($a = 1, 2, \dots$, is the serial number of neuron), where n

couplings come from n input notes, $a - 1$ couplings come from the output neuron with serial number $< a$, one coupling is the input of activation threshold is $-W_0$.

Supposes that input sample space is $D^n = [1, -1]^n = C_1 \cup C_2 \cup \dots \cup C_p$, the total number of the input samples is $S = 2^n$, where S_1 samples are assigned to be the cluster X_1 (samples belong to C_1), and S_2 samples are assigned to be the cluster X_2 (samples belong to C_2), and S_3 samples are assigned to be the cluster X_2 (samples belong to C_3), \dots , by parity of reasoning : $S = S_1 + S_2 + \dots + S_p$. Frist suppose all the connection intensity are zero between neurons (no connection), suppose the output activation function of neuron is linear, that is output

$$= \sum_{i=1}^n M_i I_i.$$

Step 1. Compute the unit direction vector of each note from origin to $D^n = [1, -1]^n = C_1 \cup C_2 \cup \dots \cup C_p$, the set can be written as M .

Step 2. Stochastically chooses α group of vectors from as input weighting values neurons which has not arranged, then put all samples in $C_1 \cup C_2 \cup \dots \cup C_p$ through the input notes in order and at the same time put them to neurons which has not arranged to carry on the computation respectively (α is user-defined learning parameter).

Step 3. Get the computed result $= \sum WI$ of input sample of each neuron which has not arranged, to each neuron respectively line up the result from big to small by size, from the line up results, find out each neuron to the similar sample computed result may separate to the same cluster of sample number from big to small. (the definition of separation is the max distance which separates two sides different cluster of samples computed result bigger than some definite value D , the established D value may affect total number of neurons and the ability of network exudes the ability.)

Step 4. Find out the not arranged neuron which can separate the sample most, record its related weighting value and separation place value $W_0 = \sum W_0 I_0$ (remark if threshold value of neuron is this value, the neuron can separate to the same cluster of sample number most, those separated same cluster sample value $\sum WI$ is bigger or equal to this value $\theta W_0 = \sum W_0 I_0$, while other all sample value $\sum WI$ is smaller than $W_0 = \sum W_0 I_0$) to spare, if the separation sample is bigger than its previous record, then covers previous record.

Step 5. From further increases separation sample number and enlarge the direction of separation distance, that is, chooses α group of vectors from unit direction vector M as α input weighting values of neurons which has not arranged, regulate the input weighting values of neurons which has not arranged, again calculates those neurons.

Step 6. Duplicate 2 to 4th step β times. Take the neuron with most record as the neuron which has arranged, follows closely the arranged neuron, (or begin arranged neuron), record the separation sample number of this neuron and the separation value $\sum W_0 I_0$, and mark the separation sample cluster (X_1 or X_2 or $X_3 \dots$ etc.) of this neuron.

Step 7. Take away the sample which is separated by this neuron from the original sample set, and take the left samples as a new sample set.

Step 8. Duplicates 2 to 7th step until the sample set is only left one kind sample.

Step 9. As well as another neuron, which arrangement follows closely outside the arranged neuron, (it will be the biggest arrangement in this nerve network, which arrange number is total number of the neuron which pattern classification needs), from the left sample computation results, we can get the smallest value in $\sum WI$, use this value subtract D as this neuron separation value, and mark the classification of the sample.

Step 10. Change the activate function of all neuron into hard limited step function, and set the corresponding threshold value as $W_0 = \sum W'I'$, where $\sum W'I'$ is the recorded separation value.

Step 11. Note W_{max} as the absolute maximum weight of the weights from the above input nodes to any neurons multiplied by the total number of input nodes N. Let the weight from the neuron of lower sequential order to that of higher sequential order equal to $+W_{max}$, if both the neurons are marked with the same cluster. Let the weight from the neuron of lower sequential order to that of higher sequential order to be $-W_{max}$, if both neurons are marked with different cluster.

After the above sequence feed-forward masking algorithms of the learning ordering have been performed, the general feed-forward network model has been determined. The neuron, marked with certain cluster (X_1 or X_2 or $X_3 \dots$ etc.) with the highest sequential order, corresponds to the output neuron of the cluster. When an input pattern belongs to this cluster, the output of this neuron is 1, otherwise it is 0. It is should be pointed out that the number of the connecting weights of the edges originating from a certain neutron to other neutrons of higher order may be less than or equal to the total number of the classified clusters, for among the neutrons of a certain cluster of higher order, it is enough to choose a neutron with the lest order as the objective to be connected. This neuron may continue to mask the previous neuron of the higher sequential order.

The determination of the weight from general feed-forward network to the two-layer feed-forward network: Let 1 be the connection weight from the neuron of the highest sequential order in cluster X_1 of the general feed-forward network output neurons to the first neuron on the first layer of two-layer feed-forward network, set 1 as the connection weight from the neuron of the highest sequential order in cluster X_2 of the general feed-forward network output neuron to the second neuron on the first layer of the two-layer feed-forward network, ..., set 1 as the connection weight from the neuron of the highest sequential order in cluster X_p of the general feed-forward network output neurons to the neutron p on the first layer of the two-layer feed-forward network, all the other weights from the general feed-forward network output neurons to the first layer of the two-layer feed-forward network are endowed with 0.

The determination of the two-layer feed-forward network model: Suppose that $(X^1, Y^1), (X^2, Y^2), \dots, (X^p, Y^p)$ are different P associative memory sample pairs from each other in D^n , $X^1 = (x_{11}, x_{21}, \dots, x_{n1})^T, \dots, Y^1 =$

$(y_{11}, \dots, y_{n1})^T, \dots$ Set each layer of the two-layer feed-forward network including n neurons. The weight matrix from the neurons on the first layer to those on the second layer of the two-layer feed-forward network is

$$\begin{pmatrix} y_{11} & y_{12} & \dots & y_{1p} \\ y_{21} & y_{22} & \dots & y_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ y_{n1} & y_{n2} & \dots & y_{np} \end{pmatrix}_{n \times p}$$

As soon as the above work has been completed, the ahead masking associative memory model with expecting sample fault-tolerant field can be determined. For this model, we have the following theorem.

Theorem 2. Let $D^n = \{(x_1, \dots, x_i, \dots, x_n)^T | x_i = 1 \text{ or } -1\}$. Suppose that $(X^1, Y^1), (X^2, Y^2), \dots, (X^p, Y^p)$ are different P associative memory sample pairs from each other in D^n , $X^1 = (x_{11}, x_{21}, \dots, x_{n1})^T, \dots, Y^1 = (y_{11}, y_{21}, \dots, y_{n1})^T, \dots$ Suppose that C_1, C_2, \dots, C_p is p compartmentalization of D^n , where $X^1 \in C_1, X^2 \in C_2, \dots, X^p \in C_p$. Therefore the cybernetic fault-tolerant field ahead masking associative memory model determined by the above method can be precisely implemented by the mapping f of the form 1 and consequently, $(X^1, Y^1), (X^2, Y^2), \dots, (X^p, Y^p)$ can be associatively memorized, and the sample fault-tolerant field X^k is

$$D(X^k) = C_k, k = 1, \dots, p,$$

such that

$$\bigcup_k D(X_k) = D^n = C_1 \cup C_2 \cup \dots \cup C_p,$$

In the case that $X^k = Y^k, k = 1, \dots, p$, the ahead masking associative memory model is a kind of auto-associative memory.

Example 2. In example 1, the fault-tolerant field $D(X^k) = C_k, k = 1, 2$ of the two modes can be established previously as the set including 2^{9-1} different modes. According to the above design method of the ahead masking associative memory model, we can correspondingly get the cybernetic fault-tolerant field ahead masking auto-associative memory model. Generally speaking, the fault-tolerant $D(X^k) = C_k, k = 1, 2$ of the two samples of this model are the maximal. The experiment indicates that under slightly noises, this auto-associative memory model can correctly associate (classify or recognize) the two different mode of Nine Square.

Corollary 1. The ahead masking associative memory model with expecting sample fault-tolerant field does not have pseudo-attractor.

Proof. Due to $D^n = \bigcup_{k=1}^p D(X^k)$, that is, the union of the fault-tolerant fields (attractive field) of the memory samples is the whole space D^n . Therefore, the

ahead masking associative memory model with expecting sample fault-tolerant field does not have pseudo-attractor.

Network capacity refers, under certain conditions of the admissible associative error probability, the maximal mode sample number which the network can save. It is generally defined as the quotient of the sample number M and the sample vector dimension n

$$\frac{M}{n},$$

The maximum memory capacity of the static state associative memory network is 1^[20]. According to the definition of network capacity, we have the following corollary.

Corollary 2. The capacity of a cybernetic fault-tolerant field ahead masking associative memory model CFTFAMAM (network) is $\frac{p}{n}$, where is input space dimension and is any positive integer.

Remark. If we do not hope that the fault-tolerant field X^1, X^2, \dots, X^p of p samples is too big, and do not hope them to be the whole space D^n , we can previously set a small fault-tolerant field $D(X^k) = C_k, k = 1, \dots, p$. Having set a point pair X^0, Y^0 , we have to find X^0 such that $X^0 \in D^n - \bigcup_{k=1}^p D(X^k), Y^0 \neq Y^1, Y^2, \dots, Y^p$. Suppose that the fault-tolerant field of X^0 is $D^n - \bigcup_{k=1}^p D(X^k)$, for $(X^0, Y^0), (X^1, Y^1), (X^2, Y^2), \dots, (X^p, Y^p)$, we then construct an ahead masking associative memory model according to the given method. Therefore the obtained associative memory model fault-tolerant field can also meet the required situation.

4 Conclusions

It is well known that setting the fault-tolerant field first and then, trying to construct an associative memory model with the set fault-tolerant field is still a challenge for the researchers. In this article, some implementation algorithms of an ahead masking associative memory model with the expecting fault-tolerant field are proposed so as to make the sample of the associative memory model fault-tolerant field easier to be implemented and to meet the desired performance. In fact, the size and the position of the associative memory model fault-tolerant fields (attractive field) can be shifted by the designer's desire completely. It can also fulfill the purpose that the designed associative model is of the exactly the sample fault-tolerant field the researchers wanted. It is obvious that the method proposed in this article has very good application prospect in pattern recognition. The specific application will be proposed in other articles latter on for the sake of the volume of this article.

References

1. Yang, G.: Models of Artificial Life. Science Press, Beijing (2005)
2. Tu, X., etc.: Theory of Biologic Control. Science Press, Beijing (1980)
3. Lee, D., Wang, W.: A Multivalued Bidirectional Associative Memory Operating on a Com-plex Domain. *Neural Networks* **11** (1998) 1623-1635
4. Wu, Y., Pados, D.A.: A Feedforward Bidirectional Associative Memory. *IEEE Transactions on Neural Networks* **4** (2000) 859-866
5. Wu, Y., Batalama, S.N.: An Efficient Learning Algorithm for Associative Memories. *IEEE Transactions on Neural Networks* **5** (2000) 1058-1066
6. Wu, Y., Batalama, S.N.: Improved One-shot Learning for Feedforward Associative Memories with Application to Composite Pattern Association. *IEEE Transactions on Systems, Man and Cybernetics, Part B* **1** (2001) 119-125
7. Chang, J., Cho, C.: Second-order Asymmetric BAM Design with a Maximal Basin of At-traction. *IEEE Transactions on Systems, Man and Cybernetics, Part A* **4** (2003) 421-428
8. Kamio, T., Morisue, M.: A Synthesis Procedure for Associative Memories Using Cellular Neural Networks with Space-invariant Cloning Template Library. *Proceedings of the International Joint Conference on Neural Networks* **2** (2003) 885-890
9. Costantini, G., Casali, D., Perfetti, R.: Neural Associative Memory Storing Gray-coded Gray-scale Images. *IEEE Transactions on Neural Networks* **3** (2003) 703-707
10. Mehmet, K., Cneyt, G., Zurada, M.: A New Design Method for the Complex-Valued Multistate Hopfield Associative Memory. *IEEE Transactions on Neural Networks* **4** (2003) 891-899
11. Mehmet, K., Cneyt, G.: A Boolean Hebb Rule for Binary Associative Memory Design. *IEEE Transactions on Neural Networks* **1** (2004) 195-220
12. Zhang, L., Wu, F., Zhang, B., Han, J.: Study of Multilayer Feedforward Neural Network and Synthetic Algorithm. *Journal of software* **7** (1995) 440-448
13. Liang, X., Wu, L., Yu, J.: An Effective Learning Algorithm of Associative Memory Neural Network. *Journal of automation* **6** (1997) 721-727
14. Zhang, D., Chen, S.: The Expanding Multi-value Index Bi-directional Associative Memory Model and Application. *Journal of software* **3** (2002) 97-702
15. Tao, Q., Cao, J., Sun, D.: Associative Memory Neural Network Model Base on Linear Programming. *Journal of computer* **4** (2001) 377-381
16. Wei, H.: The Autoassociate Memory Model of Auto-mapping Base on Structure Learning and Iterating. *Journal of software* **3** (2002) 438-446
17. Yang, G., Tu, X., Wang, S.: A Time-Varying Fault-Tolerant Field Perceptive Associative Memory Model and the Realization Algorithms. *Chinese Journal of Computers* **3** (2006) 431-440
18. Tu, X.: Artificial Fish -a Way of Artificial Life of the Computer Animation. Qinghua University Press, Beijing (2001)
19. Wang, S., Chen, X., Zeng, Y. etc.: In General Neural Network Hardware Neuron Basic Mathematical Model Discussion. *Journal of electronics* **5** (2001) 577-58
20. Huang, D.: Theory of Neuron Network Mode Recognition System. Beijing Publishing House of Electronics Industry (1996)

Classification and Diagnosis of Mechanical Faults Using the RBF Network Based on the Local Bispectra

Shuhua Xu, Benxiong Huang, and Yuchun Huang

Department of Electronics and Information Engineering,
Huazhong University of Science and Technology,
Wuhan, Hubei, China, 430074

{xush, huangbx}@mail.hust.edu.cn, supermiracle@163.com

Abstract. This paper introduces an efficient technique to design a classifier for classifying and diagnosing mechanical faults. The bispectra features of vibration signals resulting from mechanical faults are extracted and then evaluated using the Fisher's class-separability discriminant measure. The local bispectra with the most discriminant power and sensitivity are selected as the classification feature vector that can effectively represent the fault class of concern over a broad range of sample data. A RBF neural network is implemented to realize identification and diagnosis for different mechanical faults. The suggested technique is demonstrated to design a classifier for fault signals of rolling bearings that is verified to be highly accurate and robust even in the presence of excessive noise.

1 Introduction

Mechanical fault diagnosis is a challenging problem concentrated on the extraction of fault features from the available raw data as the recognition performance is strongly determined by the characterization power of the features. Providing the energy distribution for different frequency components of signals, the power spectra features were used widely. However, the power spectra would lose phase information of fault signal and fail to deal with non-minimum phase system and non-Gaussian signals. Whereas the vibration signals of mechanical faults and self-excitation signals of complicated machinery are non-Gaussian. On the other side, when machines suffer early faults, the scannel fault information would be submerged by Gaussian noise greatly, which results in great difficulty of extracting features for mechanical faults.

Nevertheless, the high order cumulants and spectra can not only retain the amplitude and phase information of non-Gaussian signals, which will bridge a gap of power spectra, but also handle Gaussian noise very efficiently[1]. Therefore, the high order spectra may extract more useful information when analyzing mechanical vibration signals. Barker had monitored rotating tool wear using high order spectral features[2]. Murray and Penman managed to extract fault feature information for vibration signals of induction motors using bispectra analysis[3]. Also, McCormick had presented two methods for diagnosing mechanical faults based on high order spectra[4], and the faults diagnosis for rotation machinery and reciprocator using

bispectra was proposed in [5]. However, the aforementioned methods take the whole bispectra or the three order spectra results as feature vector, and the classification results are not satisfactory since too many features are used by the classification network, and using the whole bispectra requires a complicated two-dimensional matching template, which brings about heavy computation. Accordingly, the integral bispectra are designed to transform the two-dimensional bispectrum into a one-dimensional template. However, the proposed radially integrated bispectra(RIB)[6], axially integrated bispectra(AIB)[7] and circularly integrated bispectra(CIB)[8] may lose the scale variance or part of phase information. Also, these integral bispectra need computation approximately, and some bispectrum values on the bifrequency plane will be lost caused by discrete computation. Moreover, the irregular distribution of the cross-terms on the bifrequency plane makes the integrated bispectra difficult to avoid numerous cross-terms. Consequently, the aforementioned methods would reduce the accuracy for diagnosis of mechanical faults.

To avoid the above problems, a method of assessing classification features is used in Section 2, and Fisher's class-separability discriminant measure is used to evaluate a large amount of bispectra features of vibration signals and extract the high discriminative local bispectra. Then, a RBF neural network is implemented to realize classification and diagnosis for different mechanical faults in Section 3. In Section 4 the experimental results for fault signals of rolling bearings demonstrate that the scheme using features evaluation combined with RBF network can not only extract fine mechanical fault information, reduce scale of the features and be computational efficient, but also has a high recognition rate for mechanical faults in low SNR.

2 Feature Evaluation Function and Extraction of the Local Bispectra

The mechanical faults diagnosis is actually a problem of pattern recognition, whereas the extraction and selection of classification features is the key of pattern recognition. The experiments and engineering applications have demonstrated that bispectra of raw mechanical vibration signals can reveal the fault information[9], [10]. Let $x(t)$ be the continuous mechanical vibration signal. Its bispectra are defined by

$$B(\omega_1, \omega_2) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} c_{3x}(\tau_1, \tau_2) e^{-j(\omega_1\tau_1 + \omega_2\tau_2)} d\tau_1 d\tau_2 \quad (1)$$

where $c_{3x}(\tau_1, \tau_2)$ is the third order correlation of $x(t)$.

Nevertheless, a large amount of bispectra features are included in the raw bispectra. Since too many inputted features will increase the scale of the classification network and affect the accuracy of classifier, it is necessary to evaluate the bispectra features, eliminate the useless ones and select a subset of features with the most discriminant power and sensitivity. The principle of evaluation method is that the less the distance between features in the same class and the larger the distance between features in different classes is, the stronger and more sensitive the feature is. Accordingly, Fisher's class-separability is used as the discriminant measure function to evaluate a large amount of bispectra of fault signals.

Suppose the training set consists of bispectra samples $\{B_k^{(i)}(\omega_1, \omega_2)\}$, $k = 1, 2, 3, \dots, N_i$, where subscript k denotes the computed bispectra of the k th observation record of fault signal, superscript i denotes the i th signal class, and N_i is the set numbers of observation records for the i th class signals, where $i = 1, 2, \dots, M$, and M is the number of fault signals classes. Let $\omega = (\omega_1, \omega_2)$ and $B(\omega) = B(\omega_1, \omega_2)$. Correspondingly, the Fisher class-separability evaluation function between the i th and j th fault classes is defined by [11].

$$E_v^{(i,j)}(\omega) = \frac{\sum_{l=i,j} p^{(l)} \left\{ \text{mean}_k \left(B_k^{(l)}(\omega) \right) - \text{mean}_l \left[\text{mean}_k \left(B_k^{(l)}(\omega) \right) \right] \right\}^2}{\sum_{l=i,j} p^{(l)} \text{var}_k \left(B_k^{(l)}(\omega) \right)}, i \neq j, \quad (2)$$

where $p^{(l)}$ is the prior probability of $B_k^{(l)}(\omega)$, $\text{mean}_k(B_k^{(l)}(\omega))$ and $\text{var}_k(B_k^{(l)}(\omega))$ are mean value and variance of all the sample bispectra at the frequency ω of the l th class of fault signal, and $\text{mean}_l[\text{mean}_k(B_k^{(l)}(\omega))]$ is the total centroid of all the sample bispectra at the frequency ω over all the fault classes. The larger $E_v^{(i,j)}(\omega)$ is, the stronger the separability between class i and class j is. Accordingly, by setting a selection threshold, the frequency subset $\{\omega_m = (\omega_{1,m}, \omega_{2,m}), m = 1, 2, 3, \dots, L\}$ with the most discriminant sensitivity among $E_v^{(i,j)}(\omega)$ for all possible combinations (i, j) can be chosen, and bispectra $B(\omega_m)$ at the local frequencies ω_m are the local bispectra, which can be utilized as the identification vector.

Through feature evaluation the extracted bispectra have not only the most discriminant power, but also can avoid the interference of the cross-terms brought in by the integral paths when using integral bispectra. Thus, the identification feature vector has stronger immunity to the cross-terms and provides the fault signal a strong individual characteristic. More importantly, the local bispectra can eliminate the useless and baneful features from the bispectra of mechanical fault signals, which can decrease the scale of RBF network greatly. Hence, the training and diagnosis is very computational efficient.

3 Classification and Diagnosis Algorithm Using RBF Network

Suppose that the k th observation record of the vibration signal of the l th class mechanical fault is $\{x_k^{(l)}(n)\}$, where $k = 1, 2, 3, \dots, N_l$, $l = 1, 2, 3, \dots, M$ is fault signal class and N_l is the number of observation records for the l th class mechanical fault. The classification and diagnosis algorithm is as follows.

Step 1: Calculate Fourier transform $X_k^{(l)}(\omega)$ for fault signal $\{x_k^{(l)}(n)\}$, and then compute bispectra $B_k^{(l)}(\omega)$ using (3).

$$B_k^{(l)}(\omega) = B_k^{(l)}(\omega_1, \omega_2) = X_k^{(l)}(\omega_1)X_k^{(l)}(\omega_2)X_k^{(l)}(\omega_1 + \omega_2) . \tag{3}$$

Step 2: Use (2) to compute the Fisher class-separability measure $E_V^{(i,j)}(\omega)$ for all class combinations (i, j) , determine the effective number of specified bispectra for between-class (i, j) , and denote it by $L^{(ij)}$. The corresponding frequencies $\{\omega_m^{(ij)}\}$, $m = 1, 2, 3, \dots, L^{(ij)}$ are the effective frequencies.

Step 3: Arrange the obtained effective frequencies $\omega_m^{(ij)}, m = 1, 2, 3, \dots, L^{(ij)}$ into the sequency $\{\omega(q), q = 1, 2, 3, \dots, L\}$, $L = \sum^{(i,j)} L^{(ij)}$, and arrange the corresponding bispectra of the k th record in class l into the sequency $\{B_k^{(l)}(q), q = 1, 2, 3, \dots, L\}$, $k = 1, 2, 3, \dots, N_l$ with the same order of frequency pairs (ω, ω_2) . Define $y_k = [B_k^{(l)}(1), \dots, B_k^{(l)}(L)]^T$ as a $L \times 1$ vector. Thus, N_l template vectors can be obtained for class l . Suppose that the number of template feature vectors for each class of fault signal is N after data fusion. Then, the classification feature vector for the l th class of fault signals is denoted by

$$Y_l = [y_i], i = lN + 1, lN + 2, \dots, (l + 1)N, l \in \{1, 2, \dots, M\}$$

Step 4: Use the feature vector to train the radial-basis function(RBF) neural network as a classifier. Let H be the $MN \times MN$ hidden node output matrix, and its element is computed by

$$h_{ij} = \exp\left(-\frac{\|y_i - y_j\|^2}{\sigma^2}\right), \tag{4}$$

where the variance σ^2 of the Gaussian kernel function is the total variance of all feature vectors $y_i, i = 1, 2, 3, \dots, MN$. Accordingly, the weight matrix of the RBF neural network is given by

$$W = (H^H H)^{-1} H^H O, \tag{5}$$

where

$$O = \begin{bmatrix} 1 \dots 1 & 0 \dots 0 & \dots & 0 \dots 0 \\ 0 \dots 0 & 1 \dots 1 & \dots & 0 \dots 0 \\ \dots & \dots & \dots & \dots \\ 0 \dots 0 & \dots & 0 \dots 0 & 1 \dots 1 \end{bmatrix}, \tag{6}$$

is the $(MN) \times M$ output matrix. Once the RBF neural network as a classifier is trained, the weight matrix W is stored.

Step 5: Implement the testing task for diagnosis using the trained RBF neural network. Let $\bar{y}_l = [y(1), y(2), \dots, y(L)]^T$ be the extracted feature vector evaluated

through a set of computed bispectra of an unknown mechanical fault class. Then, the RBF network’s hidden node output $MN \times 1$ vector $\bar{h} = [h_i]$ corresponding to \bar{y}_i can be computed as

$$h_i = \exp\left(-\frac{\|\bar{y}_i - \bar{y}_i\|^2}{\sigma_i^2}\right), \tag{7}$$

where the variance σ_i^2 in the Gaussian kernel function is the variance of the feature vector \bar{y}_i determined in the training phase. Then, the output vector of the RBF neural network is given by

$$\bar{O}_T = \mathbf{W}^T \bar{h}, \tag{8}$$

which gives the diagnosis result of the unknown mechanical fault class.

4 Experiment Results

The observation records of fault signals utilized by the experiments are recorded from a group of rolling bearings with the model `**6204`, and the rotational speed is 2416Hz during the sampling procedures. The raw mechanical vibration signals are recorded at four different working modes, which are bearing normal, outer ring damage, inner ring damage and stenting damage, and the four modes are denoted by $T_i, i=1,2,3,\dots,4$. Fig. 1 shows the typical signal waves in time domain for the four classes of raw mechanical faults.

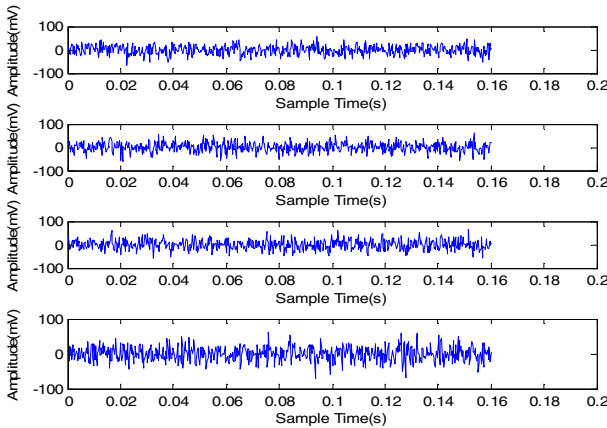


Fig. 1. Raw mechanical vibration signals for four classes of working modes (normal, outer ring damage, inner ring damage and stenting damage)

A total of 200 records of raw mechanical signals are observed and analyzed, in which every 50 records are sampled from one bearing working mode distinguishingly, and the data length of each record is 2048. According to the classification algorithm proposed in Section 3, a total of 200 feature vectors can be obtained, and they are divided into two groups of vector sets. The training set has 120 feature vectors, in which every 30 vectors are extracted from one bearing working mode, and the remaining 20 vectors extracted from the same working mode are used by the testing and diagnosis set, which has 80 feature vectors in total. In contrast to the local bispectra, the direct bispectra and radially integrated bispectra(RIB) are also used as classification feature vectors in the way of the aforementioned diagnosis procedures in Section 3, and the test results are given in Table 1.

Table 1. Recognition rates for the four classes of mechanical fault signals (S/N=10dB)

Class of Mechanical faults	Recognition rates using different identification feature vectors		
	Direct Bispectra	RIB	Local Bispectra
T1	68.3%	74.3%	90.5%
T2	70.2%	76.2%	92.3%
T3	67.7%	74.8%	89.7%
T4	69.3%	75.8%	92.9%
Average recognition rate	68.9%	75.2%	91.3%

As can be seen from Table 1, the average recognition rate is lower than 70% when direct bispectra is used as the feature vector, and the percentage can be raised to 76% approximately when RIB is used. However, when the local bispectra are utilized, the average recognition rate is over 91%. The experiment results with rolling bearings also demonstrate that the local bispectra features are able to reveal the fine difference between individual mechanical fault signals when SNR is low as 10dB.

5 Conclusions

This paper has introduced a classification technique for mechanical faults. The main objective of this technique is to represent each fault signal class by a feature vector with sufficient accuracy, and bispectra features, Fisher's class-separability discriminant evaluation function and RBF neural network are the main signal processing tools employed in the proposed technique. The experiments for fault signals of rolling bearings demonstrate that this new scheme can extract a strong individual feature, decrease the scale of RBF network greatly and improve the accuracy of diagnosis.

In this paper, the main concern was to describe an efficient technique for mechanical faults and to verify its usefulness in a realistic and reasonably challenging classifier design experiment. Future work, however, needs to extend applications to more complicated classification problems regarding the decision speed, memory requirements for reference storage as well as the number of fault classes.

References

1. Sato T., K. Sasaki.: Bispectral Holography. *Journal of A Coustical Society of America.* **62** (1977) 404–408
2. Barker R.W., Klute G.A., HinichM J.: Monitoring Rotating Tool Wear Using High Order Spectral Features. *Transactions of the ASME. J. Engineering for Industry.* **115** (1993) 23–29
3. Murray A., Penman J.: Extracting Useful Higher Order Features for Condition Monitoring Using Artificial Neural Networks. *IEEE Trans. on Signal Processing.* **45** (1997) 2821–2828
4. McCormick A.C., Nandi A.K.: Bispectral and Trispectral Features for Machine Condition Diagnosis. *IEE Proceedings -Vision, Image and Signal Processing* **146** (1999) 229–234
5. B Eugene Parker Jr, Ware H A.: Fault Diagnostics Using Statistical Change Detection in the Bispectral Domain. *Mechanical Systems and Signal Processing* **14** (2000) 561–570
6. Chandran V., Elgar S.L.: Pattern Recognition Using Invariants Defined from Higher Order Spectra-One-Dimensional Inputs. *IEEE Trans. on Signal Processing* **41** (1993) 205–212
7. Tugnait J.K.: Detection of Non-Gaussian Signals Using Integrated Polyspectrum. *IEEE Trans. on Signal Processing.* **42** (1994) 3137–3149
8. Liao X.J., Bao Z.: Circularly Integrated Bispectra: Novel Shift Invariant Features for High Resolution Radar Target Recognition. *Electronics Letters.* **34** (1998) 1879–1880
9. Fackrell J.W.A., White P.R., Hammond J K.(ed.): The Interpretation of the Bispectra of Vibration Signals I:Theory. *Mechanical Systems and Signal Processing* **9** (1995) 257–266
10. Fackrell J.W.A., White P.R., Hammond J K.(ed.): The Interpretation of the Bispectra of Vibration Signals II: Experimental Results and Application. *Mechanical Systems and Signal Processing* **9** (1995) 257–266
11. Zhang, X, Shi, Y., Bao, Z.: A New Feature Vector Using Selected Bispectra for Signal Classification With Application in Radar Target Recognition. *IEEE Trans. on Signal Processing* **49** (2001) 1875–1885

Temperature-Variation Fault Diagnosis of the High-Voltage Electric Equipment Based on the BP Neural Network

Zhen-Yu Wang, Yong-Wei Li, Peng Guo, and Xiao-Fang Yu

College of Electric and Information Engineering, Hebei University of Science and Technology, 050018, Shijiazhuang, China
wzykd@sohu.com, 05991li@163.com, guopeng993@163.com,
yuxiaofang1@163.com

Abstract. As high-voltage electric equipment has complex structure and works in harsh environment, this paper is aimed at applying Optical Fiber Sensors to a temperature-variation fault diagnosis system of high-voltage electric equipment based on the combination of neural network and expert system. Neural network has the characteristics of self-adapted, distributed storage and associative memory. Using BP neural network, we can make on-line diagnosis of temperature-variation fault of high-voltage electric equipment. All the uses above can increase the speed of diagnosis and make results be more exact.

1 Introduction

In recent years, there is an ever-growing demand for electric power and stability of high-voltage electric equipment. Temperature-variation is the major signal to indicate early hidden troubles of high-voltage electric equipment, and it is realistically significant to monitor high-voltage electric equipment. But with the harsh environment of the high-voltage monitoring, the fault monitoring becomes more difficult. FBG (Fiber Bragg Grating) sensors have the merit of high anti-jamming, high stability in communication, endurance of high-voltage, and great security [1], [2]. So in this paper it is applied to high-voltage electric equipment online monitoring to gain accuracy.

The reasons of temperature-variation fault diagnosis of high-voltage electric equipment are always complex. Taking temperature as the only factor, it is difficult to diagnose the fault. Besides temperature, we also need other factors such as current, voltage for parallel disposal and synthesized diagnosis. Neural network has the function of distributed storage and large-scale parallel disposal, so it can be used to temperature-variation fault diagnosis of high-voltage electric equipment. BP neural network now is one of the most sophisticated and widely used methods [3].

Neural network has net-input and net-output, net-output reflects the results of fault diagnosis. But when it comes to accurate temperature-variation fault diagnosis of high-voltage electric equipment, the net-output conclusions are not clear enough [4]. This paper puts forward a method to monitor temperature-variation fault diagnosis of high-voltage electric equipment by combining BP neural network with expert system [5], [6].

2 Application of FBG in High-Voltage Electric Equipment Monitoring

In this paper, we use optical fiber sensors based on FBG techniques to monitor temperature variation in high-voltage electric equipment. FBG sensors can fulfill the real distributed-measuring, high space resolution, high precision and good real-time character, so it can be well applied to temperature monitoring in high-voltage electric equipment. FBG sensors work on the C waveband, the wavelength of spectrum is from 1525nm to 1625nm. FBG sensors reflect the temperature variation by wavelength variation, every change of 100°C can make a wavelength variation of 1nm. The Bragg wavelength is given by Equation (1).

$$\lambda_{eff} = 2n_{eff} \Lambda, \tag{1}$$

where, n_{eff} is the effective refractive index of the fiber(modal index) and Λ is the Bragg grating period. We obtain Bragg wavelength shift $\Delta\lambda_B$ in temperature sensors by differentiating Equation (1) with respect to temperature:

$$\Delta\lambda_B = 2\Delta n_{eff} \Lambda + 2n_{eff} \Delta\Lambda. \tag{2}$$

3 Structure of Monitoring System Based on FBG Sensors

The system based on FBG sensor techniques can realize distributed-monitoring by different modules. There are three parts: host computer, industrial controllable computer and monitoring modules based on FBG sensors. The structure is shown in Fig. 1.

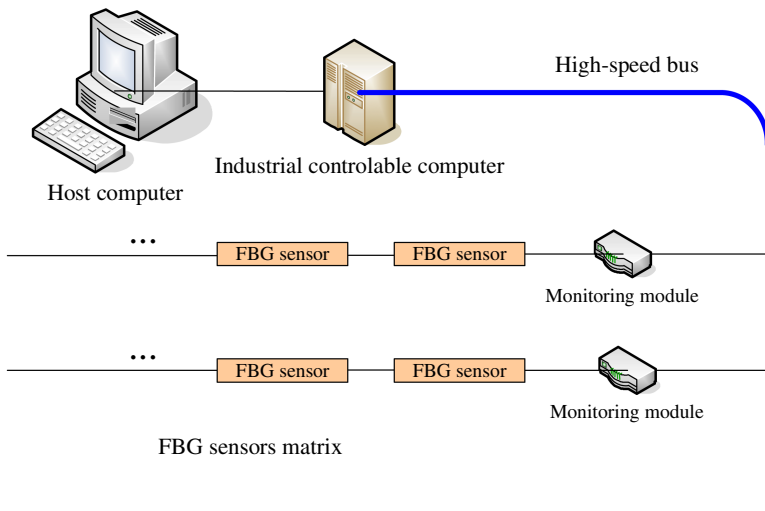


Fig. 1. Structure of temperature-variation fault monitoring system of high-voltage electric equipment

4 Combine the Neural Network with Expert System

BP neural network has good capability of mode recognition. The basic principle of BP algorithm is to revise weights from output layer according to the total error between expected output and actual output of the sample, until the error of the two becomes smaller than fixed value. The training of BP neural network is composed of the course of output by forward propagating and the course of adjusting by back propagating. In the course of forward propagating, input signals from input layer propagate to output layer by disposing in hidden layer. The state of every layer of neuron only influences the state of next layer: If cannot get expected output from output layer, the error signal of output will propagate back along with the original connected path, until it get to input layer. By revising the weights through the path, it will make the total error to be the minimum value.

The input of neural network reflects the feature of diagnosed object. The output shows possibility of fault, but it doesn't have the function of explanation. Expert system is a reasoning system based on symbol reasoning. The inference engine of expert system under the support of knowledge base and database uses different regulations to do some reasoning to get results, so it has the function of explanation. We designed a system by combining expert system with BP neural network to diagnose fault of high-voltage electric equipment.

Expert system, also called the system based on knowledge. It can solve the problem which the model is difficult to set up or not set up yet, the knowledge base is easy to expand and perfect, and has the ability of self-study and self-refresh, so it is widely applied to fault diagnosis of electric equipment. This system is made up by main 5 parts, such as knowledge base, reasoning machine, database, knowledge getting part by neural network and man-machine interface.

Reasoning machine is the organization part which controls the structure by data information in database. Using rules in knowledge base and carrying on reasoning according to certain reasoning tactics, it gets the corresponding conclusion. The database is mainly to store data during equipment running and original information

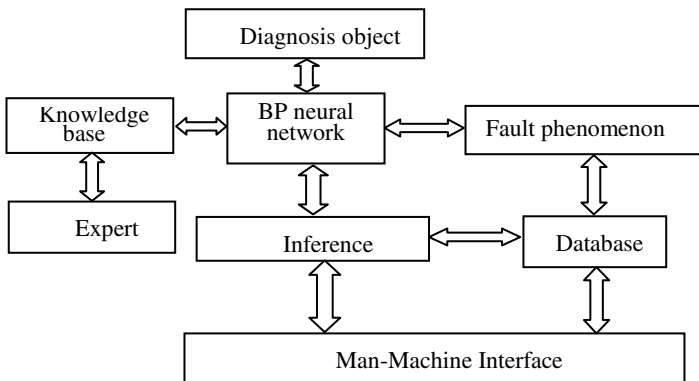


Fig. 2. The structure of temperature-variation fault diagnosis system of the high-voltage electric equipment

from fault diagnosis. The neural network has self-study function, so it is used to obtain knowledge from expert experience and realistic fault. The man-machine interface carries on media of information interchange, offering the ocular and convenient reciprocation means to users. The structure of fault diagnosis system is shown in Fig. 2.

The expression of knowledge in neural network is hidden, it is expressed by topology structure and connected values. Neural network is a net system which unified the information storing and proposing. Combining neural network with expert system, reasoning in the course of knowledge storing and answer getting are all solved, neural network is the engine of knowledge and inference.

5 Fault Diagnosis Based BP Neural Network

5.1 The Construction of Three Layers BP Neural Network Model

The three layers BP neural network is composed of an input layer, an output layer and a hidden layer. The structure is shown in Fig. 3.

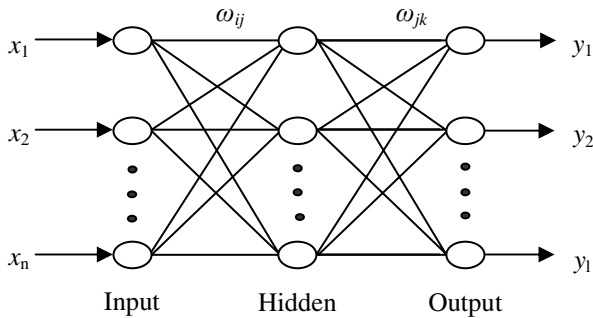


Fig. 3. Three layer neural network based on BP model

where, input vectors are $X = (x_1, \dots, x_i, \dots, x_n)$, output vectors are $Y = (y_1, \dots, y_k, \dots, y_l)$, and expected output vectors are $D = (d_1, \dots, d_k, \dots, d_l)$, the weights matrix from input layer to hidden layer and from hidden layer to output layer is W , ω_{ij} is the weights from the i node of input layer to the j node of the hidden layer, ω_{jk} is the weights of the j node of the hidden layer to the k node of output layer, and the activation function is sigmoid function:

$$f(x) = \frac{1}{1+e^{-x}} \tag{3}$$

5.2 The Training of Neural Network

The training of neural network can be divided into four steps:

(1) At the beginning of training, weights and threshold values of nodes are initialized by a group of random numbers.

(2) Input p training samples, these are (x_1, x_2, \dots, x_p) . And initializing expected outputs, there are a group of teacher values (d_1, d_2, \dots, d_p) .

(3) After calculated by the potential function of neural network, the error of actual output propagates back to input layer, which makes the neural net continue to revise weights and threshold values of nodes to reduce the function value.

(4) Making y_p close to d_p as much as possible, until it reduces to a threshold value that can be accepted or cannot reduce again.

Setting n_0 as iteration number, the revising formulas of weights and threshold values are:

$$\omega_{jk}(n_0 + 1) = \omega_{jk}(n_0) + \eta \sum_{\lambda=1}^p \delta_{jk}^\lambda o_j^\lambda, \tag{4}$$

$$\omega_{ij}(n_0 + 1) = \omega_{ij}(n_0) + \eta \sum_{\lambda=1}^p \delta_{ij}^\lambda o_i^\lambda, \tag{5}$$

where η is the length of step.

$$\delta_{jk}^\lambda = (t_k^\lambda - y_k^\lambda) y_k^\lambda (1 - y_k^\lambda), \tag{6}$$

$$\omega_{ij}(n_0 + 1) = \omega_{ij}(n_0) + \eta \sum_{\lambda=1}^p \delta_{ij}^\lambda o_i^\lambda, \tag{7}$$

when the total error E can fulfill the conditions:

$$E = \frac{1}{2} \sum_{k=1}^l (t_k^l - y_k^l)^2 < \varepsilon, \tag{8}$$

then the course can be terminated, and the training of weights and threshold values of nodes is finished.

5.3 Classification of Faults and Restoration of Characteristics

According to the characteristics of high-voltage electric equipment, temperature-variation can often reflect faults. But other factors also can affect fault diagnosis, such as environment variety, high load (mainly caused by high voltage or big current), and different equipment have different features. Here temperature-variation is the main reason of faults and we classified the faults into five different kinds: temperature-variation fault, high load resulting from high voltage, high load resulting from big current, high environment temperature and running normally. Then using the analysis of high-voltage electric equipment and normalization, we get 8 characteristics.

5.4 The Steps of Fault Diagnosis by Neural Network

BP neural network uses the policy of forward reasoning driven by data, which is reasoning from the initial state to the final state. There are four steps:

- (1) Input fault samples to every neuron of input layer.
- (2) Calculate output of neurons and regard it as input of the output layer.
- (3) Try to get the output according to Equation (3).
- (4) Using the weights function to get final results of neuron from output layer

$$F_l = \begin{cases} True, & \text{if } y_k > \varphi, \\ False, & \text{other,} \end{cases} \tag{9}$$

where, φ is the threshold value.

5.5 Fault Diagnosis

According to the characteristics of temperature-variation fault diagnosis, we choose three layers BP neural network. The number of input layer, hidden layer and output layer is: 8, 6 and 5; the steplength is 0.01, and the system error is 0.01. The number of net training is 5000. Five fault samples are shown in Table 1. Through MATLAB simulation, we get the output of 5 fault samples shown in Table 2.

Table 1. Typical fault samples

Fault samples	Characters							
	1	2	3	4	5	6	7	8
Temperature-variation fault	0.00	0.10	0.00	0.00	0.80	0.00	0.00	0.20
High load caused by high voltage	0.05	0.00	0.06	0.90	0.10	0.20	0.08	0.15
High load caused by big current	0.95	0.20	0.30	0.05	0.01	0.15	0.02	0.08
High environment temperature	0.11	0.02	0.30	0.15	0.20	0.12	0.85	0.07
Running normally	0.00	0.00	0.01	0.40	0.11	0.87	0.05	0.30

The eight characteristics of “temperature-variation fault” in Table 1. are trained by neural network, and in the five outputs of this fault, only y_2 can meet the demand:

$$y_2 = 0.8900 > \varphi = 0.85 . \tag{10}$$

So we can get the result that there is a “temperature-variation fault”.

Table 2. The target outputs of the typical fault sample

Fault samples	Output y				
	1	2	3	4	5
Temperature-variation fault	0.0001	0.8900	0.2200	0.0020	0.0020
High load caused by high voltage	0.0020	0.0012	0.0050	0.8650	0.0031
High load caused by big current	0.0000	0.0003	0.0031	0.0021	0.9501
High environment temperature	0.9773	0.3001	0.0010	0.0050	0.1200
Running normally	0.1002	0.0001	0.9100	0.0000	0.0030

6 Sections Needed to Be Improved

Although the results of combination of BP neural network and expert system are good, there are still some parts need to be further researched:

(1) BP algorithm is a kind of algorithm based on gradient descent, so the convergence speed in calculating course is slow, and if the function of connected values and weights is not chosen rightly, it may not get the convergence values.

(2) Based on neural network fault diagnosis have a good function of self-study, it can solve the problem of acquiring knowledge. But it needs lots of samples to get a stable result, and the scale of neural net cannot be too big in training.

(3) The nodes number of hidden layer must to be confirmed as soon as the topology structure of BP neural network is fixed. So it is necessary to optimize the structure of neural net.

(4) To diagnose different fault of high-voltage electric equipment, we need to build up different kinds of BP neural net, that maybe increase the convergence speed.

7 Conclusions

In this paper, we use FBG sensors to monitor the temperature-variation fault diagnosis of high-voltage electric equipment on-line based on the combination of neural network and expert system. Through MATLAB simulation, the method is proved to be effective.

Acknowledgements

This paper was supported by the National Natural Science Foundation of China (Grant No.60674107), the Natural Science Foundation of Hebei Province (Grant No.F2006000343), and the Research and Development Plan of Science and Technology of Shijiazhuang (Grant No.06713026A).

References

1. Sun, S.: The Fiber Measurement and Sensing Technology. Harbin Polytechnical University Press (2002)
2. Jung, J., Nam, H., Lee, B., Byun, J. O., Nam, S. K.: Fiber Bragg Grating Temperature Sensor with Controllable Sensitivity. *Applied optics* **38** (13) (1999) 2752-2754
3. Zhang, D., Shao, H.: A Illation Method of Fault Diagnosis Based on Neural Network. *Journal of Shanghai Jiaotong University* **33** (5) (1999) 619-621
4. Wu, L.: A Fault Diagnosis Expert System Based on Neural Network.
5. Zhang, Y., Wang, H., Zhu, Y., Yang, Z.: Study for a Fault Diagnosis Expert System Based on Artificial Neutral Network. *Measurement & Control Technology* **23** (11) (2004) 55-57
6. Fan, H., Xiao, M., Xiang, H.: Studies of the Fault Diagnosis Expert System Based Neural Nets. *Modern electric techniques* **9** (2002) 29-31

Diagnosis of Turbine Valves in the Kori Nuclear Power Plant Using Fuzzy Logic and Neural Networks

Hyeon Bae¹, Yountae Kim¹, Gyeongdong Baek¹, Byung-Wook Jung¹,
Sungshin Kim¹, and Jung-Pil Shin²

¹ School of Electrical and Computer Engineering, Pusan National University, Jangjeon-dong,
Geumjeong-gu, Busan 609-735, Republic of Korea

{baehyeon, dream0561, gdbaek, wooroogy, sskim}@pusan.ac.kr

² Department of Computer Software, The University of Aizu,

Aizu-Wakamatsu City, Fukushima 965-8580 Japan

jpshin@u-aizu.ac.jp

Abstract. This manuscript introduces a fault diagnosis system for a turbine-governor system that is an important control system in a nuclear power plant. Because the turbine governor system is operated by high oil pressure, it is very difficult to maintain the operating condition properly. The turbine valves in the turbine governor system supply an oil pressure for operation. Using the pressure change data of the turbine valves, the condition of the turbine governor control system is evaluated. This study uses fuzzy logic and neural networks to evaluate the performance of the turbine governor. The pressure data of the turbine governor and stop valves is used in the turbine governor diagnosis algorithms. The features of the pressure signals are defined to be applied in the fuzzy diagnosis system. And Fourier transformed signals of the pressure signals are used in the neural network models for diagnosis. The diagnosis results both by fuzzy logic and neural networks are compared to evaluated the performance of the designed system.

1 Introduction

The nuclear power generation was introduced in the middle of 1950's and has been continuously expanded until now. Recently, 440 nuclear power generations are charging 16% of total electric power production in the world [1]. However, the nuclear power generation is a large-scaled and complex system that firstly requires stability of the plant. Therefore, it is very difficult and complicated to control and manage the power generation. In nuclear power stations, various types of the system faults occur. Particularly, when the important functional devices are broken, the fault can be derivatively enlarged to the serious accident, that is, a radiation accident. Therefore fault diagnosis and management should be precisely achieved for stable operation of the plant.

In this study, the turbine valve system that is one of the core systems in the nuclear power plant is the target system. It is difficult to acquire data from the operating system of the turbine valve because it is a mechanically controlled device using oil

pressure and to prognose it because it has strong nonlinearity. Oil for pressure preservation leaks in several parts, so fault diagnosis and maintenance is complicated. From the year 2001 and to the end of the year 2005, in Kori nuclear power plant, the turbine operating systems were broken 30 times and at the times it was difficult to analyze find the fault causes [2-4].

In the latest studies, NPPC (national pollution prevention center) was developed at Georgia Institute of Technology. This system is an expert system that supports the operator of the nuclear power plant to find the cause of the abnormal failure applying the system operating model and relative rules [5]. EG&G Idaho company designed a rule based expert system that inspects measuring instruments and diagnoses the nuclear reaction systems in order to check the reaction conditions [6]. The diagnosis systems of the nuclear power plant are newly developed using neural networks and fuzzy logic recently. But researches related to governor are not sufficiently achieved.

In this study, the operator support system was developed to diagnose the governor operating system based on inspecting the pressure variation of the turbine valve operating system. But it is not easy to diagnose and analyze the operating systems in the field. In this study, fuzzy logic and neural networks were applied to design the diagnosis system. Both techniques are representative intelligent methods to make models and rules.

In this manuscript, Section 2 shows description of turbine operating systems of nuclear power generation, Section 3 explains normal operation and diagnosis information of the valves. Section 4 deals with fault diagnosis using fuzzy logic and Section 5 introduce fault diagnosis using neural networks. The conclusions are summarized in Section 6.

2 Turbine Operation Systems of Nuclear Power Generation

2.1 Pressurized Water Reactor (PWR)

This research dealt with the pressurized water reactor (PWR) that is a global model for power generation. This is the most common type, with over 230 in use for power generation and a further several hundred in naval propulsion. The design originated as a submarine power plant. It uses ordinary water as both coolant and moderator. The design is distinguished by having a primary cooling circuit which flows through the core of the reactor under very high pressure, and a secondary circuit in which steam is generated to drive the turbine. Pressure is maintained by steam in a pressurizer.

As shown in Fig. 1, the water is also the moderator in the primary cooling circuit, and if any of it turned to steam the fission reaction would slow down. The secondary shutdown system involves adding boron to the primary circuit. The steam drives the turbine to produce electricity, and is then condensed and returned to the heat exchangers in contact with the primary circuit [7].

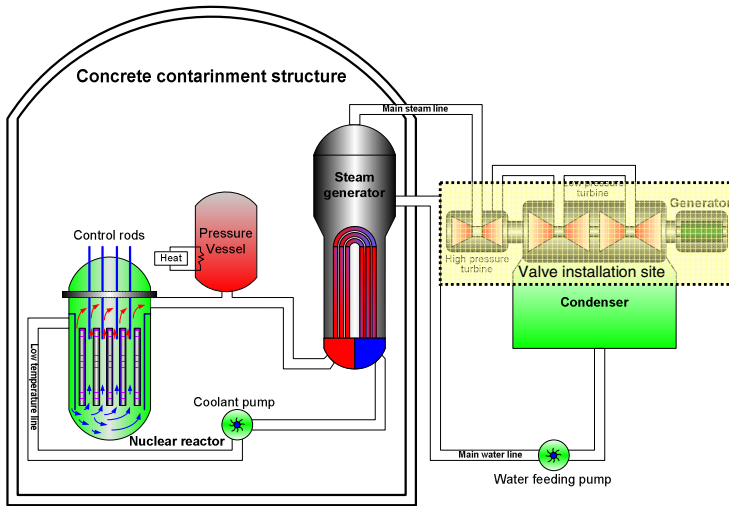
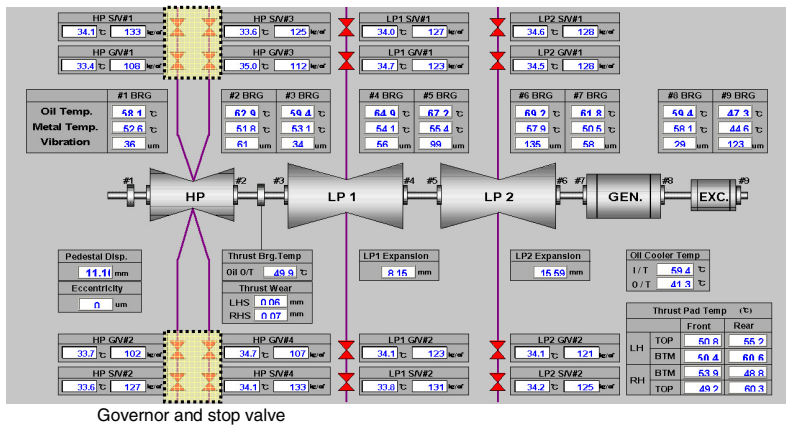


Fig. 1. A system structure of the pressurized water reactor

2.2 Summary of the Governor Operating System

2.2.1 Operating Systems in Physical Plant

Figure 2 shows a monitoring window of the power governing valve in the nuclear power plant. In this study, we want to diagnose the high pressure stop valve and governor valve as shown in the boxes. The oil supply and operating system of the turbine valve is installed in each turbine valve to drive the turbine valve. The turbine valve opens by the hydraulic servo cylinder and closes by the compressed spring. If a driver is broken, the broken driver can be repaired without turbine shut down [8].



Governor and stop valve

Fig. 2. The hydraulic lines of turbine valve driver

2.2.2 Control Characteristics of the Turbine Governor

At starting up, it is necessary that the turbine governor controls a fluid to the reference speed. At driving, the turbine governor controls it to 1800 rpm with 4 groups 16 valves (as shown in Fig. 2). The high pressure turbine valve handling about 80% of the fluid must have good responsibility with respect to the control signal and maintain variation of valve opening to low 0.5% under any circumstances.

Pressure can be changed corresponding to conditions of the governor valves and stop valves. Pressure is cyclically controlled and an internal leak continuously occurs in the driving system because of the mechanical characteristic of the system. This pressure variation affects on valve control. This can cause malfunction of the driving system. Figure 3 shows pressure signal of the valve. The slope of unloading time is closely related to the valve condition, so the values are used for valve diagnosis in this study. Loading time is the time interval that pressure reaches to the set-point (pressurization). Unloading time is the time interval that measures from the peak point to the valley point of unloading time (compression).

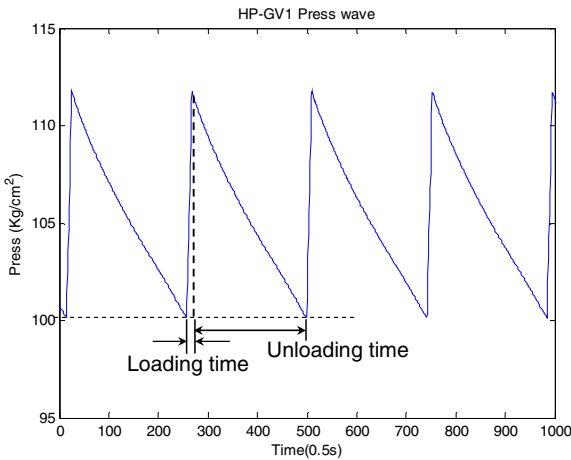


Fig. 3. Control patterns (cycling) of a governor valve based on pressure

3 Normal Operation and Diagnosis Information of the Valves

3.1 Oil Pressure of Each Valve Under Normal Operation

Oil pressure of the governor valve should be pressurized from 100 to 110kg/cm² for handling the driving system and it must be persevered more than 80kg/cm² under any circumstance. If pressure variation is bigger, then internal leak of the cylinder and valve swing can happen. If pressure is less than 80kg/cm² that surmounts spring tension of the inner cylinder, the valve can close regardless of the control signal. Thus, it is best to shorten loading time and to lengthen unloading time for optimization of the operating system.

3.2 Operation Information for Fault Diagnosis

3.2.1 Operation Information of a Stop and Governor Valve

Under normal operation, 55~60% of valve opening and 100~115 of pressure is kept and then if the dump solenoid coil is cut off, oil pressure is nosedived by loss of the power supply and then valve opening is quickly closed. Coil defect, stop interlocking driving, and inner faults of the solenoid driver are major fault causes of this phenomenon. Close interlocking is promptly closed after 4 seconds when 20% of the deviation value comparing with the reference value occurs. This fault can happen based on motor stop. Table 1 and 2 show the standard values of time and pressure.

Table 1. Standard values for the operation pressure of the stop and governor valve

Variables	Valve type	Standard value (kg/cm^2)	Field value (kg/cm^2)
Bottom pressure	Stop valve	123~127	125~126
	Governor valve	96~102	99~104
Upper pressure	Stop valve	134~140	135~140
	Governor valve	107~113	107~113

Table 2. Standard values for the operation loading time of the stop and governor valve

Variables	Valve type	Standard value (s)	Field value (s)
Loading time	Stop valve	Under 6	2.6~4.2
	Governor valve	Under 6	2.6~4.2
Unloading time	Stop valve	Over 15	27~43
	Governor valve	Over 15	Over 30

3.2.2 Diagnosis Information for the Stop and Governor Valve

Qualitative and quantitative features of loading time are finally used to present the operating conditions of the valves. Practically, field operators usually diagnose the leak condition and other defects using the slop of loading time. In this study, we want to develop the automatic diagnosis algorithm that extracts systematic rules from data based on the field information.

Basic information for rule extraction in this study is shown in Table 3 and 4. The final rules used in the fuzzy diagnosis system are generated by the information. The valve condition corresponding to loading time was defined by expert’s knowledge.

Table 3. Fault diagnosis rules for the loading time

Loading time	Fault condition
Short (under one (s))	No inner leak
Normal (from one to six (s))	Exist inner leak but normal operation is possible
Long (over six (s))	Leak of cylinder and relief, unloading, and servo valve

Table 4. Fault diagnosis rules for the system pressure

Features	Fault condition
Vertical decline	Dump solenoid valve closing
Exponential decline	Motor stop, pump broken, large quantity of cylinder leak
Cycling action	Normal operation

3.3 Measurement Signals for Valve Diagnosis

The goal of the diagnosis system is to classify the valve conditions to the four levels such as very good, good, not bed, and bed. Conditions of the valves are assessed using the valve pressure based on operating information that is mentioned above. Pressure magnitudes and patterns of each valve indicate the status of the valves. The pressure signals are generally used in the physical fields. Field operators can empirically make a decision of the valve condition using the pressure signals. Figure 4 and 5 show the pressure signals of the stop and the governor valves based on the valve condition. The stop valve has small leakage in order of SV3, SV4, SV2, and SV1 and the governor valve has small leakage in order of GV1, GV4, GV2, and GV3.

Figure 6 shows distribution of loading time and unloading time to indicate the valve conditions. In this distribution, loading time of the stop and the governor valve is not sufficient for rule generation, because the conditions can not be classified significantly. However, unloading time is a good feature to recognize the valve conditions. But GV1 and GV4 are not detected well by loading time, so in this study, the other feature was defined such as the area ratio.

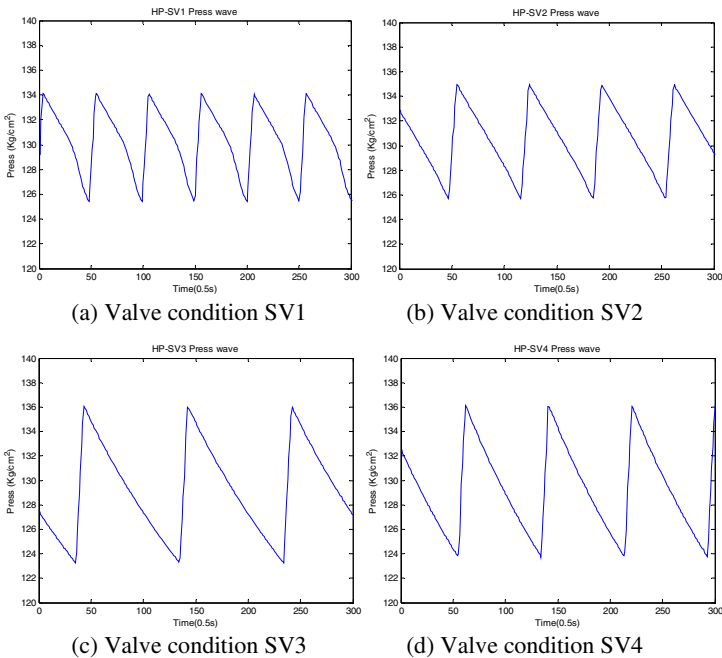


Fig. 4. The pressure feature of the high pressure turbine stop valves

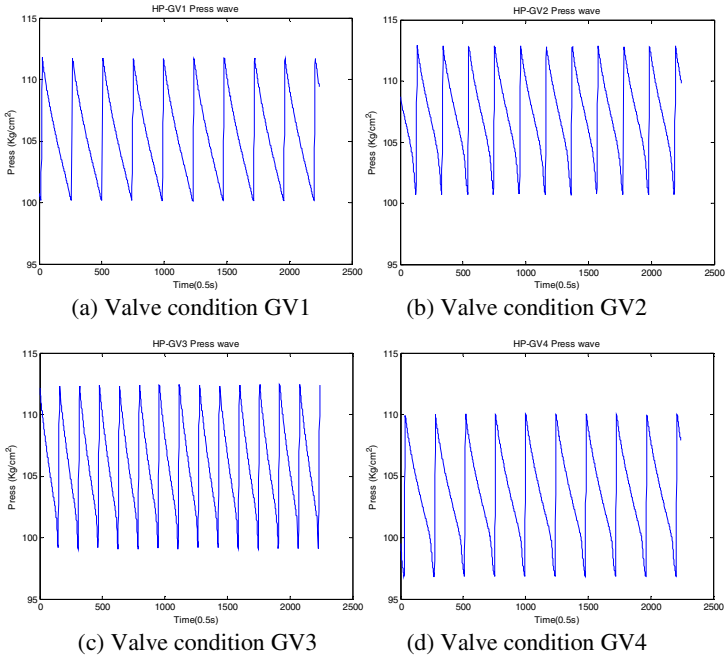


Fig. 5. The pressure feature of the high pressure turbine governing valves

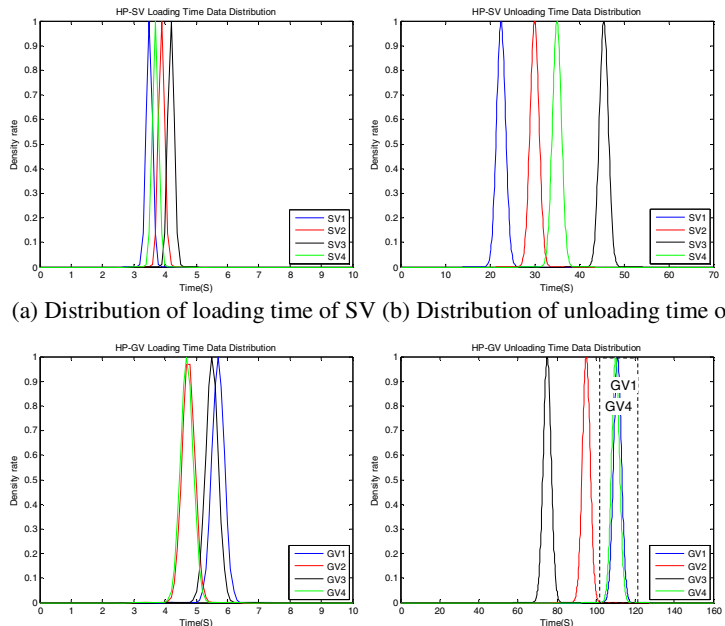


Fig. 6. The distribution of the unloading time of the stop and governor valves

4 Fault Diagnosis Using Fuzzy Logic

Because in this study, the conditions GV1 and GV4 of the governor valve can not diagnosed by loading time alone, the complementary feature is necessary. As shown in Fig. 7 (a), a triangle is built with three lines between the peak point and the valley point of loading and unloading time, and both valley points. The final pattern is calculated by difference between the physical area and the triangular area. As shown in Fig. 7(b), GV1 and GV4 can be classified by the new defined feature, that is, the area ratio (ref. Eq. (1)). Final fuzzy rules are extracted as shown in Table 5. The fuzzy rules are generally generated by operator’s knowledge. Both unloading time and the area ratio are feature variables for the rules.

$$\text{Area ratio} = \frac{\text{physical area} - \text{triangular area}}{\text{triangular area}} \tag{1}$$

The diagnosis results of the high pressure turbine valves are shown in Table 6. The high pressure turbine valve consists of the four stop valves and the four governor valves. When the output of the fuzzy diagnosis results is smaller, the valve condition is better. The diagnosis result is the same with the expert decision. Therefore the diagnosis system has good accuracy for field application. The experiment results mean that the high pressure valves can be evaluated and diagnosed by the fuzzy diagnosis system.

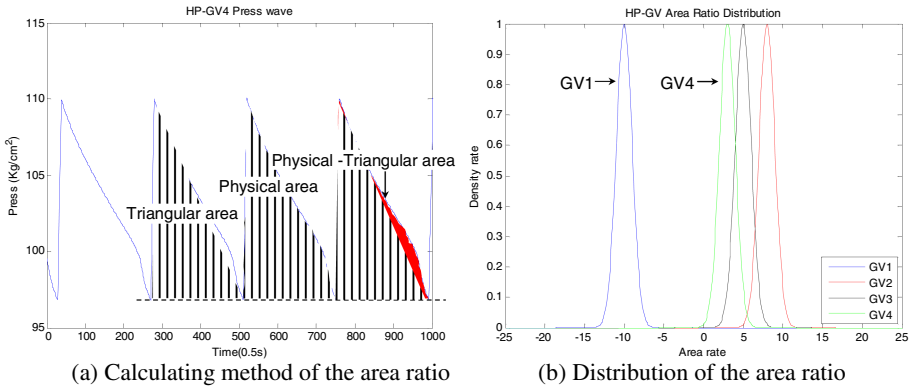


Fig. 7. The calculation of unloading area ratio and the ratio distribution of a governor valve

Table 5. Final fuzzy rules extracted from both information for the stop and governor valve

R1:	IF Unloading time is Short and Area ratio is Minus value then Oil leak is Large
R2:	IF Unloading time is Short and Area ratio is Zero value then Oil leak is Large
R3:	IF Unloading time is Short and Area ratio is Plus value then Oil leak is Large
R4:	IF Unloading time is Medium and Area ratio is Minus value then Oil leak is Medium
R5:	IF Unloading time is Medium and Area ratio is Zero value then Oil leak is Medium
R6:	IF Unloading time is Medium and Area ratio is Plus value then Oil leak is Medium
R7:	IF Unloading time is Long and Area ratio is Minus value then Oil leak is Small
R8:	IF Unloading time is Long and Area ratio is Zero value then Oil leak is Small
R9:	IF Unloading time is Long and Area ratio is Plus value then Oil leak is Medium

Table 6. The performance evaluation for the stop and governor valves

Valve	Status	Unloading time (s)	Area ratio	Fuzzy results	Expert decisions
Stop valve	SV1	22	13	0.549 (4 th)	Bad
	SV2	30	2.5	0.275 (3 rd)	Not bad
	SV3	45.5	-7.5	0.0632 (1 st)	Best
	SV4	36	-7	0.216 (2 nd)	Good
Governor valve	GV1	115	-10	0.0817 (1 st)	Best
	GV2	99	9	0.288 (3 rd)	Not bad
	GV3	75	4	0.689 (4 th)	Bad
	GV4	115	3.5	0.153 (2 nd)	Good

5 Fault Diagnosis Using Neural Networks

Fault diagnosis using fuzzy logic shows good performance for field application, but the several stages for rule extraction and rule evaluation are necessary. The fuzzy rules can be easily generated if good expert’s knowledge exists, but if no sufficient information, it is very difficult to extract rules. To solve the weak point, neural networks are applied to design the diagnosis model for comparing with fuzzy logic.

In this study, Fourier transform (FFT) is applied to extract features from the pressure data. Power values and first three frequencies of the peak points of FFT, and difference values between maximum and minimum of the pressure were used for inputs of the neural network models in diagnosis. Figure 8 shows the features that are extracted from time and frequency based signals such as pressure and FFT signals.

The transformed data are used for neural network inputs and then the fault is diagnosed by the frequency data. Backpropagation, one of the neural network structures, is applied and the node number is determined by the trial and error method. Table 7 shows the final results of valve diagnosis. The four conditions of the stop valve and the governor valve were classified clearly by the diagnosis model. As shown in Table 7, the four conditions for each valve could be identified by the neural network models using the four features. The results mean the selected features for inputs have the specific information for diagnosis.

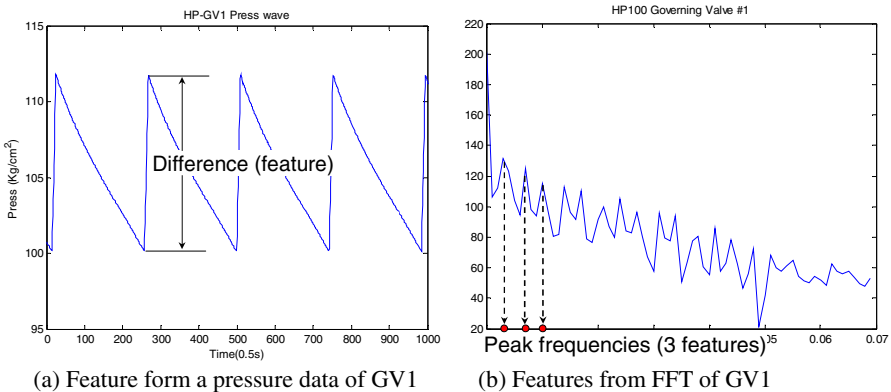


Fig. 8. The results of Fourier transform of the pressure signals

Table 7. The results of valve diagnosis using neural networks

Valve Target No	Conditions of Stop Valves								Conditions of Governor Valves							
	SV1		SV2		SV3		SV4		GV1		GV2		GV3		GV4	
	0	0	0	1	1	0	1	1	0	0	0	1	1	0	1	1
1	0	0	0.0062	1	1	0	1	1	0	0	0	1	1	0	1	1
2	0	0	0.0082	1	1	0	1	1	0	0	0	1	1	0	1	1
3	0	0	0.005	1	1	0	1	1	0	0	0	1	1	0	1	1
4	0.0001	0	0.0048	1	1	0	1	1	0	0	0	1	1	0	1	1
5	0	0	0.0046	1	1	0	1	1	0	0	0	1	1	0	1	1

6 Conclusions

This study is on the fault diagnosis of the valve oil system in the Kori nuclear power plant. The valves are operated by oil pressure. Aging conditions and faults of the valve inside can be analyzed and diagnosed using pressure information. In this study, fuzzy logic and neural networks are applied to design the diagnosis system. Unloading time and the area ratio of the oil pressure signal are used for inputs of the fuzzy diagnosis system. And the pressure signal of the valves is implemented for inputs of the neural networks. Status of the valve corresponding to structure change is immediately detected by the designed diagnosis algorithm.

Acknowledgments. This work was supported by the Second-stage of the Brain Korea 21 project in 2006 and this work was financially supported by Pusan National University in the program, Post-Doc. 2006.

References

- [1] Jang, S.H., Baek, W.P.: Nuclear Safety. Cheong Moon Gak (1998) 1-37
- [2] Development of Digital Control System of Turbine Governor of Nuclear Power Plant. Technical Report, Korea Electric Power Research Institute (2003)
- [3] Governor Control Logic Drawing of the Kori Nuclear Power Plant No. 2. Technical Report, Electric Power Research Institute (2004)
- [4] Kim, J.P., Goo, B.M., Park, J.G.: Development of Water Quality Monitoring and Diagnosis System of the Nuclear Power Station. Power Research **1** (1994) 303-315
- [5] Nelson, W.R.: REACTOR: An Expert System for Diagnosis and treatment of nuclear reactor accidents. Proceedings AAAI-82m (1982) 296-301
- [6] Nelson, W.R.: Response Trees and Expert System for nuclear reactor operation. Technical Report NUREG/CR-3631, Idaho National Engineering Laboratory, EG & G Idaho (1984)
- [7] Control System of the Turbine Governor. Technical Report, Korea Electric Power Corporation (1995)
- [8] Sub-Contractor’s Instrumentation Book 30 (Electro-Hydraulic Governor) GEC MANUAL **4** (1982)

Stochastic Cellular Neural Network for CDMA Multiuser Detection*

Zhilu Wu, Nan Zhao, Yaqin Zhao, and Guanghui Ren

School of Electronics and Information Technology, Harbin Institute of Technology,
Harbin, Heilongjiang 150001, China
{wuzhilu, yaqinzhao, rgh}@hit.edu.cn

Abstract. A novel method for the multiuser detection in CDMA communication systems based on a stochastic cellular neural network (SCNN) is proposed in this paper. The cellular neural network (CNN) can be used in multiuser detection, but it may get stuck in a local minimum resulting in a bad steady state. The annealing CNN detector has been proposed to avoid local minima; however, the near-far effect resistant performance of it is poor. So, the SCNN detector is proposed here through adding a stochastic term in a CNN. The performance of the proposed SCNN detector is evaluated via computer simulations and compared to that of the conventional detector, the stochastic Hopfield network detector, and the Annealing CNN detector. It is shown that the SCNN detector can avoid local minima and has a much better performance in reducing the near-far effect than these detectors, as well as a superior performance in bit-error rate.

1 Introduction

Code Division Multiple Access (CDMA) has been the subject of extensive research in the field of mobile radio communications. This technique permits a large number of users to communicate simultaneously on the same frequency band. However, this creates multiple access interference (MAI), which makes the conventional detector (CD) of demodulating a spread-spectrum signal in a multiuser environment not reliable and sensitive to near-far effect. For this reason multiuser detection, which can overcome this problem, is a hot topic now for CDMA systems.

The optimal multiuser detector (OMD) [1] proposed by Verdu, is shown to be near-far resistant and has the optimal performance, however, the exponential complexity in the number of users makes it impractical to use in current CDMA systems. Therefore, research efforts have concentrated on the development of suboptimal detectors, which exhibit good near-far effect resistant properties, have low computational complexity and achieve relatively high performance, such as MMSE detector [2], Hopfield neural network (HNN) detector [3], and stochastic HNN (SHN) detector [4-5].

A cellular neural network (CNN) [6] constitutes a class of recurrent and locally coupled arrays of identical dynamical cells, which can be implemented easily by very

* Project Supported by Development Program for Outstanding Young Teachers in Harbin Institute of Technology.

large scale integrated circuits (VLSI) and applied to signal processing problems. The CNN can also be used in multiuser detection, in which the energy function is related to the objective function of the OMD. Though the CNN detector can be implemented with low complexity, it may get stuck in a local minimum resulting in a bad steady state. So the Annealing CNN detector [7] has been proposed which can avoid local minima. However, the performance of the near-far effect resistance in the Annealing CNN detector is poor. In this paper, the stochastic CNN (SCNN) detector is proposed through adding a stochastic term in a CNN. The SCNN detector can also avoid local minima and has a much better performance in reducing the near-far effect than the Annealing CNN detector, as well as a superior performance in bit-error rate (BER).

2 Traditional Detection Methods

2.1 Conventional Detector

Assuming there are K users of a CDMA system in a synchronous single-path channel, the received signal can be expressed as

$$r(t) = \sum_{k=1}^K A_k(t)g_k(t)d_k(t) + n(t) , \tag{1}$$

where $A_k(t)$, $g_k(t)$, and $d_k(t)$ are the amplitude, signature code waveform, and information of the k^{th} user, respectively. $n(t)$ is additive white Gaussian noise (AWGN), with a two-sided power spectral density of $N_0/2$ W/Hz.

The CD described in (1) is a bank of K matched filters, and can be shown in Fig.1.

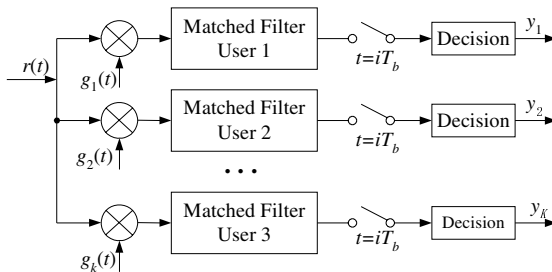


Fig. 1. The conventional detector

In Fig.1, the existence of MAI has a significant impact on the capacity and performance of the CD system because the CD follows a single-user detector strategy. As the number of interfering users increases, the amount of MAI increases.

2.2 Optimal Multiuser Detector

Verdu has shown that the OMD may be achieved by producing an estimate for the information vector transmitted based on the maximization of the logarithm of the likelihood function. The objective function of the OMD [1] is given as

$$b_{opt} = \arg \max \{ 2Y^T Ab - b^T Hb \}, \tag{2}$$

where $b \in \{+1, -1\}$, $Y^T=(y_1, \dots, y_k)$ is the row vector consisting of the sampled outputs of the matched filters, A is the diagonal matrix consisting of the corresponding received amplitudes, and $H=A^T R A$, in which R is a $K \times K$ uniform correlation matrix.

Despite the huge performance and capacity gains over the CD, the OMD is not practical. The exponential complexity in the number of users makes the cost of this detector too high.

3 Cellular Neural Network Based Multiuser Detector

Because of the exponential growth of the computational complexity of the OMD with the number of active users, many suboptimal multiuser detectors have been proposed. Detectors based on the CNN are discussed and the SCNN detector is proposed.

3.1 Cellular Neural Network

A CNN is composed of regular distributed dynamical cells, in which each cell is composed by a linear capacitor, a linear resistance, several linear and non-linear current sources controlled by voltages. And each cell can only communicate with its neighbors directly, which is called local-connection. The local mutual communication among cells makes it very convenient to realize the CNN with VLSI. Additionally, the nice continuous time domain characteristic of the CNN without delay can meet the requirements of real-time digital signal processing.

The dynamical equation describing the statue of the each cell of the one-dimensional CNN is

$$C \frac{dV_i(t)}{dt} = -\frac{1}{R_x} V_i(t) + \sum_{j \in N_r(i)} D_{ij} V_j(t) + \sum_{j \in N_r(i)} B_{ij} u_j(t) + I_i, \tag{3}$$

where $V_j(t)$ is the state of the i^{th} cell, $u_j(t)$ is the input of the j^{th} neighbor cell, I_i is the independent current source of the i^{th} cell, C is the capacitance of a linear capacitor, and R_x is the resistance of a linear resistor. The output function of the i^{th} cell is

$$V_{\gamma_i}(t) = \frac{1}{2} (|V_i(t)+1| - |V_i(t) - 1|) . \tag{4}$$

The energy function of the CNN described in (3) is given as

$$E(t) = -\frac{1}{2} \sum_i \sum_j D_{ij} V_{\gamma_i}(t) V_{\gamma_j}(t) - \sum_i I_i V_{\gamma_i}(t) + \frac{1}{2R_x} \sum_i \sum_j B_{ij} V_{\gamma_i}(t) u_j(t), \tag{5}$$

where $V_{\gamma_i}(t)$ is the output of the i^{th} cell.

It is apparent from (2) that the OMD objective function is very similar to the CNN energy function, and (2) can be rewritten as

$$\begin{aligned}
 b_{opt} &= \arg \max \{ 2Y^T Ab - b^T Hb \} \\
 &= \arg \min \{ -Y^T Ab + 1 / 2b^T Hb \} \\
 &= \arg \min \{ -Y^T Ab + 1 / 2b^T (H - E)b + 1 / 2b^T Eb \} \\
 &= \arg \min \{ -Y^T Ab + 1 / 2b^T (H - E)b \},
 \end{aligned}
 \tag{6}$$

since $b^T E b$ is always a positive number. In (6) $E=A^T A$ is the diagonal matrix containing the energy of each user.

Therefore, the OMD objective function can be directly translated into the energy function of the CNN in (5) with the templates:

$$\begin{aligned}
 D &= -(H - E), \\
 B &= 0, \\
 I &= Y^T A.
 \end{aligned}
 \tag{7}$$

And then we can get the CNN detector.

Once the network of a CNN detector is initialized, it iteratively converges to a possible stable state after which the signs of the network’s outputs are used to calculate the estimates of the transmitted signals.

However this optimization process is localized and therefore the detector will get stuck in a local minimum resulting in a bad steady state. In [7] the authors proposed an Annealing CNN detector which can avoid local minima in the CNN.

3.2 Stochastic Cellular Neural Network

Although the Annealing CNN detector can avoid local minima in the CNN and reduce the BER of CDMA systems, its performance in reducing near-far effect is still poor. In [4] the authors proposed a SHN which can avoid local minima of the HNN by adding a stochastic term in its dynamical equation and we can also use this stochastic method in a CNN to get a global optimum, so the SCNN detector is proposed here.

The distribution of the stochastic term used in the SHN detector is given as:

$$F(x) = \frac{1}{1 + e^{-ax}} .
 \tag{8}$$

In determining the change in $\alpha(k)$, a tradeoff must be made between convergence and BER performance. If $\alpha(k)$ is increased quickly over the iterations, the network converges faster at the expense of performance. If $\alpha(k)$ is increased slowly, the performance is near optimum but this occurs over a large number of iterations.

The output function used in the SCNN is a nonlinear function, defined by

$$G(x) = \frac{1 - e^{-ax}}{1 + e^{-ax}} ,
 \tag{9}$$

where α is a positive constant that controls the slope of the nonlinearity. In particular, when $\alpha \rightarrow \infty$, then $G(\cdot) \rightarrow \text{sign}(\cdot)$.

In (3), if a stochastic term given by (8) is added, the sigmoid output function given by (9) is used, the templates described in (7) are accepted and set $C=1, R_x=2.5$, we can get the iterative function of the SCNN detector, given by

$$\begin{cases} V_i(k+1) = V_{\gamma_i}(k) - \frac{1}{2.5} V_{\gamma_i}(k) + \sum_{j=1}^K D_{ij} V_{\gamma_j}(k) + I_i + v(k), \\ V_{\gamma_i}(k+1) = G(V_i(k+1)), \end{cases} \tag{10}$$

where $v(k)$ the stochastic term, $V_i(k)$ is the state of the i^{th} cell at the k^{th} iteration and $V_{\gamma_i}(k)$ is the output of the i^{th} cell at the k^{th} iteration.

4 Simulation Results

In order to evaluate the performance of the SCNN detector, a CDMA system with three users using it is designed as Fig.2.

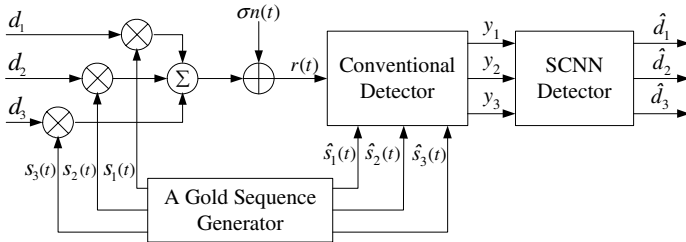


Fig. 2. A CDMA system using SCNN detector

In Fig.2, there are four users in the CDMA system and the PN sequences used are all Gold sequences of length $N=63$. The received signal $r(t)$ is handled in the CD, the outputs of which are fed into the SCNN detector, and then we can get the estimate of the baseband information transmitted of each user.

A variety of simulation experiments are presented comparing the performance of the CD, the SHN detector, the Annealing CNN detector, and the SCNN detector in the system depicted in Fig.2. The BER versus signal-noise ratio (SNR) curves with equal energy of each user, are depicted in Fig.3. It is shown that if the near-far effect is not considered, the SCNN detector achieves much better performance than the CD, the SHN detector, and the Annealing CNN detector. We also deal with the near-far effect problem. The BER curves of the first user are compared when its energy is increasing with the energy of the other two users unchanged, and the results are shown in Fig.4. From the simulation results, we can see that near-far effect resistant performance of the SHN detector and the SCNN detector is much better than the CD and the Annealing CNN detector, with the BER of the SCNN detector much lower than the SHN detector. Therefore, the overall performance of the SCNN detector is much better than the other detectors discussed and it is more suitable as a neural network based multiuser detection scheme in CDMA systems.

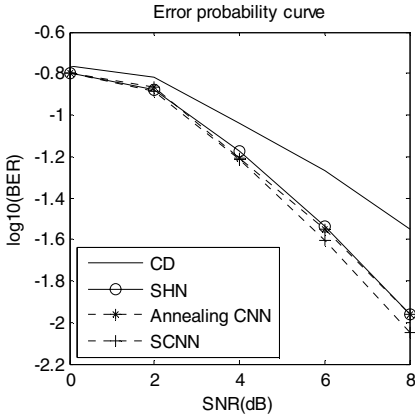


Fig. 3. Bit-Error Rate vs. Signal-Noise Ratio

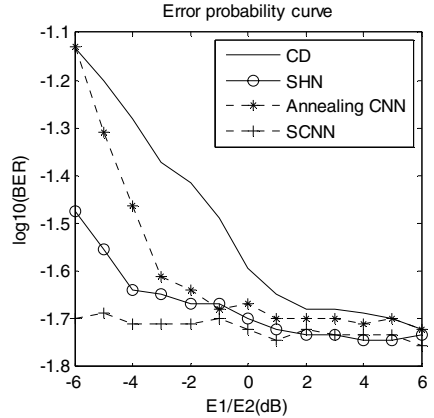


Fig. 4. Bit-Error Rate vs. Near-Far Ratio

5 Conclusions

In this paper, a novel multiuser detection method based on the SCNN for the spread spectrum communication in CDMA systems is proposed. Through adding a stochastic term, the SCNN detector can avoid local minima in the CNN detector and achieve global optimum. The simulation results show that the proposed SCNN detector offers significant performance gains compared to the CD and other recurrent neural network multiuser detectors in reducing bit-error rate and near-far effect while it can be implemented easily using VLSI.

References

- Verdu, S.: Minimum Probability of Error for Asynchronous Gaussian Multiple-Access Channels. *IEEE Trans. Info. Theory* **32** (1986) 85-96
- Xie, Z., Short, R.T., Rushforth, C.K.: A Family of Suboptimum Detectors for Coherent Multi-User Communications. *IEEE JSAC* **8** (1990) 683-690
- Kechriotis, G.I., Manolakos, E.S.: Hopfield Neural Network Implementation of the Optimal CDMA Multiuser Detector. *IEEE Transactions on Neural Networks* **7** (1996) 131-141
- Moodley, N., Mnene, S.H.: Recurrent Network Techniques in Multiuser Detection. *IEEE AFRICON* **1** (2004) 89-94
- Liu, X.D., Wang, X.X., Wu, Z.L., Gu, X.M.: Modified Hopfield Neural Network for CDMA Multiuser Detector. *ISNN 2006* **3** (2006) 88-93
- Chua, L.O., Yang, L.: Cellular Neural Networks: Theory, *IEEE Trans. on CAS* **35** (1988) 1257-1272
- Wang, H.M., Cheng, C.Q., Yu, D.H.: Cellular Neural Network Approach to a Class of Communication Problems. *The 7th Annual Joint Workshops on Modern Electronic Technology and Applications* (2002) 176-182

A New Approach of Blind Channel Identification in Frequency Domain

Chen Caiyun and Li Ronghua

School of Electronic & Information Engineering, South China University of Technology,
Guangzhou 510641, Guangdong, China
acaim@21cn.com, faace@163.com

Abstract. This paper develops a new blind channel identification method in frequency domain. Oversampled signal has the property of spectral redundancy in frequency domain which is corresponding to the cyclostationarity property in time domain. This method exploits the cyclostationarity of oversampled signals to identify possibly non-minimum phase FIR channels. Unlike many existing methods, this method doesn't need EVD or SVD of correlation matrix. Several polynomials are constructed and zeros of channels are identified through seeking for common zeros of those polynomials. It is in the similar spirit of Tong's frequency approach, but this new algorithm is much simpler and computationally more efficient. A sufficient and necessary condition for channel identification is also provided in this paper. This condition is quite similar to Tong's time domain theory but it is derived from a novel point of view.

1 Introduction

Digital communication through multi-path environment is subject to inter-symbol interference (ISI). In mobile communication, ISI will be very severe. To achieve high-speed reliable communication, channel equalization is necessary to combat ISI. Traditional equalizer works based on a training sequence. Training signals waste some band source, especially in time-varying channels where require periodically sending training sequence, or in broadcast network, where training each new-joining sub-machine will occupy much time. In recent years, blind signal processing techniques have received considerable attention [1-6] and have been reaching many application fields including channel equalization. Blind channel equalization only bases on the received data (i.e., without training sequence), so is very attractive.

Since communications channels are likely to be non-minimum phase, the identification problem was naturally addressed using higher-order statistics (HOS) of the channel output [7-9], because second order statistics (SOS) of stationary output don't include the channel's phase information. In 1991, Gardner pointed out that the SOS of cyclostationary series provides both magnitude and phase information of channels [10]. Compared to HOS, estimation of SOS needs fewer observations, thus allows faster signal processing. But Gardner's approach [10] is still based on a

slow-rate training sequence. Also in 1991, Tong et al., show that when channel output is sampled at a rate greater than the symbol rate (oversample), the output is cyclostationary and they give a method rely on the SOS only [11]. Since then, many SOS based blind identification methods have been proposed. These methods usually use SVD or EVD of output matrix for identifying channels parameters; see [12-13]. To reduce the computer complexity of such methods, LXH develop a method using QR factorization [14]. In 1995, Tong et. al., develop a frequency domain approach [15] and give some channel identifiable condition both in time and frequency domain. Tong’s frequency method provides a good insight of spectral redundancy in cyclostationary signals.

Based on the spirit of Tong’s frequency method, this paper introduces a new approach. The new approach presents the desirable properties of being computationally more efficient. A sufficient and necessary condition for this approach is given as well.

The rest of paper is organized as follows: In section 2, the cyclostationarity of oversampled signals is discussed. A new channel identification method is proposed in section 3, together with the conditions for this method to be valid. Section 4 presents the detailed new algorithm. In section 5, the complexity of the new method is analyzed and some ways for simplifying are pointed out. Finally, a conclusion is given in section 6.

2 Cyclostationarity

In PAM communication systems, when the channel is time invariant with finite response length, source signal $s(n)$ and the received signal $x(n)$ can be represented as

$$s(n) = \sum_k s_k \delta(n - kT), \tag{1}$$

$$x(n) = \sum_k s_k h(n - kT) + w(n), \tag{2}$$

where $\{s_k\}$ is the source symbol sequence, $h(\cdot)$ is the discrete channel impulse response, $w(\cdot)$ is the additive noise, $\delta(\cdot)$ is the discrete impulse function. T is the source symbol interval. The sampling interval is normalized to 1.

In the sequel, we adopt the following basic assumptions:

- 1). T is a known integer and $T > 1$, i.e. oversample rate is T .
- 2). $\{s_k\}$ is white with zero mean and unit variance, i.e. $E(s_k s_l^*) = \delta(k - l)$.
- 3). $w(\cdot)$ is zero mean, white and uncorrelated with $\{s_k\}$, and variance is σ^2 .

To observe the cyclostationarity of oversampled signals, define the source autocorrelation function as $r_s(n, m) = E(s(n)s^*(n+m))$. We begin by deriving the autocorrelation function according to (1) and assumption 2):

$$\begin{aligned}
 r_s(n, m) &= \sum_l \sum_k E(s_k s_l^*) \delta(n - lT) \delta(n + m - kT) \\
 &= \delta(m) \sum_l \delta(n - lT).
 \end{aligned}
 \tag{3}$$

Obviously $r_s(n, m)$ is a periodic function of n with period T , so $s(n)$ is a cyclostationary process. From (2), similarly, we have the autocorrelation of $x(\cdot)$ as

$$r_x(n, m) = \sum_l \sum_k h(l) h^*(k) r_s(n - l, m + k - l) + \sigma^2 \delta(m).
 \tag{4}$$

Since $r_x(n, m)$ is linear to $r_s(n, m)$, it is also periodic in n with period T . Owing to the periodicity, for channel identification purpose, we only need to use a set of autocorrelation functions in one period. In the following paper, we only consider the following T functions: $r_x(0, m)$, $r_x(1, m)$, ..., $r_x(T - 1, m)$.

3 Channel Identification

Oversampled channel with oversample rate T can be equivalently considered as T subchannels. Define subchannels $\{h_i(\cdot), i = 0, \dots, T - 1\}$ as $h_i(n) = h(nT + i)$ and correspondingly define the output of each subchannel as $x_i(n) = x(nT + i)$, the additive noise to each subchannel $w_i(n) = w(nT + i)$. Then the input-output relation of each subchannel can be expressed as

$$x_i(n) = \sum_k s_k h_i(n - k) + w_i(n), \quad i = 1, \dots, T - 1.
 \tag{5}$$

Define the co-correlation function of two subchannels' output as

$$r_{i,j}(m) = E(x_i(n) x_j^*(n + m)), \quad i, j = 1, \dots, T - 1.
 \tag{6}$$

From (2) and assumption 2), we further obtain:

$$r_{i,j}(m) = \sum_k h_i(n - k) h_j^*(n + m - k) + \sigma^2 \delta(m) \delta(i - j), \quad i, j = 0, \dots, T - 1.
 \tag{7}$$

To observe the zeros of $r_{i,j}(m)$, do Z-transform and get:

$$r_{i,j}(z) = \sum_{m=-L}^L r_{i,j}(m) z^{-m} = H_i(z) H_j^*(1/z^*) + \sigma^2 \delta(i - j).
 \tag{8}$$

For notational convenience, let

$$R_{i,j}(z) = r_{i,j}(z) - \sigma^2 \delta(i - j).
 \tag{9}$$

Then we have

$$R_{i,j}(z) = H_i(z) H_j^*(1/z^*).
 \tag{10}$$

Theorem. $H_i(z)$ can be identified from $\{R_{i,j}(z)\}$ if and only if $\{H_i(z), i = 0, \dots, T-1\}$ share no common zeros. Moreover, if $H_i(z)$ is identifiable, the zeros of $H_i(z)$ are the common zeros of the $\{R_{i,j}(z), j = 0, \dots, T-1\}$, i.e.

$$Z(H_i(z)) = \bigcap_j Z(R_{i,j}(z)), \tag{11}$$

Where $Z(H_i(z))$ stands for a set of zeros of the i^{th} subchannel.

Comments. Zeros of $H_i(z)$ are reciprocals of the zeros of $H_i^*(1/z^*)$, i.e., if $z_0 \in Z(H_i(z))$, then $(1/z_0^*) \in Z(H_i^*(1/z^*))$, so, $\{H_i(z), i = 0, \dots, T-1\}$ sharing no common zeros means that $\{H_i^*(1/z^*), i = 0, \dots, T-1\}$ sharing no common zeros.

Proof. Firstly we proof the sufficient part. Suppose $\{H_i(z), i = 0, \dots, T-1\}$ share no common zeros, i.e., $\{H_i^*(1/z^*), i = 0, \dots, T-1\}$ share no common zeros, from (10)

$$\bigcap_j Z(R_{i,j}(z)) = Z(H_i(z)) \bigcup \bigcap_j H_j^*(1/z^*). \tag{12}$$

Since $\bigcap_j H_j^*(1/z^*) = \emptyset$, we get $\bigcap_j Z(R_{i,j}(z)) = Z(H_i(z))$;

The necessity part: Suppose all subchannels share a common zero z_0 , i.e. $(1/z_0^*) \in \bigcap_j H_j^*(1/z^*)$, z_0 becomes an unidentifiable zero. The reason is when we substitute z_0 by $(1/z_0^*)$ in $H_i(z)$, we would obtain the same $Z(H_i(z)) \bigcup \bigcap_j H_j^*(1/z^*)$, then between z_0 and $(1/z_0^*)$ we can't determine which one belongs to $H_i(z)$.

4 The Algorithm

- 1) Estimate the co-correlation of each two subchannels according to (6);
- 2) Estimate the noise power σ^2 (if not known) and calculate the coefficients of $R_{i,j}(z)$ by:

$$\eta_{i,j}(m) = \begin{cases} r_{i,j}(0) - \sigma^2 & m = 0, i = j, \\ r_{i,j}(m) & \text{others.} \end{cases} \tag{13}$$

- 3) Estimate zeros of $\{H_i(z), i = 0, \dots, T-1\}$ according to (11).
- 4) Using the estimated zeros to rebuild subchannels.
- 5) From (7), power of i^{th} subchannel is given by

$$p(h_i) = \sum_k h_i(k)h_i^*(k) = r_{i,i}(0) - \sigma^2. \tag{14}$$

Normalize i^{th} subchannel’s power to $p(h_i)$, so the total channel is identified.

6) According to the estimated channel, construct an inverse filter for recovering original signals.

5 Complexity Analysis

Through the derivation, it is obvious that $T = 2$ is enough for the new algorithm. Even for systems which the oversample rate is higher than 2, it is still possible to use only two subchannels for estimation, in condition that there happened to exist two channels sharing no common zeros. For example, if $H_j(z)$ and $H_k(z)$ share no common zeros, then, for each subchannel $H_i(z)$, we have

$$\begin{aligned} & Z(R_{i,j}(z)) \cap Z(R_{i,k}(z)) \\ &= Z(H_i(z)) \cup (Z(H_j^*(1/z^*)) \cap Z(H_k^*(1/z^*))) \\ &= Z(H_i(z)). \end{aligned} \tag{15}$$

The methods can be further simplified through the following observation: If $H_i(z)$ has $k(0 < k \leq l)$ identified identical zeros, then, if $R_{i,j}(z)$ owns just $l + k$ identical zeros, the other l zeros of $R_{i,j}(z)$ are the zeros of $H_j^*(1/z^*)$ directly.

6 Conclusion

In this work, a new approach of blind channel identification is proposed. By exploiting the cyclostationarity of oversampled signals, this method identifies non-minimum phase FIR channels without resorting to HOS. This work is based on the similar spirit of Tong’s frequency approach but different. The new algorithm is simpler and computationally more efficient. A sufficient and necessary condition for the new approach is given in this paper.

Acknowledgement

This work is supported by the National Natural Science Foundation of China for Excellent Youth (Grant 60325310), the Guangdong Province Science Foundation for Program of Research Team (Grant 04205783), the Natural Science Fund of Guangdong Province, China (Grant 05103553), the Specialized Prophasic Basic Research Projects of Ministry of Science and Technology, China (Grant 2005CCA04100), Key Program of National Natural Science Foundation of China (Grant U0635001).

References

1. Xie, S.L., He, Z.S., Fu, Y.L.: A Note on Stone's Conjecture of Blind Separation. *Neural Computation* **17** (2005) 245-319
2. Li, Y.Q., Amari, S., Cichocki, A.: Probability Estimation for Recoverability Analysis of Blind Source Separation Based on Sparse Representation. *IEEE Trans. Inform. Theory* **52** (2006) 3139-3152
3. He, Z.S., Xie, S.L., Fu, Y.L.: A Novel Framework of Multi-channel Acoustic Echo Cancellation. *Progress in Natural Science* **16** (2006) 983-987
4. Li, Y.Q., Amari, S., Cichocki, A., Ho, D.W.C., Xie, S.L.: Underdetermined Blind Source Separation Based on Sparse Representation. *IEEE Trans. Signal Processing* **54** (2006) 423-437
5. He, Z.S., Xie, S.L., Fu, Y.L.: Sparse Representation and Blind Source Separation of Ill-posed Mixtures. *Science in China Series F-Information Sciences* **49** (2006) 639-652
6. Li, Y.Q., Cichocki, A., Amari, S.: Blind Estimation of Channel Parameters and Source Components for EEG Signals: A Sparse Factorization Approach. *IEEE Trans. Neural Networks* **17** (2006) 419-431
7. Godard, D.N.: Self-recovering Equalization and Carrier Tracking in Two Dimensional Data Communication Systems. *IEEE Trans. Commun. COMM-28* (1980) 1867-1875
8. Shalvi, O., Weinstein, E.: New Criteria for Blind Deconvolution of Nonminimum Phase Systems (Channels). *IEEE Trans. Inform. Theory* **36** (1990) 312-321
9. Xie, S.L., He, Z.S., Gao, Y.: *Adaptive Theory of Signal Processing*. 1st ed. Chinese Science Press, Beijing (2006) 103-129
10. Gardner, W.: A New Method of Channel Identification. *IEEE Trans. Commun.* **39** (1991) 813-817
11. Tong, L., Xu, G., Kailath, T.: Blind Identification and Equalization Based on Second-Order Statistics: A Time Domain Approach. *IEEE Trans. Inform. Theory* **40** (1994) 340-349
12. Moulines, E., Duhamel, P., Cardoso, J., Mayrargue, S.: Subspace Methods for the Blind Identification of Multichannel FIR filters. *IEEE Trans. Signal Processing* **43** (1995) 516-525
13. Roberto, L., Soura, D.: Blind Channel Equalization with Colored Sources Based on Second-Order Statistics: A Linear Prediction Approach. *IEEE Trans. Signal Processing* **49** (2006) 2050-2059
14. Li, X., Fan, H.: QR Factorization Based Blind Channel Identification and Equalization with Second-Order Statistics. *IEEE Trans. Signal Processing* **48** (2000) 60-69
15. Long, L., Xu, G., Hassibi, B., Kailath, T.: Blind Channel Identification Based on Second-Order Statistics: A Frequency-Domain Approach. *IEEE Trans. Inform. Theory* **41**(1995) 329-334

Soft Decision-Directed Square Contour Algorithm for Blind Equalization

Liu Shunlan^{1,2} and Dai Mingzeng¹

¹ School of Communication Engineering, Hangzhou Dianzi University, 310018, Hangzhou, Zhejiang, China

² National Laboratory of Information Control Technology for Communication System, 314033, Jiaxing, Zhejiang, China
liushunlan@hdu.edu.cn, daiming0916@126.com

Abstract. The recently introduced square contour algorithm (SCA) combines the benefits of the generalized Sato algorithm (GSA) and the constant modulus algorithm (CMA). It implicitly updates phase and is less likely to converge to incorrect solutions. But the SCA has relatively large residual error after the algorithm reaches its steady state for high-order constellations. A new blind equalization algorithm is proposed based on concurrent square contour algorithm (SCA) and soft decision-directed (SDD) adaptation. Like the SCA, the proposed concurrent SCA and SDD algorithm includes phase recovery and offers good convergence characteristics. Simulation results demonstrate that the proposed SCA+SDD algorithm offers practical alternatives to blind equalization of high-order QAM channels and provides significant equalization improvement over the CMA, GSA and SCA.

1 Introduction

Blind equalization improves system bandwidth efficiency by avoiding the use of a training sequence. Furthermore, for certain communication systems, training is infeasible and a blind equalizer provides a practical means for combating the detrimental effects of channel dispersion in such systems. Although various blind equalization techniques exist, the best known algorithms are the constant modulus algorithm (CMA) [1], [2], [3] and the generalized Sato algorithm (GSA) [4]. The GSA is simple to implement, but does not always give reliable initial convergence [5]. For the CMA convergence is more consistent, but an arbitrary phase rotation is imparted. The recently introduced square contour algorithm (SCA) [6], [7] combines the benefits of the GSA and CMA and it implicitly updates phase during convergence and is less likely to converge to incorrect solutions. But the SCA has relatively large residual error after the algorithm reaches its steady state for high-order constellations, such as 16- and 64-QAM signals.

The soft decision-directed (SDD) equalization algorithm [8], [9] or blind clustering algorithm, has been proposed. In this algorithm the equalizer output is modeled by M Gaussian clusters, of which mean is the symbols of the constellation set. The proposed concurrent SCA and SDD (SCA+SDD) algorithm may

be viewed as operating a SCA equalizer and a last-stage bootstrap maximum a posteriori probability (MAP) [10] blind equalizer concurrently. Like SCA, the proposed SCA+SDD algorithm also combines the benefits of the CMA and GSA. It includes phase recovery and offers good convergence characteristics. The proposed SCA+SDD algorithm has better steady-state equalization performance and faster convergence speed than the CMA, GSA and SCA. Simulations and analysis demonstrate the good performance of the proposed algorithm.

A brief overview of blind equalization model is given in Section 2. The development of the proposed SCA+SDD algorithm is provided in Section 3. In Section 4, we present some simulation results and Section 5 concludes this paper.

2 Blind Equalization Model

Consider an equalizer implemented as a linear transversal filter with $2K+1$ taps. The received signal at symbol-spaced sample k is given by

$$r(k) = \sum_{i=1}^m h_i s(k-i) + n(k), \quad (1)$$

where m is the length of the channel impulse response (CIR), h_i is the complex-valued channel tap weights, $n(k)$ is a complex-valued Gaussian white noise, and the complex-valued symbol sequence is assumed to be independently identically distributed and takes the value from the M -QAM symbol set defined by

$$S = \{s_{ql} = (2q - Q - 1) + j(2l - Q - 1), 1 \leq q, l \leq Q\}, \quad (2)$$

with $Q = \sqrt{M}$.

The output of the equalizer for the k th symbol is

$$y(k) = \sum_{l=-K}^K w_l(k) r(k-l) = \mathbf{w}^T(k) \mathbf{r}(k), \quad (3)$$

where $\mathbf{w}(k) = [w_{-K}, \dots, w_0(k), \dots, w_K(k)]^T$ is the equalizer weight vector at time k , and $\mathbf{r}(k) = [r(k+K), \dots, r(k-K)]^T$ is the input vector of the symbol-spaced samples at the k th symbol time.

An equalization algorithm attempts to minimize its particular cost function with respect to $\mathbf{w}(k)$ of the weight vector. Using a stochastic gradient algorithm to seek the minimum, we have the general form for the equalizer weight update

$$\mathbf{w}(k+1) = \mathbf{w}(k) - \mu e(k) \mathbf{r}^*(k), \quad (4)$$

where $*$ denotes complex conjugation, $e(k)$ is an error function arising from the instantaneous gradient with respect to of the particular equalization cost function, and μ is the adaptation step size.

3 The Concurrent SCA and Soft Decision-Directed Equalization

3.1 Square Contour Algorithm(SCA)

The cost function [6] for the SCA equalizer is given as

$$J_{\text{SCA}} = E \left\{ \left((|y_r(k) + y_i(k)| + |y_r(k) - y_i(k)|)^p - R_{\text{SCA}}^p \right)^2 \right\}, \tag{5}$$

where p is a positive integer and R_{SCA}^p is a constellation-dependent constant

$$R_{\text{SCA}}^p = \frac{E \left\{ \left(|s_r(k) + s_i(k)| + |s_r(k) - s_i(k)| \right)^p G \right\}}{E\{G\}}, \tag{6}$$

where

$$G = (|s_r(k) + s_i(k)| + |s_r(k) - s_i(k)|)^{p-1} \times (\text{sgn}[s_r(k) + s_i(k)](1 + j) + \text{sgn}[s_r(k) - s_i(k)](1 - j))s^*(k). \tag{7}$$

Its weight update error function is

$$e_{\text{SCA}}(k) = \left((|y_r(k) + y_i(k)| + |y_r(k) - y_i(k)|)^p - R_{\text{SCA}}^p \right) \times \left(|y_r(k) + y_i(k)| + |y_r(k) - y_i(k)| \right)^{p-1} \times \left(\text{sgn}[y_r(k) + y_i(k)](1 + j) + \text{sgn}[y_r(k) - y_i(k)](1 - j) \right). \tag{8}$$

We focus on the case $p = 2$ as a special case which yields good performance with reasonable complexity of implementation. For $p = 2$, (8) can be written as

$$e_{\text{SCA}}(k) = 4y_r(k)(4y_r^2(k) - R_{\text{SCA}}^2)X(k) + j4y_i(k)(4y_i^2(k) - R_{\text{SCA}}^2)Y(k). \tag{9}$$

Note that

$$X(k) = \frac{\text{sgn}(y_r(k))}{2} [\text{sgn}(y_r(k) + y_i(k)) + \text{sgn}(y_r(k) - y_i(k))] = \begin{cases} 1, & |y_r(k)| \geq |y_i(k)| \\ 0, & |y_r(k)| < |y_i(k)| \end{cases}, \tag{10}$$

$$Y(k) = \frac{\text{sgn}(y_r(k))}{2} [\text{sgn}(y_r(k) + y_i(k)) - \text{sgn}(y_r(k) - y_i(k))] = \begin{cases} 1, & |y_r(k)| \leq |y_i(k)| \\ 0, & |y_r(k)| > |y_i(k)| \end{cases}. \tag{11}$$

For the SCA cost function (5), the zero-error contour is a square contour, and the SCA minimizes dispersion of the equalizer output from this contour. With

the square zero-error contour, the SCA combines the reliable convergence of the CMA and the phase recovery characteristics of the GSA. As is the case for most other blind equalization algorithms, the SCA only considers the statistics of the overall signal constellation and ignores detailed knowledge of the constellation points so that the SCA has relatively large residual error after the algorithm reaches its steady state for high-order constellations, such as 16- and 64-QAM signals [7].

3.2 The Concurrent SCA and Soft Decision-Directed Algorithm

The proposed blind equalization algorithm operates a SCA equalizer and a SDD equalizer concurrently. Specifically, let

$$\mathbf{w} = \mathbf{w}_s + \mathbf{w}_d, \tag{12}$$

where \mathbf{w}_s is the weight vector of the SCA equalizer and \mathbf{w}_d is the weight vector of the SDD equalizer. At sample k , given

$$y(k) = \mathbf{w}_s^T(k)\mathbf{r}(k) + \mathbf{w}_d^T(k)\mathbf{r}(k). \tag{13}$$

When the equalizer weights have been correctly chosen, the equalizer output can be expressed as [9]

$$y(k) \approx s(k) + v(k), \tag{14}$$

where $v(k) = v_r(k) + jv_i(k)$ is approximately a Gaussian white noise. Thus, when the equalization is accomplished, the equalizer output can be modeled approximately by M Gaussian clusters with the cluster means being s_{ql} for $1 \leq q, l \leq Q$. All the clusters have an approximate covariance:

$$\begin{bmatrix} E[v_r^2(k)] & E[v_r(k)v_i(k)] \\ E[v_i(k)v_r(k)] & E[v_i^2(k)] \end{bmatrix} \approx \begin{bmatrix} \rho & 0 \\ 0 & \rho \end{bmatrix}. \tag{15}$$

Under the above conditions, the a *posteriori* probability density function (*p.d.f.*) of $y(k)$ is

$$p(\mathbf{w}, y(k)) \approx \sum_{q=1}^Q \sum_{l=1}^Q \frac{p_{ql}}{2\pi\rho} \exp \left[-\frac{|y(k) - s_{ql}|^2}{2\rho} \right], \tag{16}$$

where p_{ql} are the a *priori* probability of s_{ql} . We can divide the complex plane into $M/4$ regular regions. Each region $S_{i,l}$ contains four symbol points:

$$S_{i,l} = \{s_{pq}, p = 2i - 1, 2i, q = 2l - 1, 2l\}. \tag{17}$$

If the equalizer output $y(k)$ is within the region $S_{i,l}$, a local approximately to a *posteriori p.d.f.* of $y(k)$ is

$$\hat{p}(\mathbf{w}, y(k)) \approx \sum_{p=2i-1}^{2i} \sum_{q=2l-1}^{2l} \frac{1}{8\pi\rho} \exp \left[-\frac{|y(k) - s_{pq}|^2}{2\rho} \right], \tag{18}$$

with a *priori* probability $p_{pq}=1/4$. The purpose of the SCA sub-equalizer is to open the eye, so that the local *p.d.f.* expression (18) is approximately valid. The SDD sub-equalizer is designed to maximize the log of the local *a posteriori p.d.f.* criterion

$$\bar{J}_{\text{LMAP}}(\mathbf{w}) = E [J_{\text{LMAP}}(\mathbf{w}, y(k))], \tag{19}$$

where

$$J_{\text{LMAP}}(\mathbf{w}, y(k)) = \rho \log(\hat{p}(\mathbf{w}, y(k))). \tag{20}$$

Using a stochastic gradient algorithm to seek the minimum, we have the SDD equalizer weight update

$$\mathbf{w}_d(k+1) = \mathbf{w}_d(k) + \mu_d \frac{\partial J_{\text{LMAP}}(\mathbf{w}, y(k))}{\partial \mathbf{w}_d}, \tag{21}$$

with

$$\frac{\partial J_{\text{LMAP}}(\mathbf{w}, y(k))}{\partial \mathbf{w}_d} = \frac{\sum_{p=2i-1}^{2i} \sum_{q=2l-1}^{2l} \exp\left[-\frac{|y(k)-s_{pq}|^2}{2\rho}\right] (s_{pq} - y(k))}{\sum_{p=2i-1}^{2i} \sum_{q=2l-1}^{2l} \exp\left[-\frac{|y(k)-s_{pq}|^2}{2\rho}\right]} \mathbf{r}^*(k). \tag{22}$$

The proposed SCA+SDD algorithm can be achieved by:

$$\mathbf{w}(k) = \mathbf{w}_s(k) + \mathbf{w}_d(k), \tag{23}$$

$$y(k) = \mathbf{w}^T(k) \mathbf{r}(k), \tag{24}$$

$$\mathbf{w}_s(k+1) = \mathbf{w}_s(k) - \mu_s e_{\text{SCA}} \mathbf{r}^*(k), \tag{25}$$

$$\mathbf{w}_d(k+1) = \mathbf{w}_d(k) + \mu_d \frac{\partial J_{\text{LMAP}}(\mathbf{w}, y(k))}{\partial \mathbf{w}_d}. \tag{26}$$

It is obvious that this SDD scheme corresponds to the last stage of the bootstrap MAP scheme. The SCA sub-equalizer includes phase recovery and offers good convergence characteristics and the SDD sub-equalizer minimizes the residual SCA inter-symbol interference and increases the convergence rate.

4 Simulations

The performance of the proposed SCA+SDD, CMA, GSA and SCA blind equalizers in terms of clustering the signal constellation, suppression of inter symbol interference (ISI), was evaluated in computer simulations. The residual ISI at the output of the equalizer is defined as follows [11]:

$$\text{ISI} = \frac{\sum_i |h(i) * w(i)|^2 - |h(i) * w(i)|_{\max}^2}{|h(i) * w(i)|_{\max}^2}. \tag{27}$$

All simulation experiments described in this section employed the equalizer of transversal filter structure with 11 tap weights and the equalizers were initialized

Table 1. 7-tap channel impulse response

Tap No.	1	2	3	4	5	6	7
Real	-0.005	0.009	-0.024	0.854	-0.218	0.049	-0.016
Imaginary	-0.004	0.030	-0.104	0.520	0.273	-0.074	0.020

Table 2. Algorithm parameter settings in simulations

Example	CMA	GSA	SCA	SCA+SDD		
	μ	μ	μ	μ_s	μ_d	ρ
1	1.5×10^{-5}	1.5×10^{-4}	2×10^{-6}	2×10^{-6}	2×10^{-3}	0.6
2	8×10^{-7}	5×10^{-5}	1.2×10^{-7}	1.2×10^{-7}	5×10^{-4}	0.6

with the central tap weights set to one and all others set to zero. A typical voice-band communication channel [12] was assumed in all the simulations, with complex impulse response (CIR) shown in Table 1. Table 2 lists the algorithm parameters used in the simulation for the four blind equalizers. Choosing proper values of the parameters were chosen experientially.

Example 1. In this example, 16-QAM symbols were transmitted through the 7-tap channel, whose CIR is listed in Table 1. The noise power was adjusted such that it gave rise to a channel SNR of 30dB. 15,000 symbols were taken to estimate the channel, of which last 5000 are shown in Fig. 1 which shows the signal constellations of four blind equalizers after convergence. The results confirm that the signal constellation of the CMA has an obvious phase rotation which is recovered and corrected by the GSA, SCA and SCA+SDD and the

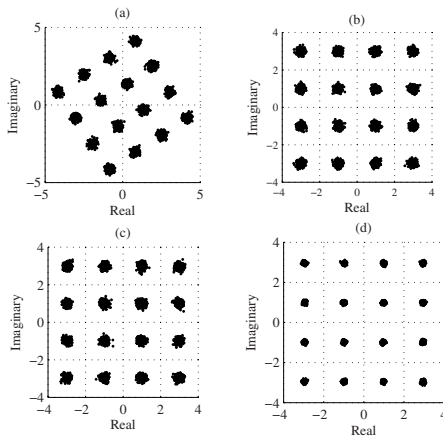


Fig. 1. Equalizer output signal constellations after convergence for example 1. (a) CMA. (b) GSA. (c) SCA. (d) SCA+SDD.

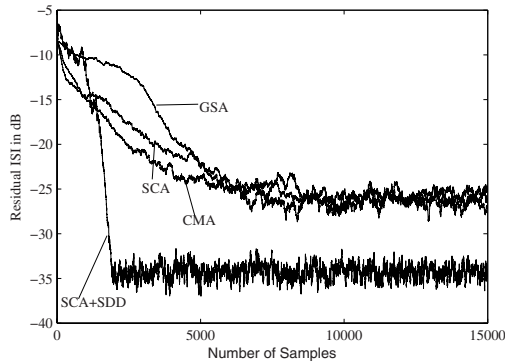


Fig. 2. Comparison of convergence performance in terms of residual ISI for example 1

clustering performance of the SCA+SDD is better than the CMA, GSA and SCA. The comparison of the convergence performance in terms of residual ISI is shown in Fig. 2. The plots clearly reveal that the SCA+SDD has better steady-state performance and faster convergence speed than the CMA, GSA and SCA.

Example 2. In this example, the transmitted data symbols were 64-QAM and the channel given in Table 1 was used with the SNR of 30dB. The equalizer output signal constellations after convergence are shown in Fig. 3. The learning curves of the four blind equalizers in terms of residual (ISI) are shown in Fig. 4. It can be seen that for this example the SCA+SDD algorithm also has faster convergence and better steady-state equalization performance. Fig. 5 compares the performance of the SCA+SDD with different values of the parameter, $\rho = 0.8$ and $\rho = 0.3$. The smaller value of ρ gives somewhat slightly smaller value of

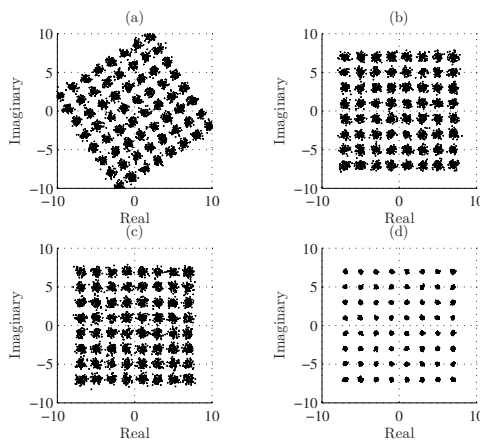


Fig. 3. Equalizer output signal constellations after convergence for example 2. (a) CMA. (b) GSA. (c) SCA. (d) SCA+SDD.

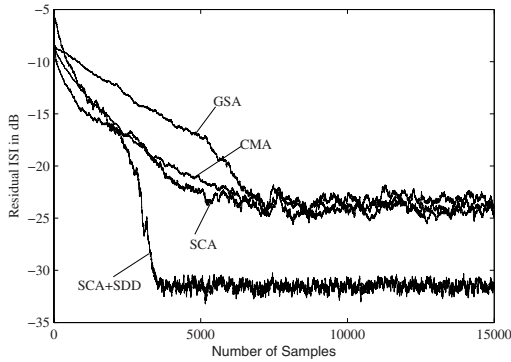


Fig. 4. Comparison of convergence performance in terms of residual ISI for example 2

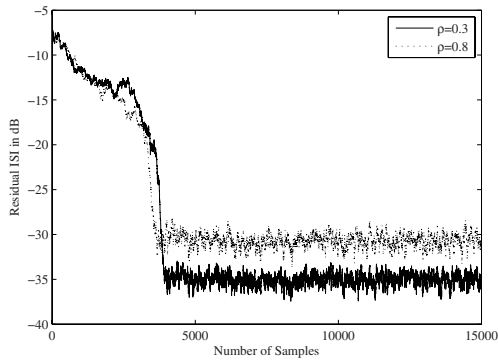


Fig. 5. Performance for the SCA+SDD with different values of the parameter, $\rho = 0.8$ and $\rho = 0.3$

residual ISI. But if a too small ρ is used, the algorithm attempts to impose a very tight control in the size of clusters and may fail to do so. On the other hand, if the value of ρ is too large, a desired degree of separation may not be achieved. As the minimum distance between the two neighboring symbol points is 2, typically ρ is chosen to be less than one.

5 Conclusion

In this paper, a new blind equalization algorithm has been proposed based on operating a SCA equalizer and a SDD equalizer concurrently. The proposed SCA+SDD algorithm combines the benefits of the CMA and GSA so that it includes phase recovery and offers good convergence characteristics. The proposed SCA+SDD algorithm has better steady-state equalization performance and faster convergence speed than the CMA, GSA and SCA. This new blind algorithm, offers practical alternatives to blind equalization of high-order QAM

channels and provides significant equalization improvement over the CMA, GSA and SCA. Additional effort is needed to extend it to more practical situation, including longer channels and equalizers.

References

1. Godard, D.: Self-Recovering Equalization and Carrier Tracking in Two-Dimensional Data Communication Systems. *IEEE Trans. Commun.* **28** (1980) 1867-1875
2. Treichler, J.R., Agree, B.G.: A New Approach to Mutipath Correction of Constant Modulus Signals. *IEEE Trans. Acoust. Speech Signal Process* **31** (1983) 459-472
3. Johnson, C. (ed.): Blind Equalization using the CM Criterion: A Review. *Proc. IEEE* **86** (1998) 1927-1950
4. Sato, Y.: A Method of Self-Recovering Equalization for Multilevel Amplitude Modulation Systems. *IEEE Trans. Commun.* **23** (1975) 679-682
5. Yang, J. (ed.): The Multimodulus Blind Equalization and Its Generalized Algorithm. *IEEE J. Sel. Areas Common.* **20** (2002) 997-1051
6. Thaiupathump, T., Kassam, S.A.: Square Contour Algorithm: A New Algorithm for Blind Equalization and Carrier Phase Recovery. *Proceedings of the 37th Asilomar Conference on Signals, Systems and Computers* (2003) 647-651
7. Thaiypathump, T., He, L., Kassam, S.A.: Square Contour Algorithm for Blind Equalization of QAM Signals. *Signal Processing* **86** (2006) 3357-3370
8. Karaoguz, J., Ardalan, S.H.: A Soft Decision-Decided Blind Equalization Algorithm Applied to Equalization of Mobile Communication Channels. *ICC92. IEEE, Chicago* **3** (1992) 1272-1276
9. Chen, S.: Low Complexity Concurrent Constant Modulus Algorithm and Soft Decision Directed Scheme for Blind Equalization. *IEE Proc.-Vis. Image Signal Process* **150** (2003) 312-320
10. Chen, S. (ed.): Multi-Stage Blind Clustering Equalizer. *IEEE Trans. Commun.* **43** (1995) 701-705
11. Shalvi, O., Weninstein, E.: New Criteria for Blind Deconvolution of Nonminimum Phase Systems (channels). *IEEE Trans IT* **36** (1990) 312-321
12. Weerackody, V., Kassam, S.A.: Dual-Mode Type Algorithms for Blind Equalization. *IEEE Trans. Commun.* **42** (1994) 22-28

Call Admission Control Using Neural Network in Wireless Multimedia Networks

Yufeng Ma and Shenguang Gong

Department of Weapon Engineering,
Naval University of Engineering, Wuhan 430033, P.R. China
050904@163.com

Abstract. Scarcity of the spectrum resource and mobility of users make Quality-of-Service (QoS) provision a critical issue in wireless multimedia networks. This paper uses neural network as call admission controller to perform call admission decision. A performance measurement is formed as a weighted linear function of new call and handoff call blocking probabilities of each service class. Simulation compares the neural network with complete sharing policy. Simulation results show that neural network has a better performance in terms of average blocking criterion.

1 Introduction

With the development in wireless communication and network technologies, multimedia applications and services are widely used in mobile cellular networks in recent years. The next generation wireless networks are expected to support multimedia services such as voice, data and video, which request different bandwidths and quality of service (QoS) requirements. How to guarantee the QoS of multi-class services is the main issue for network designers. Mobility of users, scarcity of the spectrum resource and channel fading make QoS provision more challenging task in wireless networks compared with the provisioning problem existing in wireline networks.

Call admission control (CAC) is very important technology in wireless resource management, and it is one of the key mechanisms in guarantying the QoS. It can be defined as the procedure of deciding whether or not to accept a new connection. In wireless networks, one important parameter of the QoS is call blocking probability (CBP), which indicates the likelihood of the new connection being denied. The other important parameter is call dropping probability (CDP), which expresses the likelihood of the existing connection being denied during handoff process due to insufficient resource in target cell. From the user's point of view, having a call abruptly terminated in the duration of the connection is more annoying than being blocked occasionally on a new call attempt. It is acceptance to give higher priority to handoff call. Reduction of CBP and CDP are conflicting requirements, and optimization of both is extremely complex. To guarantee an acceptable CDP is one of the main goals of QoS provisioning in wireless networks.

Imprecision, dynamic change and burst are features of the flow of wireless traffic. The fluctuation in the flow of traffic can not be predicted in advance, conventional call admission control policy can not adapt itself to the change of traffic. But intelligent control has lots of advantages to cope with complexity and uncertainty. Intelligent techniques include fuzzy logic systems, neural networks, genetic algorithm and expert systems. Without any math model developed in advance, these intelligent techniques can deal with uncertain and dynamic systems, and they can solve complex problems in communications and networks in most cases. Recently, intelligent techniques have been applied to call admission control [1-5]. Neural network is used in ATM networks in [3,4]. Fuzzy neural network is used for call admission control in wireless network in [5], but neural network is used to adjust the parameter and membership function of fuzzy controller. In this paper, we use neural network as call admission controller to perform call admission decision in wireless networks.

The remainder of this paper is organized as follows. Section 2 describes system model and goal. Section 3 gives the structure of neural network and learning algorithm. Section 4 runs simulation to compare the neural network with the complete sharing scheme. Then, it discusses the simulation results. Finally, section 5 gives conclusion of the paper.

2 System Description

2.1 System Model

In our model, we consider a cellular network with a limited number of bandwidths or channels and total channel capacity of a cell is C . There are K classes of call services. Different classes of calls may have different requirement of QoS. The arrival rates of new calls and handoff calls of class i , $i=1, 2, \dots, K$, are assumed to form a Poisson process with mean λ_{ni} and λ_{hi} , respectively. The channel holding times of new calls of class i are assumed to follow an exponential distribution with mean $1/\mu_{ni}$. The cell residence time, i.e., the length of time that a user stays in a cell during a visit, is assumed to follow an exponential distribution with mean $1/\mu_{hi}$ and μ_{hi} denotes the call handoff rate. b_i denotes the number of channels required to accommodate the class i calls.

2.2 The Goal

Let CBP_i denote the new call blocking probability for class- i users. Let CDP_i denote the handoff blocking probability for class- i users. We define a weighted linear function of CBP_i and CDP_i as average blocking criterion.

$$P = \sum_{i=1}^K \{\omega_{ni} CBP_i + \omega_{hi} CDP_i\}, \quad (1)$$

where ω_{ni} and ω_{hi} represent the weighting factor of new call and handoff of class i . $\omega_{hi} > \omega_{ni}$, because handoff failure is considered less desirable to user compared with new call attempt failure.

Let CDP_{id} denote an upper bound for the handoff blocking probability for class- i users. QoS requirement is defined as follows:

$$CDP_i \leq CDP_{id}. \tag{2}$$

The goal is to minimize average blocking criterion defined in (1), with the constraints defined in (2).

3 Neural Network Call Admission Controller

3.1 Neural Network Based Call Admission Control

We define system state of a cell as a vector

$$x = \{x_{n1}, x_{n2}, \dots, x_{nK}, x_{h1}, x_{h2}, \dots, x_{hK}\}, \text{ for } i = 1, 2, \dots, K, \tag{3}$$

where x_{ni} and x_{hi} denotes the number of new calls and handoff calls of class- i in the cell, respectively. The state space is given by

$$X = \left\{ x : x_{ni} \geq 0, x_{hi} \geq 0, \sum_{i=1}^K b_i(x_{ni} + x_{hi}) \leq C \right\}. \tag{4}$$

An action is defined as

$$a = \{a_{n1}, a_{n2}, \dots, a_{nK}, a_{h1}, a_{h2}, \dots, a_{hK}\}, \quad a_{ni}, a_{hi} \in \{0, 1\}. \tag{5}$$

If $a_{ni}=1$, it stands for the acceptance for a new call arrival in class- i . In other words, if the next call request is a new call and it is from class- i , it will be accepted. If $a_{ni}=0$, it stands for the rejection for a new call arrival in class- i .

If $a_{hi}=1$, it stands for the acceptance for a handoff call arrival in class- i . In other words, if the next call request is a handoff call and it is from class- i , it will be accepted. If $a_{hi}=0$, it stands for the rejection for a handoff call arrival in class- i .

The action space is defined by

$$A = \{a : a_{ni} = 0 \text{ if } x + e_{ni} \notin X; \quad a_{hi} = 0 \text{ if } x + e_{hi} \notin X\}, \tag{6}$$

where e_{ni} and e_{hi} are vectors of zeros, except for an one in the i -th position. The system state and action can be represented as

$$f : X \rightarrow A = f(X). \quad (7)$$

In other words, the call admission policy can be interpreted as a mapping. The functional mapping divides the $2K$ -dimensional state space into two regions: the accept region and the reject region.

We use BP neural network as call admission controller. The neural network structure for call admission control is shown in Fig.1. It includes input layer, hidden layer and output layer. The inputs are the numbers of new calls and handoff calls of class- i ($i=1,2,\dots,K$). The output is call admission decision. We choose training data set based on upper limit (UL) policy that has a threshold for a class- i call originating in a cell. An incoming handoff call of any class is accepted only if there are enough available channels for the call. During the learning phase of neural network, the resulting QoS will be measured and compared to the target QoS. If the QoS is still hold, the output a_{ni} or a_{hi} is 1, i.e., the call will be accepted. Otherwise, the request is rejected. After the learning phase, the neural network can be used in the recalled mode to perform call admission control.

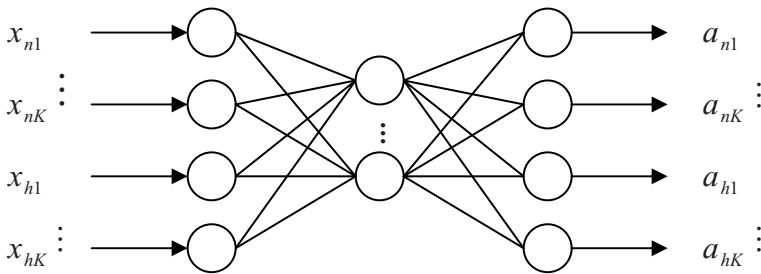


Fig. 1. Structure of neural network

3.2 Learning Algorithm

For each training data set, starting at the input nodes, neural network can compute to obtain the actual outputs of call admission control decision, and CDP_i can be measured. CDP_i will be compared to the target QoS. In order to meet the QoS requirement, we define the error function as

$$E(n) = \frac{1}{2n} \sum_{i=1}^n (CDP_{i,target} - CDP_i)^2. \quad (8)$$

On the other hand, from the output node, back propagation operation is used to compute back propagation error for hidden nodes. Then, weight vector parameters will be updated. When $E(n)$ is small enough, the learning process will stop and weight vector will be stored.

4 Simulation

4.1 Simulation Parameters

In order to evaluate the performance of neural network, we implement and simulate complete sharing scheme for comparison. We let $C=50, K=2, b_1=1, b_2=2$. We assume that the handoff call arrival rate is proportional to the new call arrival rate by $\lambda_{hi} = \alpha\lambda_{ni}$ for every class ($i=1,2$). We set $\alpha = 0.5, \lambda_{n1} = 2\lambda_{n2}, \lambda_{h1} = 2\lambda_{h2}$. $1/\mu_{ni}$ and $1/\mu_{hi}$ is assumed to follow exponential distribution with mean 200 seconds and 100 seconds, respectively. We choose weighting factor $\omega_{n1}=1, \omega_{n2}=2, \omega_{h1}=5, \omega_{h2}=10$. We set thresholds for CDP_1 and CDP_2 as 0.01 and 0.02, respectively. UL is set to (13,5). We choose number of hidden nodes is 9 in neural network.

4.2 Simulation Results

The measures obtained through the simulation are CBP and CDP of every service class and average blocking criterion. We simulate when new call arrival rate changes from 0.08 calls per second to 0.15 calls per second. The measures are plotted as a function of the new call arrival rate.

Simulation curves of the CBP_1 and CBP_2 of the NN and CS schemes are shown in Fig.2. We can notice that the values of the CBP_1 increase as the traffic load increases for two schemes. The CBP_1 of the CS scheme are lower than that of NN. The curves of the CDP_1, CDP_2 are shown in Fig.3. The CDP_1 of CS scheme are higher than NN. The CDP_1, CDP_2 of CS scheme are far higher than thresholds for CDP_1 and CDP_2 respectively. The CS scheme can not meet the requirement of QoS. The CDP_1, CDP_2 of NN are lower than their thresholds.

Simulation curves of average blocking criterion are shown in Fig.4. We can see from the figure, the average blocking criterion of NN is lower than CS scheme. It indicates that the NN scheme has a better performance. It can adapt to changes in the network load.

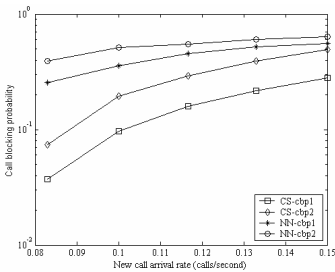


Fig. 2. Call blocking probability

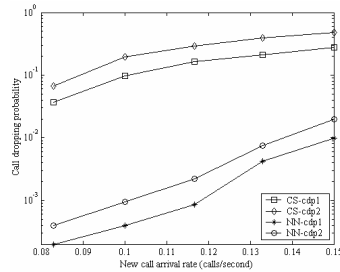


Fig. 3. Call dropping probability

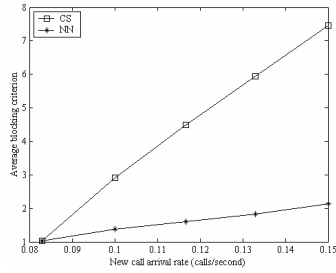


Fig. 4. Average blocking criterion

5 Conclusion

In this paper, we use neural network as call admission controller to perform call admission decision in wireless multimedia networks. Simulation results show the NN has a better performance. It outperforms the complete sharing scheme.

References

1. Ko, Y.-C., Park, S.-C., Chun, C.-Y., Lee, H.-W., Cho, C.-H.: An Adaptive QoS Provisioning Distributed Call Admission Control Using Fuzzy Logic Control. *IEEE ICC* (2001) 356–360
2. Xiao, Y., Chen, C.L.P., Wang, Y.: A Near Optimal Call Admission Control with Genetic Algorithm for Multimedia Services in Wireless/Mobile Networks. *IEEE NAECON* (2000) 787-792
3. Pi, Y.M., Liu, Z.M.: Kohonen Neural Network Based Admission Control in ATM Telecommunication Network. *IEEE ICCT* (1996) 905-908
4. Zhang, L., Liu, Z. M.: A Novel Neural Estimator for Call Admission Control and Buffer Design in ATM Network. *IEEE ISCAS* (1998) 514-517
5. Lo, K.-R., Chang, C.-J., Shung, C.B.: A Neural Fuzzy Resource Manager for Hierarchical Cellular Systems Supporting Multimedia Services. *IEEE Trans. Veh. Technol.* **52** (2003) 1196–1206

A Neural Network Solution on Data Least Square Algorithm and Its Application for Channel Equalization

Jun-Seok Lim

Dept. of Electronics Eng., Sejong University,
Kunja, Kwangjin, 98,143-747, Seoul, Korea
jslim@sejong.ac.kr

Abstract. Using the neural network model for oriented principal component analysis (OPCA), we propose a solution to the data least squares (DLS) problem, in which the error is assumed to lie in the data matrix only. In this paper, We applied this neural network model to channel equalization. Simulations show that DLS outperforms ordinary least square in channel equalization problems.

1 Introduction

Linear least squares (LS) problems involve finding “good” approximate solutions to a set of independent, but inconsistent, linear equations

$$\mathbf{A} \mathbf{x} = \mathbf{b}, \quad (1)$$

where \mathbf{A} is an $m \times n$ complex data matrix, \mathbf{b} is a complex $m \times 1$ observation vector, and \mathbf{x} is a complex $n \times 1$ prediction vector, which is optimally chosen to minimize some kind of squared error measure [1]. It is usually assumed that the underlying noiseless data satisfy (1) with equality. Different classes of LS problems can be defined in terms of the type of perturbation necessary to achieve equality in the system of equations described by (1). For example, in the ordinary least squares (OLS) problem, the error (or perturbation) is assumed to lie in \mathbf{b} [2].

$$\mathbf{A}\mathbf{x}_{OLS} = (\mathbf{b} + \mathbf{r}), \quad (2)$$

where \mathbf{r} is the residual error vector that corresponds to a perturbation in \mathbf{b} . The OLS solution vector \mathbf{x}_{OLS} is chosen so that the Euclidean (or Frobenius) norm of \mathbf{r} is minimized. It is implicitly assumed in the OLS problem that \mathbf{A} is completely errorless, and therefore the columns of \mathbf{A} are not perturbed in the solution [1]. On the other hand, the total least squares (TLS) problem assumes error in both \mathbf{A} and \mathbf{b} [3].

$$(\mathbf{A} + \mathbf{E})\mathbf{x}_{TLS} = (\mathbf{b} + \mathbf{r}). \quad (3)$$

The TLS solution vector is chosen so that the Euclidean norm of $[\mathbf{E} \ \mathbf{r}]$ is minimal. Another interesting case that is described and solved in this correspondence assumes that errors occur in \mathbf{A} but not in \mathbf{b} . This is called the data least squares (DLS) problem because the error is assumed to lie in the data matrix \mathbf{A} as indicated by

$$(\mathbf{A} + \mathbf{E})\mathbf{x}_{DLS} = \mathbf{b}. \tag{4}$$

DeGroat, et. al. in [2] developed a close form solution to (4) and demonstrated that it outperforms OLS and TLS in case of noisy data matrix. However, the solution is a kind of batch type algorithm.

In this paper, we propose a neural network model for DLS solution with a neural network model for oriented principal component analysis (OPCA). We applied this neural network model to channel equalization. Simulations show that the proposed DLS network outperforms ordinary least square in channel equalization problems.

2 Generalized Total Least Square Problem

Given an unknown system with finite impulse response and assuming that both the input and output are corrupted by the Gaussian white noise, the system should be estimated from the noisy observation of the input and output, as in Fig.1.

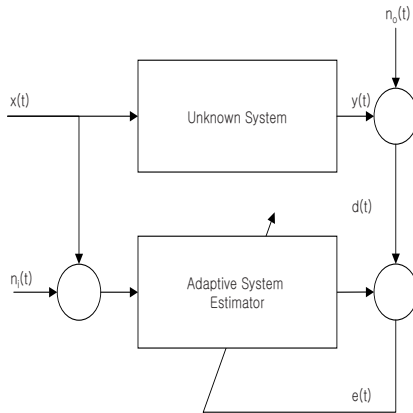


Fig. 1. The model of generalized total least square

The unknown system is described by

$$\mathbf{h} = [h_0, h_1, \dots, h_{N-1}]^H \in C^{N \times 1}, \tag{5}$$

where \mathbf{h} may be time-varying or time-invariant. The desired signal is given by

$$d(n) = \mathbf{x}^H(n)\mathbf{h} + n_o(n), \tag{6}$$

where the output noise $n_o(n)$ is a Gaussian white noise with variance σ_o^2 and independent of the input signal, and the noise free input vector is represented as

$$\mathbf{x}(n) = [x(n), x(n - 1), \dots, x(n - N + 1)]^T. \tag{7}$$

The noisy input vector of the system estimator is given by

$$\tilde{\mathbf{x}}(n) = \mathbf{x}(n) + \mathbf{n}_i(n) \in C^{N \times 1}, \tag{8}$$

where $\mathbf{n}_i(n) = [n_i(n), n_i(n - 1), \dots, n_i(n - N + 1)]^T$ and the input noise $n_i(n)$ is the Gaussian white noise with variance σ_i^2 .

Notice that the input noise may originate from the measured error, interference, quantized noise and so on. Hence, we adopt a more general signal model than the least squares based estimation. Moreover, the augmented data vector is defined as

$$\bar{\mathbf{x}}(n) = [\tilde{\mathbf{x}}^T(n), d(n)]^T \in C^{(N+1) \times 1}. \tag{9}$$

The correlation matrix of the augmented data vector has the following structure

$$\bar{\mathbf{R}} = \begin{bmatrix} \tilde{\mathbf{R}} & \mathbf{p} \\ \mathbf{p}^H & c \end{bmatrix}, \tag{10}$$

where $\mathbf{p} = E\{\tilde{\mathbf{x}}(n)d^*(n)\}$ and $c = E\{d(n)d^*(n)\}$, $\tilde{\mathbf{R}} = E\{\tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^H(n)\} = \mathbf{R} + \sigma_i^2\mathbf{I}$, $\mathbf{R} = E\{\mathbf{x}(n)\mathbf{x}^H(n)\}$. We can further establish that $\mathbf{p} = \mathbf{R}^H\mathbf{h}$ and $c = \mathbf{h}^H\tilde{\mathbf{R}}\mathbf{h} + \sigma_o^2$.

The constrained Rayleigh quotient is defined as

$$J(\mathbf{w}) = \frac{[\mathbf{w}^T, -1]\bar{\mathbf{R}}[\mathbf{w}^T, -1]^H}{[\mathbf{w}^T, -1]\bar{\mathbf{D}}[\mathbf{w}^T, -1]^H}, \tag{11}$$

where $\bar{\mathbf{D}} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \gamma \end{bmatrix}$ with $\gamma = \frac{\sigma_o^2}{\sigma_i^2}$ [3]. The generalized total least square solution is obtained by solving

$$\min_{\mathbf{w}} J(\mathbf{w}). \tag{12}$$

DLS is a special case in (11) with $\gamma=0$ [3].

3 Oriented Principal Component Analysis (OPCA)

In this section we extend the standard principal component analysis problem by introducing OPCA [4] which corresponds to the generalized eigenvalue problem of two random signals and bears the same relationship to generalized eigenvalue decomposition (GED) as PCA bears to ordinary eigenvalue decomposition (ED).

More precisely, the goal is to find the direction vector w that maximizes the signal-to-signal ratio

$$J_{OPC} = \frac{E\{(\tilde{w}^H x_1)^2\}}{E\{(\tilde{w}^H x_2)^2\}} = \frac{\tilde{w}^H R_1 \tilde{w}}{\tilde{w}^H R_2 \tilde{w}}, \tag{13}$$

where $R_1 = E\{x_1 x_1^H\}$ and $R_2 = E\{x_2 x_2^H\}$. We assume that R_2 is strictly positive definite, hence nonsingular. Quite often $\{x_{1k}\}$ and $\{x_{2k}\}$ are stationary stochastic processes, where $R_1 = E\{x_{1k} x_{1k}^H\}$ and $R_2 = E\{x_{2k} x_{2k}^H\}$ and OPCA is still defined by (13). As usual there is little difference between random vectors and stationary random processes, and hence the term OPCA is used for both cases interchangeably.

The optimal solution to (13) will be called the principal oriented component of the pair (x_1, x_2) . The adjective "oriented" is justified by the fact that the principal component of x_1 is now steered by the distribution of x_2 : it will be oriented toward the directions where v has minimum energy while trying to maximize the projection energy of x_1 . J_{opc} is nothing but the generalized Rayleigh quotient for the matrix pencil (R_1, R_2) , so the principal oriented component is the principal generalized eigenvector of the symmetric generalized eigenvalue problem [5].

$$R_1 \tilde{w} = \lambda R_2 \tilde{w}. \tag{14}$$

4 Neural Network Model for Oriented Principal Component (OPC) Extraction

We initially focus on the extraction of the first component. The maximum value of J_{opc} in (13) is the principal generalized eigenvalue λ_1 . Therefore, the function

$$V(\tilde{w}) = \frac{1}{2}(\lambda_1 - J_{OPC}(\tilde{w})) \tag{15}$$

is such that $V(\tilde{w}) > 0$, and $V(\tilde{w}) = 0$ only for $\tilde{w} = e_1$, so V may serve as a Lyapunov energy function for a system to be proposed. The proper gradient descent algorithm would be

$$\frac{d\tilde{w}}{dt} = -\nabla V = \frac{1}{\tilde{w}^H R_2 \tilde{w}} \left(R_1 \tilde{w} - \frac{\tilde{w}^H R_1 \tilde{w}}{\tilde{w}^H R_2 \tilde{w}} R_2 \tilde{w} \right) \tag{16}$$

with the globally asymptotically stable fixed point $\tilde{w} = e_1$.

In fact, even the simpler equation

$$\frac{d\tilde{w}}{dt} = \left(R_1 \tilde{w} - \frac{\tilde{w}^H R_1 \tilde{w}}{\tilde{w}^H R_2 \tilde{w}} R_2 \tilde{w} \right), \tag{17}$$

is stable since

$$\frac{dV}{dt} = \frac{d\tilde{w}^H}{dt} \nabla V = -\frac{1}{\tilde{w}^H R_2 \tilde{w}} \left\| R_1 \tilde{w} - \frac{\tilde{w}^H R_1 \tilde{w}}{\tilde{w}^H R_2 \tilde{w}} R_2 \tilde{w} \right\|^2 \leq 0, \tag{18}$$

and again the point $\tilde{\mathbf{w}} = \mathbf{e}_1$ is the globally asymptotically stable attractor. From (18), we can indirectly conclude that the proposed algorithm converges asymptotically.

5 Neural Network for Data Least Square (DLS) Solution

We can apply the neural network based method in section 4 to solution of DLS. If we modify (11) and (12), the object function for DLS becomes

$$\tilde{J}(\mathbf{w}) = \frac{\tilde{\mathbf{w}}^H \bar{\mathbf{D}} \tilde{\mathbf{w}}}{\tilde{\mathbf{w}}^H \bar{\mathbf{R}} \tilde{\mathbf{w}}} = \frac{[\mathbf{w}^H, -1] \bar{\mathbf{D}} [\mathbf{w}^T, -1]^T}{[\mathbf{w}^H, -1] \bar{\mathbf{R}} [\mathbf{w}^T, -1]^T}. \tag{19}$$

The DSL solution can be derive as (20). Applying the recursive algorithm in section 3 for the maximization of (20) yields

$$\max_{\tilde{\mathbf{w}}} \tilde{J}(\tilde{\mathbf{w}}), \text{ and then } \mathbf{w} = \tilde{\mathbf{w}}(1 : N) / (-\tilde{\mathbf{w}}(N + 1)), \tag{20}$$

where $\tilde{\mathbf{w}}(1 : N)$ is a vector with the elements from the 1-st to the N-th, and $\tilde{\mathbf{w}}(N + 1)$ is the (N+1)-th element in $\tilde{\mathbf{w}}$. Applying the OPCA to (19) yields

$$\Delta \tilde{\mathbf{w}} = \frac{1}{(\tilde{\mathbf{w}}^H \bar{\mathbf{R}} \tilde{\mathbf{w}})^2} ((\tilde{\mathbf{w}}^H \bar{\mathbf{R}} \tilde{\mathbf{w}}) \bar{\mathbf{D}} \tilde{\mathbf{w}} - (\tilde{\mathbf{w}}^H \bar{\mathbf{D}} \tilde{\mathbf{w}}) \bar{\mathbf{R}} \tilde{\mathbf{w}}). \tag{21}$$

$$\begin{aligned} \tilde{\mathbf{w}}(n) &= \tilde{\mathbf{w}}(n - 1) + \beta((\tilde{\mathbf{w}}^H(n - 1) \bar{\mathbf{R}}(n) \tilde{\mathbf{w}}(n - 1)) \bar{\mathbf{D}} \tilde{\mathbf{w}}(n - 1) \\ &\quad - (\mathbf{w}^H(n - 1) \mathbf{w}(n - 1)) \bar{\mathbf{R}}(n - 1) \tilde{\mathbf{w}}(n - 1)), \end{aligned} \tag{22}$$

where $\bar{\mathbf{R}}(n) = \lambda_f \bar{\mathbf{R}}(n - 1) + \bar{\mathbf{x}}(n) \bar{\mathbf{x}}^H(n)$ and λ_f is a forgetting factor. The algorithm is summarized in table 1.

Table 1. OPCA based Data Least Square (NN-DLS) Algorithm

<ol style="list-style-type: none"> 1. Initialize $\lambda_f, \beta, \bar{\mathbf{x}}(0) = [\mathbf{x}^T(0), d(0)]$, $\tilde{\mathbf{w}}(0) = [\mathbf{w}^T(0), -1]$ with the $\mathbf{w}(0) \in \mathbb{C}^{N \times 1}$ to a random vector 2. Fill the matrix $\mathbf{Q}(0) \in \mathbb{C}^{N \times N}$ with small random values 3. Initialize scalar variables $C(0)$ to zero <p>For $j > 0$</p> <ol style="list-style-type: none"> 4. Compute $z(j) = \tilde{\mathbf{w}}^H(j - 1) \bar{\mathbf{x}}(j)$ 5. Update the weight vector as $\tilde{\mathbf{w}}(j) = \tilde{\mathbf{w}}(j - 1) + \beta (z^2(j) \bar{\mathbf{D}} \tilde{\mathbf{w}}(j - 1) - (\mathbf{w}^H(j - 1) \mathbf{w}(j - 1)) z(j) \bar{\mathbf{x}}(j))$ 6. Normalize the weight vector 7. $\mathbf{w}(j) = \tilde{\mathbf{w}}(1 : n - 1) / (-\tilde{\mathbf{w}}(n + 1))$ <p>loop</p>
--

If we cancel the estimation of autocorrelation matrix, we can obtain a simpler update equation as follows.

$$\tilde{\mathbf{w}}(n) = \tilde{\mathbf{w}}(n-1) + \beta (z^2(n)\bar{\mathbf{D}}\tilde{\mathbf{w}}(n-1) - (\tilde{\mathbf{w}}^H(n-1)\tilde{\mathbf{w}}(n-1))z(n)\bar{\mathbf{x}}(n)), \quad (23)$$

where $z(n) = \tilde{\mathbf{w}}^H(n-1)\bar{\mathbf{x}}(n)$. By this simplification, the algorithm needs $7N+9$ of multiplications in each iteration.

6 A Channel Equalization Application

In this section, we demonstrate the usefulness of the DLS neural network model by comparing it with the optimal method and OLS methods in a channel equalization problem. The channel equalization problem is graphically described by the block diagram in Fig. 2. Basically, the solution vector, $\mathbf{w} = [w_1, w_2, \dots, w_p]^T$ represents an FIR approximate inverse filter to the channel characteristic $H(z)$. The output of the inverse (equalization) filter can be written in matrix form using the output of the channel as input to the finite impulse response (FIR) equalization filter. The output of the equalized channel should be approximately equal to the original input

$$\begin{bmatrix} \tilde{s}_{p-1} \\ \tilde{s}_p \\ \vdots \\ \tilde{s}_{N-1} \end{bmatrix} = \begin{bmatrix} v_{p-1} & \cdots & v_1 & v_0 \\ v_p & \cdots & v_2 & v_1 \\ \vdots & \vdots & \vdots & \vdots \\ v_{N-1} & \cdots & v_{N-p+1} & v_{N-p} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_p \end{bmatrix} \approx \begin{bmatrix} s_{p-1} \\ s_p \\ \vdots \\ s_{N-1} \end{bmatrix}, \quad (24)$$

where p is the FIR filter order; and N is the total number of output samples. In this problem, we assume that the left side in (24) is known without error because the input training signal is assumed to be known without error. It is easy to see that (24) has the form of (4).

For the simulation, a well-known complex nonminimum-phase channel model introduced by Cha and Kassam [6] is used to evaluate the proposed neural

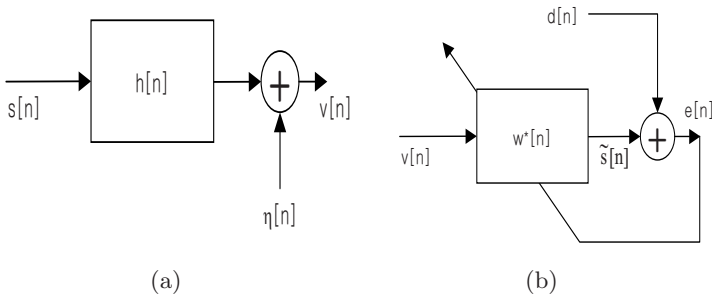


Fig. 2. Transmission and Equalization model: (a) received signal model, (b) equalizer model ($s[n]$: transmitted signal, $h[n]$: channel model, $\eta[n]$: additive noise, $v[n]$: received signal, $d[n]$: training signal)

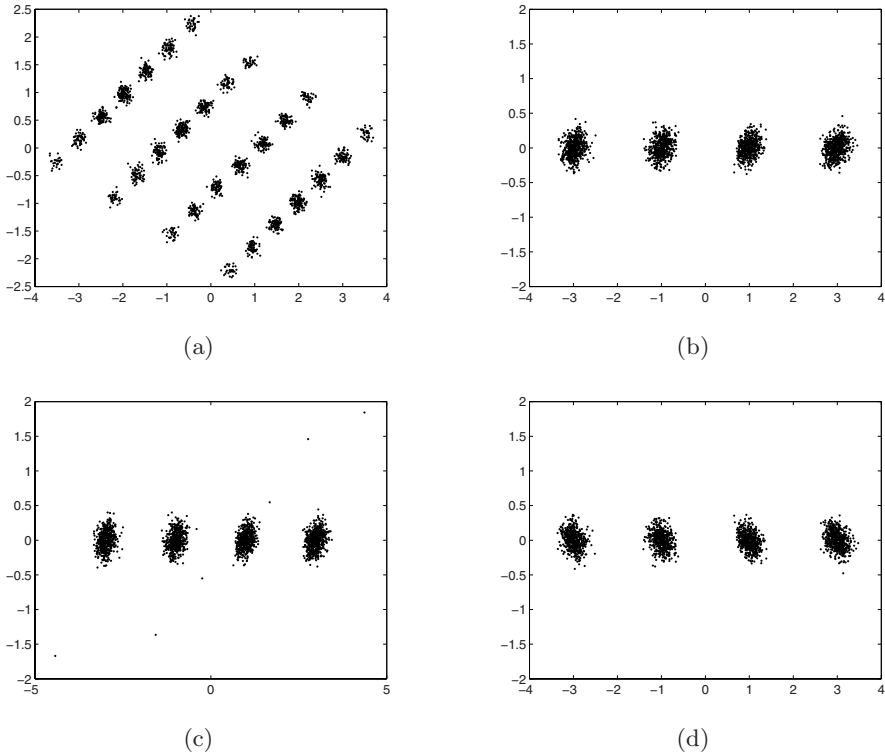


Fig. 3. Equalization results in 4-PAM signaling: (a) constellation of received signals (b) constellation of outputs of optimal equalizer (c) constellation of outputs of RLS equalizer (d) constellation of outputs of the proposed equalizer

network based DLS equalizer performance for 4-PAM signaling. The channel output $v(n)$ (which is also the input of the equalizer) is given by

$$v(n) = (0.34 - j0.27)s(n) + (0.87 + j0.43)s(n-1) + (0.34 - j0.21)s(n-2) + \eta(n) \quad (25)$$

Where $\eta(n)$ is white Gaussian noise. 4-PAM symbol sequence $s(n)$ is passed through the channel and the sequence $s(n)$ is valued from the set $\{\pm 1, \pm 3\}$. All the equalizers, the recursive least square (RLS) based equalizer and the proposed neural network based equalizer, are trained with 1000 data symbols at 15 dB SNR. The RLS is a recursive algorithm for the OLS problem. The order of equalizer was set to 9.

Fig. 3 (a) shows the distribution of the input data of the different equalizers. This figure shows received signals scattered severely due to transmission channel effect. Figures 3 (b), (c) and (d) show the scatter diagrams of the outputs of the three equalizers, optimal one, RLS based one and the proposed one, respectively. From these figures, we can see that the constellation from the proposed algorithm is almost the same as the equalized signals by the optimal equalizer which is

derived from the Wiener solution, while RLS based equalizer produced widely scattered constellation. It leads to the conclusion that the proposed algorithm outperforms the RLS algorithm. Moreover, it works almost the same as optimal equalizer.

7 Conclusion

In this paper, we proposed a neural network model based data least square (DLS) solution. Channel equalization simulations were performed to compare the proposed algorithm with the algorithms in OLS and we found better performance over OLS methods.

References

1. Golub, G.H., Van Loan, C.F.: An Analysis of the Total Least Squares Problem. *SIAM J. Numer. Anal.* **17** (1980) 883- 893
2. DeGroat, R.D., Dowling, E.M.: The Data Least Squares and Channel Equalization. *IEEE Trans. Signal Processing* **41** (1993) 407-411
3. Davila, C.E.: An Efficient Recursive Total Least Squares Algorithm for FIR Adaptive Filtering. *IEEE Trans. Signal Processing* **42** (1994) 268-280
4. Diamantaras, K.I., Kung, S.Y.: *Principal Component Neural Networks: Theory and Applications*. Wiley, New York, N.Y. (1996)
5. Deprettere, E.F.: *SVD and Signal Processing*. Elsevier Science Publishers, New York, (1973) 209-232
6. Cha, I., Kassam, S.A.: Channel Equalization using Adaptive Complex Radial Basis Function Networks. *IEEE J. Sel. Area. Comm.* **13** (1995) 122-131

Multiple Receive-Antennas Aided and RBF-Based Multiuser Detection for STBC Systems

Hongwu Liu and Ji Li

School of Information Science and Technology,
Southwest Jiaotong University,
Chengdu 610031, Sichuan, China
{Hongwu.Liu, JeeLee}@yahoo.com.cn

Abstract. In this paper, authors propose a multiuser detection (MUD) scheme using a radial basis function (RBF) network for a space-time block coding (STBC) systems with multiple receive-antennas. The neuron centers of the RBFs are jointly carried out in space and time domains, which lead to a powerful technique to combat the interference resulting from different sources. Simulations compare this detector with traditional receiver structures such as the minimum mean square error (MMSE) receiver and maximum likelihood detection (MLD). It shows that this RBF-based space-time MUD can be flexibly trade off between the bit error rate (BER) performance, the center initialization-rate and the number of the receive-antennas.

1 Introduction

Without the channel state information (CSI) available at the transmitter, space-time block coding (STBC) can utilize the information in the spatial and time domains simultaneously, bring the diversity gain to the systems[1]-[3]. Multiuser Detection (MUD) for STBC systems has been widely studied in recent years [2][3]. Specifically, the signals emitted from the different transmit antennas of a same STBC transmitter can be viewed as if they were originated from different virtual users so that STBC detection in the case of single user can also be regarded as a MUD problem. Due to multiple users and multiple transmit antennas (virtual users), STBC systems face serious multi-access interference (MAI) in the uplink and limit the capacity. The maximum likelihood detection (MLD) can achieve satisfactory bit error rate (BER) performance but the computation complexity varies exponentially with the number of users. Therefore, the MUD technique is a non-deterministic polynomial (NP)-hard problem, which requires unforeseeable huge computing power in order to find a global optimum solution.

Our proposal of the utilization of a radial basis function (RBF) network as MUD detector for STBC systems is inspired by two facts, in which RBF techniques and modifications are used [4]. In the first, an RBF network is used as an equalizer [5]. The second includes the MUD in code division multiple access

(CDMA) systems [6][7]. However, because of the complexity in terms of neuron centers, research has been extended to suboptimum receivers with more reasonable computational complexity [8]. Furthermore, a RBF-base MUD for the BLAST has been proposed for the point-to-point MIMO communications [9]. However, to the best of our knowledge, using RBF network to MUD in a STBC multiuser system has not been generally investigated and presented. This paper proposes a multiple receive-antennas aided and RBF-based MUD for the STBC systems, which including the RBF-based MUD for single receive-antenna STBC systems as a special case. Specifically, the neuron centers of the RBFs are jointly carried out in space and time domains, considering of all the diversity aspects. Computer simulations are used to verify and compare the performance of this detector against the traditional receiver structures such as the minimum-mean-square error (MMSE) receiver and MLD.

2 System Model

We consider the uplink of the STBC systems, where K active users transmit on the same frequency in the same cell. Without loss of generality, the STBC code g_2 in [1] is selected in this study:

$$g_2 = \begin{bmatrix} x_1 & x_2 \\ -x_2^* & x_1^* \end{bmatrix}. \tag{1}$$

For each user, 2 transmit antennas is employed. At the base-station receiver, there are M receive antennas spacing far away enough with each other to ensure the signals arrived at different antennas undergoing independent fading. The transceiver block diagram is shown in Fig.1. The input symbol-stream of the k th user is firstly STBC coded and form 2 symbol-streams. Then these 2 symbol-streams are transmitted out from 2 antennas, respectively. When only the k th user is active, the signal received at the m th antenna during 2 consecutive symbol periods can be expressed as

$$\begin{bmatrix} r_m(2t-1) \\ r_m^*(2t) \end{bmatrix} = \begin{bmatrix} h^{(m,1,k)} & h^{(m,2,k)} \\ h^{(m,2,k)*} & -h^{(m,1,k)*} \end{bmatrix} \begin{bmatrix} s_k(2t-1) \\ s_k(2t) \end{bmatrix} + \begin{bmatrix} n_m(2t-1) \\ n_m^*(2t) \end{bmatrix}. \tag{2}$$

where $h^{(m,n,k)}$ is the path gain of user k from it's n th transmit antenna to the m th receive antenna at the base-station, $s_k(2t-1)$ and $s_k(2t)$ are the symbols transmitted by the k th user in the t th STBC block, $n_m(2t-1)$ and $n_m(2t)$ are the noise part received at the m th antenna during $(2t-1)$ -th and $2t$ -th symbol periods. The above signal model takes the cost of conjugating half of the CSI and received signals. Moreover, it is impossible to apply (2) for all the generalized STBC designs, such as the following STBC code:

$$S = \begin{bmatrix} x_1 & x_2 & x_3 & 0 \\ -x_2^* & x_1^* & 0 & -x_3 \\ x_3^* & 0 & -x_1^* & -x_2 \\ 0 & -x_3^* & x_2^* & -x_1^* \end{bmatrix}. \tag{3}$$

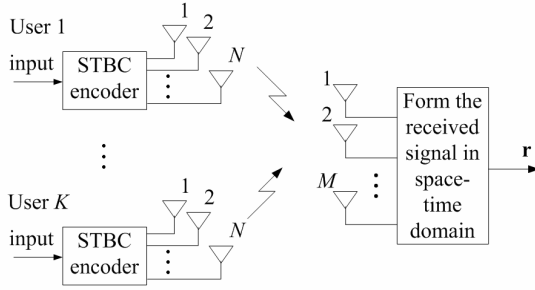


Fig. 1. The transceiver block diagram of an uplink STBC MUD system

Therefore, a redundant form is introduced in the following. For the k th user, the path gain for the m th receive antenna is

$$\mathbf{h}^{(m,k)} = [h^{(m,1,k)} \ h^{(m,2,k)}]^T . \tag{4}$$

The signal symbols per STBC block can be denoted as

$$\mathbf{s}_k = [\text{Re}(s_k(2t - 1)) \ \text{Re}(s_k(2t)) \ \text{Im}(s_k(2t - 1)) \ \text{Im}(s_k(2t))]^T . \tag{5}$$

The signal received at the m th receive antenna is $\mathbf{r}_m = [r_m(2t - 1) \ r_m(2t)]^T$ and noise at the m th receive antenna is $\mathbf{n}_m = [n_m(2t - 1) \ n_m(2t)]^T$. Using STBC encoding, the received signal at the m th receive antenna can be obtained by the following transformation, which is applicable to any STBC design:

$$\mathbf{r}_m(t) = g_2 \mathbf{h}^{(m,k)} + \mathbf{n}_m(t) = \mathbf{H}_{m,k} \mathbf{s}_k + \mathbf{n}_m(t) . \tag{6}$$

where $\mathbf{H}_{m,k}$ for g_2 design is defined as

$$\mathbf{H}_{m,k} = \begin{bmatrix} h^{(m,1,k)} & h^{(m,2,k)} & h^{(m,1,k)} & h^{(m,2,k)} \\ h^{(m,2,k)} & -h^{(m,1,k)} & -h^{(m,2,k)} & h^{(m,1,k)} \end{bmatrix} . \tag{7}$$

When all K users are active, the signal received at the m th antenna is

$$\mathbf{r}_m(t) = \sum_{k=1}^K \mathbf{H}_{m,k} \mathbf{s}_k(t) + \mathbf{n}_m(t) = \mathbf{H}_m \mathbf{s}(t) + \mathbf{n}_m(t) . \tag{8}$$

where $\mathbf{H}_m = [\mathbf{H}_{m,1}, \dots, \mathbf{H}_{m,K}]$, and $\mathbf{s}(t) = [\mathbf{s}_1^T(t), \dots, \mathbf{s}_K^T(t)]^T$ can be taken as a vector containing the $2K$ virtual users' transmitted symbols' real and image parts. The signal out from all M antennas after 2 consecutive symbol periods can be expressed as

$$\mathbf{r}(t) = \mathbf{H} \mathbf{s}(t) + \tilde{\mathbf{n}}(t) . \tag{9}$$

where $\tilde{\mathbf{n}}(t) = [\mathbf{n}_1^T(t), \dots, \mathbf{n}_M^T(t)]^T$, and $\mathbf{H} = [\mathbf{H}_1^T, \dots, \mathbf{H}_M^T]^T$ is a $2M \times 4K$ matrix.

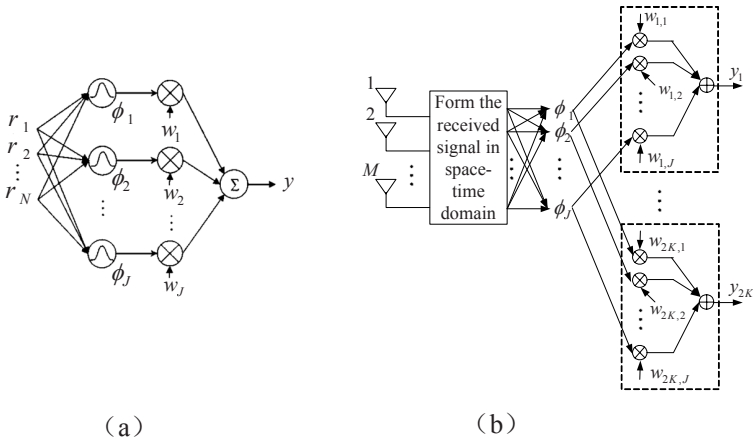


Fig. 2. Structures the RBF network and RBF-based MUD. (a): RBF network. (b): RBF-based MUD.

3 RBF-Based Multiuser Detection

The structure of an RBF network is composed of two layers, as shown in Fig. 2(a), with the output obtained as

$$y = \sum_{j=1}^J w_j \exp \left(-\frac{\|\mathbf{r} - \mathbf{c}^j\|^2}{\sigma_j^2} \right). \tag{10}$$

where \mathbf{c}^j , σ_j^2 , w_j and \mathbf{r} are the j th center of an N -dimensional vector, the degree of the spread of the j th basis function, the weight of the j th basis function, and the input signal of an N -dimensional vector, respectively. Moreover, $\|\cdot\|$ denotes the Euclidean norm of the vector. Input data are connected to the hidden layer in which the number of RBFs is J . In the hidden layer, the nonlinear output of a basis function is obtained by using the spread of a basis function and the Euclidean distance between a center vector and an input vector. The output y of an RBF network is then obtained by linear summation of the nonlinear functions' outputs, which are multiplied by weights, as shown in (10). Generally, centers and weights are trained by a *supervised k-means clustering algorithm* or an *unsupervised clustering algorithm* and a *least mean square (LMS) algorithm*, respectively [5]. In this paper, the Gaussian function in (10) is chosen as RBF.

Fig.2(b) shows the structure of the proposed space-time MUD receiver for STBC g_2 , where an RBF network is applied to the space-time domain. When the number of bits per symbol is p and number of symbols per STBC block is q , the required number of basis functions is $J = (2^p)^{Kq}$, which stands for the complete set of user data [6]. In other words, the center vectors of the RBF network represent the noise free input vectors of receive antennas for possible Kq virtual users data bit combinations. Using the channel response matrix \mathbf{H}

(or the estimated one), the j th center vector of the complex value for the STBC g_2 is initialized as

$$\mathbf{c}^j = \mathbf{H}\mathbf{s}^j = \begin{bmatrix} \sum_{k=1}^{2K} H_{1,k} s_k^j \\ \sum_{k=1}^{2K} H_{2,k} s_k^j \\ \vdots \\ \sum_{k=1}^{2K} H_{2M,k} s_k^j \end{bmatrix}, \quad 1 \leq j \leq J. \tag{11}$$

where $H_{l,k}$ stands for the element of \mathbf{H} in the l th row and k th column, s_k^j is the k th element of \mathbf{s}^j , and \mathbf{s}^j is the j th combination of $2K$ active virtual users' symbols. Thus, each center takes the sum of the users' data and spans in the space-time domain. The data output from the space-time domain is used as the input to the basis functions. The basis function outputs are multiplied by the weight vector, \mathbf{w}_k , assigned to the k th virtual user, which can be defined as

$$\mathbf{w}_k = [w_{k,1}, w_{k,2}, \dots, w_{k,J}]^T, \quad 1 \leq k \leq 2K. \tag{12}$$

where $w_{k,j}$, the weight of the j th basis function for the k th virtual user, is initialized by s_k^j . To apply this to other users, we need only change the weight vector to that particular user's weight vector [7][8]. Then the linear combining is used to obtain the decision variables for the transmitted symbols. As a result, the estimated symbols vector of all users at time t is expressed as

$$\hat{\mathbf{s}}(t) = [\hat{s}_1(t), \hat{s}_2(t), \dots, \hat{s}_{2K}(t)]^T = \begin{bmatrix} \sum_{j=1}^J w_{1,j} \exp\left(-\frac{\|\mathbf{r}(t) - \mathbf{c}^j\|^2}{\sigma_j^2}\right) \\ \sum_{j=1}^J w_{2,j} \exp\left(-\frac{\|\mathbf{r}(t) - \mathbf{c}^j\|^2}{\sigma_j^2}\right) \\ \vdots \\ \sum_{j=1}^J w_{2K,j} \exp\left(-\frac{\|\mathbf{r}(t) - \mathbf{c}^j\|^2}{\sigma_j^2}\right) \end{bmatrix}. \tag{13}$$

During data transmission, *supervised learning* no longer applies and the *decision-directed learning algorithm* is used for adaptation. This adaptation process for an RBF network involves computing the squared Euclidean distance between the centers and the received signal vector, selecting the minimum squared Euclidean distance and moving the corresponding center closer to the received signal vector. That is to say, each distorted set of the $2K$ virtual users' symbols is trained by each RBF center. Therefore, the phase components of each center will estimate the phases of a channel under slow-varying fading channel conditions [7].

4 Simulation Results

The bit-error rate (BER) performances of the proposed RBF-based MUD for STBC systems are investigated by the simulation. In the simulations, the path gain for different transmit-receive antenna-pair of different users are assumed to be independent. For all path gains, the normalized maximum Doppler shift

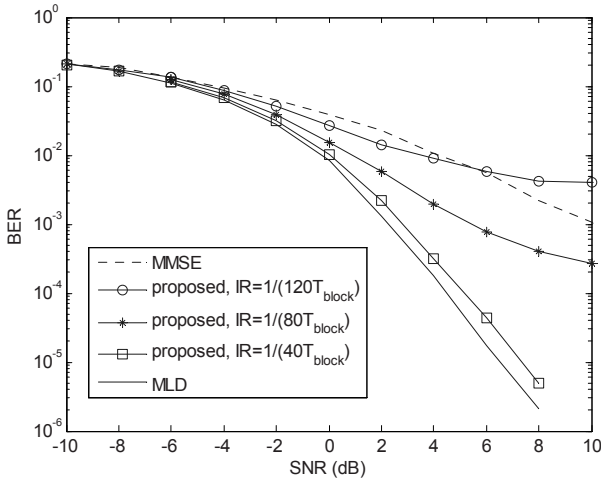


Fig. 3. BER performance of the RBF-based MUD for STBC g_2 , $f_d T_s = 0.001$, BPSK modulation, $M=4$, $K=4$

is $f_d T_s = 0.001$, where f_d and T_s are the maximum Doppler shift and symbol period, respectively. For MMSE and MLD, the AWGN noise variance σ^2 is assumed to be known. For the RBF center initialization, each center spread is set to be $\sigma_j^2 = \sigma^2$. The perfect CSI is assumed for MMSE and MLD detectors, while for the proposed RBF-based MUD, the perfect CSI can only be available at the centers initialization process, as performed in (11). Furthermore, to prevent the RBF-based MUD from becoming a MLD, the initialization of RBF centers can only be carried out at a rate less than $1/(qT_s)$, where qT_s stands for one STBC block period. In the simulation, the initialization-rate (IR) are defined as $IR = 1/(40qT_s)$, $IR = 1/(80qT_s)$ and $IR = 1/(120qT_s)$, respectively. During data transmission, RBF centers are adapted by the *decision-directed learning algorithm* [5] with a learning rate of 0.2.

Fig.3 shows the average BER performance vs. the signal-to-noise-ratio (SNR) when the number of user is 4 ($K=4$), and the number of receive antennas is 4 ($M=4$). As can be seen from Fig.3, when the IR is $1/(40qT_s)$, the RBF-based MUD has very close BER performance as that of MLD. When the IR becomes smaller, the BER performance becomes worse. When the IR is $1/(120qT_s)$, the BER of the RBF-based MUD is worse than the MMSE receiver for $SNR > 5$ dB. However, even when the IR is $1/(80qT_s)$, the RBF-based MUD has better BER performance than MMSE (Noting at this IR, the center initialization process happens per 80 STBC blocks).

Fig.4 shows the average BER performance vs. the SNR when the number of receive antennas is 3 ($M=3$). As can be seen from Fig. 4, the BER of RBF-based MUD with $IR= 1/(40qT_s)$ is close to that of MLD, and the BER of RBF-based MUD increases when the IR becomes slow. Besides, the BER of MMSE receiver is worse than that of the RBF-based MUD with $IR= 1/(120qT_s)$, noting the later

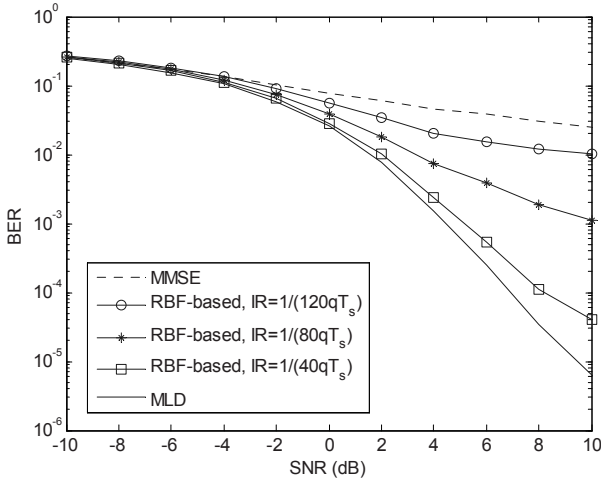


Fig. 4. BER performance of the RBF-based MUD for STBC g_2 , $f_d T_s = 0.001$, BPSK modulation, $M=3$, $K=4$

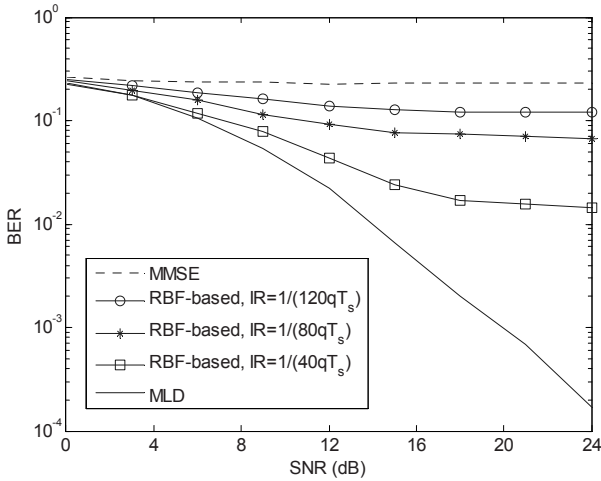


Fig. 5. BER performance of the RBF-based MUD for STBC g_2 , $f_d T_s = 0.001$, BPSK modulation, $M=1$ (single receive-antenna), $K=4$

initializes the centers per 120 STBC blocks. The reason for the BER deterioration of MMSE receiver is that the channel matrix \mathbf{H} in (9) becomes a fat matrix when M is 3 and K is 4.

Fig.5 shows the average BER vs. the SNR when single receive antenna is employed ($M=1$), which is the special case of the proposed MUD. As can be

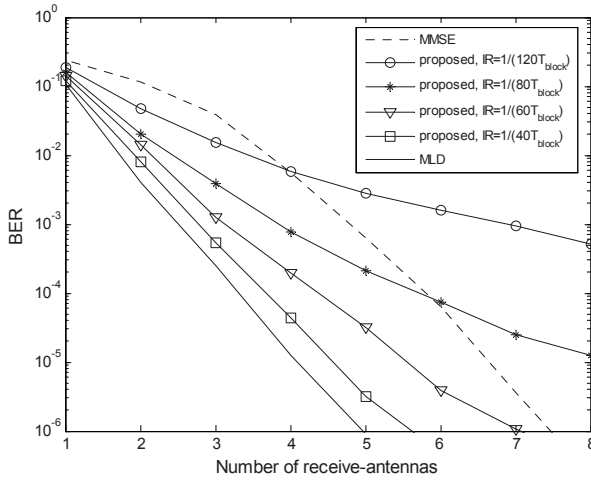


Fig. 6. BER versus the number of receive-antennas, SNR=6dB

seen from Fig. 5, the MMSE receiver can not recovery the signal for serious fat channel matrix effect, while the RBF-based MUD can still achieve a favourable BER performance.

Fig.6 shows the average BER versus the number of the receive antennas when the SNR is 6dB. As can be seen from Fig.6, when the number of receive-antennas becomes smaller, the BER of MMSE receiver becomes worse more rapidly than the proposed scheme. When the number of receive antennas is less than 4 and the IR is larger than or equal to $1/(120qT_s)$, the proposed scheme has smaller BER than the MMSE receiver. On the other hand, as can be seen from Fig.6, the proposed scheme can achieve very close BER as the MLD when the IR and the number of receive antennas are properly chosen.

5 Conclusion

A multiple receive-antennas aided and RBF-based multiuser receiver for the uplink of STBC systems is proposed. The neuron centers of the RBFs are jointly carried out in space and time domains, which lead to a powerful technique to combat the interference resulting from different sources. Simulation results verify that: the proposed RBF-based MUD has similar BER performance as MLD when the centers initialization-rate is high, and has superior BER performance to the MMSE receiver for quite slow centers initialization-rate. The proposed scheme can also flexibly trade off between the BER performance, the centers initialization-rate and the number of the receive-antennas.

References

1. Tarokh, V., Jafarkhani, H., Calderbank, A.R.: Space-time Block Coding for Wireless Communications: Performance Results. *IEEE Journal on Selected Areas in Communications* **17(3)** (1999) 451-460
2. Iraj, S., Lilleberg, J.: EM-based Multiuser Detection and Parallel Interference Cancellation for Space-time Block Coded WCDMA Systems Employing 16-QAM over Multipath Fading Channels. In: *Proc. 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*. Lisboa, Portugal (2002) 688-692
3. Lu, B., Wang, X.: Iterative Receivers for Multiuser Space-time Coding Systems. In: *Proc. IEEE International Conference on Communications*. New Orleans, La, USA (2000) 302-306
4. Mulgrew, B.: Applying Radial Basis Functions, *IEEE Signal Processing Mag.* (1996) 50-65
5. Chen, S., Mulgrew, B., Grant, P.M.: A Clustering Technique for Digital Communications Channel Equalization Using Radial Basis Function Networks. *IEEE Trans. on Neural Networks* **4(4)** (1993) 570-579
6. Cruickshank, D.G.M.: Radial Basis Function Receivers for DS-CDMA. *Electronics Letters* **32(3)** (1996) 188-190
7. KyunByoung, K., Sooyong, C., Changeon, K., Daesik, H.: RBF Multiuser Detector with Channel Estimation Capability in a Synchronous MC-CDMA System. *IEEE Trans. on Neural Networks* **12(6)** (2001) 1536-1539
8. KyunByoung, K., Sooyong, C., Changeon, K., Daesik, H.: A Simplified Multiuser Receiver of DS-CDMA System. In: *Proc. IEEE IJCNN01* (2001) 1977-1982
9. KyunByoung, K., Sooyong, C., Changeon, K., Daesik, H.: A Novel RBF-based Detector for MIMO Systems over Rich-scattering Fading Channels. In: *Proc. IEEE GLOBECOM02* (2002) 434-438

Location Management Cost Estimation for PCS Using Neural Network

Demin Li¹, Jie Zhou², Jiacun Wang³, and Guoliang Wei¹

¹ College of Information Science and Technology, Donghua University, Shanghai
201620, China

deminli@dhu.edu.cn

² School of Sciences, Donghua University, Shanghai 201620, China

zhoujie@dhu.edu.cn

³ Department of Software Engineering, Monmouth University, West Long Branch,
NJ 07762, USA

jwang@monmouth.edu

Abstract. The fluid-flow model is suitable for vehicle traffic on highways but not for pedestrian movements at variable velocities. Considering the change of a mobile's velocity within a short time is limited due to physical restrictions, therefore a mobile user's future velocity is likely to be correlated with its past and current velocity. Since Gauss-Markov model captures the essence of the correlation of a mobile's velocity in time, we propose a Gauss-Markov process based fluid model that it is suitable for both vehicle traffic on highways and pedestrian in street, and presents the total cost estimation of location management for the Gauss-Markov process based fluid model. Considering the importance of memory level in the Gauss-Markov model, we estimate the parameter using neural network. Considering the measurements of updating and paging cost is not consecutive but contain missing observations, we propose the methods of location management total cost estimation with missing measurement for PCS, and utilize the Kalman filter to deduce the steady covariance of the estimation error of total cost per unit time. A numerical example is given to illustrate the use of the proposed approach.

1 Introduction

Location management, which keeps track of the mobile terminals moving from place to place in personal communication system (PCS) networks, is a key issue in PCS networks. There are two basic operations in location management [8]: *location update and paging*. Location update is the process through which system tracks the location of mobile terminals. The mobile terminal reports its up-to-date location information dynamically. A paging area (PA) may include one or more cells. When an incoming call arrives the system searches for the mobile terminal by sending polling signals to cells in the PA. This searching process is referred to as paging. To perform location update or paging will incur a significant amount of cost (e.g., wireless bandwidth and processing power at the

mobile terminals, the base stations, and databases), which should be minimized in the systems.

The main issue of location management scheme is to decrease the waste of wireless bandwidth by reducing the active traffic between the network and the mobile host. Generally, if the size of a location area (LA) increases, the location update load decreases, while the paging load increases due to the increased number of cells for paging. Thus, a tradeoff between location update cost and paging cost is required [13].

Three kinds of dynamic location management schemes have been proposed [10], namely, distance based, movement based, and time based. Under the distance-based scheme, the location update is performed whenever the distance (in term of number of cells) between the current cell of the terminal and the last cell in which the update is performed, where the distance is threshold. Under the movement-based scheme, the location update is performed whenever the mobile terminal completes movements between cells, where the movement is threshold. Under the time-based scheme, the location update is performed every units of time, where the time is threshold. It has been pointed out that the movement-based location update method may be the most practical, because it is effective and can be easily implemented under the framework of current PCS networks [2].

Fluid flow model is commonly used movement-based mobility models in the literatures [9], [14], [15], and is more suitable for users with high mobility, infrequent speed, and direction changes. Under the fluid-flow model, the average location update rate is equal to the average number of crossings of the boundary of region LA per unit time, and the motion speed of a mobile terminal (MT) is *constant or average value* [11], [3], [7], [17]. For pedestrian movements in which mobility is generally confined to a limited geographical area such as residential and business building, the velocity of a MT should be *variable*. The fluid-flow model is suitable for vehicle traffic on highways but not for pedestrian movements at variable velocities. Furthermore, the change of a mobile's velocity within a short time is limited due to physical restrictions. Therefore, a mobile user's future velocity is variable and likely to be correlated with its past and current velocity.

Gauss-Markov model [7] represents a wide range of user mobility patterns, including, as its two extreme cases, the random walk [5] and the constant velocity fluid-flow models [6]. Since Gauss-Markov model captures the essence of the correlation of a mobile's velocity in time, we propose a Gauss-Markov process based fluid model that it is suitable for both vehicle traffic on highways and pedestrian in street.

Consider the measurements of updating and paging cost are not consecutive but contain missing observations, and the missing observations [12] are caused by a variety of reasons, e.g., a certain failure in the measurement, intermittent sensor failures, and accidental loss of some collected data, or some of the data may be jammed or coming from a high noise environment, etc. For these reasons, this paper presents an approach to the total cost estimation of variable velocity

mobile location management for PCS with missing measurements. The key to the cost estimation is the design of an estimator. A Kalman filter of linear uncertain discrete stochastic system with missing measurement is thus developed to serve this purpose.

The rest of this paper is organized as follows. In Section 2, we describe a Gauss-Markov process based fluid model. The model is a linear uncertain discrete-time stochastic system. Sections 3 presents the total cost of location management for the Gauss-Markov process based fluid model. The design of Kalman filter for total cost estimation with missing measurement is studied in 4 Section. Numerical example is presented in Section 5, and Section 6 concludes this paper.

2 System Description

Due to physical restrictions, mobile users often move at one speed one moment and at another speed the next. We postulate that the mobile process is a fluid process at inconstant speed that satisfies Gauss-Markov process. The fluid Model characterizes aggregate movement behavior as the flow of a fluid. Under the fluid-flow model, the direction of an MT’s movement in the LA is uniformly distributed in the range of $(0, 2\pi)$ [15]. Let \bar{v} be average speed (km/hr), and $S(d)$, $L(d)$ be area and perimeter of a LA including d rings respectively. A ring is a circle made up of cells around one cell. The average location update rate is equal to the average number of crossings of the boundary of region LA per unit time, i.e.

$$N_{upd} = \frac{L(d)\bar{v}}{\pi S(d)} \tag{1}$$

1-D discrete version of Gauss-Markov mobility model [7]

$$v_n = \alpha v_{n-1} + (1 - \alpha)\mu + \sigma\sqrt{1 - \alpha^2}w_{n-1} \tag{2}$$

Where v_n is a mobile velocity during the n th period, α is the memory level of the Gauss-Markov mobility model. The parameter is easily determined by training an artificial neural network [16] from the mobile database of PCS in mobile switching center. σ^2 is the variance of v_n , w_n is an uncorrelated Gaussian process with zero mean, unit variance and is independent of v_n . Let $u_n = v_n - \mu$, $\beta = \sigma\sqrt{1 - \alpha^2}$ we obtain the following simple and clear form

$$u_n = \alpha u_{n-1} + \beta w_{n-1} \tag{3}$$

Substituting \bar{v} in (1) by $u_n + \mu$ in (3), we obtain a Gauss-Markov process based fluid model as follows

$$N_{upd} = \frac{L(d)(u_n + \mu)}{\pi S(d)} \tag{4}$$

3 The Total Cost Study for Location Management Per Unit Time

3.1 Cost of Location Update

Let the cost for performing a location update be δ_{upd} , which accounts for the wireless and wire line bandwidth utilization and the computational cost for processing a location update in for crossing a LA. Consider (4), and then the expected cost of location updates per unit time is expressed as

$$c_{upd} = N_{upd}\delta_{upd} = \frac{L(d)(u_n + \mu)}{\pi S(d)}\delta_{upd} \tag{5}$$

3.2 Cost of Paging

Let the cost for polling a cell be δ_{poll} . As was mentioned earlier, all the cells in the PA are paged when an incoming call arrives. Consider the PCS networks with hexagonal cell configurations. The number of cells in the paging process with movement-based location update scheme, denoted by N_{poll} , is upper bounded as follows:

$$N_{poll} \leq (1 + \sum_{i=1}^{d-1} 6i) \tag{6}$$

We use the upper bound in (6) as an approximation to N_{poll} . The expected paging cost per call arrival, denoted by c_{paging} , is given by

$$\begin{aligned} c_{paging} &= N_{poll}\delta_{poll} \\ &= (1 + \sum_{i=1}^{d-1} 6i)\delta_{poll} \\ &= [1 + 3d(d - 1)]\delta_{poll} \end{aligned} \tag{7}$$

The Poisson process is a good model used to describe the arrivals of incoming phone calls per unit time. In the analysis, we assume that the call arrival to each mobile terminal is a Poisson process with rate λ . With these parameters, the expected paging cost per unit time

$$c_{paging} = [1 + 3d(d - 1)]\lambda\delta_{poll} \tag{8}$$

3.3 Total Cost Per Unit Time

Consider $L(d) = 6d$, $S(d) = \frac{3\sqrt{3}d^2}{2}$. To sum up (5), (8), we obtain the clear and simple formulation for the total cost per unit time for movement-based location update scheme as follows:

$$\begin{aligned} c_n &= c_{paging} + c_{poll} \\ &= \frac{4\delta_{upd}}{\sqrt{3}\pi d}(u_n + \mu) + [1 + 3d(d - 1)]\lambda\delta_{poll} \\ &= \frac{4\delta_{upd}}{\sqrt{3}\pi d}u_n + \frac{4\mu}{\sqrt{3}\pi d}\delta_{upd} + [1 + 3d(d - 1)]\lambda\delta_{poll} \end{aligned} \tag{9}$$

Let $\eta = \frac{4\delta_{upd}}{\sqrt{3\pi d}}$, $\kappa = \frac{4\mu}{\sqrt{3\pi d}}\delta_{upd} + [1 + 3d(d - 1)]\lambda\delta_{poll}$ in (9), we obtain

$$c_n = \eta u_n + k \tag{10}$$

4 Filter Design for Linear Uncertain Discrete Time Stochastic System with Missing Measurement

We have obtained the following linear uncertain discrete stochastic system in section 2

$$u_n = \alpha u_{n-1} + \beta w_{n-1} \tag{11}$$

with the measurement equation or total cost per unit time

$$c_n = \eta u_n + k \tag{12}$$

Consider missing measurement, the (10) follow as

$$c_n = \gamma_n \eta u_n + \kappa \tag{13}$$

where γ_n is a Bernoulli distributed white sequence, is assumed to be independent of w_n , u_0 , and is taking values on 0 or 1 with

$$\text{Prob}\{\gamma_n = 1\} = \bar{\gamma} \tag{14}$$

where $\bar{\gamma}$ is a known positive constant. Let

$$\tilde{\gamma}_n = \gamma_n - \bar{\gamma} \tag{15}$$

$$e_n = u_n - \hat{u}_n \tag{16}$$

where $\tilde{\gamma}_n$ is a scalar zero mean stochastic variable with variance $\sigma_{\tilde{\gamma}}^2 = (1 - \bar{\gamma})\bar{\gamma}$, \hat{u}_n stands for the state estimate of u_n , the estimate error of total cost per unit time as follow

$$\begin{aligned} \tilde{c}_n &= c_n - (\bar{\gamma}\eta\hat{u}_n + \kappa) \\ &= (\gamma_n\eta u_n + \kappa) - (\bar{\gamma}\eta\hat{u}_n + \kappa) \\ &= \bar{\gamma}\eta e_n + \tilde{\gamma}_n\eta u_n \end{aligned} \tag{17}$$

The linear filter considered in this note is of the following structure:

$$\hat{u}_{n+1} = g\hat{u}_n + q\tilde{c}_n \tag{18}$$

where g, q are the filter parameters to be scheduled, and subsequently the state error

$$\begin{aligned} e_{n+1} &= u_{n+1} - \hat{u}_{n+1} \\ &= (\alpha - g - q\tilde{\gamma}_n\eta)u_n + (g - \bar{\gamma}q\eta)e_n + \beta w_n \end{aligned} \tag{19}$$

Define

$$\begin{aligned} x_{n+1}^f &= \begin{pmatrix} u_{n+1} \\ e_{n+1} \end{pmatrix} \\ A_n &= \begin{pmatrix} \alpha & 0 \\ \alpha - g - \eta q \tilde{\gamma}_n & g - \eta q \bar{\gamma} \end{pmatrix} \\ B &= \begin{pmatrix} \beta & 0 \\ 0 & \beta \end{pmatrix} \\ w_n^f &= \begin{pmatrix} w_n \\ w_n \end{pmatrix} \end{aligned}$$

Considering (3) and (19), we obtain the following augmented system:

$$x_{n+1}^f = A_n x_n^f + B w_n^f \tag{20}$$

Define

$$X_n = E[x_n^f (x_n^f)^T] = \begin{pmatrix} X_n^{uu} & X_n^{ue} \\ X_n^{eu} & X_n^{ee} \end{pmatrix} \tag{21}$$

Consider

$$E[w_n^f (w_n^f)^T] = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \triangleq E_0 \tag{22}$$

$$A \triangleq \begin{pmatrix} \alpha & 0 \\ \alpha - g & g - \eta q \bar{\gamma} \end{pmatrix} \tag{23}$$

and $\tilde{\gamma}_n$ is independent of w_n, u_n, e_n , and w_n is white.

We obtain by (20), (21) and (22)

$$X_{n+1} = A X_n A^T + B E_0 B^T \tag{24}$$

If the state of (24) is mean square bounded, the steady-state covariance X defined by

$$X \triangleq \lim_{n \rightarrow +\infty} X_n \tag{25}$$

exists and satisfied the following discrete-time modified Lyapunov equation:

$$X = A X A^T + B E_0 B^T \tag{26}$$

Theorem 1. If the state in (24) is mean square bounded, $|\alpha| < 1$, there exist a unique symmetric positive-semidefinite solution to (26) if and only if the Kalman filter parameters g, q satisfy

$$|g - q \bar{\gamma} \eta| < 1 \tag{27}$$

Proof. : It follows from [4] that, there exist a unique symmetric positive-semidefinite solution to (26) if and only if

$$\rho\{A \otimes A\} < 1 \tag{28}$$

where, ρ is the spectral radius and \otimes is the Kronecker product. Consider

$$A \otimes A = \begin{pmatrix} \alpha^2 & 0 \\ (\alpha - g)(\alpha + g - \eta q \bar{\gamma}) & (g - \eta q \bar{\gamma})^2 \end{pmatrix} \tag{29}$$

Substitute (29) into (28), the (28) is equivalent to

$$\max\{\alpha^2, (g - \eta q \bar{\gamma})^2\} < 1 \tag{30}$$

Consider $|\alpha|$, the (30) is equivalent to

$$|g - \eta q \bar{\gamma}| < 1 \tag{31}$$

This completes the proof of this theorem.

Theorem 2. The covariance of estimation error of total cost per unit time is equivalent to variance the estimation error of the state in equation (3), in other words $\widetilde{C}_n = (\eta \bar{\gamma})^2 X_n^{ee}$

Proof. We know from (17) that

$$\tilde{c}_n = \bar{\gamma} \eta e_n + \tilde{\gamma} \eta u_n = (\tilde{\gamma} \eta \ \bar{\gamma} \eta) \begin{pmatrix} u_n \\ e_n \end{pmatrix} \tag{32}$$

Define

$$\Psi \triangleq \begin{pmatrix} \tilde{\gamma} \\ \bar{\gamma} \end{pmatrix} \tag{33}$$

Substitute (33) into (32), we obtain

$$\tilde{c}_n = (\tilde{\gamma} \eta \ \bar{\gamma} \eta) \begin{pmatrix} u_n \\ e_n \end{pmatrix} = \eta \Psi^T x_n^f \tag{34}$$

Then covariance of estimation error of total cost per unit time follows as

$$\widetilde{C}_n = \varepsilon[\tilde{c}_n (\tilde{c}_n)^T] = \eta^2 \varepsilon(\Psi^T x_n^f (x_n^f)^T \Psi) \tag{35}$$

Considering γ_n , u_n is uncorrelated and γ_n is white noise sequences, we obtain from (35) that

$$\begin{aligned} \widetilde{C}_n &= \eta^2 \varepsilon(\Psi^T x_n^f (x_n^f)^T \Psi) \\ &= \eta^2 (0, \bar{\gamma}) \begin{pmatrix} X_n^{uu} & X_n^{ue} \\ X_n^{eu} & X_n^{ee} \end{pmatrix} \begin{pmatrix} 0 \\ \bar{\gamma} \end{pmatrix} \\ &= (\eta \bar{\gamma})^2 X_n^{ee} \end{aligned} \tag{36}$$

The theorem is proofed.

We obtain the corollary from theorem 1 and 2 as follows

Corollary 1. If $|\alpha| < 1$ and parameters g, q in the filter (18) satisfy (27), the steady covariance of estimation error of total cost per unit time exists and satisfies (26).

Proof. If $|\alpha| < 1$ and parameters g, q satisfy (27), by theorem 1, there exist a unique symmetric positive-semi-definite solution to (26). Considering the relation of \tilde{C}_n, X_n^{ee} in theorem (2), this completes the proof of this corollary.

As summary, we give our main results as follows

Corollary 2. Let $\tilde{C} \triangleq \lim_{n \rightarrow +\infty} \tilde{C}_n, \varepsilon \triangleq g - \eta q \bar{\gamma}$, If the state of (24) is mean square bounded $|\alpha| < 1$ and $|\varepsilon| < 1$, then the steady covariance of estimation error of total cost per unit time

$$\tilde{C} = \frac{(\eta\beta\bar{\gamma})^2}{(1 - \varepsilon\alpha)} \tag{37}$$

Proof. If the state of (24) is mean square bounded, $|\alpha| < 1, |\varepsilon| < 1$, then, it follows directly from theorem 1 and the Lyapunov equation (26) as follows

$$X^{ee} = \frac{\beta^2}{(1 - \varepsilon\alpha)} \tag{38}$$

Consider the equation $\tilde{C}_n = (\eta\bar{\gamma})^2 X_n^{ee}$ in theorem 2 and $\tilde{C} \triangleq \lim_{n \rightarrow +\infty} \tilde{C}_n, X^{ee} = \lim_{n \rightarrow +\infty} X_n^{ee}$, it follows immediately (37) This completes the proof of this corollary.

5 Numerical Example

In this section, we demonstrate how to design the Kalman filter through an example.

In a personal communication system the number of rings of location area $d = 4$. The velocity, v_n of a mobile user is a Gauss-Markov based fluid process with mean $\mu = 4.32km/h$ or $\mu = 1.2m/s$ memory level $\alpha = 0.8, \sigma^2$ is the variance of $v_n, \sigma = 0.2$, the cost for performing a location update, $\delta_{upd} = 1$, the cost for polling a cell, $\delta_{poll} = 2$, and the probability for complete observation is assumed to be 0.7, in other words $\bar{\gamma} = 0.7$. The packet rate is $\lambda = 2$. We can calculate from $\eta = \frac{4\delta_{upd}}{\sqrt{3\pi d}}$ that $\eta = \frac{4\delta_{upd}}{\sqrt{3\pi d}} = \frac{1}{\sqrt{3\pi}}$. Substituting $\eta = \frac{1}{\sqrt{3\pi}}$ and $\bar{\gamma} = 0.7$ into (31) gives $\left|g - \frac{0.7}{\sqrt{3\pi}}q\right| < 1$.

The purpose of this example is to design the Kalman filter parameters, g and q , such that the steady covariance of estimation error of total cost per unit time satisfies $\tilde{C} \leq 0.0007$. Let $q = 10\sqrt{3\pi}$, then $g = 7.8, \varepsilon = 0.8$, and the estimation error of total cost per second $\tilde{C} = \frac{(\eta\beta\bar{\gamma})^2}{(1 - \varepsilon\alpha)} = 0.00067$. The average total cost $\kappa = 148.127$ packets per second. This example shows how to estimate location management cost for PCS with missing measurements. Based on

Eq. (37) that if the memory level $0 \leq \alpha < 1$ and $|g - q\eta\bar{\gamma}| < 1$, the steady covariance of estimation error of total cost per second $\tilde{C} = \frac{(\eta\beta\bar{\gamma})^2}{(1-\varepsilon\alpha)} = \frac{(\eta\sigma\bar{\gamma})^2(1-\alpha^2)}{1-(g-q\eta\bar{\gamma})\alpha}$, then $\lim_{\alpha \rightarrow 0} \tilde{C} = (\eta\sigma\bar{\gamma})^2$, $\lim_{\alpha \rightarrow 1} \tilde{C} = 0$, and $\lim_{\bar{\gamma} \rightarrow 0} \tilde{C} = 0$, $\lim_{\bar{\gamma} \rightarrow 1} \tilde{C} = \frac{(\eta\beta)^2}{(1-\varepsilon\alpha)}$.

We can see from this example that the approach proposed possesses both effectiveness and flexibility.

6 Conclusion

This paper discusses the total cost estimation of mobile location management for PCS with missing measurements. Mobile nodes are assumed to move at variable velocity. Considering the fact that the change of a mobile's velocity within a short time period is necessarily limited due to physical restrictions, which leads a mobile node's future velocity to be correlated with its past and current velocity, we propose to use a Gauss-Markov mobility model to capture the correlation of a mobile's velocities. A Kalman filter of a linear uncertain discrete stochastic system is designed to estimate the total cost, and the formulas for steady state covariance of the estimation error of total cost computation are presented. The design of the proposed Kalman filter is illustrated by a numerical example.

Possible future research directions to location management in PCS include location management for time delay with measurement missing PCS, location management for special routing protocols, and location management for QoS of mobile decision support in PCS.

Acknowledgments. This work is partially supported by NSFC granted number 70271001, China Postdoctoral Fund granted number 2002032191 and Shanghai Fund of Science and Technology.

References

1. Cao, Y., Lam, J., Hu, L.: Delay-dependent Stochastic Stability and H_∞ Analysis for Time-delay Systems with Markovian Jumping Parameters. *J. Franklin Institute* **340** (2003) 423-434
2. Akyildiz, I.F., Ho, J., Lin, Y.: Movement-based Location Update and Selective Paging for PCS Networks. *IEEE/ACM Trans. Networking* **4** (1996) 629-638
3. Akyildiz, I.F., Wang, W.: A Dynamic Location Management Scheme for Next-Generation Multitier PCS Systems. *IEEE Trans. Wireless Communications* **1**(1) (2002) 178-189
4. DeKoning, W.L.: Optimal Estimation of Linear Discrete Time Systems with Stochastic Parameters. *Automatica* **20** (1984) 113-115
5. Ho, J.S.M., Akyildiz, I.F.: Mobile User Location Update and Paging under Delay Constraints. *ACM-Baltzer J. Wireless Networks* **1** (1995) 413-425
6. Lin, Y.B.: Reducing Location Update Cost in a PCS Network. *IEEE/ACM Trans. Networking* **5** (1997) 25-33
7. Liang, B., Haas, Z.J.: Predictive Distance-based Mobility Management for Multi-dimensional PCS Networks. *IEEE/ACM Trans. Networking* **11**(5) (2003) 718-732

8. Li, J., Pan, Y., Jia, X.: Analysis of Dynamic Location Management for PCS Networks. *IEEE Trans. Vehicular Technology* **51(5)** (2002) 1109-1119
9. Markoulidakis, J.G., Lyberopoulos, G.L., Anagnostou, M.E.: Traffic Model for Third Generation Cellular Mobile Telecommunications Systems. *ACM-Baltzer J. Wireless Networks* **4** (1999) 389-400
10. Bar-Noy, A., Kessler, I., Sidi, M.: Mobile Users: To Update or Not to Update? *ACM-Baltzer J. Wireless Networks* **1(2)** (1995) 175-186
11. Thomas, R., Gilbert, H., Mazziotto, G.: Influence of the Movement of the Mobile Station on the Performance of the Radio Cellular Network. In: *Proc. 3rd Nordic Seminar*. Copenhagen, Denmark (1988) 9-14
12. Wang, Z., Ho, D.W.C., Liu, X.: Variance-constrained Filtering for Uncertain Stochastic Systems with Missing Measurements. *IEEE Trans. Automatic Control* **48(7)** (2003) 1254-1258
13. Wong, V., Leung, V.C.M.: Location Management for Next-generation Personal Communications Networks. *IEEE Network* (2000) 18-24
14. Wan, G., Lin, E.: Cost Reduction in Location Management Using Semirealtime Movement Information. *ACM-Baltzer J. Wireless Networks* **5(4)** (1999) 245-256
15. Xie, H., Tabbane, S., Goodman, D. J.: Dynamic Location Area Management and Performance Analysis. In: *Proc. 42nd IEEE Vehicular Technology Conf.* (1992) 536-539
16. Cao, J., Tao, Q.: Estimation of the Domain of Attraction and the Convergence Rate of a Hopfield Associative Memory and an Application. *J. Comput. Syst. Sci.* **60(1)** (2000) 179-186
17. Zhou, J., Tang, B., Li, D.: Partition Digraph for Location Area Management in Mobile Computing Environment. *International Journal of Nonlinear Sciences and Numerical Simulation* **5(4)** (2004) 393-396

Neural-Based Separating Method for Nonlinear Mixtures

Ying Tan

National Laboratory on Machine Perception,
Department of Intelligence Science,
Peking University, Beijing, 100871, China
ytan@pku.edu.cn

Abstract. A neural-based method for source separation in nonlinear mixture is proposed in this paper. A cost function, which consists of the mutual information and partial moments of the outputs of the separation system, is defined to extract the independent signals from their nonlinear mixtures. A learning algorithm for the parametric RBF network is established by using the stochastic gradient descent method. This approach is characterized by high learning convergence rate of weights, modular structure, as well as feasible hardware implementation. Successful experimental results are given at the end of this paper.

1 Introduction

Recently, blind source separation in signal processing has received considerable attention from researchers, due to its numerous promising applications in the area of communications and speech signal processing, medical signal processing including ECG, MEG and EEG, and monitoring [1,2]. A number of blind separation algorithms have been proposed based on different separation models. These algorithms play increasingly important roles in many applications. The study of blind signal processing techniques is of both theoretical significance and practical importance.

Generally speaking, Blind source separation is to recover unobservable independent sources (or “signals”) from several observed data masked by linear or nonlinear mixing. Most existing algorithms for linear mixing models stem from the theory of the independent component analysis (ICA) [3]. Therefore, a solution to blind source separation problem exists and this solution is unique up to some trivial indeterminacies (permutation and scaling) according to the basic ICA theory [3]. Even though the nonlinear mixing model is more realistic and practical, most existing blind separation algorithms developed so far are valid for linear models. For nonlinear mixing models, many difficulties occur and both the linear ICA theory and existing linear demixing algorithms are no longer applicable because of the complexity of nonlinear characteristics [4].

So far several authors studied the difficult problem of the nonlinear blind source separation and proposed a few efficient demixing algorithms [4,5,6,7,8,9].

Deco [5] studied a very particular scenario of volume-conserving nonlinear transforms. Pajunen et al. [6] proposed model-free methods which used Kohonen’s self-organizing map (SOM) to extract independent sources from nonlinear mixture, but suffers from the exponential growth of network complexity and interpolation error in recovering continuous sources. Burel [7] proposed a nonlinear blind source separation algorithm using two-layer perceptrons by the gradient descent method to minimize the mutual information (measure of dependence). Subsequently, Yang et al. [8] developed an information backpropagation (BP) algorithm for Burel’s model by natural gradient method. In their model cross nonlinearities is included. Taleb et al. [9] proposed an entropy-based direct algorithm for blind source separation in post nonlinear mixtures. Recently, authors [12,13] proposed several algorithms and approaches for separation of nonlinear mixture of sources.

The purpose of this paper is to investigate a radial-basis function (RBF) neural network model for blind-style demixing of nonlinear mixtures in the presence of cross nonlinearities. Since the instinct unsupervised learning of the RBF network and blind signal processing are in essence unsupervised learning procedures, therefore the study of the RBF-based separation system seems natural and reasonable.

2 Nonlinear Mixture Model

A generic nonlinear mixture model for blind source separation can be described as $\mathbf{x}(t) = \mathbf{f}[\mathbf{s}(t)]$, where $\mathbf{x}(t) = [x_1(t), x_2(t), \dots, x_n(t)]^T$ is the vector of observed random variables, superscript T denotes the transposition, $\mathbf{s}(t) = [s_1(t), s_2(t), \dots, s_n(t)]^T$ is the vector of the latent variables called the independent source vector, \mathbf{f} is an unknown multiple-input and multiple-output (MIMO) mapping from R^n to R^n called nonlinear mixing transform (NMT). If the mixing function \mathbf{f} is linear, this model reduces to the linear mixing. In order for the mapping to be invertible we assume that the nonlinear mapping \mathbf{f} is monotone. The left part of Fig. 1 shows the general model of blind source separation system which contains both channel and cross-channel nonlinearities.

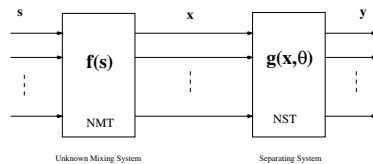


Fig. 1. Nonlinear mixing and separating systems for blind signal separation

The separating system $\mathbf{g}(\cdot, \theta)$ in the right part of Fig. 1, also called nonlinear separation transform (NST), is used to recover the original signals from the nonlinear mixture $\mathbf{x}(t)$ without the knowledge of the source signals $\mathbf{s}(t)$ and

the mixing nonlinear function $\mathbf{f}(\cdot)$. Obviously, this problem is untractable, in particular for nonlinear mixing system, unless conditions are imposed on the nonlinear function $\mathbf{f}(\cdot)$. At first, the existence of the solution for the NST can be guaranteed. According to related nonlinear ICA theories, the nonlinear ICA problem always has at least one solution. That is, given a random vector \mathbf{x} , there is always a function \mathbf{g} so that the components of $\mathbf{y} = [y_1, \dots, y_n]^T$ given by $\mathbf{y} = \mathbf{g}(\mathbf{x})$ are independent [64].

Unfortunately, this kind of mapping is not at all unique. It is shown in [4] that a unique solution subjected to a rotation can be obtained under the assumptions that the problem is a two-dimensional one, mixing function is a conformal mapping, and the densities of the independent components are known and have bounded support. In order to obtain a unique solution of the model in Fig. 1, we assume that $\mathbf{f}(\cdot)$ is invertible and its inverse $\mathbf{f}^{-1}(\cdot)$ exists and can be uniquely approximated by a parametric RBF network shown in Fig. 2 of the following section. In addition, we add some constraints on the output; i.e., the moment matching between the outputs of the separating system and sources. According to Fig. 1, the output of the nonlinear separating system can be written as

$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t), \boldsymbol{\theta}) = \mathbf{g}(\mathbf{f}(\mathbf{s}(t)), \boldsymbol{\theta}) = \mathbf{s}(t) \tag{1}$$

where $g(\cdot, \boldsymbol{\theta}) = f^{-1}(\cdot)$ denotes a parametric fitting function class, $\boldsymbol{\theta}$ is a parameter vector to be determined.

Generally speaking, $g(\cdot, \boldsymbol{\theta})$ can be altered by varying $\boldsymbol{\theta}$. If we find such $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$ that $g(\cdot, \hat{\boldsymbol{\theta}})$ is a good approximation of the inverse of the nonlinear mixing function $f^{-1}(\cdot)$, then a good separation of nonlinear mixture is achieved.

3 Nonlinear Separation Based on an RBF Network

3.1 The RBF Neural Network

Fig. 2 shows an n -input and n -output RBF network model. It consists of three layers; i.e., input layer, hidden layer and output layer. The neurons in hidden layer are of local response to its input and called RBF neurons while the neurons of the output layer only sum their inputs and are called linear neurons. The RBF network of Fig. 2 is often used to approximate an unknown continuous function $\phi : R^n \rightarrow R^n$ which can be described by the affine mapping

$$\mathbf{u}(\mathbf{x}) = \mathbf{B}K(\mathbf{x}, \mathbf{p}) \tag{2}$$

$$K(\mathbf{x}, \mathbf{p}) = [1, \exp(-(\mathbf{x}-\boldsymbol{\mu}_1)^T(\mathbf{x}-\boldsymbol{\mu}_1)/\sigma_1^2), \dots, \exp(-(\mathbf{x}-\boldsymbol{\mu}_M)^T(\mathbf{x}-\boldsymbol{\mu}_M)/\sigma_M^2)]^T. \tag{3}$$

where $\mathbf{B} = [\alpha_{ij}]$ is a $n \times M$ weight matrix of the output layer, $K(\mathbf{x}, \mathbf{p})$ is Gaussian kernel function vector of the RBF network, which consists of the locally receptive functions. $\mathbf{p} = (\boldsymbol{\mu}_1, \sigma_1, \dots, \boldsymbol{\mu}_M, \sigma_M)^T$ is the parameter set of the kernel function. Here we let the first component of $K(\mathbf{x}, \mathbf{p})$ be 1 for taking the bias into account.

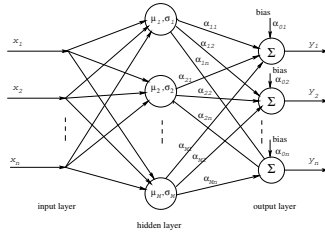


Fig. 2. The radial basis function networks with n output units

3.2 Nonlinear Separation System Based on RBF Network

Since the local response power of RBF networks offers great classification and approximation capabilities, the Gaussian RBF network is used as a good function approximator in many modelling applications. If we let \mathbf{S} be a compact subset in R^n and $\mathbf{p}(\mathbf{x})$ be a continuous target vector on \mathbf{S} , then for any $\epsilon > 0$ there exist M centroids $\boldsymbol{\mu}_i = [\mu_{i1}, \dots, \mu_{in}]^T$ and an $n \times M$ constant matrix \mathbf{B} such that $\mathbf{r}(\mathbf{x}, \boldsymbol{\theta}) = \mathbf{B} \cdot K(\mathbf{x}, \mathbf{p})$ satisfies $|\mathbf{r}(\mathbf{x}, \boldsymbol{\theta}) - \mathbf{p}(\mathbf{x})| < \epsilon$ for all $\mathbf{x} \in \mathbf{S}$. This approximation ability of RBF networks directly stems from the classic Stone-Weierstrass theorem and is closely related to Parzen’s approximation theory. Therefore, the inverse of the nonlinear mixing model can be modeled by using an RBF network. Such architecture is preferred over multilayer perceptrons (MLP) as an RBF network has better capability for functional representation. Since its response is linearly related to its weights, learning in an RBF network is expected to train faster while its local response power offers a good approximation capability. As a result, we can reach

$$\mathbf{y} = \hat{\mathbf{B}}K[\mathbf{f}(\mathbf{s}), \hat{\mathbf{p}}] \propto \mathbf{s} \tag{4}$$

where $\mathbf{g}(\cdot, \hat{\boldsymbol{\theta}}) = \hat{\mathbf{B}}K[\cdot, \hat{\mathbf{p}}]$, $\hat{\mathbf{B}}$ and $\hat{\mathbf{p}}$ are the final estimates of parameters \mathbf{B} and \mathbf{p} of the RBF network such that the inverse of \mathbf{f} is well approximated by the RBF network.

3.3 Cost Function

In order to deal with the nonlinear separation problem effectively, we define a cost function, or contrast function, which is the objective function for signal separation, as

$$C(\boldsymbol{\theta}) = I(\mathbf{y}) + \sum_{i_1 \dots i_n} c_{i_1 \dots i_n} [M_{i_1 \dots i_n}(\mathbf{y}, \boldsymbol{\theta}) - M_{i_1 \dots i_n}(\mathbf{s})]^2 \tag{5}$$

where $I(\mathbf{y})$ is mutual information of the outputs of the separation system, $M_{i_1 \dots i_n}(\mathbf{y}, \boldsymbol{\theta})$ and $M_{i_1 \dots i_n}(\mathbf{s})$ are the $i_1 \dots i_n$ -th moments of \mathbf{y} and \mathbf{s} , respectively, $c_{i_1 \dots i_n}$ are constants which are used to balance the mutual information and the matching of moments.

According to information theory and related the Kullback-Leibler divergence, mutual information $I(\mathbf{y})$ in Eq. (5) is expressed as

$$I(\mathbf{y}) = \sum_{i=1}^n H(y_i) - H(\mathbf{y}) \tag{6}$$

where $H(\mathbf{y}) = -E[\log(p_{\mathbf{y}}(\mathbf{y}))]$ is the joint entropy of random vector \mathbf{y} , $H(y_i) = -E[\log(p_{y_i}(y_i))]$ is the entropy of random variable y_i , the i th component of \mathbf{y} , and $E(\cdot)$ denotes the expectation operator.

The $i_1 \cdots i_n$ th moment of \mathbf{y} is defined as

$$M_{i_1 \cdots i_n}(\mathbf{y}) = E(y_1^{i_1} \cdots y_n^{i_n}) - E(y_1^{i_1}) \cdots E(y_n^{i_n}). \tag{7}$$

It can be seen from Eqs. (5)- (7) that the constrast function defined in Eq. (5) is always non-negative, and reaches zero if and only if both mutual information is null and a perfect matching of moments between the outputs of the separation system and original sources is achieved. Therefore, independent outputs with the same moments as that of original sources can be found by minimizing the contrast function by adjusting the parameters of the RBFN separating system, i.e.,

$$\hat{\boldsymbol{\theta}} = \arg \min_{\boldsymbol{\theta}} \{I(\mathbf{y}) + \sum_{i_1 \cdots i_n} c_{i_1 \cdots i_n} [M_{i_1 \cdots i_n}(\mathbf{y}, \boldsymbol{\theta}) - M_{i_1 \cdots i_n}(\mathbf{s})]^2\}. \tag{8}$$

3.4 Learning Algorithm of the Separating RBF Network

In order to derive the unsupervised learning algorithm of all the parameters of the separating RBF network, we employ the gradient descent method. First of all, we compute the gradient of the contrast function of Eq. (5) with respect to the parameter $\boldsymbol{\theta}$ and obtain

$$\frac{\partial C(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial I(\mathbf{y})}{\partial \boldsymbol{\theta}} + \sum_{i_1 \cdots i_n} 2c_{i_1 \cdots i_n} [M_{i_1 \cdots i_n}(\mathbf{y}, \boldsymbol{\theta}) - M_{i_1 \cdots i_n}(\mathbf{s})] \frac{\partial M_{i_1 \cdots i_n}(\mathbf{y}, \boldsymbol{\theta})}{\partial \mathbf{y}} \frac{\partial \mathbf{y}}{\partial \boldsymbol{\theta}} \tag{9}$$

where mutual information can be further rewritten as

$$I(\mathbf{y}) = \sum_{i=1}^n H(y_i) - E\{\log |\frac{\partial \mathbf{g}(\mathbf{x}, \boldsymbol{\theta})}{\partial \mathbf{x}}|\} - H(\mathbf{x}) \tag{10}$$

where $|\partial \mathbf{g}(\mathbf{x}, \boldsymbol{\theta})/\partial \mathbf{x}|$ is the determinant of the Jacobian matrix of $\mathbf{g}(\mathbf{x}, \boldsymbol{\theta})$ with respect to vector \mathbf{x} .

In Eq. (10), the computation of $H(y_i)$ needs to use the pdf of y_i which is unknown. By applying the Gram-Charlier expansion suggested by Amari et al. [10] to express each marginal pdf of \mathbf{y} , the marginal entropy can be approximated as

$$H(y_i) \approx \frac{1}{2} \log(2\pi e) - \frac{(k_3^i)^2}{2 \times 3!} - \frac{(k_4^i)^2}{2 \times 4!} + \frac{3}{8} (k_3^i)^2 k_4^i + \frac{1}{16} (k_4^i)^3 \tag{11}$$

where $k_3^i = m_3^i$, $k_4^i = m_4^i - 3$ and $m_k^i = E[(y_i)^k]$, $j = 1, \dots, n$.

Since $H(\mathbf{x})$ does not contain any parameters of the separating RBF network, it becomes null when taking gradient with respect to the parameters.

Regarding different concrete parameters \mathbf{B} , $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ of the parameter set $\boldsymbol{\theta}$ of the RBF network, we have the following gradient equations of the separated signal \mathbf{y}

$$\frac{\partial \mathbf{y}}{\partial \mathbf{B}} = K(\mathbf{x}, \mathbf{t}), \tag{12}$$

$$\frac{\partial \mathbf{y}}{\partial \boldsymbol{\mu}} = \mathbf{B} \cdot \text{diag}[\mathbf{v}_1 \circ K(\mathbf{x}, \mathbf{t})], \tag{13}$$

$$\frac{\partial \mathbf{y}}{\partial \boldsymbol{\sigma}} = \mathbf{B} \cdot \text{diag}[\mathbf{v}_2 \circ K(\mathbf{x}, \mathbf{t})]. \tag{14}$$

where $\mathbf{v}_1 = [2(\mathbf{x} - \boldsymbol{\mu}_1)/\sigma_1^2, \dots, 2(\mathbf{x} - \boldsymbol{\mu}_M)/\sigma_M^2]^T$, $\mathbf{v}_2 = [2\|\mathbf{x} - \boldsymbol{\mu}_1\|^2/\sigma_1^3, \dots, 2\|\mathbf{x} - \boldsymbol{\mu}_M\|^2/\sigma_M^3]^T$, function $\text{diag}[\cdot]$ denotes diagonal matrix, symbol \circ denotes Hadamard product which is the multiplication of corresponding pairs of elements between two vectors.

Finally, from Eqs. (9), (12)–(14), we can easily calculate the gradients of the constrast function with respect to each parameter of parameter set $\boldsymbol{\theta}$, then give the following learning updating formula:

$$\delta \mathbf{B} = -\eta \frac{\partial C(\boldsymbol{\theta})}{\partial \mathbf{B}}, \quad \delta \boldsymbol{\mu} = -\eta \frac{\partial C(\boldsymbol{\theta})}{\partial \boldsymbol{\mu}}, \quad \delta \boldsymbol{\sigma} = -\eta \frac{\partial C(\boldsymbol{\theta})}{\partial \boldsymbol{\sigma}} \tag{15}$$

where $\boldsymbol{\mu} = [\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_M]^T$ and $\boldsymbol{\sigma} = [\sigma_1, \dots, \sigma_M]^T$; η denotes the positive learning rate; $\delta \mathbf{B}$, $\delta \boldsymbol{\mu}$ and $\delta \boldsymbol{\sigma}$ indicate the adjustments of \mathbf{B} , $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, respectively.

3.5 Performance Index and Algorithm Description

From Eq. (10), by omitting the unknown $H(\mathbf{x})$, an index to measure the independence of the outputs of the separation system is defined as

$$J_i = \sum_{i=1}^n H(y_i) - E\left\{\log \left| \frac{\partial \mathbf{g}(\mathbf{x}, \boldsymbol{\theta})}{\partial \mathbf{x}} \right| \right\} \tag{16}$$

Even though the index J_i may be negative, the lower the value of J_i is, the more independent the outputs of the separating system is. The smallest negative value of J_i is just equal to the reciprocal of $H(\mathbf{x})$. In a similar manner, according to Eq. (10), a performance index measuring moment match up to the k -th order between the outputs of the separation system and original sources can also be directly defined as

$$J_m^k = \sum_{i_1 \dots i_n \leq k} [M_{i_1 \dots i_n}(\mathbf{y}, \boldsymbol{\theta}) - M_{i_1 \dots i_n}(\mathbf{s})]^2 \tag{17}$$

The maximum value of k is chosen such that the inverse of the mixing nonlinear transform can be uniquely approximated by an RBF network through the minimization of the cost function. In actual implementation, usually only up to forth-order moment is enough for this purpose by experiments. We expect both J_i and J_m^k are at their minima simultaneously, so the two indices can be combined into one overall index as follows

$$J = J_i + \alpha J_m^k \tag{18}$$

where α is a proportionality constant weighting the two quantities.

In addition, a stopping criterion to terminate the iterative process is defined as a relative change amount of the overall index e_r is less than a predetermined small positive constant ϵ . What follows summarizes the steps of the learning algorithm.

- 1: Given initialization parameters $\mathbf{B}, \boldsymbol{\mu}, \boldsymbol{\sigma}$, for our RBF network, choose a small learning rate η and index balance number α as well as the order number k of the moment to be matched.
- 2: Adjust parameters, $\mathbf{B}, \boldsymbol{\mu}, \boldsymbol{\sigma}$, of the RBF network by using Eqs (12)-(15).
- 3: Compute the performance indices J_i, J_m^k, J according to Eqs. (16) to (18) by making use of current model parameters.
- 4: Verify the termination criterion ($e_r < \epsilon$) and exit, otherwise go to step 2.

Note that in the approach we initialize the RBF network randomly even though a good initialization greatly benefits a gradient descent learning algorithm.

4 Simulation Results

Consider a two-channel nonlinear mixture with a cubic nonlinearity:

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \mathbf{A}_2 \begin{bmatrix} (\cdot)^3 \\ (\cdot)^3 \end{bmatrix} \mathbf{A}_1 \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} \tag{19}$$

where mixing matrices \mathbf{A}_1 and \mathbf{A}_2 are nonsingular and given as

$$\mathbf{A}_1 = \begin{pmatrix} 0.25 & 0.86 \\ -0.86 & 0.25 \end{pmatrix}, \quad \mathbf{A}_2 = \begin{pmatrix} 0.5 & 0.9 \\ -0.9 & 0.5 \end{pmatrix}$$

Obviously, the inverse of the mixing model exists and can be expressed as

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \mathbf{A}_1^{-1} \begin{bmatrix} \text{sgn}(\cdot)(\cdot)^{1/3} \\ \text{sgn}(\cdot)(\cdot)^{1/3} \end{bmatrix} \mathbf{A}_2^{-1} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

where sgn function is to take the sign of the argument.

The source vector $\mathbf{s}(t)$ consists of a sinusoidal signal and an amplitude-modulated signal; i.e., $\mathbf{s}(t) = [0.5 * [1 + \sin(6\pi t)] \cos(100\pi t), \sin(20\pi t)]^T$.

An RBF network shown in Fig. 2 is used to separate this nonlinear mixture. In this experiment we choose six hidden neurons with Gaussian kernel function. The moment matching is taken up to third-order. An example of the evolution curves for the learning algorithm is shown in Fig. 3. The learning curve is smooth and it converges after 500 iterations.

The value of the performance index after convergence of the learning algorithm is very small so that the separated signals obtained by our model are seen to be mutually independent.

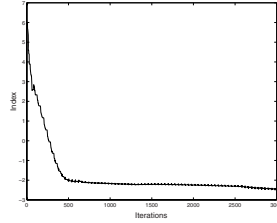


Fig. 3. Learning curve of the proposed algorithm

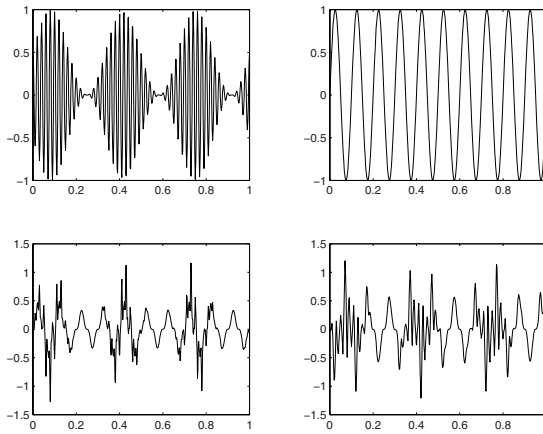


Fig. 4. Two source signals (above) and their nonlinear mixtures (below)

Fig 4 shows the two source signals $\mathbf{s}(t)$ and the input signals $\mathbf{x}(t)$ of the separating system of Fig 1, i.e., the mixture of the sources. Fig 5 show the signals separated by the proposed approach. For convenient comparison, we also have plotted the separating results of the linear demixing algorithm proposed in reference [11] for this case into Fig 6. It can be seen that the linear demixing algorithm fails to separate the nonlinear mixture but our proposed model and algorithms can give a clear separation of this nonlinear mixture. We also used the other specific nonlinear algorithms such as the SOM algorithm [6], but no useful results have been obtained.

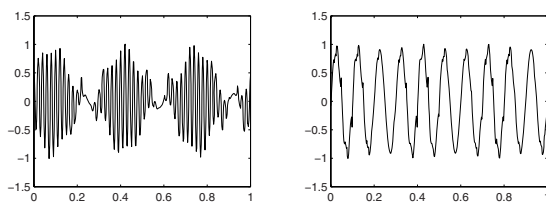


Fig. 5. Separated signals of our proposed method

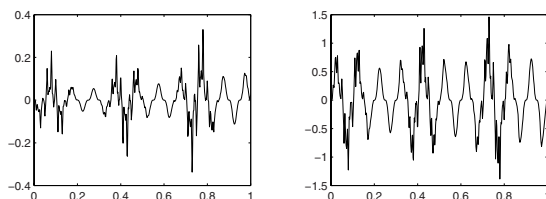


Fig. 6. Separated signals of adaptive algorithm for linear mixture case

5 Concluding Remarks

A neural-based separating approach is established to separate nonlinearly mixed sources in terms of a novel cost function which consists of mutual information and cumulants' matching. because of the local response of RBF networks, this proposed method is characterized by fast learning convergence rate of weights, natural unsupervised learning characteristics, modular network structure as well as suitable hardware implementation. All of these properties make it be an effective candidate for real-time multi-channel separation of nonlinear mixtures of sources. Extensive simulation results verified the validation of our methods.

Acknowledgments. This Project was supported by National Natural Science Foundation of China, Grant No. 60673020.

References

1. Papadias, C.B., Paulraj, A.: A Constant Modulus Algorithm for Multi-user Signal Separation in Presence of Delay Dpread Using Antenna Arrays. *IEEE Signal Processing Letters* **4** (1997) 178-181
2. Herault, J., Jutten, C.: Space or Time Adaptive Signal Processing by Neural Network Models. Proc. AIP Conf., Snowbird, UT, 1986, in *Neural Networks for Computing*, J. S. Denker, Ed. New York: Amer. Inst. Phys. (1986) 206-211
3. Jutten, C., Herault, J.: Blind Separation of Sources, Part I: An Adaptive Algorithm Based on Neuromimetic Architecture. *Signal Processing* **24** (1991) 1-20
4. Hyvarinen, A., Pajunen, P.: Nonlinear Independent Component Analysis: Existence and Uniqueness Results. *Neural Networks* **12** (1999) 429-439

5. Deco, G., Brauer, W.: Nonlinear Higher-order Statistical Decorrelation by Volume-conserving Neural Architectures. *Neural Networks* **8** (1995) 525-535
6. Pajunen, P., Hyvarinen, A., Karhunen, J.: Nonlinear Blind Source Separation by Self-organizing Maps. *Progress in Neural Information Processing: Proceedings of ICONIP'96*, 2, Springer (1996) 1207-1210
7. Burel, G.: Blind Separation of Sources: a Nonlinear Neural Algorithm. *Neural Networks* **5** (1992) 937-947
8. Yang, H.H., Amari, S., Cichocki, A.: Information Back-propagation for Blind Separation of Sources from Non-linear Mixture. *Proc. IEEE IJCNN, Houston, USA* (1997) 2141-2146
9. Taleb, A., Jutten, C., Olympieff, S.: Source Separation in Post Nonlinear Mixtures: an Entropy-based Algorithm. *Proc. ESANN'98* (1998) 2089-2092
10. Amari, S., Cichocki, A., Yang, H.H.: A New Learning Algorithm for Blind Signal Separation. *Advances in Neural Information Processing Systems*, 8, eds. David S. Touretzky, Michael C. Mozer and Michael E. Hasselmo, MIT Press: Cambridge, MA (1996) 757-763
11. Cichocki, A., Unbehauen, R.: Robust Neural Networks with On-line Learning for Blind identification and Blind Separation of Sources. *IEEE Trans. Circuits and Systems I* **43** (1996) 894-906
12. Tan, Y., Wang, J., Zurada, J.M.: Nonlinear Blind Source Separation Using Radial Basis Function Networks. *IEEE Transaction on Neural Networks* **12(1)** (2001) 124-134
13. Tan, Y., Wang, J.: Nonlinear Blind Separation Using Higher-Order Statistics and A Genetic Algorithm. *IEEE Transaction on Evolutionary Computation* **5(6)** (2001) 600-612

Adaptive Natural Gradient Algorithm for Blind Convolutive Source Separation

Jian Feng^{1,2}, Huaguang Zhang^{1,2}, Tieyan Zhang¹, and Heng Yue²

¹ School of Information Science and Engineering, Northeastern University,
Shenyang 110004, China

fjneu@163.com, hg_zhang@21cn.com, zty@syepi.edu.cn

² Key Laboratory of Process Industry Automation, Ministry of Education,
Northeastern University, Shenyang 110004, China

yueheng1968@163.com

Abstract. An adaptive natural gradient algorithm for blind source separation based on convolutional mixture model is proposed. The proposed method makes use of cost function as optimum criterion in separation process. The update formula of separation matrix is deduced. The learning steps for blind source separation algorithm are given, and high capability of the proposed algorithm has been demonstrated. The simulations results have shown the validity, practicability and the better performance of the proposed method. This technique is suitable for many applications in real life systems.

1 Introduction

Blind signal separation (BSS) is the problem of recovering mutually independent unobserved signals (sources) from multiple observed data masked by linear or nonlinear mixing. Since the late 1980s, blind source separation in signal processing has attracted significant coverage from researchers, due to its wide number of applications in diverse fields, such as communications and speech signal processing ^{[1][2]}, medical signal processing including ECG, MEG, and EEG ^[3], and machine fault diagnosis ^[4]. After Herault ^[5] and Weinstein ^[6], various methods for blind separation have been proposed.

BSS consists of recovering signals from different physical sources from several observed combinations independently of the propagation medium. BSS is also a promising tool for nondestructive control systems condition monitoring by signals analysis, as it is intended to retrieve the signature of a single device from combinations of several working status. In this way, BSS can be seen as a pre-processing step that improves the diagnosis.

The paper is organized as follows. In section 2, we describe the blind source separation problem. Section 3 describes the optimization criterion of objective function in BSS process and in section 4 the learning algorithm for achieving blind source separation are derived. The advantages in terms of performance of proposed method are shown by simulation results in section 5.

2 Problem Definition

BSS consists in recovering signals of different physical sources from the observation of several combinations of them. Typically, the observations are obtained as the output of a set of sensors, where each sensor receives a different combination of source signals. The adjective "blind" indicates that the source signals are not observed and also that no information is available about the combinations and the noises. This approach is usually used when it is too difficult to model the transfer from the sources to the sensors. The lack of knowledge about the combinations and the sources is compensated by the hypothesis of mutually independent sources. Nevertheless, signals of different physical sources generally satisfy this condition. This assumption allows exploitation of the spatial diversity provided by many sensors and is the fundamental basis of BSS.

BSS can be very computationally demanding if the number of source signals is large (say, of order 100 or more). In particular, this is the case in biomedical signal processing applications such as electroencephalographic/magnetoencephalographic data processing, where the number of sensors can be larger than 100 and where it is desired to extract only some "interesting" sources. Fortunately, blind signal extraction method overcomes somewhat this difficulty. The general model of BSS with noise free model is shown in Fig. 1.

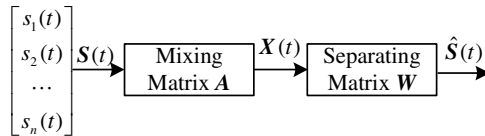


Fig. 1. BSS general scheme with noise free model

Suppose that the statistically independent source signals $s_i(t)$, $i = 1, \dots, n$, $t = 1, \dots$, are picked up by m sensors ($m \geq n$) through an unknown propagation medium (mixing matrix A) at the discrete time t . The coefficients a_{ij} in mixing matrix A ($m \times n$) are time-independent if the propagation delay through the medium is very short (in relation to the sampling period). But in some real life systems, the propagation medium is accounted for transmission with severe filtering and delays. Thus the model used for this application is a linear time dependent mixture, called a convolutional mixture. Each a_{ij} represents the linear transfer function from the i th to the j th sensor and is given by $a_{ij}(z) = \sum_{k=0}^{P-1} a_{ij}(k) \cdot z^{-k}$, where P is the length of mixing filter, z^{-1} is the backward-shift operator. The observation vector $X(t) = [x_1(t), \dots, x_n(t)]$ is therefore a mixture of the n sources:

$$x_i(t) = \sum_{j=1}^n a_{ij}(z) s_j(t). \tag{1}$$

Using z transform, this equation can be expressed in vector notation as follows

$$\mathbf{X}(t) = \mathbf{A}(t)\mathbf{S}(t), \tag{2}$$

where $\mathbf{X}(t) = [x_1(t), \dots, x_m(t)]^T$, $\mathbf{S}(t) = [s_1(t), \dots, s_n(t)]^T$, and $\mathbf{A} = [a_{ij}(z)]$.

Sometimes, mixture $\mathbf{X}(t)$ is also considered as linear combination of $\mathbf{X}(t)$ and additive noise, but noise is also supposed to be independent of the sources. Both the sources and the noises are required to be stationary and have a zero mean.

So, the aim of BSS is to estimate a stable inverse system of \mathbf{A} , a separating filter matrix $\mathbf{W} (n \times m)$ to make the output signals statistically independent. Each $w_{ij}(t)$ is

given by $w_{ij}(z) = \sum_{l=0}^{L-1} w_{ij}(l) \cdot z^{-l}$, where L is the length of separating filter, z^{-1} is the backward-shift operator. The output $\hat{\mathbf{S}}(t)$ of the separating filter matrix \mathbf{W} is also an optimal estimate of source $\mathbf{S}(t)$, such as

$$\hat{\mathbf{S}}(t) = \mathbf{W}(t)\mathbf{X}(t), \tag{3}$$

where

$$\mathbf{W}(z) = \sum_{l=0}^{L-1} \mathbf{W}(l)z^{-l}. \tag{4}$$

3 Optimization Criterion

Usually, a kind of cost function is chose as objective function in BSS process. When cost function obtains its maximum or minimum, we consider that separation process is done. The selection of cost function lies on certain BSS problem. The following rules are necessary for choice: (1) cost function should include as much characteristic information of source signals as possible. (2) There exists extremum in the cost function, and signals are linear irrelevant or mutual independent. (3) The computational consumption is as lower as possible. Because autocorrelation property of non-stationery signals is time-dependent, ideal cost function should include correlation information as much as possible at any time. But its computational consumption is too expensive to fulfill on-line. In this paper, we choose the cost function as in [7], such as

$$\mathbf{J}(t) = \sum_{q=0}^d \mathbf{J}_q(t) = \sum_{q=0}^d \| E\{\hat{\mathbf{S}}(t)\hat{\mathbf{S}}^T(t-q)\} - \mathbf{D}_q(t) \|_F^2, \tag{5}$$

where $E\{\cdot\}$ denotes mathematical expectation, q is delay time coefficient at time t , d is total number of all delay, $\mathbf{D}_q(t)$ is a diagonal matrix in relation to q , $\|\cdot\|_F$ denotes Frobenius norm of matrix. Frobenius norm of matrix \mathbf{A} is defined as

$$\|\mathbf{A}\|_F = \sqrt{\sum_i^m \sum_j^n |a_{ij}|^2}. \tag{6}$$

The cost function (5) is a covariance matrix of output signals $\hat{\mathbf{S}}(t)$. When the Frobenius norm of its non-diagonal components takes the minimum or equals to zero, blind signal separation can be achieved.

Wu obtained adaptive separation algorithm based on linear instantaneous mixture model using conventional gradient descent method. According to the same idea, in convolutional mixture model, adaptive iterative equation of the separating filter matrix \mathbf{W} can be obtained as follows:

$$\begin{aligned}\Delta \mathbf{W}_{i+1}^T(l) &= \mathbf{W}_{i+1}^T(l) - \mathbf{W}_i^T(l) \\ &= -\eta \sum_{q=0}^d \{[\mathbf{C}_{l,q}(t) + \mathbf{C}_{l,q}^T(t)] \mathbf{W}_i^T(z) \bullet [E\{\hat{\mathbf{S}}(t) \hat{\mathbf{S}}^T(t-q)\} - \mathbf{D}_q(t)]\},\end{aligned}\quad (7)$$

where $\mathbf{C}_{l,q}(t) = E\{X(t-l)X^T(t-q)\}$, η is learning rate parameter(stepsize) to determine the learning speed.

4 Adaptive Separating Algorithm

Conventional gradient descent algorithm allows a fast convergence without introducing an additional algorithmic delay. But it always receives local optimum solution and the computational complexity is increased. In non-orthogonal space, Conventional gradient descent algorithm can not receive optimum performance.

Amari^[8] proposed a kind of natural gradient algorithm with low computational consumption and proved its asymptotical stability. We can deduce the cost function (5) according to natural gradient principle. Such will be stated next.

At time t , the separating filter matrix \mathbf{W} satisfies

$$\hat{\mathbf{S}}(t) = \mathbf{W}(t)X(t), \hat{\mathbf{S}}(t-q) = \mathbf{W}(t)X(t-q). \quad (8)$$

Let \mathbf{M} be

$$\mathbf{M} = E\{\hat{\mathbf{S}}(t)\hat{\mathbf{S}}^T(t-q)\} - \mathbf{D}_q(t), \quad (9)$$

the cost function (5) can be rewritten as follows

$$\mathbf{J}(t) = \sum_{q=0}^d tr(\mathbf{M}^T \mathbf{M}), \quad (10)$$

where operator $tr(\cdot)$ denotes matrix trace, the sum of the elements of the principal diagonal of a matrix.

Calculating the derivative in equation (10),

$$\frac{d\mathbf{J}(t)}{d\mathbf{M}} = 2 \sum_{q=0}^d \mathbf{M}. \quad (11)$$

At time t , an adaptive iterative formula can be obtained as follows

$$\Delta \mathbf{M} = \mathbf{M}(t+1) - \mathbf{M}(t) = 2\eta_t \sum_{q=0}^d [\mathbf{A}_q(t) - \hat{\mathbf{S}}(t)\hat{\mathbf{S}}^T(t-q)], \quad (12)$$

where $\mathbf{A}_q(t) = diag(\hat{\mathbf{S}}(t)\hat{\mathbf{S}}^T(t-q))$, η_t is stepsize factor at time t . We can get following equation from (9):

$$\frac{d\mathbf{M}}{d\mathbf{W}^T(l)} = (\mathbf{C}_{l,q}(t) + \mathbf{C}_{l,q}^T(t))\mathbf{W}^T(z) = \mathbf{W}^{-1}(z)(\mathbf{P}_{l,q}(t) + \mathbf{P}_{l,q}^T(t)), \quad (13)$$

where $\mathbf{P}_{l,q}(t) = E\{\hat{\mathbf{S}}(t-l)\hat{\mathbf{S}}^T(t-q)\}$.

If we substitute instantaneous value for expectation, the relation between \mathbf{M} and $\mathbf{W}(l)$ can be expressed in form of differential equation:

$$\Delta\mathbf{M} = \mathbf{W}^{-1}(z)[\hat{\mathbf{S}}_{l,q}(t) + \hat{\mathbf{S}}_{l,q}^T(t)]\Delta\mathbf{W}^T(l), \quad (14)$$

where $\hat{\mathbf{S}}_{l,q}(t) = \hat{\mathbf{S}}(t-l)\hat{\mathbf{S}}^T(t-q)$.

Combining (12) and (14), we can obtain

$$\Delta\mathbf{W}_{i+1}^T(l) = 2\eta_i \sum_{q=0}^d \{[\hat{\mathbf{S}}_{l,q}(t) + \hat{\mathbf{S}}_{l,q}^T(t)]^{-1}\mathbf{W}_i(z)(l)[\mathbf{A}_q(t) - \hat{\mathbf{S}}_q(t)]\} \quad (15)$$

Because of decreased correlation in iterative process, correlation matrix inclines diagonal. Substituting $\mathbf{A}_q(t) = \text{diag}(\hat{\mathbf{S}}_{l,q}(t))$ for $\hat{\mathbf{S}}_{l,q}(t)$, we can obtained adaptive iterative formula of separation update matrix $\Delta\mathbf{W}_{i+1}^T(l)$ as follows:

$$\Delta\mathbf{W}_{i+1}^T(l) = \eta_i \sum_{q=0}^d \{\mathbf{A}_{l,q}^{-1}(t)\mathbf{W}_i(z)[\mathbf{A}_q(t) - \hat{\mathbf{S}}_q(t)]\} \quad (16)$$

Similarly, the learning steps for BSS can be given as follows:

- 1) Choose the length of separation filter L and delay coefficient q , initialize separation matrix.
- 2) Compute output $\hat{\mathbf{S}}(t)$ according to (3).
- 3) Compute update matrix $\Delta\mathbf{W}_{i+1}^T(l)$ according to (16).
- 4) Construct separation matrix (4).
- 5) Compute the cost function $\mathbf{J}(t)$ according to (5) by making use of current separation matrix parameter.
- 6) Verify that $\mathbf{J}(t)$ is smaller than the selected threshold and exit, otherwise go to step 2).

5 Simulation Results

The model and algorithms proposed in this paper have been implemented by Matlab toolbox. A number of simulations have been performed to fully evaluate the validity and performance of the algorithms for stationary source signals with complex convolutional mixture.

As an exemplary simulation, we consider random mixtures of 5 normalized sources with zero mean and unit variance. We performed 100 times simulations and used the histogram to distinguish the desired sources among those estimated. In each simulation, we ran the extraction algorithm one or several times until all the sources signals were extracted. The obtained results were that, in 18% of the experiments in which

we extracted all the desired sources with just the first run of the algorithm, this quantity increases to 87% of the experiments if a second run is allowed and to 96% after the third run. Thus, we can prove the proposed algorithm has a high capability. Better results have been also obtained for instantaneous mixture model.

6 Conclusion

In this paper, we adopt an adaptive natural gradient algorithm to solve blind delayed source separation problem. We choose cost function as criterion. We also deduced the update formula of separation matrix. This technique is suitable for convolutional mixture model in some real life systems. Simulations demonstrate the effectiveness of the proposed approach.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under grant no. 60572070 to Zhang Tieyan. This work is also supported in part by Open Project Foundation of Key Laboratory of Process Industry Automation, Ministry of Education China under grant no. PAL200503, and the China Postdoctoral Science Foundation under grant no. 20060400962, Natural Science Foundation of Liaoning Province to Feng Jian.

References

1. Papadias C.B., Paulraj A.: A Constant Modulus Algorithm for Multi-user Signal Separation in Presence of Delay Spread Using Antenna Arrays. *IEEE Signal Processing Letter* **4** (1997) 178–181
2. Veen A.J., Talvar S., Paulraj A.: A Subspace Approach to Blind Space-time Signal Processing for Wireless Communication Systems. *IEEE Trans. Signal Processing* **45** (1997) 173–190
3. Makeig S., Bell A., Jung T.P., et al.: Independent Component Analysis in Electroencephalographic Data. *Advances in Neural Information Processing System* **8** (1996) 145 ~ 151
4. Li Zh. N., He Y.Y., Chu F. L.: Application of the Blind Source Separation in Machine Fault Diagnosis: A review and prospect. *Mechanical Systems and Signal Processing* in press
5. Herault J., Jutten C.: Space and Time Adaptive Signal Processing by Neural Network Models. *Proceedings of International Conference on Neural Network for Computing*, Snowbird, USA, July 1986, 206–211
6. Weinstein E., Feder M., Oppenheim A.V.: Multi-channel Signal Separation by Decorrelation. *IEEE Trans. Speech Audio Process* **4** (1993) 405–413
7. Wu H.C., Principe J. C.: A Unifying Criterion for Blind Source Separation and Decorrelation: Simultaneous Diagonalization of Correlation Matrices. *Proceedings of the 1997 IEEE Workshop on Neural Networks for Signal processing*, Amelia Island, Sept. USA, 1997, 496-505
8. Amari S.: Natural Gradient Works Efficiently in Learning. *Neural Computations* **10** (1998) 251-276

Blind Source Separation in Post-nonlinear Mixtures Using Natural Gradient Descent and Particle Swarm Optimization Algorithm

Kai Song, Mingli Ding, Qi Wang, and Wenhui Liu

Department of Automatic Test and Control
Harbin Institute of Technology, Harbin, 150001, China
{Kaisong, dingml}@hit.edu.cn

Abstract. Extracting independent source signals from their nonlinear mixtures is a very important issue in many realistic models. This paper proposes a new method for solving nonlinear blind source separation (NBSS) problems by exploiting particle swarm optimization (PSO) algorithm and natural gradient descent. First, we address the problem of separation of mutually independent sources in post-nonlinear mixtures. The natural gradient descent is used to estimate the separation matrix. Then we define the mutual information between output signals as the fitness function of PSO. The mutual information is used to measure the statistical dependence of the outputs of the demixing system. PSO can rapidly obtain the globally optimal coefficients of the higher order polynomial functions. Compared to conventional NBSS approaches, the main characteristics of this method are its simplicity, the rapid convergence and high accuracy. In particular, it is robust against local minima in search for inverse functions. Experiments are discussed to demonstrate these results.

1 Introduction

Nowadays BSS has been widely studied because it has potential applicability in many areas such as medical data processing, speech recognition and radar signal communication [1], [2]. In the blind source separation, the source signals and the parameter of transfer channel are unknown, by using only observed signals, the unknown source signals can be separated and estimated.

So far the theory of BSS has been successfully approached in linear instantaneous mixing model. Various algorithms have been proposed, for example, Independent Component Analysis (ICA), Principle Component Analysis (PCA), high-order statistical cumulants and others [3], [4], [5].

Nonlinear blind source separation (NBSS) is a much more recent research topic since the mixing models in actual cases are usually nonlinear [6], [7]. Nevertheless, few methods mention both the accuracy and convergence velocity in the NBSS. Yang and Amari [7] proposed an information back propagation algorithm (BP) for training network parameters by a two-layer perceptron. One drawback of this method is slow convergence rate yield from the highly nonlinear relationship between the output and

learning weights of the network. An approach using RBF network proposed by Tan and Wang [8] can recover source signals and the convergence rate is very fast. However, this method degrades greatly when high-level noise is present with nonlinear distortion. In [9], Bayesian ensemble learning algorithm is proposed for a post-nonlinear mixture. Genetic algorithms are introduced in [10], [11], which minimize nonlinear mixing degree of signals using GA to achieve blind signal separation. However, in the post-nonlinear (PNL) mixtures, the optimized functions which are the invertible functions of nonlinear mixture functions usually have high dimensions and many local minima, so that it is difficult to achieve global optimization.

This paper mainly researches on a blind source separation method of nonlinear mixed signals, which is based on particle swarm optimization (PSO) algorithm and natural gradient descent. The natural gradient descent method is applied to estimate the separation matrix. PSO which enhances the global search velocity and search ability is used to obtain the globally optimal higher-order polynomial coefficients.

2 Nonlinear Blind Source Separation

The nonlinear mixing model is described by PNL mixtures. $x(t) = f[A s(t)]$, where $x(t) = [x_1(t), x_2(t), \dots, x_n(t)]^T$ is an n dimension mixed signal vector, $s(t) = [s_1(t), s_2(t), \dots, s_n(t)]^T$ is an n dimension unknown source signal vector, A is an $n \times n$ unknown full-rank matrix, and $f = [f_1, f_2, \dots, f_n]^T$ is the set of unknown invertible nonlinear mixture functions. It is supposed that the source signals $s_1(t), s_2(t), \dots, s_n(t)$ are non-Gaussian and mutually statistically independent. Generally speaking, their means are zero, and their variances equal unity. Mixed signals are described as:

$$x_i(t) = f_i \left(\sum_{j=1}^n a_{ij} s_j(t) \right) \tag{1}$$

where $i = 1, 2, \dots, n$.

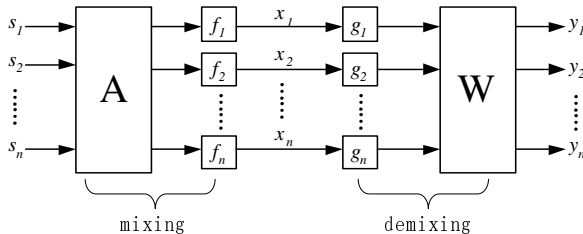


Fig. 1. Nonlinear mixing and demixing process

Fig.1. shows that the separation model is a demixing process. First we need to approximate the inverse function f_i^{-1} of each transfer channel, transform the nonlinear mixing model into a linear model, and then apply the natural gradient descent method

to estimate the separation matrix w , finally achieve the estimations y_i of source signals.

$$y_i(t) = \sum_{j=1}^n w_{ij} g_j(x_j(t)) \tag{2}$$

The inverse function f_i^{-1} may be approximated by higher-order odd function polynomials g_j :

$$g_j(x_j) = \sum_{k=1}^p g_{jk} x_j^{2k-1} \tag{3}$$

where g_{jk} is a parameter to be determined by PSO.

The detailed steps of NBSS are expressed as follows. Firstly we use a natural gradient rule to compute separation matrix w [12]:

$$\Delta W = -\frac{\partial I(y_1, \dots, y_n)}{\partial W} W^T W = \eta [I - \varphi(y) y^T] W \tag{4}$$

where I is a unit matrix, η denotes study ratio, $y = [y_1, \dots, y_n]^T$ is an n dimension output signal vector. $\varphi(y) = [\varphi_1(y_1), \dots, \varphi_n(y_n)]$ denotes the activation function [6]. Its each element $\varphi_i(y_i)$ is described as [6]:

$$\begin{aligned} \varphi_i(y_i) &= F_1(k_3^i, k_4^i) y_i^2 + F_2(k_3^i, k_4^i) y_i^3 \\ F_1(k_3^i, k_4^i) &= \frac{9}{4} k_3^i k_4^i - \frac{1}{2} k_3^i \\ F_2(k_3^i, k_4^i) &= \frac{3}{2} (k_3^i)^2 + \frac{3}{4} (k_4^i)^2 - \frac{1}{6} k_4^i \end{aligned} \tag{5}$$

where $k_3^i = E(y_i^3), k_4^i = E(y_i^4) - 3$.

Then the mutual information is applied to define fitness function and the mutual information between y_i is described:

$$I(y_1, \dots, y_n) = -H(y_1, \dots, y_n) + \sum_{i=1}^n H(y_i) \tag{6}$$

Values near to zero of mutual information between y_i imply independence between output signals being statistically independent [6], [10], [11]. According to differential entropy relation, namely

$$H(y_1, \dots, y_n) = \log |\det(W)| + \sum_{j=1}^n E[\log |g_j'(x_j)|] \tag{7}$$

where $E\{.\}$ denotes the expression operator.

Using some algebra, and suppose that $g_{j1} = 1$ (this only scales the polynomial), we can get

$$\log |g_j'(x_j)| \approx \sum_{k=1}^p g_{jk} (2k-1) x_j^{2k-2} \tag{8}$$

Then substituting (7), (8) into (6), we can get the mutual information between y_i :

$$I(y) = -\log |\det(W)| - \sum_{i=1}^n E[\sum_{k=1}^p (2k-1)g_{ik}x_i^{2k-2}] + \sum_{i=1}^n H(y_i) \tag{9}$$

In (9), each marginal entropy $H(y_i)$ is calculated by the Gram-Charlier expansion, namely

$$H(y_i) \approx \frac{\log(2\pi e)}{2} - \frac{(k_3^i)^2}{2 \cdot 3!} - \frac{(k_4^i)^2}{2 \cdot 4!} + \frac{3}{8}(k_3^i)^2 k_4^i + \frac{1}{16}(k_4^i)^3 \tag{10}$$

where $k_3^i = E(y_i^3), k_4^i = E(y_i^4) - 3$.

Finally the fitness function that we compute in particle swarm optimization algorithm will be mutual information between output signals y_i :

$$fitness_function(y) = I(y) \tag{11}$$

3 Particle Swarm Optimization Algorithm

Particle swarm optimization (PSO) is an evolutionary computation technique, introduced by Kennedy and Eberhart [13], which developed out of work simulating the movement of flocks of birds. In past several years, PSO has been successfully applied in many research and application areas [14].

Each particle is treated as a volumeless particle in n-dimensional search space. The position and the velocity of the i th particle in the n-dimensional search space is represented as $x_i = (x_{i1}, x_{i2}, \dots, x_{in})$ and $v_i = (v_{i1}, v_{i2}, \dots, v_{in})$ respectively. The best position (*pbest*) of the i th particle is recorded and represented as $p_i = (p_{i1}, p_{i2}, \dots, p_{in})$. The global best particle (*gbest*) is denoted by $p_g = (p_{g1}, p_{g2}, \dots, p_{gn})$, which represents the best particle found so far at time t in the entire swarm. The modified velocity and position of each particle can be calculated using the current velocity and the distance from p_{ij} to p_{gj} , as follows:

$$v_{ij}(t+1) = w \cdot v_{ij}(t) + c_1 r_1 (p_{ij}(t) - x_{ij}(t)) + c_2 r_2 (p_{gj}(t) - x_{ij}(t)) \tag{12}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \quad (i = 1, 2, \dots, l, j = 1, 2, \dots, n) \tag{13}$$

where l is number of particles in a group. m is number of members in a particle. t is pointer of iterations (generations). c_1 and c_2 are constants respectively called social and cognitive coefficients distributed in the range of $[0, 2]$. w is called the inertia weight. r_1 and r_2 are two independent random numbers uniformly distributed in the range of $[0, 1]$.

In (12), the inertial weight w provides the necessary diversity to the swarm by changing the momentum of particles and hence avoids the stagnation of particles at local optima. The setting for the value of w is $w = w_{max} - t \cdot \frac{w_{max} - w_{min}}{t_{max}}$. Where t_{max} is the maximum iteration number.

The detailed procedure of PSO is described as follows:

- Step 1 Initialize a population of particles with random positions and velocities.
- Step 2 Evaluate the fitness value of each particle.
- Step 3 For each particle, compare its current fitness value with the fitness value of its previous best position ($pbest$). If current fitness value is better, then update $pbest$ and its fitness value with the current position and fitness value.
- Step 4 Determine the global best particle of current swarm with the best fitness value. If the fitness value is better than the fitness value of global best position ($gbest$), then update $gbest$ and its fitness value with the position and fitness value of the current best particle.
- Step 5 Update the velocity and position of each particle according to (12) and (13).
- Step 6 If a predefined stopping criterion is met, then output $gbest$ and its fitness value, otherwise go back to Step 2.

Hence the global optimum solution (minimum) of the swarm is obtained, so are the polynomial coefficients.

4 Simulation Results

In this section we discuss simulation and experiment results of the proposed algorithm. To verify the validity and performance of the proposed algorithm, several computer simulations are conducted to test the PSO-based method to blind separation of independent sources from their post-nonlinear mixtures.

4.1 Example 1

Consider the mixture of two independent random source signals: a sinusoidal signal $\sin(2\pi)$ and a random noise signal $\text{rand} [(1 N)]$ uniformly distributed in interval $[-1, 1]$. There are 200 samples from each signal ($N=200$). The mixture matrix is chosen randomly as $A = \begin{pmatrix} 0.7373 & -0.5177 \\ 0.2582 & 0.9562 \end{pmatrix}$. The nonlinear transfer functions are chosen randomly as $f_1(x) = \tanh(x)$, $f_2(x) = \tanh(0.8x)$.

Fig.2 and 3 show source signals and their mixtures. The separated signals which are the estimations of source signals are obtained by a demixing model. Our goal is to

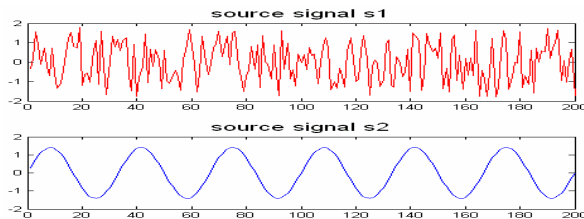


Fig. 2. Source signals

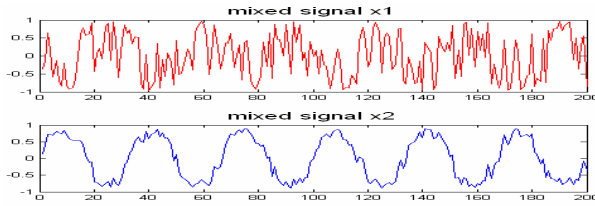


Fig. 3. Mixed signals

obtain the separation matrix W and the higher-order odd function polynomial g_j which is the approximation of f_i^{-1} . Then we transform the nonlinear mixing model into a linear model to extract independent source signals. The detailed process is described as follows:

First, according to natural gradient descent method in (4), (5), we obtain separation matrix W :

$$W = \begin{pmatrix} 0.26142 & 0.14159 \\ -0.074525 & 0.21158 \end{pmatrix}$$

The iteration curves of separation matrix coefficients are shown in Fig.4. Then we define the mutual information between output signals as the fitness function of PSO. The mutual information is used to measure the statistical dependence of the outputs of the demixing system. Values near to zero of mutual information imply independence between output signals being statistically independent. So we utilize (9), (10), and (11) to calculate the minimum of the fitness function of PSO. The inverse function f_i^{-1} may be approximated by higher-order odd function polynomial g_j whose optimal parameters g_{jk} are obtained by PSO. g_{jk} denote the coordinate of the global best particle of current swarm with the best fitness value (minimum).

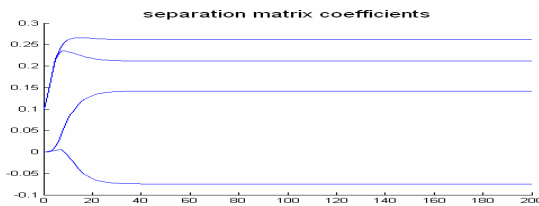


Fig. 4. Separation matrix coefficients using natural gradient descent method

The parameters of PSO are summarized as follows: The population size of the swarm is 20. The number of iterations is 40. The order of odd function polynomial is defined as 5, namely $k=1, 2, 3$. Using some algebra, we suppose that $g_{j1}=1$ (this only predigests the polynomial). So the dimension of search space is selected as 4. Constants $c_1=c_2=2$. r_1 and r_2 are two independent random numbers uniformly

distributed in the range of [0, 1]. The inertia weight w is reduced linearly from 0.9 to 0.4 during the search. At last, we obtain the optimal parameters g_{jk} by PSO, which are shown in Table 1.

Table 1. The polynomial parameters obtained by PSO-based method

g_{jk}	x^1	x^3	x^5
$g_1(x)$	1	0.78418	-0.54287
$g_2(x)$	1	0.74199	-0.016257

Fig.5 shows the separated signals obtained with the proposed method (PSO-based). Many simulations have been conducted to produce such results, so Fig.5 reports the performance for one of many runs. It is shown from the Fig.5 that PSO-based method achieves the successful separation of the two source signals from their nonlinear mixtures. However, theoretical results already prove that it is not possible to separate the sources without nonlinear distortion in the general case [6]. Therefore, in order to evaluate and compare the performance of BSS, the source to distortion ratio (SDR) is used to verify the similarity between the source signals s_i and separated signals y_i with N samples. Now, we defined SDR as

$$SDR_i(t) = SDR(y_i(t), s_i(t)) = 10 \log \left(\frac{\sum_{t=1}^N [s_i(t)]^2}{\sum_{t=1}^N [y_i(t) - s_i(t)]^2} \right) \tag{14}$$

where the bigger $SDR_i(t)$ is, the better the effect of separated signals is. Each source to distortion ratio (SDR) is calculated according to (14). $SDR_1 = 14.245 \text{ dB}$, $SDR_2 = 14.567 \text{ dB}$.

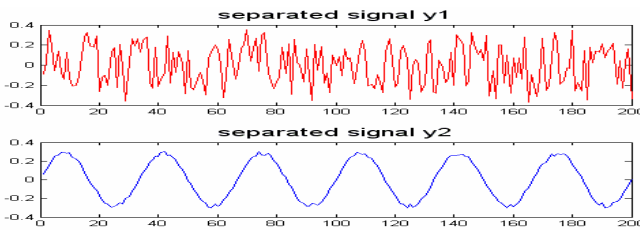


Fig. 5. Separated signals by PSO-based method

Apart from the PSO-based method, we also give experimental results of GA-based method for comparison. In the GA-based method, the odd function polynomial coefficient g_{jk} is obtained by GA. The parameters of GA are summarized as follows: The encoding is generated as 20-bit binary string. The population size is $N = 20$. The number of generations is $Generations = 40$. We use roulette selection and single point

crossover. The crossover rate $P_c = 0.8$ and mutation rate $P_m = 0.04$. These values were selected because of their better performance when compared with other combinations that were evaluated as well. In order to compare the performances of the two methods, we track the evolution curve of GA and the search process of PSO in Fig.6. In Fig.7 we show the separated signals obtained by the GA-based method.

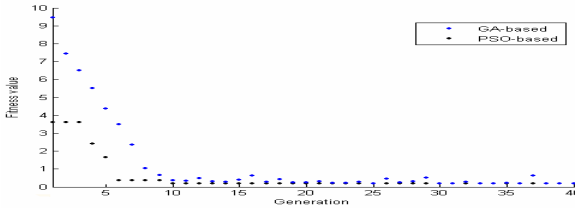


Fig. 6. Comparison of the evolution and search process of the two methods

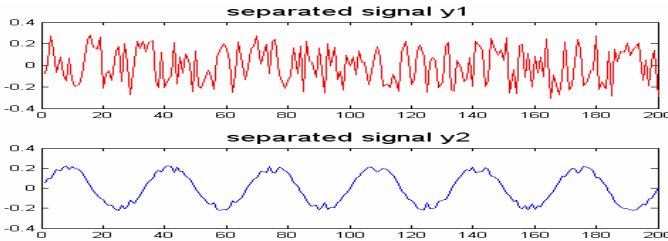


Fig. 7. Separated signals by GA-based method

It is obvious that PSO-based method has much better performance and faster convergence velocity to optimal solution than the traditional GA-based method. In the search and evolution process, GA need set some key parameters such as population size, probability of mutation, probability of crossover, population initialization, etc. If these parameters were not presettted suitable, efficiency of GA would lower. Furthermore, during calculation, GA need convert parameters from solution space to genetic space and from genetic space to solution space several times. This translation brings additional time-consumption. PSO does not need genetic operation such as crossover and mutation, so it is fast to obtain the global optimization and available to achieve real-time blind source separation.

4.2 Example 2

To further test the practical applicability of the proposed method, we consider a “cocktail party” problem. The signals match up with two different persons saying the word “hello” in English respectively. We obtain the speech signals from Brain Science Institute RIKEN (www.bsp.brain.riken.jp). These two different speech signals are mixed by randomly mixture matrix and randomly inverse nonlinear function which are as same as Example 1. All parameters of PSO are also as same as example 1. Figs.8 shows the original speeches. Fig.9 shows their mixtures.

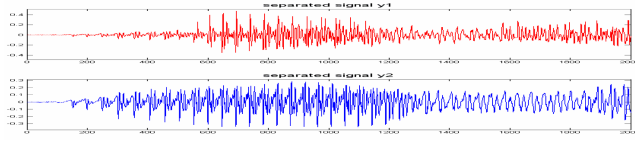


Fig. 8. Original speeches S corresponding to two different persons saying the word hello

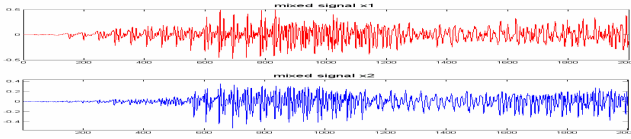


Fig. 9. Mixed signals X after a post-nonlinear mixture

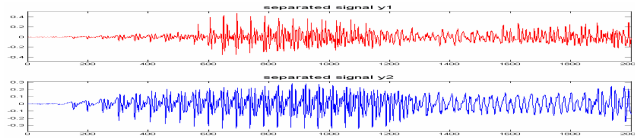


Fig. 10. Separated speech signals y by PSO-based method

Table 2. Performance comparison of the two algorithms

Algorithm type	Population size	Iteration Number	$\frac{1}{n} \sum_{i=1}^n SDR_i$	Time of convergence for NBSS
PNL mixtures of a sine signal and random noise signal				
PSO-based	20	40	14.406 dB	625 ms
GA-based	20	40	8.109 dB	2432 ms
PNL mixtures of two speech signals				
PSO-based	20	40	17.775 dB	1828 ms
GA-based	20	40	12.803 dB	7242 ms

Fig.10 shows the separated results by the proposed PSO-based method for one of the simulations. As can be seen, the separated signals are very similar to the source signals, up to possible scaling factors and permutations of the sources (e.g., separated signal y_2 match up with source signal s_1 , but it changes also in amplitude. Separated signal y_1 match up with source signal s_2 , but it also overturns its amplitude). To save space, the separated results by GA-based method are not shown due to its somewhat similarity to that in Example 1. At last, some conclusions such as SDR and time of convergence for NBSS can be drawn from the experimental performance results of the two approaches shown in Table 2.

5 Conclusions

A nonlinear blind source separation method using natural gradient descent and particle swarm optimization algorithm has been proposed in this paper. Simulation results show that the proposed method can obtain clearer estimations of source signals from their PNL mixtures. The traditional approaches to NBSS for example GA have the drawback that it may not achieve the separation accuracy and convergence velocity simultaneously. The proposed method enhances both the global search velocity and local search ability in search for inverse function.

References

1. Cichocki, A., Amari, S.I.: Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications. John Wiley and Sons, New York (2002)
2. Li, Y., Cichocki, A., Amari, S.: Blind Estimation of Channel Parameters and Source Components for EEG Signals: A Sparse Factorization Approach. *IEEE Transactions on Neural Networks* **17** (2) (2006) 419-431
3. Hyvärinen, A., Oja, E.: Independent Component Analysis: Algorithms and Applications. *Neural Networks* **13** (4-5) (2000) 411-430
4. Zhang, L., Amari, S., Cichocki, A.: Equi-convergence Algorithm for Blind Separation of Sources with Arbitrary Distributions. *Lecture Notes in Computer Science*, **2085** Springer-Verlag (2001) 826-833
5. Bell, A., Sejnowski, T.J.: An Information-maximization Approach to Blind Separation and Blind Deconvolution. *Neural Computation* (1995) 1129-1159
6. Taleb, A., Jutten, C.: Source Separation in Post-nonlinear Mixtures. *IEEE Transactions on Signal Processing* **47** (10) (1999) 2807-2820
7. Yang, H.H., Amari, S., Cichocki, A.: Information-theoretic Approach to Blind Separation of Sources in Nonlinear Mixture. *Signal Processing* **64** (1998) 291-300
8. Tan, Y., Wang, J., Zurada, J.M.: Nonlinear Blind Source Separation using a Radial Basis Function Network. *IEEE Transactions on Neural Networks* **12** (2001) 124-134
9. Jutten, C., Karhunen, J.: Advances in Blind Source Separation (BSS) and Independent Component Analysis (ICA) for Nonlinear Mixtures. *Int. J. of Neural Systems*, **14** (5) (2004) 267-292
10. Tan, Y., Wang, J.: Nonlinear Blind Source Separation using Higher Order Statistics and A Genetic Algorithm. *IEEE Trans. Evol. Comput.* **5** (2001) 600-612
11. Rojas, F. (ed.): Blind Source Separation in Post-Nonlinear Mixtures Using Competitive Learning, Simulated Annealing, and a Genetic Algorithm. *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews* **34** (4) (2004) 407-416
12. Amari, S.I.: Natural Gradient Works Efficiently in Learning. *Neural Computation* **10** (1998) 251-276
13. Clerc, M., Kennedy, J.: The Particle Swarm: Explosion, Stability, and Convergence in A Multi-dimensional Complex Space. *IEEE Trans. Evol. Comput.* **6** (2002) 58-73
14. Abraham, A., Grosan, C., Ramos, V.: Swarm Intelligence and Data Mining. *Studies in Computational Intelligence*. Springer-Verlag, Germany (2006)

Echo State Networks for Real-Time Audio Applications

Stefano Squartini, Stefania Cecchi, Michele Rossini, and Francesco Piazza

A3Lab, DEIT, Università Politecnica delle Marche,
Via Breccie Bianche 31, 60131 Ancona, Italy
sts@deit.univpm.it
a3lab.deit.univpm.it

Abstract. This paper deals with the employment of Echo State Networks for identification of nonlinear dynamical systems in the digital audio field. The real contribution of the work is that such networks have been implemented and run in real-time on a specific PC based software platform for the first time, up to the authors knowledge. The nonlinear dynamical systems to be identified in the audio applications here addressed are the mathematical model of a commercial Valve Amplifier and the low-frequency response of a loud-speaker. Experimental results have shown that, at a certain frequency sampling rate, the ESNs considered (after the training procedure performed off-line) are able to tackle the real-time tasks successfully.

1 Introduction

Neural Networks [1] have been extensively employed in the literature to face different problems in several application fields, likely related to the Digital Signal Processing (DSP) area. Numerous distinct architectures and learning algorithms have been proposed on purpose, in dependence on the task under study. In particular, it often happens that the learning system is asked to have dynamical mapping capabilities, i.e. the ability of storing and updating context information occurring at arbitrarily distant time instants. Common static networks, as the FeedForward Neural Networks (FFNN), are not well-suited to tackle the problem and typically the focus is directed to the Recurrent Neural Networks (RNN) because they have an internal state that can represent context information. Gradient based algorithms are widely used for their simplicity and low computational cost as learning algorithms: back-propagation through time (BPTT) and real-time recurrent learning (RTRL) are well-known examples [1].

However, those types of algorithms have shown to be not sufficiently powerful to discover contingencies spanning long temporal distances. Indeed, as a consequence of the vanishing gradient effect [1, 2], either the system gets information latching being resistant to noise or, alternatively, it is efficiently trainable by gradient descent learning algorithm, but not both. Several solutions have been proposed to mitigate this effect [2], as the Recurrent Multiscale Architecture (RMN) [3], which significantly reduces the impact of the vanishing gradient

even maintaining the usage of BPTT algorithm. However, the Echo State Network (ESN) [4], [5], recently appeared in the literature, seems to be the most effective solution from this perspective. Indeed the approach followed in ESNs consists in providing a large set of basis functions through a network of fixed recurrent connections and in combining them through a static linear or nonlinear adaptive mapper for optimal input representation. This allows dealing with dynamical properties of the input time series avoiding training the network feedback synapses, and so resulting in a strongly simplified learning procedure with immunity to the vanishing gradient effect. Such a property has been experimentally verified by some of the authors in a recent paper [3], by comparing ESNs, RMNs and common globally RNNs performances when applied to a specific benchmark.

ESNs properties have been also tested in the literature on more complicated and realistic tasks, as identification of NARMA systems [6], neural activity mapping [7], mobile robot modeling and control [8], Q-function modeling in reinforcement learning [9], speech recognition [10]. However, up to author's knowledge this work represents the first effort to evaluate their capabilities in real-world audio tasks, where we can experience nonlinear and dynamical systems to identify, taking also real-time constraints into account. Here, the modeling of a commercial Valve Amplifier and the identification of a loud-speaker low-frequency response are the audio applications addressed and ESNs have been employed for their fulfillment. Once performed the training procedure offline, we implemented the adapted networks on the Nu-Tech framework, a suitable SW platform for real-time audio processing directly on the PC hardware. As expected, the related real-time constraints result in some restrictions on the network parametrization, which, however, does not affect the effectiveness of the approach in the tasks under study, as shown by the computer simulations carried out.

2 Echo State Networks

The basic working principle of ESNs is that, under certain conditions, its activation state $\mathbf{x}(n)$ is a function of past input values $\mathbf{u}(n)$, $\mathbf{u}(n-1)$, ..., so it can be interpreted as an echo of the input history. Let us introduce an adequate terminology to describe such a kind of network. It has K input lines, N internal neurons and L output units. There are four types of synaptic weights: input, internal, output, output-internal. They are described by the corresponding weight matrices W^{in} , W , W^{out} , W^{back} , whose dimensions are respectively $N \times K$, $N \times N$, $L \times (K + N + L)$, $N \times L$. Connections between input and output lines and among output units are allowed. There are no specific assumptions on the topology of internal neural block, namely reservoir; in particular we are not constrained to consider a layer architecture. However it is expected that the internal connections form recurrent paths in order to have a state space behavior. The block diagram of an ESN is depicted in Fig 1. The activation state of the reservoir is given by:

$$x(n+1) = f(W^{in}u(n+1) + Wx(n) + W^{back}y(n)), \quad (1)$$

where $f = (f_1, \dots, f_N)$ are the activation functions of the internal units (usually sigmoidal). The out-put equation is:

$$y(n + 1) = f^{out}(W^{out}(u(n + 1), x(n + 1), y(n))), \tag{2}$$

where $f^{out} = (f_1^{out}, \dots, f_L^{out})$ are the activation functions of the output units (usually sigmoidal). In other words, it can be said that the echo functions are the basis functions that the output static mapper has to select for an optimal input representation. Therefore, the Echo State Property has to be satisfied. If we want the state to depend on the past inputs W^{back} must be neglected first. Then, as shown in [4], [5], a sufficient condition is contractivity of W . Nevertheless a weaker operative condition holds in practice: the spectral radius $|\lambda_{max}|$ of W is less than unity. Sparseness and randomness of W connections are two important requirements to have sufficiently rich dynamics, for the final network to yield the desired mapping. Concerning the learning algorithms, it must be underlined that the reservoir weights are fixed. This allows getting a relevant simplification of the adaptation process, since we do not have to worry about adapting the recurrent connections, the main reason of vanishing gradient occurrence in gradient based algorithms. The only part of ESN subject to learning is the static mapper, for which we can use methods developed in the literature for static NNs. In particular, if the output lines have no feedback weights and the related nonlinearities are invertible, linear regression algorithms might be employed, avoiding iterative procedure based on gradient calculation. According to these assumption, and neglecting the direct input-output synapses, (2) becomes

$$\tilde{y}(n) = (f^{out})^{-1}(y(n)) = W^{out} \mathbf{x}(n), \tag{3}$$

where $\mathbf{x}(n)$ is the state vector. If we consider a training observation range equal to $[1 \cdot \dots \cdot T_{tr},]$, (3) becomes:

$$\tilde{\mathbf{y}} = W^{out} \mathbf{X}. \tag{4}$$

By applying Singular Value Decomposition (SVD) we can obtain the optimal W^{out} in terms of the available observations.

3 Implementation Issues

A graphical tool for dealing with ESNs has been developed in C++ (Fig.2), and it can be basically seen as composed by three different parts. The first is related to the determination of the Echo State Network , i.e. the number of internal units, the spectral radius, the internal matrix connectivity, the output-internal weight presence and the type of activation function. The second part refers to the training algorithm that could be based on the gradient based algorithms (like the conjugate gradient), or, as aforementioned, on the linear regression approach, performed through the Singular Value Decomposition (SVD).

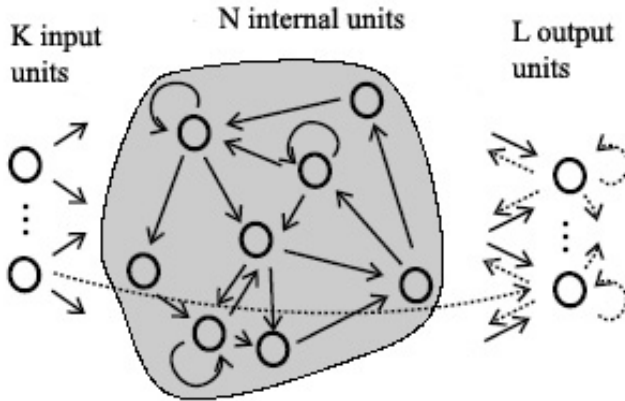


Fig. 1. Echo State Network block diagram: the input line, the reservoir and the output line (static mapper).

The last part gives out the values of the ESN parameters in different operating conditions (initialization, training, generalization). In the learning phase, beginning from the input and target signal, it is important to define the samples number requested to improve the forgetting time of the starting state. In order to have satisfying performances, the samples number should be greater than the network dimensions and the spectral radius. Furthermore it is possible to add white noise in order to avoid instability problems and generally improve the achievable results. At the end of the training phase, the output weight matrix and the correlation matrix of the internal states could be displayed and analyzed to evaluate the generalization performances of the network. It must be said, that in all computer simulations performed, the linear regression method has been employed.

Once trained, the ESN can be suitably saved in a proper format and then used for real time applications. This has been accomplished through the Nu-Tech Platform [11]. This software allows to implement and test real time DSP algorithms in multi-channel scenarios: the Nu-Tech framework is basically composed by two elements, i.e. the interface to the PC sound card and the PlugIn architecture. The former allows handling the audio streams (frame-by-frame) from the I/O sound card channels also through an accurate management of the latency times. The latter lets the user develop his own C/C++ algorithms within the graphical routing scheme reproducing the sound-card MIMO structure.

In our case study, an ESN Nu-Tech PlugIn has been realized as a standard C++ *dll* file able to operate within the Nu-Tech interface (Fig. 3). Such a PlugIn can process the audio streaming according to the parametrization related to the trained Echo State Network contained in the proper file coming from the aforementioned C++ based tool. It must be remarked that the combination of the graphical tool and the Nu-Tech framework presents significant pros from

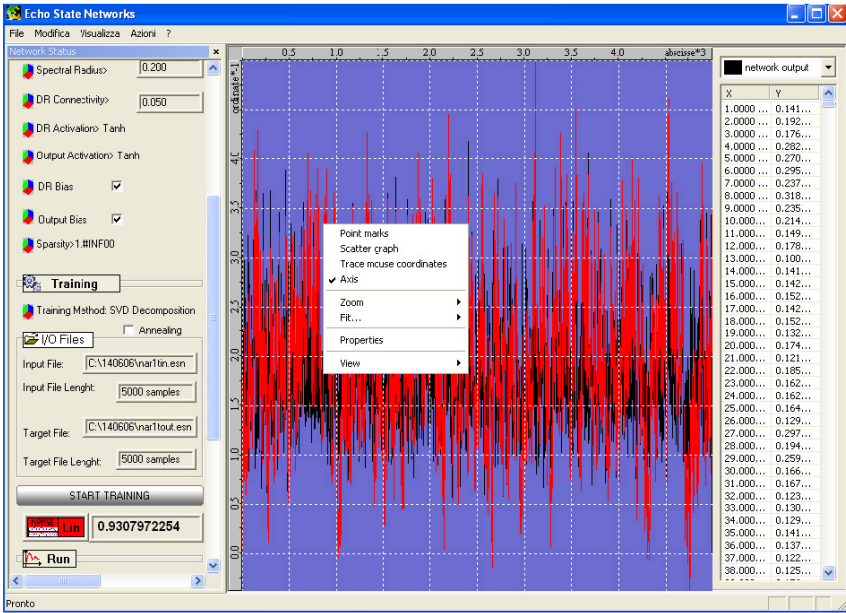


Fig. 2. Graphical tool for ESN generation, initialization, training and testing

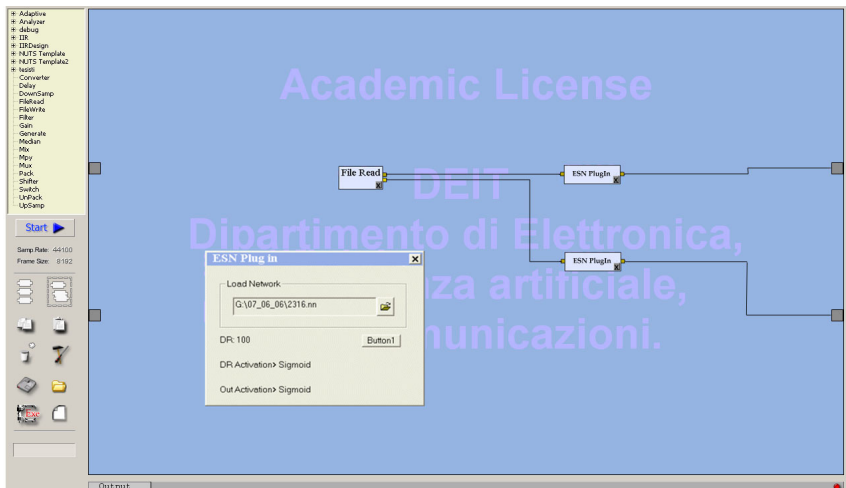


Fig. 3. Nu-Tech Platform with ESN PlugIn

a pure technological point of view: indeed we can easily adapt and run suitable ESNs for real-time applications by just dealing with the available friendly user-interfaces.

Looking at the real-time processing constraints, it must be said that they induce some restrictions on the ESN parametrization allowed. Looking at them from the perspective of the applications described in the following, we can say that the activation function of the internal units must be a sigmoidal function in order to have a lower computational cost and the maximum limit of dynamic reservoir dimension must be 200 units with a connectivity factor of 2.5% to avoid clicks during streaming.

4 Experimental Results

In this section some experimental results related to the field of audio processing will be presented and analyzed. The Echo State Networks have been created, initialized, and trained by using the C++ based graphical tool. Then, for the generalization phase mono wave files (sampled at 44100 Hz) have been used as the inputs feeding the trained networks running in real-time on the Nu-Tech platform. To evaluate the algorithm performances, a normalized mean square error has been defined as follows

$$NRMSE = \sqrt{\frac{\sum_{i=1}^T (y[i] - d[i])^2}{T \cdot \sigma^2}}, \quad (5)$$

where y is the output of the network, d is the desired response of the system to identify, σ^2 is the target variance and T the observation time range. For each experimental results a table will be shown with the ESN parameters and NRMSE calculated for the training and testing phases.

4.1 Linear Dynamical Systems

As starting case study, we consider a linear system identification problem. In this case we have considered the behavior of a signal filtered by a FIR filter of order M . To simulate the filter behavior, the Echo State Network has to calculate $M+1$ parameters storing M past input values. The filter lengths are 50, 80, 100 samples and the dynamical reservoir dimension strictly depends on this. The activation function can be linear or sigmoidal taking into account its linear zone and scaling the input values. The spectral radius is very high (0.97, 0.99) to improve the store capacity of the network. The results are shown in the Table [1](#). As we can see the best results are achieved for 200 internal units with a connectivity of 5.2% and high spectral radius.

4.2 Modelling of a Commercial Valve Amplifier

Almost a century after their introduction, vacuum tube amplifiers are still appreciated for their special sound qualities. It is known indeed that valve amplifiers are highly rated by audiophiles and musicians, and often preferred to their digital counterpart [\[12\]](#). So recently several works have been orientated to

Table 1. Linear system identification experimental results. FO is the filter order, DR is the dimension of the dynamical reservoir, CP is the connectivity percent, SR is the spectral radius, NRMSE_{tr} is the normalized mean square error calculate for the training phase and NRMSE for ESN application.

<i>FO</i>	<i>DR</i>	<i>CP</i>	<i>SR</i>	<i>NRMSE_{tr}</i> <i>mean</i>	<i>NRMSE_{tr}</i> <i>st.dev</i>	<i>NRMSE</i> <i>mean</i>	<i>NRMSE</i> <i>st.dev</i>
50	120	8%	0.97	0.0240	$9.78 \cdot 10^{-4}$	0.048	$9.65 \cdot 10^{-4}$
80	200	5.2%	0.99	0.0054	$1.4 \cdot 10^{-3}$	0.0079	$1.42 \cdot 10^{-3}$
100	330	3.3%	0.99	0.0143	$3.4 \cdot 10^{-3}$	0.0217	$3.2 \cdot 10^{-3}$

the non linear digital modeling of a Tube Amplifier. There are mainly two approaches: the former refers to the application of a mathematical model derived from the study of the equivalent circuit, the latter is based on the characterization of a real Valve Amplifier through non linear identification techniques. In this work, as a term of comparison, we have used a free commercial VST PlugIn [13] that implements the behavior of a real tube amplifier. First of all the training phase has been based on a target signal filtered by the VST PlugIn, then the behavior of the ESN has been tested. The results are shown in Table 2. As we can see, the best results have been achieved using the sigmoidal activation function both for the internal and the output weights. Moreover, the generalization performances do not improve if we increase the size of the dynamical reservoir.

Table 2. Modelling a commercial Valve Amplifier: experimental results. AF is the type of the activation function, DR is the dimension of the dynamical reservoir, CP is the connectivity percent, NRMSE_{tr} is the normalized mean square error calculate for the training phase and NRMSE for ESN application.

<i>AF</i>	<i>DR</i>	<i>CP</i>	<i>bias</i>	<i>NRMSE_{tr}</i> <i>mean</i>	<i>NRMSE_{tr}</i> <i>st.dev</i>	<i>NRMSE</i> <i>mean</i>	<i>NRMSE</i> <i>st.dev</i>
<i>Ell</i>	150	5.2%	<i>Low</i>	0.0022	$5.23 \cdot 10^{-4}$	0.0057	$1.8 \cdot 10^{-4}$
<i>Ell</i>	100	8%	<i>Low</i>	0.0040	$8.54 \cdot 10^{-4}$	0.0069	$1.6 \cdot 10^{-3}$
<i>Ell</i>	100	8%	<i>Med</i>	0.0037	$1.40 \cdot 10^{-3}$	0.0104	$3.3 \cdot 10^{-3}$
<i>Ell</i>	100	8%	<i>High</i>	0.0041	$2.20 \cdot 10^{-3}$	0.0195	$1.2 \cdot 10^{-3}$
<i>Atan</i>	100	8%	<i>Low</i>	0.0116	$4.99 \cdot 10^{-3}$	0.0294	$2.6 \cdot 10^{-3}$
<i>Tanh</i>	100	8%	<i>Low</i>	0.0218	$1.40 \cdot 10^{-3}$	0.1392	$1.09 \cdot 10^1$

4.3 Identification of a Loudspeaker Low-Frequency Response

In the last decade, several efforts have been made to model the non linear response of loudspeaker in order to reduce the non linear distortion especially at low frequency. The principal causes of non linearities in loudspeaker include non linear suspension and non-uniform flux density. The main results refer

to a model derived from an equivalent circuit of a loudspeaker system. In this work we have considered a mathematical model of a Loudspeaker to analyzed its Low Frequency response as in [14]. The loudspeaker has the following parameters:

$$\begin{aligned}
 x(k+1) &= \begin{bmatrix} -0.1 & 0 & -0.2 \\ 0 & 1 & 1 \\ 0.6 & -0.5 & -0.15 \end{bmatrix} x(k) + \begin{bmatrix} 0.4 \\ 0 \\ 0 \end{bmatrix} u(k) \\
 &+ \begin{bmatrix} -0.04x_2(k)x_3(k) - 0.05x_2^2(k)x_3(k) \\ 0 \\ -0.08x_2^3(k) + 0.01x_1(k)x_2(k) + 0.02x_1(k)x_2^2(k) \end{bmatrix},
 \end{aligned}
 \tag{6}$$

$$y(k) = (0 \ 1 \ 0)^T x(k).
 \tag{7}$$

This relation derived from two differential equation associated to the mechanical and electrical equivalent circuit of the loudspeaker taking into account the distortions constraints. In the training phase, noise signal low pass filtered at 1kHz has been used. The reservoir dimension vary from 120 to 200 units with a connectivity of 2-1.5%; the activation functions are sigmoidal while the radius spectrum varies from 0.8 to 0.97. The results are shown in table 3. After the training, the neural network has been tested with a white noise signal and sweep signal (20Hz - 1kHz), played in the wave format within Nu-Tech. Again, the usage of sigmoidal activation functions both for the internal and the output weights with higher spectral radius allows to achieve the best results. Furthermore increasing the dimension of the dynamical reservoir does not yield better results.

Table 3. Identification of a Loudspeaker Low-Frequency response: experimental results . AF is the type of the activation function, DR is the dimension of the dynamical reservoir, CP is the connectivity percent, rs is the spectral radius, NRMSE_{tr} is the normalized mean square error calculate for the training phase, NRMSE_t for ESN application with noise input and NRMSE_s for ESN application with sweep input.

AF	DR	CP	rs	NRMSE _{tr} mean/std	NRMSE _{tr} mean/std	NRMSE _t mean/std	NRMSE _s mean/std
Ell	120	2%	0.97	0.0049 2.8 10 ⁻³	0.0482 4.9 10 ⁻³	0.0181 9.1 10 ⁻³	2.521 10 ¹⁶ 2.01 10 ¹⁶
Ell	120	2%	0.8	0.0027 6.36 10 ⁻⁴	0.0562 1.6 10 ⁻³	0.0112 4.8 10 ⁻³	7.49 10 ¹⁷ 1.03 10 ¹⁶
Tanh	120	1.8%	0.8	0.00078 1.17 10 ⁻⁴	0.0618 9.23 10 ⁻⁴	0.0067 9.64 10 ⁻⁴	2.63 10 ¹⁷ 3.05 10 ¹⁷
Ell	200	1.5%	0.92	0.0038 3.1 10 ⁻³	0.0579 1.8 10 ⁻³	0.0138 2.2 10 ⁻³	2.653 10 ¹⁵ 3.753 10 ¹⁵

5 Conclusions

In this paper we have faced the problem of implementing the Echo State Networks in the Nu-Tech framework for real-time audio applications. Up to the authors' knowledge this represent the first attempt in this direction, and the achieved results seem to be encouraging: indeed, even though the real-time constraints induce some restrictions on the ESN parametrization, the performed ESNs (once adequately trained off-line) are able to solve the tasks successfully. Deep studies are actually ongoing on the possibility of implementing the training phase also in real-time, paying attention to the further to the ESN parametrization limitations which inevitably arise. Moreover, future work could be done to evaluate the applicability of other types of Neural Networks with memory (as RNNs) and compare them to the ESNs in terms of performances in real-time audio applications.

Acknowledgments. This work was supported by the European Commission as sponsor of the hArtes Project number 035143.

References

1. Haykin, S.: *Neural network - A Comprehensive foundation*. Englewood Cliffs, NJ: Prentice Hall (1999)
2. Hochreiter, S., Bengio, S., Frasconi, P., Schmidhuber, J.: *Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-Term Dependencies*. A Field Guide to Dynamic Recurrent Network, Chapter 7, J.F. Kolen and S.C. Kremer, Eds. IEEE Press (2001)
3. Squartini, S., Paolinelli, S., Piazza, F.: *Comparing Different Recurrent Neural Architectures on a Specific Task from Vanishing Gradient Effect Perspective*. IEEE Int. Conf. Networking, Sensing and Control, Miami, USA (2006) 380-385
4. Jaeger, H.: *The Echo State Approach to Analysing and Training Recurrent Neural Networks*. German National Research Center for Information Technology, Fraunhofer Institute for Autonomous Intelligent Systems, Tech. Rep. (2001) GMD Report 148
5. Jaeger, H.: *Short Term Memory in Echo State Networks*. GMD report 152, German National Research center Information Technology (2002)
6. Jaeger, H.: *Adaptive Nonlinear System Identification with Echo State Networks*. Advances in Neural Information Processing Systems, 2002, Becker, S., Thrun, S., Obermayer, K. Cambridge, MA: MIT Press (2003) 593-600
7. Rao, Y.N., Kim, S.P., Sanchez, J.C., Erdogmus, D., Principe, J.C., Carmena, J.M., Lebedev, M.A., Nicolelis, M.A.: *Learning Mappings in Brain Machine Interfaces with Echo State Networks*. IEEE Int. Conf. Acoustics, Speech and Signal Processing **5** (2005) 233-236
8. Ploger, P.G. , Arghir, A., Gunther, T., Hosseiny, R.: *Echo State Networks for Mobile Robot Modeling and Control*. Lecture Notes in Artificial Intelligence of the 7th Robot Soccer World Cup **3020** (2003)
9. Bush, K., Anderson, C.: *Modeling Reward Functions for Incomplete State Representations via Echo State Networks*. Proc. Int. Joint Conf. Neural Networks, Montreal, Canada (2005) 2995-3000

10. Skowronski, M.D., Harris, J.G.: Minimum Mean Squared Error Time Series Classification using an Echo State Network Prediction Model. *IEEE Int. Symp. Circuits and Systems*, Island of Kos, Greece (2006) 3153-3156
11. Squartini, S., Ciavattini, E., Lattanzi, A., Zallocco, D., Bettarelli, F., Piazza, F.: NU-Tech: Implementing DSP Algorithms in a Plug-in based Software Platform for Real Time Audio applications. Presented at the 118th AES Convention, Barcelona, Spain (2005)
12. Van der Veen, M.: Universal System and Output Transformer for Valve Amplifiers. Presented at the 118th AES Convention, Barcelona, Spain (2005)
13. <http://www.voxengo.com/product/tubeamp/> Voxengo Tube Amplifier VST PlugIn.
14. Gao, F.X.Y., Snelgrove, W.M.: Adaptive Linearization of a Loudspeaker. *Int. Conf. Acoustics, Speech and Signal Processing* **5** (1991) 3589-3592

Blind Separation of Positive Signals by Using Genetic Algorithm

Mao Ye^{1,2}, Zengan Gao², and Xue Li³

¹ School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, Sichuan 610054, P.R. China

yem_mei29@hotmail.com

² Institute of Knowledge Management and Business Intelligence, School of Economics and Management, Southwest Jiaotong University, Chengdu, Sichuan 610031, P.R. China

³ School of Information Technology and Electronic Engineering, The University of Queensland, Brisbane, Queensland 4072, Australia

xueli@itee.uq.edu.au

Abstract. When the source signals are known to be independent, positive and well-grounded which means that they have a non-zero pdf in the region of zero, a few algorithms have been proposed to separate these positive sources. However, in many practical cases, the independent assumption is not always satisfied. In this paper, a new approach is proposed to separate a class of positive sources which are not required to be independent. These source signals can be separated very quickly by using genetic algorithm. The objective function of genetic algorithm is derived from uncorrelated and some special assumptions on such positive source signals. Simulations are employed to illustrate the good performance of our algorithm.

1 Introduction

Blind source separation (BSS) recovers n original source signals from m observed signals from sensors without any prior information of sources and mixing process. The simplest instantaneous BSS model is the following:

$$x(k) = As(k), \quad (1)$$

where $x(k) = [x_1(k), \dots, x_m(k)]^T$ is a $m \times 1$ vector of mixed observations, $s(k) = [s_1(k), \dots, s_n(k)]^T$ represents the samples of the unobserved source signals, and $A \in R^{m \times n}$ ($m \geq n$) is a constant but unknown mixing matrix with full rank. Assume the weight matrix $W \in R^{n \times m}$ and the output $y(k) = Wx(k)$ at time k . The goal of a BSS algorithm is to find a weight matrix W such that $y(k)$ is a permutation of source signals up to a scaling factor.

The problem of blind source separation has been studied by many authors in recent years since the pioneering work by Jutten and Herault [8]. Many BSS algorithms are proposed based on independent component analysis (ICA) models,

the infomax method, and the nonlinear principal component analysis (PCA), etc. (for details, see e.g. [5,7]). Since nonnegative assumption on the sources is very natural for many practical applications (for a review, see e.g. [10,12]), recently, a few authors considered the nonnegative assumption on source signals [10,11,12,9]. If the source signals are positive, independent and well-grounded which means that they have a non-zero pdf in the region of zero, E. Oja and M. Plumbley have proposed a few algorithms to separate these sources [10,11,12]. And the global convergence has been proved in [15]. Although these algorithms have good performance of separation [10], they cannot be applied to nonindependent positive source signals. When the independent assumption is replaced by the following special nonnegative **assumption A**: 'for each source, there is at least one value of the acquisition variable for which this source presents a non-null response, to exclusion of all other sources', Naanaa and Nuzillard proposed a new method to separated these sources by solving a constrained optimization problem [9]. In many applications, the source signals fall into this category, for example, the nuclear magnetic resonance (NMR) spectroscopy signals. However, their method needs to compute a constrained optimization problem for each observed signal. If the number of sampled observed signals is increased, the efficiency of their algorithm will decrease quickly.

In this paper, we propose an effective and efficient BSS algorithm for a class of positive source signals by using evolutionary computation techniques. This class of positive signals are uncorrelated and satisfy assumption A. Moreover, if the mixing matrix is orthonormal, the mixing matrix can be computed by using our algorithm very quickly. Simulations confirm the utility of our algorithm. The techniques used in this paper may give some hints to the research of more general positive source separation.

2 Problem Statements

Suppose we have unobserved source signals $s = [s_1(k), s_2(k), \dots, s_n(k)]^T$ for $k \leq N$ where the real-valued component $s_i(k)$ are positive, uncorrelated, and have bounded non-zero variance. And the source signals satisfy the following condition [9]:

Assumption A. For each $i \in \{1, 2, \dots, n\}$ there exists an $j_i \in \{1, 2, \dots, N\}$ such that $s_i(j_i) > 0$ and $s_k(j_i) = 0$ ($k = 0, \dots, i-1, i+1, \dots, n$).

Without loss of generality, assume the sources have unit variances $\sigma_{s_i}^2 = E\{(s_i(k) - \bar{s}_i)^2\} = 1$. Otherwise, we can scale the variance of source signal to be unit. For example, let the sources are scaled to $s'(k) = Gs(k)$ such that $x(k) = A's'(k)$ with $A' = AG$ where $G = \text{diag}(g_1, \dots, g_n)$ and $g_i = \sigma_{s_i}^{-1}$. The scaled signals $s'(k)$ also satisfy the assumption A as well as the source signals $s(k)$.

In this paper, we will not handle the original observed signals $x(k)$ directly. By performing a pre-whitening step [3,11], the matrix $Q \in R^{n \times m}$ is generated such that the covariance matrix $C_z = E\{(z(k) - \bar{z})(z(k) - \bar{z})^T\} = I_n$ where $z(k) = Qx(k)$ and \bar{z} is the mean of $z(k)$ for $1 \leq k \leq N$. This pre-whitening

method does not change the nonnegativity of the sources. When the number of source signals is unknown, it can be estimated if the observed signals are sufficiently sampled [4].

Since the covariance matrix of source signals $C_s = I_n$, so the covariance matrix of observed signals $x(k)$ is given by $C_x = E\{(x(k) - \bar{x})(x(k) - \bar{x})^T\} = AC_sA^T = AA^T$. Since A is bounded and has rank n , C_x is a positive definite symmetric matrix which can be decomposed as $C_x = U_x\Lambda_xU_x^T$ where $U_x \in R^{n \times n}$ is the eigenvector matrix and Λ_x is the diagonal eigenvalue matrix respectively. Let $Q = RA_x^{-1/2}U_x^T$ where R is any $n \times n$ orthonormal rotation matrix with $RR^T = R^TR = I_n$, then $C_z = QC_xQ^T = I_n$ which means that the square matrix QA is orthonormal.

After the above pre-whitening procedure, equation (1) is changed to the following equation,

$$z(k) = Bs(k), \tag{2}$$

for $1 \leq k \leq N$ and $B = QA$ is an $n \times n$ orthonormal matrix. Suppose $W \in R^{n \times n}$ is an orthonormal matrix with $WW^T = W^TW = I$ and $y(k) = Wz(k) = WBs(k) = Us(k)$ where $U = WB$. The goal of our algorithm is to find an orthonormal matrix such that U is a permutation matrix, i.e. $y(k)$ is a permutation of $s(k)$.

3 The Algorithm

Let us first present a useful lemma which has already been proved in [11].

Lemma 1. *Let $U = [\mu_{ij}]$ be an $n \times n$ orthonormal matrix such that $UU^T = U^TU = I_n$. Then all elements of U are nonnegative if and only if U is a permutation matrix, i.e. a matrix for which, given a sequence $\{j_i | 0 \leq i \leq n\}$ of n distinct integers $0 \leq j_i \leq n$, we have $\mu_{ij} = 1$ if $j = j_i$ and $\mu_{ij} = 0$ otherwise.*

Assume the matrix $B = [B_1, B_2, \dots, B_n]$, equation (2) can be rewritten as the following,

$$z(k) = \sum_{i=1}^n s_i(k)B_i, \tag{3}$$

for $1 \leq k \leq N$. Because the source signals satisfy the Assumption A, each signal s_i there exists at least one time j_i such that $s_i(j_i) > 0$ and $s_k(j_i) = 0$ for $k \neq i$. At time j_i , equation (3) is changed to

$$z(j_i) = s_i(j_i)B_i. \tag{4}$$

This means that each column of B will be collinear with at least one of the observed signals $z(k) (1 \leq k \leq N)$.

Because the matrix B is orthonormal, i.e. $BB^T = I_n$ and $B^TB = I_n$, the problem to estimate the mixing matrix is changed to find n orthogonal vectors

V_1, V_2, \dots, V_n in the normalized observed signals $z(k)$ for $1 \leq k \leq N$ such that the following function is minimized,

$$\Gamma(V) = \|VV^T - I_n\|_F + \|V^T V - I_n\|_F, \tag{5}$$

where $V = [V_1, V_2, \dots, V_n]$ and $\|\cdot\|_F$ is the standard Frobenius matrix norm. Next, we will show the solution matrix V is unique.

Lemma 2. *Assume the source signals $s(k)$ are positive for $1 \leq k \leq N$ and satisfy assumption A, for $z(k) = As(k)$ where $1 \leq k \leq N$, there is only one $n \times n$ orthonormal matrix V up to a permutation matrix.*

Proof. Suppose we have an orthonormal matrix V , the columns of which belong to the normalized observed signal set. For simplicity, suppose the time index of the corresponding observed signals is $[t_1, t_2, \dots, t_n]$, i.e. $V = [z(t_1)/\|z(t_1)\|, z(t_2)/\|z(t_2)\|, \dots, z(t_n)/\|z(t_n)\|]$. It means that

$$V = A\bar{S},$$

where the elements of matrix \bar{S} are positive. Since the matrices V and A are orthonormal, and \bar{S} is a positive matrix, by Lemma 1, the matrix V will be the same as the matrix A up to a permutation matrix.

Similarly as the method in [9], we first form a data set \bar{Z} by discarding those observed signals $z(k)$ where $\|z(k)\| < \epsilon$ and normalizing every observed signal vector. Then from data set \bar{Z} , construct a new data set \hat{Z} in which all signals are mutually noncolinear. After that, re-index the vectors in \hat{Z} . Suppose the index set $Q = \{1, 2, \dots, \bar{N}\}$. Now, our goal is to find n distinct orthogonal vectors $\hat{z}(k_i) \in \hat{Z}, k_i \in Q$ such that the matrix $V = [\hat{z}(k_1), \dots, \hat{z}(k_n)]$ is orthonormal.

If we find these n orthogonal vectors directly, in the worst case we need compute $C_{\bar{N}}^n$ times. If the number of sampled observed signals is large, it is unreasonable for practical computation. Naturally, we consider the evolutionary search techniques to find these orthogonal vectors.

Define the feasible solution in the form of string (or chromosome) $[k_1, k_2, \dots, k_n]$, where $k_1, k_2, \dots, k_n \in Q$ are integers. This means that $[\hat{z}(k_1), \dots, \hat{z}(k_n)]$ is the matrix we wanted. For example, consider a four-dimensional problem with $\bar{N} = 1000$. If a feasible solution is $[23, 45, 899, 934]$, it means that the vectors in \hat{Z} with index $\{23, 45, 899, 934\}$ will construct an orthonormal matrix.

Let the function Γ be the objective function of our genetic algorithm. The evolutionary search starts with a population of p random solutions and iteratively used the processes of selection, crossover, and mutation to perform a combination of hill climbing, solution recombination, and random search over the possible index combinations. The process was continued until some criteria are satisfied [6]. In each stage of the algorithm, a set of best index solutions (the most minimum Γ values) were kept. At the end of the algorithm, these solutions were reported. The technique to keep the best solutions in each iteration can guarantee the convergence to the optimal solution in some sense, which was shown in [13][14]. The genetic algorithm can be summarized as follows:

Algorithm 1. Evolutionary-Search (Number of feasible solutions: m , Dimensionality: n)

```

Begin
  S=Initial seed population of  $p$  strings;
  BestSet=null;
  While no(termination_criterion) do begin
    NewS=Selection(S);
    NewS=CrossOver(NewS);
    NewS=Mutation(NewS);
    S=Reinsertation(S,NewS);
    Update Bestset to be the  $m$  solutions in  $BestSet \cup S$  with most minimum
       $\Gamma$  values;
  End;
  return BestSet;
end

```

The initial population is achieved by generating p strings using a random integer generator that uniformly distributes numbers in the range $[1, \bar{N}]$. Selection process determines the individuals for reproduction and the number of offsprings that an individual can produce. Here, we select 90 percent of individuals to produce new individuals, and keep 10 percent of individuals which have minimum values. During the selection phase, each individual of current population is assigned a fitness value derived from the corresponding objective function value. Then, the selection algorithm selects individuals for reproduce on the basis of their relative fitness values. In our algorithm, the fitness values are calculated using linear ranking method with pressure 2 in [2] as follows

$$F(x_i) = 2 - Max + 2(Max - 1) \frac{x_i - 1}{p - 1}, \quad (6)$$

where Max is always chosen in $[1.1, 2]$ which is used to determine the selective pressure such that no individuals generate an excessive number of offsprings. And x_i is the position of the i th individual in the re-ordered population based on their corresponding objective function values.

The popular rank selection principle is used. The idea is to replicate copies of a solution by ordering them by rank and biasing the population in favor of the higher-rank solution. This method is very stable since it results in a global hill climbing of an entire population of solutions. For our implementation, we use the stochastic universal sampling (SUS) method in [2]. The procedure of SUS algorithm is as follows: First, obtain a cumulative sum of the fitness value vector corresponding to the population, and generate equally spaced numbers between 0 and sum ; then, one random number is generated, all the others used being equally spaced from the point; in the end, the indexes of the selected individuals are determined by comparing the generated numbers with the cumulative sum vector.

Crossover (recombination) phase produces the new strings which are recombined by parts of parents. In our algorithm, we use the simplest form of crossover, i. e. single-point crossover. Consider an example with dimensionality 5 and $\bar{N} = 1000$. There are two strings: $A_1 = [23\ 34\ 78\ 899\ 901]$ and $A_2 = [45\ 92\ 37\ 309\ 267]$. Assume the switch position is randomly generated as 3, then the new strings will be $A'_1 = [23\ 34\ 78\ 309\ 267]$ and $A'_2 = [45\ 92\ 37\ 899\ 901]$. For the default setting, 70 percent of old strings will be recombined to reproduce new offsprings.

Mutation phase produces the new strings by randomly modifying the elements of strings in the whole population. Mutation is used to uncover the good strings that may be lost in the actions of selection and crossover. Consider an example for our problem with dimensionality 5, there is a string $[23\ 79\ 345\ 898\ 900]$ which is changed to $[23\ 89\ 345\ 898\ 900]$ by mutation. In our algorithm, mutation is randomly applied with probability 0.3 based on the number of population.

Reinsertion function handles the case that the number of new population produced by selection, crossover and mutation are fewer or more than that of old population, and keeps the most fit old individuals. Our technique is that the new individuals are fitness-based reinserted into the old population. The least fit old individuals are replaced and discarded.

The algorithm to separate the uncorrelated positive source signals which satisfy **assumption A** can be summarized as the following.

Algorithm 2. GA-BSS (Observed signal set: Z , Dimensionality: n)

1. Discard those columns $z(k)$ from the observed signal set $Z = \{z(k), k = 1, \dots, \}$ such that $\|z(k)\| < \epsilon$ and form a new data set \bar{Z} by normalizing the remaining signal vectors.

2. Form the matrix \hat{Z} consisting of all mutually noncolinear vectors of \bar{Z} , and reindex the vectors in the new data set.

3. Call the function Evolutionary-Search(Number: m , Dimensionality: n) and return the best index set $BestSet$.

4. Select observed signal vectors from $BestSet$ which has the minimum value of objective function Γ and form the matrix \bar{A} .

5. Replace each column in \bar{A} by the average of all vectors in \bar{Z} that are colinear to it.

6. Compute the estimate of source signals: $\bar{s}(k) = \bar{A}^{-1}z(k), 1 \leq k \leq N$.

In the algorithm GA-BSS, step 1 and step 2 come from the algorithm LP-BSS in [9]. These two steps can reduce the range of search and the effects of noise. Two vectors e_i and e_j are defined to be collinear if their angle θ does not exceed a tolerance threshold.

4 Simulations and Discussions

The efficiency of GA-BSS algorithm will be illustrated by using a few experiments. Since only LP-BSS and GA-BSS algorithms can be applied to recover positive source signals exactly when they are not guaranteed to be independent,

we only compare the performance of GA-BSS with that of LP-BSS. Other BSS methods cannot recover the source signals exactly, for example, the standard ICA algorithm, nonlinear PCA algorithm, etc.

Without loss of generality, suppose the mixing matrix A is an orthonormal and constant matrix. For recovering the source signals, the purpose of both algorithms is to find an orthonormal matrix \bar{A} such that A and \bar{A} are equivalent up to a permutation matrix P , i.e. $\bar{A} \approx AP$. To evaluate the performance of separation, two same performance indexes are used as in [9]. Let $D = A'\bar{A}$, the first Comon's index is defined in [3] as follows:

$$e(D) = \sum_i \left| \sum_j |d_{ij}| - 1 \right|^2 + \sum_j \left| \sum_i |d_{ij}| - 1 \right|^2 + \sum_i \left| \sum_j |d_{ij}|^2 - 1 \right|^2 + \sum_j \left| \sum_i |d_{ij}|^2 - 1 \right|^2,$$

where d_{ij} the element of matrix D . The second Choi's index in [9] is defined by

$$e'(G) = \frac{1}{2(n-1)} \sum_{i=1}^n \left(\sum_{k=1}^n \frac{|g_{ik}|^2}{\max_j |g_{ij}|^2} - 1 + \sum_{k=1}^n \frac{|g_{ki}|^2}{\max_j |g_{ji}|^2} - 1 \right),$$

where g_{ij} is the element of matrix $G = \bar{A}^{-1}A$. These two indexes are zero if and only if the matrices A and \bar{A} are equivalent.

In the first experiment, we will show the scalability of GA-BSS and LP-BSS algorithms. There are three 252x252 images and the pixel intensities were scaled to unit variance. The source sequence $s(k)$ is the sequence of pixel values of the images from top left to bottom right and $1 \leq k \leq 63504$. Since the source images have nonnegative pixel values, so they are suitable for positive source signal separation. The original images used are shown in Fig.1.



Fig. 1. Source images used for the positive BSS algorithms

Let $s(k)(k = 1, \dots, N)$ are the source signals where $N < 63504$. We first de-correlate these signals, then let only one signal has positive value and other

signals equal to zero at some randomly chosen time k such that the final source signals satisfy assumption A. The observed signals $x(k)$ ($k \leq N$) are obtained by mixing these source signals using a randomly generated orthonormal matrix. These mixed signals are fair to test both algorithms because all of them can recover the source signals exactly. Under the same accuracy with respect to these two performance indexes, the evaluation time of GA-BSS and LP-BSS are draw in Fig.2 when the number N is from 2000 to 20000. It can be seen that GA-BSS are more efficient than that of LP-BSS with large number of observed signal data. This is because LP-BSS algorithm need compute the scores of every observed signal vectors by using revised simplex method to decide which vectors can construct the mixing matrix. As the number of observed signal vectors is increasing, the execution time will increase quickly.

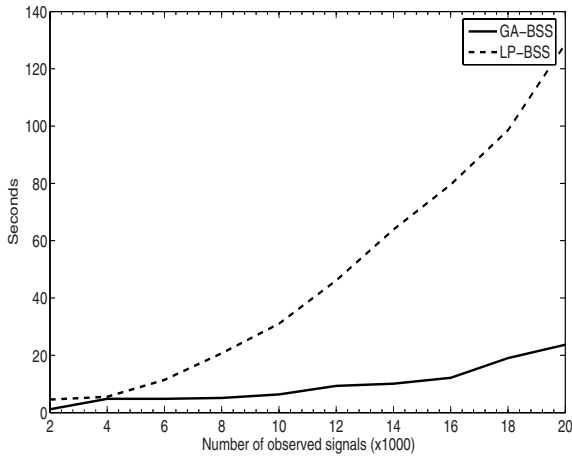


Fig. 2. The evaluation time (seconds) of two BSS algorithms with respect to different sizes of observed signal data

Table 1. The performance of GA-BSS and LP-BSS algorithms

Algorithm	Comon's index	Choi's index	Evaluation time(s)
GA-BSS	3.3746e-005	2.8725e-006	2.3434
LP-BSS	9.2540e-005	6.2286e-006	2.7540

In the second experiment, the same source signals in [9] are used, which are NMR Spectroscopy signals. The observed signals are constructed by mixing the source signals using a randomly generated orthonormal matrix. Table 1 shows the evaluation results of GA-BSS and LP-BSS algorithms. It can be seen that GA-BSS algorithm are more efficient and accurate than that of LP-BSS algorithm.

5 Conclusions

We proposed an effective and efficient BSS algorithm for a class of positive source signals. This class of positive signals are uncorrelated and satisfy assumption A. Moreover, if the mixing matrix is orthonormal, our algorithm can handle the case that the positive source signals are partially correlated and satisfy assumption A. Our main contribution is that our algorithm can recover the mixing matrix very quickly based on evolutionary search techniques. Simulations confirm the efficiency and effectiveness of our algorithm. Our studies may give some hints to the research of blind separation of more general positive signals.

Acknowledgments

This work was supported by Program for New Century Excellent Talents in University and Application Research Foundation of Science and Technology Bureau of Sichuan Province of China, Grant No. 2006J13-065.

References

1. Aggarwal, C.C., Orlin, J.B., Tai, R.P.: Optimized Crossover for the Independent Set Problem. *Operat Res* **45** (2) (1997) 226-234
2. Baker, J.E.: Adaptive Selection Methos for Genetic Algorithms. *Proc. ICAGA 1* (1985) 101-111
3. Comon, P.: Independent Component Analysis-A New Concept? *Signal Process* **36** (3) (1994) 287-314
4. Cickocki, A., Karhunen, J., Kasprzak, W., Vigarior, R.: Neural Networks for Blind Separation with Unknown Number of Sources. *Neurocomputing* **24** (1) (1999) 55-93
5. Cichocki, A., Amar, S.-i.: *Adaptive Blind Signal and Image Processing*. John Wiley and Sons (2002)
6. De Jong, K.A.: *Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD dissertaion, Univerisity of Michigan (1975)
7. Hyvärinen, A., Karhunen, J., Oja, E.: *Independent Component Analysis*. John Wiley and Sons (2001)
8. Jutten, C., Herault, J.: Blind Separation of Sources Part I: And Adaptive Algorithm Based on Neuromimetic Architecture. *Signal Process* **24** (1991) 1-10
9. Naanaa, W., Nuzillard, J.M.: Blind Source Separation of Positive and Partially Correlated Data. *Signal Process* **85** (2005) 1711-1722
10. Oja, E., Plumbley, M.D.: Blind Separation of Positive Sources by Globally Convergent Gradient Search. *Neural Computation* **16** (9) (2004) 1811-1825
11. Plumbley, M.D.: Conditions for Nonnegative Independent Component Analysis. *IEEE Signal Processing Letters* **9** (6) (2002) 177-180
12. Plumbley, M.D., Oja, E.: A 'Nonnegative PCA' Algorithm for Independent Component Analysis. *IEEE Trans. Neural Networks* **15** (1) (2004) 66-76

13. Rudolph, G.: Convergence Analysis of Canonical Genetic Algorithms. *IEEE Trans. Neural Networks* **5** (1994) 96-101
14. Suzuki, J.: A Markov Chain Analysis on Simple Genetic Algorithms. *IEEE Trans. System, Man and Cybernetics* **25** (1995) 655-659
15. Ye, M.: Global Convergence Analysis of a Discrete Time Nonnegative ICA Algorithm. *IEEE Trans. Neural Networks* **17** (1) (2006) 253-256

A Speech Enhancement Method in Subband

Xiaohong Ma, Xiaohua Liu, Jin Liu, and Fuliang Yin

School of Electronic and Information Engineering,
Dalian University of Technology, Dalian 116023, China
maxh@dlut.edu.cn

Abstract. The speech enhancement method based on blind source separation and post-processing in subband [1] is an effective method in noise and reverberation environments. Performance analysis and computer simulations indicate that its performance is degraded under uncorrelated or mild correlated noise cases, and sometimes it might cause distortion of the enhanced signals. To apply the method in real environment, some improvements have been made on it. These are that adaptive noise cancellers are only used in the subbands with poor separation results and the independent component analysis (ICA) operations in low frequency bands are replaced by the efficient time-frequency masking method. Experimental results show the effectiveness of the proposed method.

1 Introduction

The speech signals received by microphone array are inevitably interfered by noise from the environment and the reverberation of the rooms in the speech communication system. Multi-microphone techniques are a growing field in speech enhancement since beamforming and related techniques have a great potential for noise reduction. Spatial information is often exploited to enhance the speech signal in a noisy environment. The conventional methods for spatial processing techniques include delay-sum beamforming, adaptive beamforming [2] and post-filter algorithm [3] and so on. Unfortunately, the performance of the beamforming based methods will degrade if any a priori information about the acoustical environment and the sources involved is unknown. Besides, a large number of microphones are generally required for good speech enhancement performance [1]. Unlike beamforming technique, ICA method [1], [5], [6] which estimates original source signals using only the mixed signals observed in each input channel is an unsupervised adaptive technique [6]. Comparing with beamforming approach, ICA [4] has two advantages: it only needs a small number of microphones and does not require a priori information about the sources. Clearly, the weakness of the beamforming algorithm is the advantage of the ICA algorithm [1]. The method suggested by Low, S.Y. et al [1] has achieved high performance in depressing noise and reverberation. Regrettably, its performance was degraded under uncorrelated or mild correlated noise cases. To apply the method in real environment, some improvements have been made on it. These are that adaptive

noise cancellers are only used in the subbands with poor separation results and time-frequency masking method [7], [8] instead of ICA is used in low frequency bands.

2 The Model of Array Signals

Ideal model and practical model are two widely used models for speech enhancement. The former only deals with environmental noise, while the latter considers environmental noise and reverberation simultaneously. A microphone array with L microphones was placed in the acoustic enclosure environment where speech and noise coexist. In the practical model, the signals $x_i(t)(t = 1, 2, \dots, L)$ received by the i th microphone can be modeled as

$$x_i(t) = g_i * s(t) + n_i(t) \tag{1}$$

where $x_i(t)$ is the nonstationary speech signal, $n_i(t)$ represents the stationary noise, g_i is the room impulse response between the speech source and the i th microphone and $*$ denotes convolution.

3 The Proposed Method

3.1 The Structure of the Proposed System

The proposed speech enhancement scheme is shown in Fig.1. There are seven main modules, namely, analysis filter bank, time-frequency masking, subband ICA, kurtosis calculation, judgment, adaptive noise canceller (ANC) and synthesis filter bank.

Comparing Fig. 1 with the Fig. 1 in [1], we can see that the modules including the analysis and synthesis filter banks to decompose and reconstruct signals, the

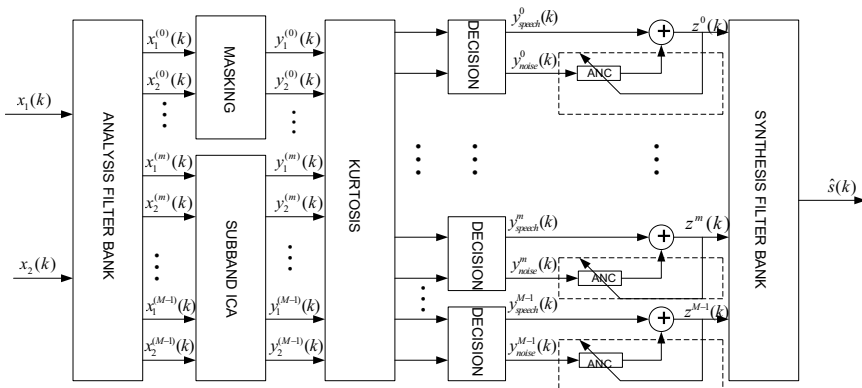


Fig. 1. Proposed speech enhancement scheme

subband ICA to approximately separate the speech signal and noise, and the kurtosis to ascertain which output contains the most dominant interference are the same and have been discussed by Low, S.Y. et al [1] in detail. But the remainder modules are different from [1]. Hence, in the following subsections, we will analyze the performance of adaptive noise canceller, then propose the rule to decide in which subbands the ANC is executed. Furthermore, we will investigate the time-frequency masking method.

3.2 The Performance of the Adaptive Noise Canceller [9]

The architecture of the adaptive noise canceller is shown in Fig.2. The signal-to-noise ratio (SNR) at the reference input is represented as follows [9]

$$\rho_{ref}(z) = \left(\Phi_{ss}(z) |J(z)|^2 \right) / \left(\Phi_{nn}(z) |H(z)|^2 \right) \tag{2}$$

where $\Phi_{ss}(z)$ and $\Phi_{nn}(z)$ denote the power spectrum of the speech signal and noise respectively. When the noises at the primal input and the reference input are correlated with each other, the SNR at the output is

$$\rho_{out}(z) = 1/\rho_{ref}(z) \tag{3}$$

Next, the signal distortion $D(z)$ is defined as

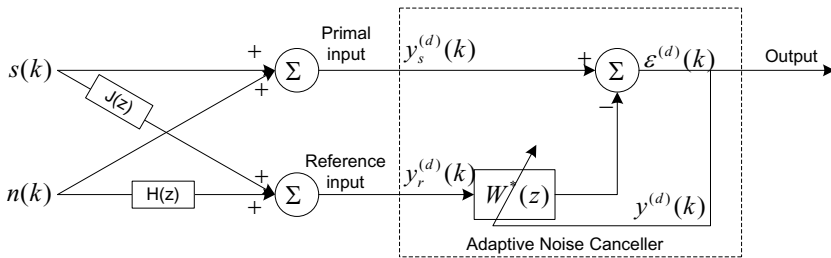


Fig. 2. Diagram of the adaptive noise canceller

$$D(z) \approx \left(\Phi_{ss}(z) |J(z)/H(z)|^2 \right) / \Phi_{ss}(z) = |J(z)/H(z)|^2 \tag{4}$$

The SNR at the primal input is defined as follows

$$\rho_{pri} = \Phi_{ss}(z)/\Phi_{nn}(z) \tag{5}$$

Naturally, the signal distortion $D(z)$ can be calculated as

$$D(z) \approx \rho_{ref}(z)/\rho_{pri}(z) \tag{6}$$

Equation (6) indicates that it causes small signal distortion if SNR at the primal input is high while SNR at the reference input is low.

Finally, the power spectrum of noise at the output can be represented as follows [9]

$$\Phi_{outputnoise}(z) \approx \Phi_{nn}(z) |\rho_{ref}(z)| |\rho_{pri}| \quad (7)$$

Equation (7) indicates that the smaller the product of $|\rho_{ref}(z)|$ and $|\rho_{pri}|$ is, the better result can be gotten by ANC.

3.3 The Rule to Execute Adaptive Noise Canceller

Unfortunately, equation (7) might not be guaranteed in [1] because the performance of the subband ICA can not be controlled. As a result, when the subband ICA performs well and results in a high value of $|\rho_{pri}|$, the degradation of the enhanced signal occurs.

Considering the conclusion drawn above, a scheme to resolve this problem is proposed. The speech signal with super-Gaussian distribution usually has a larger value in term of kurtosis, while the noise with Gaussian-like distribution always has a smaller value. Thus, we can safely say that the difference between the values of the two outputs from kurtosis block is large in the subbands where the ICA algorithm successfully separates the mixtures, while the difference is small in the subbands where the ICA algorithm fails to separate the mixtures. Based on this point, a rule to decide in which subbands the ICA algorithm performs well is designed as

$$\xi^{(d)} = (\xi_{speech}^d - \xi_{ref}^d) / \xi_{speech}^d \quad (8)$$

where ξ_{speech}^d denotes the larger kurtosis value in the d th subband, ξ_{ref}^d is the smaller kurtosis value. If $\xi^{(d)} > th$, where th is a threshold, the ICA algorithm performs well in this subband. In this case, the adaptive noise canceller is not needed; otherwise, it is executed.

4 Time-Frequency Masking

Binary time-frequency masks are powerful tools for the separation of sources from mixtures. The algorithm depends on two major points [7]: (1) the existence of an invertible transformation T that transforms the signals to a domain on which they have disjoint representations; (2) finding functions F and G that provide the means of labelling on the transform domain. Further discussion about this algorithm can be found in [7].

Let us discuss the motivation for replacing the ICA algorithm in low frequency bands by efficient time-frequency masking. Firstly, there is an inevitable scaling problem which is difficult to deal with in ICA algorithm. However, the problem doesn't exist in the time-frequency masking algorithm. Secondly, as a fact, most energy of the speech signal is contained in low frequency bands, so it is important

to keep the speech energy in these bands from losing. The time-frequency masking method performs better than ICA method in this aspect. The efficiency of time-frequency masking can be seen in section 5. The purpose of time-frequency masking is to obtain another version of output that contains more energy of speech signal and less energy of noise.

5 Experiments

To illustrate the efficiency of the time-frequency masking method, some simulation experiments have been carried out. The noises received by two microphones are uncorrelated with each other. The performance comparison of ICA and time-frequency masking is shown in Fig.3. As expected, the waveform of speech enhanced by time-frequency masking method (Fig.3 (f)) indicates that most energy of speech is successfully kept while noises are perfectly depressed. However, ICA method shown in Fig.3 (d) fails to do so.

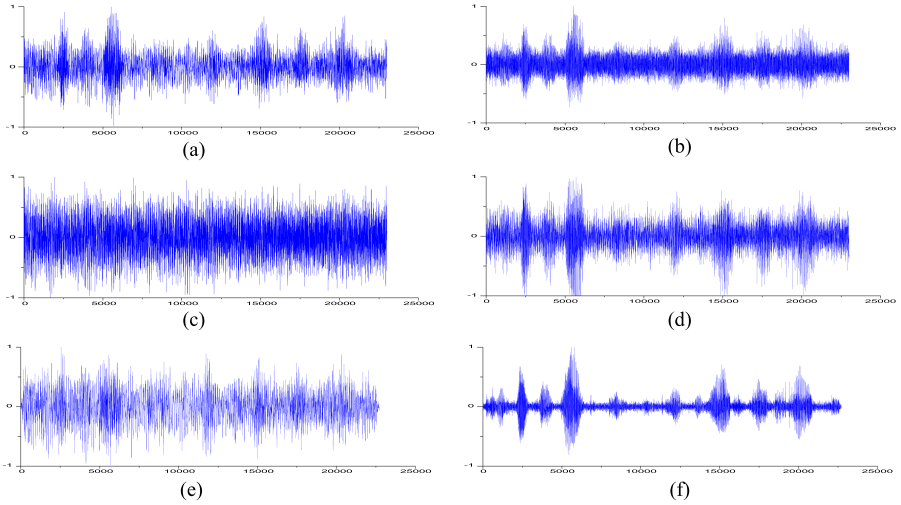


Fig. 3. Performance comparison of ICA and time-frequency masking, (a)-(b) are signals received by two microphones respectively. (c)-(d) are separated outputs from ICA algorithm. (e)-(f) are the outputs of time-frequency masking method.

To illustrate the performance of the proposed method, we carried out extensive experiments using noisy signals recorded in an actual room acoustics environment with dimensions of $5m \times 4m \times 4m$. The microphone array with two microphones was placed on a table $1.2m$ away from a woman speaker. Here, the sampling rate was 16 KHz. The noise produced by computer and room reverberations are the main interferences. Fig.4 (a)-(b) show the waveforms of the received signals. The former with higher SNR value seems to bring in larger speech distortion, while the later with lower SNR value can better represent

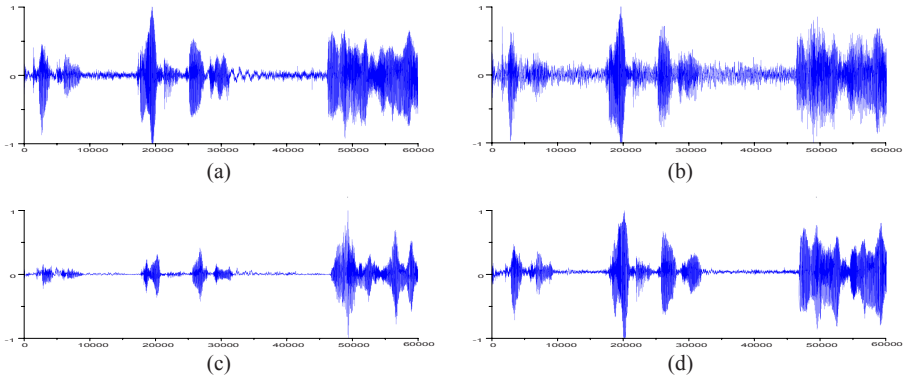


Fig. 4. A example of speech enhancement by the proposed method. (a)-(b) are signals received by the microphones respectively. (c) speech enhanced by Low, S.Y. method. (d) speech enhanced by the proposed method.

the practical speech signal. The reason for it may be that the parameters of the two microphones are not identical. Fig.4 (c) is the result using the original method [1]. The waveform of speech enhanced by the proposed method is shown in Fig.4 (d). Comparing Fig.4 (c) with Fig.4 (d), it can be seen clearly that the proposed method causes less distortion on the enhanced speech signal than the original method, meanwhile successfully depresses noises, and moreover, the hearing test verifies this point. The SNR comparisons between the original method and the proposed method are shown in Table 1. Comparing with the input SNR of Mic2, the output SNR enhanced by original method results in an improvement of 10.8385 dB, while the proposed method results in an improvement of 13.8420 dB. Comparing with the input SNR of Mic1, the two methods both yield SNR improvements too, but the proposed method performs better. Furthermore, we have carried out six other experiments, and the same conclusion has been achieved.

Table 1. SNR comparison of the original and the proposed method

Input SNRs (dB)		Output SNRs (dB)	
Mic1	Mic2	Original method	Proposed method
16.4835	8.3949	19.2334	22.2369

6 Conclusion

A new speech enhancement method in subband was proposed in this paper. The performance of the ANC was analyzed in detail, and a rule to decide in which subbands the ANC would be executed was proposed. Besides, the efficient time-frequency method was introduced in the system. Simulation results show the

effectiveness of the proposed method. Especially, the listening tests show that it causes less distortion on the enhanced speech signal than original method [1].

Acknowledgments. This work is supported by the National Natural Science Foundation of China under Grant No. 60575011, No. 60372082 and Liaoning Province Natural Science Foundation of China under Grant No. 20052181.

References

1. Low, S.Y., Nordholm, S., Togneri, R.: Convolutional Blind Signal Separation with Post-Processing. *IEEE Transactions on Speech and Audio Processing* **12** (5) (2004) 539-648
2. Griffiths, L.J., Jim, C.W.: An Alternative Approach to Linearly Constrained Adaptive Beamforming. *IEEE Transactions on Antennas and Propagation* **30**(1) (1982) 27-34
3. Gannot, S., Bershtein, D. Weinstein, E.: Signal Enhancement Using Beamforming and Nonstationarity with Applications to Speech. *IEEE Transactions on Signal Processing* **49** (8) (2001) 1614-1626
4. Hyvarinene, A.: Fast and Robust Fixed-point Algorithms for Independent Component Analysis. *IEEE Transactions on Neural Networks* **10** (3) (1999) 626-634
5. Asano, F., Ikeda, S., Ogawa, M. et al : Combined Approach of Array Processing and Independent Component Analysis for Blind Separation of Acoustic Signals. *IEEE Transactions on Speech and Audio Processing* **11** (3) (2003) 204-215
6. Saruwatari, H., Kawamura, T., Lee, A. et al: Blind Source Separation Based on a Fast-Convergence Algorithm Combining ICA and Beamforming. *IEEE Transactions on Speech and Audio Processing* **14** (2) (2006) 666-678
7. Yilmaz, Q., Rickard, S.: Blind Separation of Speech Mixtures via Time-Frequency Masking. *IEEE Transactions on Signal Processing* **52** (7) (2004) 1830-1847
8. Savada, H., Araki, S., Mukai, R. et al.: Blind Extraction of Dominant Target Sources Using ICA and Time-Frequency Masking. *IEEE Transactions on Audio, Speech, and Language Processing* **14** (6) (2006) 2165-2173
9. Widrow, B., Stearns, A.D. Adaptive Signal Processing. Prentice-Hall, New Jersey (1985)

Sequential Blind Signal Extraction with the Linear Predictor

Yunxia Li and Zhang Yi

Computational Intelligence Laboratory,
School of Computer Science and Engineering,
University of Electronic Science and Technology of China,
Chengdu 610054, P.R. China
yunxiali@uestc.edu.cn, zhangyi@uestc.edu.cn

Abstract. The sequential blind signal extraction method with a linear predictor is proposed. The Kalman filter is introduced to overcome the problem of the choice of the linear predictor coefficients. Using a deflation technique, the proposed algorithm is able to sequentially recover the source signals one-by-one. Simulation results verify the validity and performance of the proposed algorithm.

1 Introduction

Over the past decade blind source separation(BSS) has received much research attention. The objective of BSS is to recover source signals from their mixture without prior information on the source signals and mixing channel. This class of signal processing techniques can be used in many areas such as communications, medical signal processing, speech recognition, image restoration, etc. [1,2].

There are two approaches for recovering the original source signals, namely, the simultaneous separation approach [2,3,4,5] and the extraction approach. In the extraction approach, the source signals are extracted one-by-one by eliminating the the already extracted sources from their mixtures with deflation techniques. Many blind source extraction (BSE) algorithms use the property of sparseness [6] or high-order statistics [7,8,9] to extract a specific signal. But they often have high computation load. Thus the versatile extraction algorithms based on second-order statistics (SOS) [10,11,12] become popular. A class of SOS approaches employing a linear predictor was analyzed [13], which proposed a new method to perform one source extraction based on the normalized mean square prediction error. The key to the success of the proposed algorithm is the choice of the linear predictor coefficients.

We therefore address this problem by introducing the Kalman filter [14] to estimate the linear predictor coefficients. The simulation results confirm the validity of the proposed method.

2 Proposed Algorithm

The observed sensor signals at discrete time t can be expressed through the following linear model

$$x(t) = As(t) \quad t = 0, 1, 2, \dots \tag{1}$$

where $x(t)$ is an $n \times 1$ sensor vector, $s(t)$ is an $n \times 1$ unknown mixing matrix, and the subscript t denotes time index.

To cope with ill-conditioned cases, we first decorrelate the sensor signals $x(t)$ by a linear transformation, i.e., $x_1(t) = Vx(t)$ such that $E \{x_1(t)x_1^T(t)\} = I_n$, where V is a prewhitening matrix.

To extract one of the sources, we can employ a linear neural network cascaded with a linear predictor, where the input-output relation of the network and the prediction error are given, respectively, as follows:

$$y_1(t) = w_1^T(t)x_1(t), \tag{2}$$

and

$$e_1(t) = y_1(t) - B_1^T \tilde{y}_1(t), \tag{3}$$

where the predictor coefficient vector $B_1 = [b_1, b_2, \dots, b_P]^T$ with P length and $\tilde{y}_1(t) = [y_1(t-1), y_1(t-2), \dots, y_1(t-P)]^T$. It has been pointed out in [13] a single source can be recovered by minimizing the normalized mean square prediction error under the constraint $\|w_1(t) = 1\|$. This way, the cost function was proposed as

$$J_1(w) = \frac{E\{e_1^2(t)\}}{E\{y_1^2(t)\}}. \tag{4}$$

After prewhitening process, the cost function $J_1(w)$ becomes

$$J_1(w) = E\{e_1^2(t)\}. \tag{5}$$

It yields the online update rule in [13]:

$$w_1(t+1) = w_1(t) - \mu e_1(t) \tilde{x}_1(t), \tag{6}$$

where μ is learning rate and

$$\tilde{x}_1(t) = x_1(t) - \sum_{p=1}^P b_p x_1(t-p). \tag{7}$$

The update is followed by the normalization of the demixing vector

$$w_1(t+1) = w_1(t+1) / \sqrt{w_1^T(t+1)w_1(t+1)}. \tag{8}$$

Unfortunately, the algorithm’s performance strongly depends on the choice of the linear predictor coefficients B [13]. However, due to the blind nature of the problem, the AR coefficients of the source signals are not known in advance. Therefore it is difficult to find such an optimal linear predictor. Thus they are just generated randomly, which, as a result, has negative influence upon the source extraction. In this paper, we use the Kalman filter [14] to estimate the

linear predictor coefficients so as to improve the extraction performance. For simplicity, we transform the eqn. (3) to be

$$y_1(t) = B_1^T \tilde{y}_1(t) + e_1(t). \tag{9}$$

Since B_1 , the linear predictor coefficients vector, is unknown but invariable, it is can be described by

$$B_1(t + 1) = B_1(t). \tag{10}$$

In state-space expression, the eqn. (9) and the eqn. (10) are called observation and state equations, respectively. Denote by $\hat{B}_1(t)$ the estimation of $B_1(t)$ and by $Q(t)$ the estimation covariance matrix. $\hat{B}_1(t)$ and $Q(t)$ are given by the Kalman filter equations

$$K_1(t) = Q_1(t) \tilde{y}_1(t) [\tilde{y}_1^T(t) Q_1(t) \tilde{y}_1(t) + \sigma_1^2]^{-1}, \tag{11}$$

$$\hat{B}_1(t + 1) = \hat{B}_1(t) + K_1(t) v_1(t), \tag{12}$$

and

$$Q_1(t + 1) = Q_1(t) - K_1(t) \tilde{y}_1^T(t) Q_1(t), \tag{13}$$

where $v_1(t) = y_1(t) - \hat{B}_1^T \tilde{y}_1(t)$ and σ_1^2 indicates the variance of $e_1(t)$. As it is well known the Kalman filter is optimal in estimating B_1 when the observation and state equations are of the form in (9) and (10). Thus the unknown linear predictor coefficients vector B_1 used to obtain $e_1(t)$ in (3) and $\tilde{x}_1(t)$ in (7) is substituted by $\hat{B}_1(t)$. Therefore the eqns. (3)-(8) combined with the Kalman filter equations (11)-(13) are used to extract the first source signal. Finally, the extraction process could be easily generalized for extraction of next sources, say, y_2, \dots, y_n , in cooperation with the deflation procedure described below.

Now let us suppose that y_j has been extracted, where the subscript j also indicates the total number of source signals being extracted so far. We exploit the knowledge of y_j to generate the new input vector x_{j+1} which will not include the already extracted signals y_j . This can be easily carried out by the linear transformation

$$x_{j+1}(t) = x_j(t) - \tilde{w}_j(t) y_j(t). \tag{14}$$

The problem now is to get the appropriate $\tilde{w}_j(t)$. Since $y_j(t)$ is the estimated source signal we seek the $\tilde{w}_j(t)$ so as to remove the extracted signal from mixture signal. Although the ideal condition is that the components of $x_{j+1}(t)$ are independent to $y_j(t)$ respectively, the weaker condition, decorrelation, can find out the appropriate $\tilde{w}_j(t)$. Assume $y_j(t)$ and $x_{j+1}(t)$ are uncorrelated, it holds

$$\begin{aligned} & E \{y_j(t)(x_j(t) - \tilde{w}_j(t)y_j(t))\} \\ &= E \{y_j(t)\} E \{x_j(t) - \tilde{w}_j(t)y_j(t)\} \\ &= 0. \end{aligned} \tag{15}$$

Then we have

$$E \{y_j(t)x_j(t)\} - \tilde{w}_j(t)E \{y_j^2(t)\} = 0. \tag{16}$$

It yields

$$\tilde{w}_j(t) = \frac{E \{y_j(t)x_j(t)\}}{E \{y_j^2(t)\}}. \tag{17}$$

The deflation algorithm (17) is also derived in [15], which obtains it by minimizing the energy function $\tilde{J}_j(\tilde{w}_j(t)) = \frac{1}{2}\|x_{j+1}\|^2$.

The deflation rule (14) and (17) can be continued until all of the estimated source signals are recovered. In every extraction operation the weight learning rules (3)-(8) combined with the Kalman filter equations (11)-(13) are used to extract the source signal.

3 Simulations

In our simulations, we use four source signals s_1, \dots, s_4 as shown in Fig. 1. They can be found in the file ABio7.mat provided by the ICALAB toolbox with the book [2]. The source signals were mixed by the randomly mixing matrix

$$A = \begin{bmatrix} -0.0001 & 0.0269 & -0.0392 & 0.0155 \\ -0.0172 & -0.0034 & -0.0112 & 0.0230 \\ 0.0543 & -0.0253 & -0.0705 & -0.0148 \\ 0.0125 & 1.0118 & 0.0459 & 0.0008 \end{bmatrix}.$$

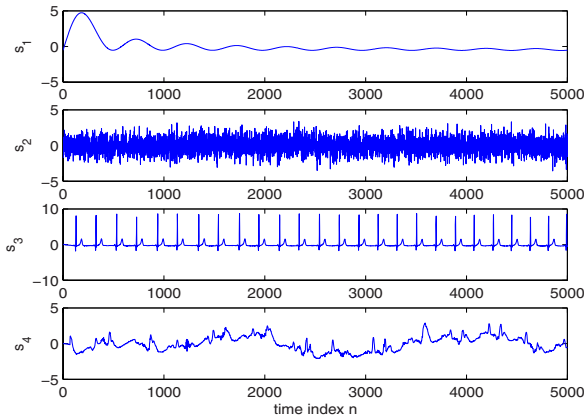


Fig. 1. Source signals

Firstly, we use two algorithms to extract the first signal. One is the algorithm in [13], for which the linear predictor coefficient vector B is randomly given. Simulation results indicates the extracting performance varies with the choice of B . The relative good result is shown in Fig. 2(a). Another algorithm is our

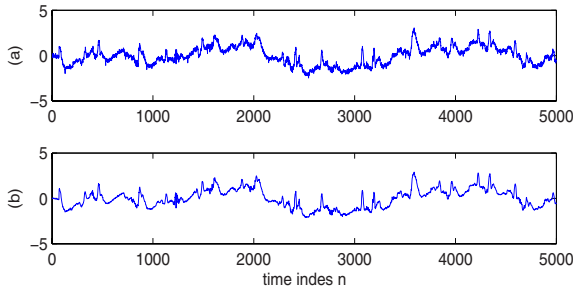


Fig. 2. The first extracted signal:(a) is recovered by the algorithm [13]; (b) is by our algorithm

algorithm, for which B is given by the Kalman filter. The extracted signal is given in Fig. 2(b). Obviously, the first extracted signal by our algorithm is clearer, while the one by [13] is slightly mixed by other source signals.

Fig. 3 shows the sequential extraction results by our proposed extraction algorithm and the deflation method ([17]). The extracted signals successfully recover the source signals except for the amplitude and permutation ambiguity.

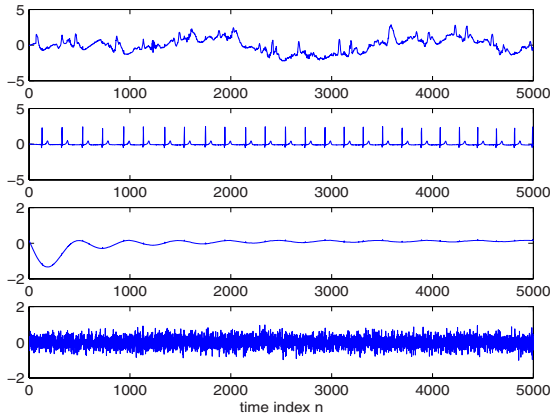


Fig. 3. The sequentially extracted signals by our approach

4 Conclusions

Developing algorithm by linear predictor is one of the trends in BSE. However, the linear predictor coefficients heavily affect the extraction results. We introduce the Kalman filter to estimate the linear predictor coefficients. This measure can help to improve the extraction performance. Then we propose a deflation technique to accomplish recovering source signals sequentially. The simulation results verify the new extraction and deflation method.

References

1. Baldonado, M., Chang, C.C.K., Gravano, L., Paepcke, A.: The Stanford Digital Library Metadata Architecture. *Int. J. Digit. Libr.* **1** (1997) 108–121
2. Cichocki A., Amari S.: *Adaptive Blind Signal and Image Processing*. Wiley, New York, (2003)
3. Hyvärinen A., Karhunen J., Oja E.: *Independent Component Analysis*. Wiley, New York, (2001)
4. Choi S., Cichocki A., Amari S.: Equivariant nonstationary source separation. *Neural Networks* **15** (2002) 121-130
5. Cichocki A., Unbehauen R.: Robust Neural Networks with On-line Learning for Blind Identification and Blind Separation of Sources. *IEEE Transaction on Circuits and systems*, **43** (11) (1996) 894-906
6. Zibulevsky, M., Zeevi, Y.Y.: Extraction of a Source from Multichannel Data Using Sparse Decomposition. *Neurocomputing* **49** (1-4) (2002) 163-173
7. Liu, W., Mandic, D.P.: A Normalised Kurtosis Based Blind Source Extraction Algorithm for Noisy Mixture. *ICASSP* (2006) 641-644
8. Zhang, Z.L., Yi, Z.: Extraction of a Source Signal Whose Kurtosis Value Lies in a Specific Range. *Neurocomputing*, **69** (2006) 900-904
9. Cichocki, A., Thawonmas, R., Amari, S.: Sequential Blind Signal Extraction in Order Specified by Stochastic Properties. *Electronics Letters* **33** (1) (1997) 64-65
10. Cichocki, A., Thawonwas, R.: On-line Algorithm for Blind Signal Extraction of Arbitrarily Distributed, but Temporally Correlated Sources Using Second Order Statistics. *Neural Processing Letters* **12** (2000) 91-98
11. Zhang, Z.L., Yi, Z.: Robust Extraction of Specific Signals with Temporal Structure. *Neurocomputing* **69** (2006) 888-893
12. Barros, A.K., Cichocki, A.: Extraction of Specific Signals with Temporal Structure. *Neural Computation* **13** (9) (2001) 1995-2003
13. Liu, W., Mandic, D.P., Cichocki, A.: A Class of Novel Blind Source Extraction Algorithms Based on a Linear Predictor. *Proc. IEEE International Symposium on Circuits and Systems* (2005) 3599-3602
14. Anderson, B.D., Moore, J.B.: *Optimal Filtering*, Englewood Cliffs. NJ:Prentice-Hall, 1979
15. Cichocki, A., Rutkowski, T., Barros, A.K., Oh, S.H.: A Blind Extraction of Temporally Correlated but Statistically Dependent Acoustic Signals. *Proceedings of the 2000 IEEE Signal Processing Society Workshop* (2000) 455-464

Fast Code Detection Using High Speed Time Delay Neural Networks

Hazem M. El-Bakry¹ and Nikos Mastorakis²

¹Faculty of Computer Science & Information Systems,
Mansoura University, Egypt
helbakry20@yahoo.com

²Department of Computer Science,
Military Institutions of University Education (MIUE)-Hellenic Naval Academy,
Greece

Abstract. This paper presents a new approach to speed up the operation of time delay neural networks for fast code detection. The entire data are collected together in a long vector and then tested as a one input pattern. The proposed fast time delay neural networks (FTDNNs) use cross correlation in the frequency domain between the tested data and the input weights of neural networks. It is proved mathematically and practically that the number of computation steps required for the presented time delay neural networks is less than that needed by conventional time delay neural networks (CTDNNs). Simulation results using MATLAB confirm the theoretical computations.

1 Introduction

Recently, time delay neural networks have shown very good results in different areas such as automatic control, speech recognition, blind equalization of time-varying channel and other communication applications. The main objective of this research is to reduce the response time of time delay neural networks. The purpose is to perform the testing process in the frequency domain instead of the time domain. Our approach was successfully applied for sub-image detection using fast neural networks (FNNs) as proposed in [1,2,3]. Furthermore, it was used for fast face detection [7,9], and fast iris detection [8]. Another idea to further increase the speed of FNNs through image decomposition was suggested in [7].

FNNs for detecting a certain code in one dimensional serial stream of sequential data were described in [4,5]. Compared with conventional neural networks, FNNs based on cross correlation between the tested data and the input weights of neural networks in the frequency domain showed a significant reduction in the number of computation steps required for certain data detection [1,2,3,4,5,7,8,9,11,12]. Here, we make use of our theory on FNNs implemented in the frequency domain to increase the speed of time delay neural networks. The idea of moving the testing process from the time domain to the frequency domain is applied to time delay neural networks. Theoretical and practical results show that the proposed FTDNNs are faster than CTDNNs. In section 2, our theory on FNNs for detecting certain data in one dimensional matrix is described. Experimental results for FTDNNs are presented in section 3.

2 Fast Code Detection Using Cross Correlation in the Frequency Domain

Finding a certain code/data in the input one dimensional matrix is a searching problem. Each position in the input matrix is tested for the presence or absence of the required code/data. At each position in the input matrix, each sub-matrix is multiplied by a window of weights, which has the same size as the sub-matrix. The outputs of neurons in the hidden layer are multiplied by the weights of the output layer. When the final output is high, this means that the sub-matrix under test contains the required code/data and vice versa. Thus, we may conclude that this searching problem is a cross correlation between the matrix under test and the weights of the hidden neurons.

The convolution theorem in mathematical analysis says that a convolution of f with h is identical to the result of the following steps: let F and H be the results of the Fourier Transformation of f and h in the frequency domain. Multiply F and H^* (conjugate of H) in the frequency domain point by point and then transform this product into the spatial domain via the inverse Fourier Transform. As a result, these cross correlations can be represented by a product in the frequency domain. Thus, by using cross correlation in the frequency domain, speed up in an order of magnitude can be achieved during the detection process [1,2,3,4,5,7,8,9,14]. In the detection phase, a sub matrix I of size $1 \times n$ (sliding window) is extracted from the tested matrix, which has a size $1 \times N$, and fed to the neural network. Let W_i be the matrix of weights between the input sub-matrix and the hidden layer. This vector has a size of $1 \times n$ and can be represented as $1 \times n$ matrix. The output of hidden neurons $h(i)$ can be calculated as follows:

$$h_i = g \left(\sum_{k=1}^n W_i(k) I(k) + b_i \right) \quad (1)$$

where g is the activation function and $b(i)$ is the bias of each hidden neuron (i). Equation 1 represents the output of each hidden neuron for a particular sub-matrix I . It can be obtained to the whole input matrix Z as follows:

$$h_i(u) = g \left(\sum_{k=-n/2}^{n/2} W_i(k) Z(u+k) + b_i \right) \quad (2)$$

Eq.2 represents a cross correlation operation. Given any two functions f and d , their cross correlation can be obtained by:

$$d(x) \otimes f(x) = \left(\sum_{n=-\infty}^{\infty} f(x+n) d(n) \right) \quad (3)$$

Therefore, Eq. 2 may be written as follows [1]:

$$h_i = g(W_i \otimes Z + b_i) \quad (4)$$

where h_i is the output of the hidden neuron (i) and $h_i(u)$ is the activity of the hidden unit (i) when the sliding window is located at position (u) and $(u) \in [N-n+1]$.

Now, the above cross correlation can be expressed in terms of one dimensional Fast Fourier Transform as follows [1]:

$$W_i \otimes Z = F^{-1} \left(F(Z) \bullet F^* \left(W_i \right) \right) \tag{5}$$

Hence, by evaluating this cross correlation, a speed up ratio can be obtained comparable to conventional neural networks. Also, the final output of the neural network can be evaluated as follows:

$$O(u) = g \left(\sum_{i=1}^q W_o(i) h_i(u) + b_o \right) \tag{6}$$

where q is the number of neurons in the hidden layer. O(u) is the output of the neural network when the sliding window located at the position (u) in the input matrix Z. W_o is the weight matrix between hidden and output layer.

The complexity of cross correlation in the frequency domain can be analyzed as follows:

1- For a tested matrix of 1xN elements, the 1D-FFT requires a number equal to Nlog₂N of complex computation steps [13]. Also, the same number of complex computation steps is required for computing the 1D-FFT of the weight matrix at each neuron in the hidden layer.

2- At each neuron in the hidden layer, the inverse 1D-FFT is computed. Therefore, q backward and (1+q) forward transforms have to be computed. Therefore, for a given matrix under test, the total number of operations required to compute the 1D-FFT is (2q+1)Nlog₂N.

3- The number of computation steps required by FNNs is complex and must be converted into a real version. It is known that, the one dimensional Fast Fourier Transform requires (N/2)log₂N complex multiplications and Nlog₂N complex additions [13]. Every complex multiplication is realized by six real floating point operations and every complex addition is implemented by two real floating point operations. Therefore, the total number of computation steps required to obtain the 1D-FFT of a 1xN matrix is:

$$\rho = 6((N/2)\log_2 N) + 2(N\log_2 N) \tag{7}$$

which may be simplified to:

$$\rho = 5N\log_2 N \tag{8}$$

4- Both the input and the weight matrices should be dot multiplied in the frequency domain. Thus, a number of complex computation steps equal to qN should be considered. This means 6qN real operations will be added to the number of computation steps required by FNNs.

5- In order to perform cross correlation in the frequency domain, the weight matrix must be extended to have the same size as the input matrix. So, a number of zeros = (N-n) must be added to the weight matrix. This requires a total real number of computation steps = q(N-n) for all neurons. Moreover, after computing the FFT for the weight matrix, the conjugate of this matrix must be obtained. As a result, a real number of computation steps = qN should be added in order to obtain the conjugate of

the weight matrix for all neurons. Also, a number of real computation steps equal to N is required to create butterflies complex numbers ($e^{-jk(2\pi n/N)}$), where $0 < K < L$. These $(N/2)$ complex numbers are multiplied by the elements of the input matrix or by previous complex numbers during the computation of FFT. To create a complex number requires two real floating point operations. Thus, the total number of computation steps required for FNNs becomes:

$$\sigma = (2q+1)(5N \log_2 N) + 6qN + q(N-n) + qN + N \quad (9)$$

which can be reformulated as:

$$\sigma = (2q+1)(5N \log_2 N) + q(8N-n) + N \quad (10)$$

6- Using sliding window of size $1 \times n$ for the same matrix of $1 \times N$ pixels, $q(2n-1)(N-n+1)$ computation steps are required when using CTDNNs for certain code detection or processing (n) input data. The theoretical speed up factor η can be evaluated as follows:

$$\eta = \frac{q(2n-1)(N-n+1)}{(2q+1)(5N \log_2 N) + q(8N-n) + N} \quad (11)$$

3 Simulation Results

Time delay neural networks accept serial input data with fixed size (n). Therefore, the number of input neurons equals to (n). Instead of treating (n) inputs, our new approach is to collect all the input data together in a long vector (for example $100 \times n$). Then the input data is tested by time delay neural networks as a single pattern with length L ($L=100 \times n$). Such a test is performed in the frequency domain as described in section II. Complex-valued neural networks have many applications in fields dealing with complex numbers such as telecommunications, speech recognition and image processing with the Fourier Transform [6,10]. Complex-valued neural networks mean that the inputs, weights, thresholds and the activation function have complex values. In this section, formulas for the speed up ratio with different types of inputs will be presented. The special case of only real input values (i.e. imaginary part=0) will be considered. Also, the speed up ratio in the case of a one and two dimensional input matrix will be concluded. The operation of FNNs depends on computing the Fast Fourier Transform for both the input and weight matrices and obtaining the resulting two matrices. After performing dot multiplication for the resulting two matrices in the frequency domain, the Inverse Fast Fourier Transform is calculated for the final matrix. Here, there is an excellent advantage with FNNs that should be mentioned. The Fast Fourier Transform is already dealing with complex numbers, so there is no change in the number of computation steps required for FNNs. Therefore, the speed up ratio in the case of complex-valued time delay neural networks can be evaluated as follows:

1) In case of real inputs

A) For a one dimensional input matrix

Multiplication of (n) complex-valued weights by (n) real inputs requires $(2n)$ real operations. This produces (n) real numbers and (n) imaginary numbers. The addition

of these numbers requires $(2n-2)$ real operations. Therefore, the number of computation steps required by conventional neural networks can be calculated as:

$$\theta = 2q(2n-1)(N-n+1) \tag{12}$$

The speed up ratio in this case can be computed as follows:

$$\eta = \frac{2q(2n-1)(N-n+1)}{(2q+1)(5N\log_2 N) + q(8N-n) + N} \tag{13}$$

The theoretical speed up ratio for searching short successive (n) data in a long input vector (L) using complex-valued time delay neural networks is shown in Tables 1, 2, and 3. Also, the practical speed up ratio for manipulating matrices of different sizes (L) and different sized weight matrices (n) using a 2.7 GHz processor and MATLAB is shown in Table 4.

Table 1. The theoretical speed up ratio for time delay neural networks (1D-real values input matrix, $n=400$)

Length of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
10000	4.6027e+008	4.2926e+007	10.7226
40000	1.8985e+009	1.9614e+008	9.6793
90000	4.2955e+009	4.7344e+008	9.0729
160000	7.6513e+009	8.8219e+008	8.6731
250000	1.1966e+010	1.4275e+009	8.3823

Table 2. The theoretical speed up ratio for time delay neural networks (1D-real values input matrix, $n=625$)

Length of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
10000	7.0263e+008	4.2919e+007	16.3713
40000	2.9508e+009	1.9613e+008	15.0452
90000	6.6978e+009	4.7343e+008	14.1474
160000	1.1944e+010	8.8218e+008	13.5388
250000	1.8688e+010	1.4275e+009	13.0915

Table 3. The theoretical speed up ratio for time delay neural networks (1D-real values input matrix, $n=900$)

Length of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
10000	9.823 e+008	4.2911e+007	22.8933
40000	4.2206e+009	1.9612e+008	21.5200
90000	9.6176e+009	4.7343e+008	20.3149
160000	1.7173e+010	8.8217e+008	19.4671
250000	2.6888e+010	1.4275e+009	18.8356

Table 4. Practical speed up ratio for time delay neural networks (1D-real values input matrix)

Length of input matrix	Speed up ratio (n=400)	Speed up ratio (n=625)	Speed up ratio (n=900)
10000	17.88	25.94	35.21
40000	17.19	25.11	34.43
90000	16.65	24.56	33.59
160000	16.14	24.14	33.05
250000	15.89	23.76	32.60

B) For a two dimensional input matrix

Multiplication of (n^2) complex-valued weights by (n^2) real inputs requires $(2n^2)$ real operations. This produces (n^2) real numbers and (n^2) imaginary numbers. The addition of these numbers requires $(2n^2-2)$ real operations. Therefore, the number of computation steps required by conventional neural networks can be calculated as:

$$\theta = 2q(2n^2 - 1)(N - n + 1)^2 \quad (14)$$

The speed up ratio in this case can be computed as follows:

$$\eta = \frac{2q(2n^2 - 1)(N - n + 1)^2}{(2q + 1)(5N^2 \log_2 N^2) + q(8N^2 - n^2) + N} \quad (15)$$

The theoretical speed up ratio for detecting $(n \times n)$ real valued submatrix in a large real valued matrix $(N \times N)$ using complex-valued time delay neural networks is shown in Tables 5, 6, 7. Also, the practical speed up ratio for manipulating matrices of different sizes $(N \times N)$ and different sized weight matrices (n) using a 2.7 GHz processor and MATLAB is shown in Table 8.

Table 5. The theoretical speed up ratio for time delay neural networks (2D-real values input matrix, n=20)

Size of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	3.1453e+008	4.2916e+007	7.3291
200x200	1.5706e+009	1.9610e+008	8.0091
300x300	3.7854e+009	4.7335e+008	7.9970
400x400	6.9590e+009	8.8203e+008	7.8898
500x500	1.1091e+010	1.4273e+009	7.7711

Table 6. The theoretical speed up ratio for time delay neural networks (2D-real values input matrix, n=25)

Size of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	4.3285e+008	4.2909e+007	10.0877
200x200	2.3213e+009	1.9609e+008	11.8380
300x300	5.7086e+009	4.7334e+008	12.0602
400x400	1.0595e+010	8.8202e+008	12.0119
500x500	1.6980e+010	1.4273e+009	11.8966

Table 7. The theoretical speed up ratio for time delay neural networks (2D-real values input matrix, n=30)

Size of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	5.4413e+008	4.2901e+007	12.6834
200x200	3.1563e+009	1.9608e+008	16.0966
300x300	7.9272e+009	4.7334e+008	16.7476
400x400	1.4857e+010	8.8201e+008	16.8444
500x500	2.3946e+010	1.4273e+009	16.7773

Table 8. Practical speed up ratio for time delay neural networks (2D-real values input matrix)

Size of input matrix	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	17.19	22.32	31.74
200x200	17.61	22.89	32.55
300x300	16.54	23.66	33.71
400x400	15.98	22.95	34.53
500x500	15.62	22.49	33.32

2) In case of complex inputs

A) For a one dimensional input matrix

Multiplication of (n) complex-valued weights by (n) complex inputs requires (6n) real operations. This produces (n) real numbers and (n) imaginary numbers. The addition of these numbers requires (2n-2) real operations. Therefore, the number of computation steps required by conventional neural networks can be calculated as:

$$\theta = 2q(4n-1)(N-n+1) \tag{16}$$

The speed up ratio in this case can be computed as follows:

$$\eta = \frac{2q(4n-1)(N-n+1)}{(2q+1)(5N\log_2N) + q(8N-n) + N} \tag{17}$$

Table 9. The theoretical speed up ratio for time delay neural networks (1D-complex values input matrix, n=400)

Length of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	9.2111e+008	4.2926e+007	21.4586
200x200	3.7993e+009	1.9614e+008	19.3706
300x300	8.5963e+009	4.7344e+008	18.1571
400x400	1.5312e+010	8.8219e+008	17.3570
500x500	2.3947e+010	1.4275e+009	16.7750

The theoretical speed up ratio for searching short complex successive (n) data in a long complex-valued input vector (L) using complex-valued time delay neural

networks is shown in Tables 9, 10, and 11. Also, the practical speed up ratio for manipulating matrices of different sizes (L) and different sized weight matrices (n) using a 2.7 GHz processor and MATLAB is shown in Table 12.

Table 10. The theoretical speed up ratio for time delay neural networks (1D-complex values input matrix, n=625)

Length of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	1.4058e+009	4.2919e+007	32.7558
200x200	5.9040e+009	1.9613e+008	30.1025
300x300	1.3401e+010	4.7343e+008	28.3061
400x400	2.3897e+010	8.8218e+008	27.0883
500x500	3.7391e+010	1.4275e+009	26.1934

Table 11. The theoretical speed up ratio for time delay neural networks (1D-complex values input matrix, n=900)

Length of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	1.9653e+009	4.2911e+007	45.7993
200x200	8.4435e+009	1.9612e+008	43.0519
300x300	1.9240e+010	4.7343e+008	40.6410
400x400	3.4356e+010	8.8217e+008	38.9450
500x500	5.3791e+010	1.4275e+009	37.6817

Table 12. Practical speed up ratio for time delay neural networks (1D-complex values input matrix)

Length of input matrix	Speed up ratio (n=400)	Speed up ratio (n=625)	Speed up ratio (n=900)
10000	37.90	53.58	70.71
40000	36.82	52.89	69.43
90000	36.34	52.47	68.69
160000	35.94	51.88	68.05
250000	35.69	51.36	67.56

B) For a two dimensional input matrix

Multiplication of (n^2) complex-valued weights by (n^2) real inputs requires $(6n^2)$ real operations. This produces (n^2) real numbers and (n^2) imaginary numbers. The addition of these numbers requires $(2n^2-2)$ real operations. Therefore, the number of computation steps required by conventional neural networks can be calculated as:

$$\theta = 2q(4n^2 - 1)(N - n + 1)^2 \quad (18)$$

The speed up ratio in this case can be computed as follows:

$$\eta = \frac{2q(4n^2 - 1)(N - n + 1)^2}{(2q + 1)(5N^2 \log_2 N^2) + q(8N^2 - n^2) + N} \quad (19)$$

The theoretical speed up ratio for detecting (nxn) complex-valued submatrix in a large complex-valued matrix (N×N) using complex-valued neural networks is shown in Tables 13, 14, and 15. Also, the practical speed up ratio for manipulating matrices of different sizes (N×N) and different sized weight matrices (n) using a 2.7 GHz processor and MATLAB is shown in Table 16.

Table 13. The theoretical speed up ratio for time delay neural networks (2D-complex values input matrix, n=20)

Size of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	6.2946e+008	4.2916e+007	14.6674
200x200	3.1431e+009	1.9610e+008	16.0281
300x300	7.5755e+009	4.7335e+008	16.0040
400x400	1.3927e+010	8.8203e+008	15.7894
500x500	2.2197e+010	1.4273e+009	15.5519

Table 14. The theoretical speed up ratio for time delay neural networks (2D-complex values input matrix, n=25)

Size of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	8.6605e+008	4.2909e+007	20.1836
200x200	4.6445e+009	1.9609e+008	23.6856
300x300	1.1422e+010	4.7334e+008	24.1301
400x400	2.1198e+010	8.8202e+008	24.0333
500x500	3.3973e+010	1.4273e+009	23.8028

Table 15. The theoretical speed up ratio for time delay neural networks (2D-complex values input matrix, n=30)

Size of input matrix	Number of computation steps required for classic complex-valued neural networks	Number of computation steps required for fast complex-valued neural networks	Speed up ratio
100x100	1.0886e+009	4.2901e+007	25.3738
200x200	6.3143e+009	1.9608e+008	32.2021
300x300	1.5859e+010	4.7334e+008	33.5045
400x400	2.9722e+010	8.8201e+008	33.6981
500x500	4.7904e+010	1.4273e+009	33.5640

Table 16. Practical speed up ratio for time delay neural networks (2D-complex values input matrix)

Size of input matrix	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	38.33	46.99	62.88
200x200	39.17	47.79	63.77
300x300	38.44	48.86	64.83
400x400	37.92	47.23	65.99
500x500	37.32	46.89	64.89

4 Conclusion

New FTDNNs have been presented. Theoretical computations have shown that FTDNNs require fewer computation steps than conventional ones. This has been achieved by applying cross correlation in the frequency domain between the input data and the input weights of time delay neural networks. Simulation results have confirmed this proof by using MATLAB. This algorithm can be successfully applied to any application that uses time delay neural networks.

References

- [1] El-Bakry, H. M., Zhao, Q.: A Modified Cross Correlation in the Frequency Domain for Fast Pattern Detection Using Neural Networks. *International Journal of Signal Processing* **1** (2004) 188-194
- [2] El-Bakry, H. M., Zhao, Q.: Fast Object/Face Detection Using Neural Networks and Fast Fourier Transform. *International Journal of Signal Processing* **1** (2004) 182-187
- [3] El-Bakry, H. M., Zhao, Q.: Fast Pattern Detection Using Normalized Neural Networks and Cross Correlation in the Frequency Domain. accepted and under publication in the *EURASIP Journal on Applied Signal Processing*
- [4] El-Bakry, H. M., Zhao, Q.: A Fast Neural Algorithm for Serial Code Detection in a Stream of Sequential Data. *International Journal of Information Technology* **2** (2005) 71-90
- [5] El-Bakry, H.M., Stoyan, H.: FNNs for Code Detection in Sequential Data Using Neural Networks for Communication Applications. Proc. of the First International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA 2004 Orlando, Florida, USA, Vol. IV, 150-153.
- [6] Hirose, A.: Complex-Valued Neural Networks Theories and Applications. Series on innovative Intelligence **5** (2003)
- [7] El-Bakry, H.M.: Face detection using fast neural networks and image decomposition. *Neurocomputing Journal*, **48** (2002) 1039-1046
- [8] El-Bakry, H.M.: Human Iris Detection Using Fast Cooperative Modular Neural Nets and Image Decomposition. *Machine Graphics & Vision Journal (MG&V)* **11** (2002) 498-512
- [9] El-Bakry, H.M.: Automatic Human Face Recognition Using Modular Neural Networks," *Machine Graphics & Vision Journal (MG&V)* **10** (2001) 47-73
- [10] Jankowski, S., Lozowski, A., Zurada, M.: Complex-valued Multistate Neural Associative Memory. *IEEE Trans. on Neural Networks* **7** (1996) 1491-1496
- [11] El-Bakry, H.M., Zhao, Q.: Fast Pattern Detection Using Neural Networks Realized in Frequency Domain. Proc. of the International Conference on Pattern Recognition and Computer Vision, The Second World Enformatika Congress WEC'05, Istanbul, Turkey, (2005) 89-92
- [12] El-Bakry, H.M., Zhao, Q.: Sub-Image Detection Using Fast Neural Processors and Image Decomposition. Proc. of the International Conference on Pattern Recognition and Computer Vision, The Second World Enformatika Congress WEC'05, Istanbul, Turkey, (2005) 85-88
- [13] Cooley, J.W., Tukey, J.W.: An algorithm for the machine calculation of complex Fourier series. *Math. Comput* **19** (1965) 297-301
- [14] Klette, R., Zamperon: Handbook of image processing operators. John Wiley & Sons Ltd (1996)

Multiscale Estimation to the Parameter of Multidimension Time Series*

Cheng-Lin Wen¹, Guang-Jiang Wang², Chuan-Bo Wen², and Zhi-Guo Chen²

¹ Hangzhou Dianzi university,
Hangzhou 310018, China
wenc1@hdu.edu.cn

² School of Computer and Information Engineering, Henan University,
Kaifeng475001, China
wgjiang2006@hdu.edu.cn

Abstract. During preceding theory study and engineering application, we dealt with the parameter estimation of one-dimension long memory process actually, and rarely take into account high dimensions. There are few papers about it. In this paper, using the decorrelation property of discrete wavelet transform, high dimension situation (mainly 2D) is simplified to 1D and corresponding referers are improved according to new idea, combining with matrix transform. So the computation complexity is reduced effectively and estimation precision is satisfied. Some experiment results show that this algorithm has a better general performance.

1 Introduction

In theory study and engineering, there is a sort of time series or stochastic process, called long memory processes with the property that the process which is widely separated still has a nonnegligible correlation [1]. Just for their wide existence in many fields, research of this type of processes has great scientific value and broad application background [2, 3]. Especially estimation of process parameters is very popular in current research.

For the long term correlation of long memory process, there exists a problem in Traditional Maximum Likelihood Estimation (TMLE), which is the great computational burden because of the determination and matrix inverting are computationally expensive to evaluate, even to moderate the length of time series. This leads to less efficient implement to the theory. Wavelets are useful tools to handle the signal, and good at analyzing the object both in time and frequency domains locally. Due to decorrelating even highly correlated series and having the orthonormal feature, wavelet is studied in scientific work broadly, and be given an honorary name-- "mathematic microscope" [4].

* This work is partially supported by NSFC (60434020, 60374020), International Cooperative Project Foundation (0446650006), Henan Outstanding Youth Science Fund (0312001900), and Ministry of Education Science Foundation (205092).

The paper developed a new method combining wavelet transform with TMLE, named MMLE [5], simplified previous given two computations with local analysis of wavelet, and applied it to a kind of long multivariate memory processes with broad application background. The approach decreases the computation burden when estimate the parameter with satisfied precision. For the convenience, the paper mainly aimed at two-dimension long memory process without losing generaltion.

2 Model for Long Memory Process and Traditional Estimation to Parameters

2.1 Model for Long Memory Process [6]

Definition 1. we call $\{\mathbf{x}(k)\}$ as long memory process, if the spectral density function (SDF) of stochastic process $\{\mathbf{x}(k)\}$ satisfies next two conditions,

$$(1) S_{\mathbf{x}}(f) \propto |f|^{-\alpha}$$

$$(2) S_{\mathbf{x}}(f) \rightarrow \infty, \quad \text{当 } \alpha < 0, \quad f \rightarrow 0$$

Definition 2. pure power low time series (PPLTS)

We say that discrete parameter time series $\{\mathbf{x}(k)\}$ is a pure power law time series if its SDF has the form.

$$S_{\mathbf{x}}(f) = C_S |f|^{-\alpha}, \quad -\frac{1}{2} \leq f \leq \frac{1}{2}, \quad C_S > 0$$

For $-1 < \alpha < 0$, the PPLTS is the stationary long memory time series.

Definition 3. cross-spectrum

The cross-spectrum matrix of two dimension time series $\{\mathbf{z}(k)\}$ can be obtained via the formula

$$S_{\mathbf{z}}(f) = \begin{bmatrix} S_{\mathbf{x}} & S_{\mathbf{xy}} \\ S_{\mathbf{yx}} & S_{\mathbf{y}} \end{bmatrix},$$

where $S_{\mathbf{xy}} = S_{\mathbf{yx}}^* = \frac{1}{2\pi} \sum_{t=-\infty}^{\infty} e^{-itf} \cdot \gamma_{\mathbf{xy}}(t)$ is the cross spectral density, and $\gamma_{\mathbf{xy}}$ is cross covariance.

Cross covariance matrix of the stationary time series is written be

$$\mathbf{P}(\tau) = E \left\{ [\mathbf{z}(k) - E\{\mathbf{z}(k)\}] [\mathbf{z}(k + \tau) - E\{\mathbf{z}(k + \tau)\}]^T \right\}$$

where τ is the integer, symbol $E\{\cdot\}$ denotes the expectation; Superscript T denotes transpose. Without losing generally, we only studied the time series with zero mean in paper.

2.2 Traditional Maximum Likelihood Estimation to Parameters of Two-Dimensions Process

Suppose now that we have a two dimension time sequence $\{z(k)\} = \begin{Bmatrix} x(k) \\ y(k) \end{Bmatrix}$, where two components $\{x(k)\}$ 、 $\{y(k)\}$ with zero mean and unknown parameters $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_n]$, and its cross covariance matrix is

$$\begin{aligned} P_z(\tau) &= E\{z(k)z^T(k+\tau)\} \\ &= \begin{bmatrix} E\{x(k)x^T(k+\tau)\} & E\{x(k)y^T(k+\tau)\} \\ E\{y(k)x^T(k+\tau)\} & E\{y(k)y^T(k+\tau)\} \end{bmatrix} \\ &= \begin{bmatrix} P_x(\tau) & P_{xy}(\tau) \\ P_{yx}(\tau) & P_y(\tau) \end{bmatrix} \end{aligned} \tag{1}$$

where $P_{xy}(\tau) = P_{yx}(\tau)$, and suppose $P_x(\tau)$ 、 $P_{xy}(\tau)$ 、 $P_y(\tau)$ 、 $P_{yx}(\tau)$ are descending functions about variate τ .

Define a portion of a time series

$$Z(k, k+N-1) = [z^T(k), z^T(k+1), \dots, z^T(k+N-1)]$$

or

$$Z(k, k+N-1) = \begin{bmatrix} X(k, k+N-1) \\ Y(k, k+N-1) \end{bmatrix}$$

where

$$\begin{aligned} X(k, k+N-1) &= [x(k), x(k+1), \dots, x(k+N-1)]^T \\ Y(k, k+N-1) &= [y(k), y(k+1), \dots, y(k+N-1)]^T \end{aligned}$$

The realization of time series is a two dimension scalar with corresponding length.

$$Z(k, k+N-1) = [z^T(k), z^T(k+1), \dots, z^T(k+N-1)]$$

or

$$Z(k, k+N-1) = \begin{bmatrix} X(k, k+N-1) \\ Y(k, k+N-1) \end{bmatrix}$$

where

$$\begin{aligned} X(k, k+N-1) &= [x(k), x(k+1), \dots, x(k+N-1)]^T \\ Y(k, k+N-1) &= [y(k), y(k+1), \dots, y(k+N-1)]^T \\ x(k), y(k) &\in R^1; k = 1, 2, \dots \end{aligned}$$

Under the assumption that $Z(k, k+N-1)$ obeys a multivariate Gaussian distribution, the likelihood function [7] is given by

$$L(\alpha|Z(k, k+N-1)) \equiv \frac{1}{(2\pi)^{N/2} |P_Z|^{1/2}} e^{-\frac{1}{2} Z P_Z^{-1} Z} \tag{2}$$

where

$$P_Z = E\{Z(k, k+N-1)Z^T(k, k+N-1)\}$$

Now maximization of (2) is equivalent to minimization of the log likelihood function

$$\begin{aligned} l(\boldsymbol{\alpha}|Z(k, k+N-1)) &\equiv -2\log(L(\boldsymbol{\alpha}|Z(k, k+N-1))) - N\log(2\pi) \\ &= \log(|\mathbf{P}_Z|) + Z^T \mathbf{P}_Z^{-1} Z. \end{aligned} \quad (3)$$

We can obtain estimated parameter $\hat{\boldsymbol{\alpha}} = [\hat{\alpha}_1, \hat{\alpha}_2, \dots, \hat{\alpha}_n]$ by solving likelihood function (3).

In theory, we can get the estimation of parameter α utilizing traditional likelihood estimation method. However, one of the bottlenecks of TMLE is its great computational burdens. $L(\boldsymbol{\alpha}|z)$ can be very time consuming, because determination $|\mathbf{P}_Z|$ and inverse of \mathbf{P}_Z are computationally expensive to evaluate, even for moderate N [8]. So researches about decreasing the computation complexity of likelihood function are valuable both on academic and practical.

3 Multiscale Estimation for Two-Dimensions Time Series

3.1 Basic Theory of Wavelet Transform

A wavelet filter $\{h_l : l=0, \dots, L-1\}$, where L is the width of the filter and must be an even integer, must satisfy the following three properties:

$$\sum_{l=0}^{L-1} h_l = 0 \quad (4)$$

$$\sum_{l=0}^{L-1} h_l^2 = 1 \quad (5)$$

$$\sum_{l=0}^{L-1} h_l^2 = 1 \quad (6)$$

for all nonzero integers n .

The required second filter is the ‘quadrature mirror’ filter (QMF) $\{g_l\}$ that corresponds to $\{h_l\}$ [4]:

$$g_l = (-1)^{l+1} h_{L-1-l} \quad (7)$$

According (5)、(7), the filters satisfy

$$\sum_{l=0}^{L-1} g_l^2 = 1 \quad (8)$$

$$\sum_{l=0}^{L-1} h_l g_l = 0 \tag{9}$$

3.2 Multiscale Transform for Two-Dimension Time Series

In order to represent wavelet transform theory conveniently, we express a time series segment $\mathbf{Z}(k, k + N - 1) = [\mathbf{z}^T(k), \mathbf{z}^T(k + 1), \dots, \mathbf{z}^T(k + N - 1)]^T$ as

$$\begin{aligned} \mathbf{Z}^{(m)} &:= \mathbf{Z}_V^{(m)} := L_M \cdot \mathbf{Z}(k, k + N - 1) = \begin{bmatrix} \mathbf{X}(k, k + N - 1) \\ \mathbf{Y}(k, k + N - 1) \end{bmatrix} \\ &=: \begin{bmatrix} \mathbf{X}_V^{(m)} \\ \mathbf{Y}_V^{(m)} \end{bmatrix} = [x_V^m(0), \dots, x_V^m(N - 1), y_V^m(0), \dots, y_V^m(N - 1)]^T \end{aligned} \tag{m = M, M - 1, \dots, L}$$

where subscript V denotes the smoothing space where get time series. Let superscript M be finest scale and L_M is the linear operator. Suppose $\bar{\mathbf{H}}_i$ and $\bar{\mathbf{G}}_i$ are the scale operator and wavelet operator to make wavelet transform for one dimension time series $\mathbf{X}(k, k + N - 1)$ and $\mathbf{Y}(k, k + N - 1)$ separately. So the decomposition transformation from (m) th scale to $(m - 1)$ th scale can be written be

$$\mathbf{Z}_V^{(m-1)} = \mathbf{H}_m \mathbf{Z}_V^{(m)} \tag{10}$$

$$\mathbf{Z}_D^{(m-1)} = \mathbf{G}_m \mathbf{Z}_V^{(m)} \tag{11}$$

and we can recover $\mathbf{Z}_V^{(m)}$ using the reconstruction equation

$$\mathbf{Z}_V^{(m)} = H_{m-1}^* \mathbf{Z}_V^{(m-1)} + G_{m-1}^* \mathbf{Z}_D^{(m-1)} \tag{12}$$

where superscript $*$ denotes matrix conjugate transpose, then the smooth operator and detailed operator in (10)-(12) can be expressed as[9]

$$\mathbf{H}_m = \text{diag}\{\bar{\mathbf{H}}_{m-1}, \bar{\mathbf{H}}_{m-1}\} \tag{13}$$

$$\mathbf{G}_m = \text{diag}\{\bar{\mathbf{G}}_{m-1}, \bar{\mathbf{G}}_{m-1}\} \tag{14}$$

Moreover the smooth sequence and detailed sequence in (12) have explicit form

$$\begin{aligned} \mathbf{Z}_V^{(m-1)} &= \begin{bmatrix} \mathbf{X}_V^{(m-1)} \\ \mathbf{Y}_V^{(m-1)} \end{bmatrix} = [x_V^{m-1}(0), \dots, x_V^{m-1}(N/2), y_V^{m-1}(0), \dots, y_V^{m-1}(N/2)]^T \\ \mathbf{Z}_D^{(m-1)} &= \begin{bmatrix} \mathbf{X}_D^{(m-1)} \\ \mathbf{Y}_D^{(m-1)} \end{bmatrix} = [x_D^{m-1}(0), \dots, x_D^{m-1}(N/2), y_D^{m-1}(0), \dots, y_D^{m-1}(N/2)]^T \end{aligned}$$

We claim that

$$\gamma = [(\mathbf{X}_D^{M-1})^T, (\mathbf{X}_D^{M-2})^T, \dots, (\mathbf{X}_D^L)^T, (\mathbf{X}_V^L)^T, (\mathbf{Y}_D^{M-1})^T, (\mathbf{Y}_D^{M-2})^T, \dots, (\mathbf{Y}_D^L)^T, (\mathbf{Y}_V^L)^T] \tag{15}$$

Then the decomposition and reconstruction can be written be

$$\gamma(M) = \mathbf{W}_Z \mathbf{Z}(M) . \tag{16}$$

$$\mathbf{Z}(M) = \mathbf{W}_Z^* \gamma(M) . \tag{17}$$

where \mathbf{W}_Z is called wavelet operator

$$\mathbf{W}_Z = \begin{bmatrix} \mathbf{G}_{M-1} \\ \mathbf{G}_{M-2} \mathbf{H}_{M-1} \\ \mathbf{G}_{M-3} \mathbf{H}_{M-2} \mathbf{H}_{M-1} \\ \vdots \\ \mathbf{G}_L \mathbf{H}_{L+1} \mathbf{H}_{L+2} \cdots \mathbf{H}_{M-1} \\ \mathbf{H}_L \mathbf{H}_{L+1} \cdots \mathbf{H}_{M-1} \end{bmatrix} . \tag{18}$$

and because of orthonormal ,satisfies

$$\mathbf{W}_Z^* \mathbf{W}_Z = \mathbf{I} . \tag{19}$$

Theorem 1. Let $\gamma = \mathbf{W}_Z \mathbf{Z}$, we can obtain

$$\mathbf{P}_\gamma = E\{\gamma\gamma^T\} = (\mathbf{W}_Z)E\{\mathbf{Z}\mathbf{Z}^T\}(\mathbf{W}_Z)^T = (\mathbf{W}_Z)P_Z(\mathbf{W}_Z)^T$$

3.3 Multiscale Analysis to the Property of Two-Dimension Sequence

Theorem 2

$$P_{ab,CF}^{(M-1)}(\tau) = E\left\{a_C^M(k)b_F^T(k+\tau)\right\} , \quad \text{where } a, b \in (x,y) , \quad C, F \in (V,D) ,$$

$$P_{ab,CF}^{(M-1)}(\tau) = \delta_{CF} P_{ab,CF}^{(M-1)}(2\tau) .$$

It's convenient that we just made the decomposition once and get the explicit form of (1)

$$\mathbf{P}_\gamma^{(M-1)} = E \begin{bmatrix} X_V^{(M-1)}(X_V^{(M-1)})^T & X_V^{(M-1)}(X_D^{(M-1)})^T & X_V^{(M-1)}(Y_V^{(M-1)})^T & X_V^{(M-1)}(Y_D^{(M-1)})^T \\ X_D^{(M-1)}(X_V^{(M-1)})^T & X_D^{(M-1)}(X_D^{(M-1)})^T & X_D^{(M-1)}(Y_V^{(M-1)})^T & X_D^{(M-1)}(Y_D^{(M-1)})^T \\ Y_V^{(M-1)}(X_V^{(M-1)})^T & Y_V^{(M-1)}(X_D^{(M-1)})^T & Y_V^{(M-1)}(Y_V^{(M-1)})^T & Y_V^{(M-1)}(Y_D^{(M-1)})^T \\ Y_D^{(M-1)}(X_V^{(M-1)})^T & Y_D^{(M-1)}(X_D^{(M-1)})^T & Y_D^{(M-1)}(Y_V^{(M-1)})^T & Y_D^{(M-1)}(Y_D^{(M-1)})^T \end{bmatrix}$$

According above knowledge, we got the result

$$\mathbf{P}_\gamma^{(M-1)}(\tau) = \begin{bmatrix} \mathbf{P}_x^M(2\tau) & 0 & \mathbf{P}_{xy}^M(2\tau) & 0 \\ 0 & \mathbf{P}_x^M(2\tau) & 0 & \mathbf{P}_{xy}^M(2\tau) \\ \mathbf{P}_{yx}^M(2\tau) & 0 & \mathbf{P}_y^M(2\tau) & 0 \\ 0 & \mathbf{P}_{yx}^M(2\tau) & 0 & \mathbf{P}_y^M(2\tau) \end{bmatrix} . \tag{20}$$

3.4 Multiscale Covariance Analysis

Lemma 1 [10]

Suppose that block matrixes satisfy $M = \begin{bmatrix} A_1 & A_2 \\ \mathbf{0} & A_4 \end{bmatrix}$ or $M = \begin{bmatrix} A_1 & \mathbf{0} \\ A_3 & A_4 \end{bmatrix}$, moreover

$A_1 \in R^{m \times m}$, $A_4 \in R^{m \times m}$, show that we also have $|M| = |A_1| \cdot |A_4|$.

Lemma 2 [10]

Suppose that block matrix is $M = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix}$, and A_1 is invertible, show that we have

$$|M| = \left| A_1 (A_4 - A_3 A_1^{-1} A_2) \right|.$$

Lemma 3 [11]

Suppose that block matrix is $M = \begin{bmatrix} A_1 & A_2 \\ A_3 & A_4 \end{bmatrix}$, and A_1 is m dimension and invertible.

Then let $B_1 = A_4 - A_3 A_1^{-1} A_2$, $B_2 = A_1 - A_2 A_4^{-1} A_3$, show that we have

$$M^{-1} = \begin{bmatrix} B_2^{-1} & -A_1^{-1} A_2 B_1^{-1} \\ -A_4^{-1} A_3 B_2^{-1} & B_1^{-1} \end{bmatrix}$$

According above knowledge, we obtained the result

$$|\mathbf{P}_\gamma| = \left| \mathbf{P}_{XX} (\mathbf{P}_{YY} - \mathbf{P}_{YX} \mathbf{P}_{XX}^{-1} \mathbf{P}_{XY}) \right|. \tag{21}$$

Then \mathbf{P}_γ became a diagonal matrix.

With above theorem 3.3, we got the result

$$\mathbf{P}_\gamma^{-1} = \begin{bmatrix} \mathbf{B}_2^{-1} & \mathbf{P}_{XX}^{-1} \mathbf{P}_{XY} \mathbf{B}_1^{-1} \\ -\mathbf{P}_{YY}^{-1} \mathbf{P}_{YX} \mathbf{B}_1^{-1} & \mathbf{B}_1^{-1} \end{bmatrix}. \tag{22}$$

Where $\mathbf{B}_1 = \mathbf{P}_{YY} - \mathbf{P}_{YX} \mathbf{P}_{XX}^{-1} \mathbf{P}_{XY}$, $\mathbf{B}_2 = \mathbf{P}_{XX} - \mathbf{P}_{XY} \mathbf{P}_{YY}^{-1} \mathbf{P}_{XY}$, then \mathbf{P}_γ^{-1} is simplified.

Remark 1. Using MMLE, when we estimated parameters, not only correlation in each component become weaker but also between components. The non-diagonal elements in every diagonal block are very small, so we can approximate every small matrix by a diagonal matrix. Computation complexity decreases to $O(N)$ from $O(N^3)$.

3.5 Simulation and Precision Analysis

In this subsection, the setup of simulation environment and the statistics are listed in table 1 and 2 respectively. Studied object is a long memory process- pure power process(PPLP)

Table 1. Simulation environment

selected parameters α	the length of sampled series N	repeating times M	selected wavelet filter
$\alpha_1 = -0.4$	256	1024	Daubechies(4)
$\alpha_2 = -0.75$			

Table 2. Researched statistic

Sample Mean $\bar{\alpha}$	ample bias $\tilde{\alpha}$	Sample standard deviation (SD)	Root Mean Square Error (RMSE)
$\frac{1}{M} \sum_{i=1}^M \hat{\alpha}^{(i)}$	$ \bar{\alpha} - \alpha $	$\left(\frac{1}{M} \sum_{i=1}^M (\hat{\alpha}^{(i)} - \bar{\alpha})^2 \right)^{1/2}$	$\left(\frac{1}{M} \sum_{i=1}^M (\hat{\alpha}^{(i)} - \alpha)^2 \right)^{1/2}$

Table 3. Performance comparison

α	Traditional MLE	MMLE	Precision losing
sample bias ($\tilde{\alpha}$)	0.0056	0.0157	2.525%
standard bias (SD)	0.0577	0.0625	7.682%
Root Mean Square Error RMSE	0.0591	0.0644	8.229%
sample bias ($\tilde{\alpha}$)	0.0089	0.0276	2.493%
standard bias (SD)	0.0330	0.0348	5.455%
Root Mean Square Error RMSE	0.0366	0.0389	6.284%
Complexity comparison	10^7	10^2	

From table 3, we can find the precision loss is rather small if using the method we propose in paper, but, compared with traditional MLE, computation complexity decreases greatly. So MSMLE and MSLSE’s performance is quite well. And the Multi-Scale estimation method can be used as a good alternative of parameter estimation.

References

1. Gramnger, C.W.J.:An Introduction to Long Memory Time Series Models, Stochastic process Applications **8** (1981)87-92
2. Geweke, Porter-Hydak, S.: The Estimation and Application of Long Memory Time Series Models, Journal of Time Series Analysis **4** (1983) 221-238
3. Bjorn, V. (ed.): Multiresolution Methods for Financial Time Series Prediction, Computational Intelligence for Financial Engineering, Proceedings of the IEEE/IAFE (1995)

4. Wen, C., Wang, S.: The Multi-Scale Maximum Likelihood Estimation of Long Memory Processes, ICNN&B **1** (2005) 312-317,.
5. Wen, C., Zhou, D.: Multiscale Theory and Its Application [M]. Beijing: Tinghua press, (2002)
6. Percival, D.B., Andrew, T.W.: Wavelet Methods for Time Series Analysis [M], China Machine Press (2004)
7. Zhong-Ke Shi (ed.): Algorithm of Optimal estimation. Beijing : Science Press (2001).
8. Daubechies.: Ten Lectures on Wavelets, Philadelphia, Pennsylvania: Society for industrial and Appl. Math., (1992).
9. Wen, C., Wang, G.: Study on Multiscale Least Squares and Application in Parameter Identification, ICMLC I (2006) 1625-1629
10. Du, S.: Arithmetic Study of Block Matrix Determination. Sichuan Arts and Science College (Natural Science Edition) **9** (1999)
11. Ming-Zhen Su(ed.): Algorithm Research of Matrix Inversing. Education College Transaction **5** (1999).

Gaussianization Based Approach for Post-Nonlinear Underdetermined BSS with Delays

Alessandro Bastari, Stefano Squartini, Stefania Cecchi, and Francesco Piazza

A3Lab, DEIT, Università Politecnica delle Marche,
Via Breccie Bianche 31, 60131 Ancona, Italy
{a.bastari,sts@deit,s.cecchi,f.piazza}@univpm.it

Abstract. The principal aim of this work is to demonstrate, from an empirical point of view, the effectiveness of a previously proposed technique for the Blind Source Separation (BSS) with Post Non Linear (PNL) underdetermined instantaneous mixing model (uBSS), in the more complex and realistic case where delayed sources are considered in the linear part of the mixing (PNL-uBSS with delays). The proposed approach is composed of two consecutive stages: in the first stage the inverse nonlinearities are estimated by Gaussianization of the mixtures; in the second stage source signals are extracted from the linearized mixtures using a three step approach already known in the literature for linear delayed uBSS. An improved technique based on Extended Gaussianization is also provided for the estimation of inverse nonlinearities. Experimental results using synthetic mixtures of real world data (speech signals) are given to prove the effectiveness of the proposed approach.

1 Introduction

The Blind Source Separation (BSS) problem deals with the identification and separation of unknown source signals, only knowing a certain number of their mixtures, when the generative mixing model is supposed to be unknown as well. The scientific community has paid more and more attention to the BSS problem in the last decades, because of its potential applications in several fields of signal processing, as in telecommunications or in bioinformatics, just to name a few.

To make the BSS problem solvable, some further assumptions on the generating mixing model and/or knowledge on the source statistics (e.g. the source a priori distribution) have to be introduced. Different formulations and proposed approaches can be found in the literature, specially focusing on linear quadratic mixing models [1], where as many mixtures as the sources to be recovered are supposed to be available for the separation process. In such a case, under the only assumption of statistically independent source signals, the separation problem becomes formally equivalent to a Independent Component Analysis (ICA) problem, so that classical ICA techniques can be used [2]. A more challenging but complex situation is obtained considering nonlinear mixing models. Indeed it

has been shown [3] that for these kinds of model, also under the only assumption of statistically independent source signals, the original signals can be recovered only up to a nonlinear distortion, that is not allowable in most practical applications. For this reason only particular kinds of nonlinearities must be used. A particularly interesting nonlinear mixing model is the Post Non Linear (PNL) model [3], that can be used in all real situations where a system with a linear transmission channel and with sensors introducing nonlinear memoryless distortions (e.g.: saturation-like distortions) has to be modeled. The quadratic PNL BSS problem is separable (i.e. it admits unique solution up to the usual permutation and scaling indeterminacies) under the only assumption of statistically independent source signals.

In this work the underdetermined (number of mixtures smaller than number of sources) PNL mixing model with delays is studied, and it can be considered a more realistic improvement of a previously described technique [4]. It is well known that in the overcomplete case also when the mixing model is supposed to be available, the separation problem cannot be considered solved. Sources are usually recovered by an estimation process exploiting information about the a priori amplitude distribution. Therefore the separation procedure can be split in two consecutive stages: in the first one nonlinearities are compensated, obtaining a linear underdetermined BSS problem; in the succeeding stage classical linear BSS techniques can be used. Here nonlinearities are recovered by Gaussianization of the known mixtures and a subsequent step for compensation of residual nonlinearities (Extended Gaussianization); a three step approach for sparse sources [5] is used for the linear delayed demixing. It should be highlighted since now that the Gaussianization technique does not lead to an exact estimation of nonlinearities, but it is particularly useful when the nonlinearities does not allow to get any estimation of the mixing model, making the separation not feasible. Experimental results given in Sec. 4 support these remarks.

2 Delayed Post Nonlinear BSS Mixing Model

Let $s_1(t), \dots, s_n(t)$, for $t = 1, \dots, T$, be the n unknown source signals to be recovered only knowing the m mixtures signals $x_1(t), \dots, x_m(t)$. Denoting $\mathbf{s}(t)$ and $\mathbf{x}(t)$ the source and the mixture vectors respectively, the nonlinear delayed mixing model can be written in the most general way as:

$$\mathbf{x}(t) = \begin{bmatrix} h_1(s_1(t - \tau_{11}) \cdots s_n(t - \tau_{1n})) \\ \vdots \\ h_m(s_1(t - \tau_{m1}) \cdots s_n(t - \tau_{mn})) \end{bmatrix}, \mathbf{H}(\cdot) = \begin{bmatrix} h_1(\cdot) \\ \vdots \\ h_m(\cdot) \end{bmatrix} : \mathbf{R}^n \rightarrow \mathbf{R}^m,$$

where $\mathbf{H}(\cdot)$ is an unknown nonlinear mapping to be estimated. If a generic non linearity $\mathbf{H}(\cdot)$ is used, also considering instantaneous ($\tau_{ij} = 0, \forall i, j$) and complete ($m = n$) mixing for independent sources, we can not get a solution to the BSS problem. A specific kind of nonlinearity has to be chosen to make the problem solvable: following the notation in Fig. 1, the generic mixture signal is

supposed to be generated from a first linear mixing with delays, followed by a component wise non linearity $H(\cdot)$, according to the following generative model:

$$x_i(t) = h_i \left(\sum_{j=1}^n \beta_{ij} s_j(t - \tau_{ij}) \right), \quad \text{for } i = 1, \dots, m, \quad (1)$$

where τ_{ij} and β_{ij} are unknown real-valued parameters, describing attenuations and delays from the j -th source to the i -th sensor respectively. The n sources $s_j(t)$ are supposed statistically independent and sparse (modeled by a Laplacian distribution), whilst the nonlinearities $h_j(\cdot)$ must be invertible and derivable and $\mathbf{B} = [\beta_{ij}]$ a full rank matrix. Using the PNL model, the non linearity recovery stage is independent from the successive linear demixing, so that the last stage can be accomplished by using standard linear techniques for delayed BSS.

3 A Practical Approach to Separation

In this paragraph a new approach for independent source estimation for delayed PNL-uBSS problem is introduced. The separation procedure is composed of two successive stages: in the first one Gaussianization and Extended Gaussianization techniques are used to get the inverse nonlinearities g_i (Fig. 1), such that $g_i \circ h_i$ are as much linear as possible. After nonlinearity compensation a linear delayed uBSS problem needs to be solved; for this purpose in this work a three step approach [5] is used, made of: 1) geometric technique (potential function) for attenuation matrix \mathbf{B} recovery; 2) iterative technique for differential delays recovery; 3) Maximum Likelihood (ML) based technique for source estimation.

3.1 First Stage: Non Linearity Recovery Using Gaussianization

The non-linearity recovery step is based on two fundamental considerations. The first one is that the amplitude distribution of u_i (due to Central limit Theorem) is closer to a Gaussian distribution than the independent components s_i , whilst the applied nonlinearity h_i tends to distort it. Therefore a practical statistical approach for nonlinearity compensation could be defined that makes use of nonlinear functions g'_i to transform the mixture signals x_i in new signals v_i , with Gaussian amplitude distribution (Fig. 1). This transformation is called Gaussianization and has been proposed in [7].

The second fundamental consideration is that (due to the practical limitation of Central Limit Theorem) the amplitude distribution of u_i is not exactly Gaussian, but we assume it as close-to-Gaussian distribution. In order to obtain a better nonlinearity compensation we can determine a second function g''_i using a parametric model based on the Cornish-Fisher Expansion [6], so that the output signals z_i is as similar as possible to statistical amplitude distribution of u_i . The global Post Non-Linear compensation functions are obtained by composition of either g'_i and g''_i ($g_i = g'_i \circ g''_i$). After nonlinearity compensation the signals z_i could be considered as linear mixture with delays, so that we could apply all known approaches to solve Linear uBSS with delays problems.

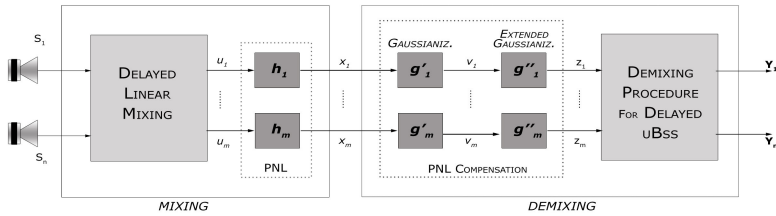


Fig. 1. PNL-uBSS with delays problem: mixing model and separation structure

Gaussianization. Gaussianization is an invertible transformation $T(\xi)$ that transforms a random variable into a standard Gaussian random variable with zero-mean and unitary variance. The Gaussianization transformation [7], [9] can be written as

$$T(\xi) = \Phi^{-1}(F(\xi)), \tag{2}$$

where $\Phi(\cdot)$ is the Standard Normal Distribution and $F(\cdot)$ is the cumulative distribution function (CDF) of input signal. $F(\cdot)$ is an unknown function, and it can be estimated in different ways; for the sake of easiness, we can use empirical CDF (ecdf) methods

$$\hat{F}(\xi) = \frac{1}{N+1} \sum_{k=1}^N I(\xi^{(k)} \leq \xi), \tag{3}$$

where N is the number of samples, $\xi^{(k)}$ is the k -th sample, and $I(\xi^{(k)} \leq \xi)$ is a function with value 1 if $\xi^{(k)} \leq \xi$ and 0 otherwise. A low-computational implementation method for Gaussian transformation, limiting the influence of outliers is the following

$$v^{(k)} = \Phi^{-1} \left(c \frac{\text{rank}(u^{(k)})}{N+1} + \frac{1}{2}(1-c) \right), \tag{4}$$

where $\text{rank}(u^{(k)})$ indicates the position of $u^{(k)}$ in sorted (from smallest to largest) sample vector, and c value is application dependent, but not critical.

Cornish-Fisher Expansion to Improve Nonlinearity Compensation. It is known that for an ideal Gaussian distribution all cumulants of order higher than two are equal to zero, whereas a distribution close to Gaussian, such as x_i amplitude distribution, can be modeled using an estimation of them. With Extended Gaussianization method, we want to create a parametric function to model the remaining nonlinearity after Gaussian transformation, using v_i as input and skewness and kurtosis as parameters. Cornish and Fisher have pro-pounded a particular expansion in which the terms are polynomial functions

of the standard Gaussian quantiles, with functions of known cumulants as coefficients [6]. The four-term Cornish-Fisher (C-F) expansion for α -quantile I is

$$\xi(\eta_\alpha) \simeq \eta_\alpha + \frac{1}{6}(\eta_\alpha^2 - 1)k_3 + \frac{1}{24}(\eta_\alpha^3 - 3\eta_\alpha)k_4 - \frac{1}{36}(2\eta_\alpha^3 - 5\eta_\alpha)k_3^2, \quad (5)$$

where k_3 and k_4 are skewness and kurtosis of x respectively, η_α is the α -quantile of the standard Gaussian distribution, and $\xi(\eta_\alpha)$ is zero-mean and of unitary variance. Since the distribution of z_i is assumed to be close to Gaussian and v_i is the outcome of the Gaussian transformation, the C-F expansion can be used to approximate z_i in terms of v_i

$$z_i \simeq v_i + \frac{1}{6}(v_i^2 - 1)k_{3,i} + \frac{1}{24}(v_i^3 - 3v_i)k_{4,i} - \frac{1}{36}(2v_i^3 - 5v_i)k_{3,i}^2. \quad (6)$$

In [6] the parameters k_3 and k_4 come out from the adaptation process responsible for finding the independent components contained in the PNL mixtures, assuming to be in the complete case. In our overcomplete case, we give a practical estimation of them by using the ones related to the v_i variables, exploiting the approximation limits of the Gaussianization process. This is not analytically exact but has the advantage of being very effective in terms of computational load and improvement of performances.

3.2 Second Stage: Three-Step Approach to Linear Delayed uBSS Problem

After the recovery of nonlinearities we have to face a linear underdetermined and delayed BSS problem, to solve which a three step approach described in [5] has been implemented. A brief overview of this technique can be found in the remainder of this section; refer to [5] or [10] for further details.

Supposing a perfect compensation of nonlinearities, from (1) we get

$$z_i(t) = \sum_{j=1}^n \beta_{ij} s_j(t - \tau_{ij}), \quad \text{for } i = 1, \dots, m, \quad (7)$$

and our aim is to recover both the mixing parameters and the source signals. In the first step the Short Time Fourier Transform (STFT) $Z_i[k]$ (where k is the frequency index) of linearized mixtures z_i in (7) is computed over K -sample frames. It can be shown that, under the assumptions of independent and sparse source signals s_i , and exploiting linearity and the circular shift property of STFT, the representation of $Mag(\mathbf{Z})$ (amplitude of \mathbf{Z} , where $\mathbf{Z} = [\mathbf{Z}_i]$) tends to cluster along certain directions, corresponding to the columns of the attenuation matrix \mathbf{B} , that can be thus be recovered, using a potential based clustering technique on \mathbf{Z} amplitudes; since sources can always be recovered up to a permutation and a scaling factor, the columns of the attenuation matrix \mathbf{B} are chosen of unitary norm. In the two-dimensional case ($m = 2$), a potential function based approach can be used [5]. Since the columns of B have been chosen of unitary norm, the norm of vectors \mathbf{Z} turns to be fixed.

The second step deals with delays recovery. Since both \mathbf{Z} and delay matrix $\mathbf{T} = [\tau_{ij}]$ are unknown, only differential delays of source j between different sensors can be determined. As can be easily understood from (7), all the information about delays is contained in S_j phase. Considering the two-dimensional case ($m = 2$), if a phase correction $\exp(-\mathbf{j}2\pi\mathbf{k}\mathbf{R}/\mathbf{K})$ (where $\mathbf{R} = [\rho_j/2; -\rho_j/2]$) and ρ is the differential delay of j -th source between the two sensors) is applied, a cluster reappears in the scatterplot of both the real and the imaginary part of mixture amplitudes. The employed approach consists, thereby, in iteratively adjusting the delay between the two sensor signals, until a cluster along B_j direction (already known) reappears. At each iteration step the potential function along the direction relative to source j is evaluated and the $\bar{\rho}$ value that maximizes this function is used as an estimation of the corresponding differential delay. This procedure must be executed for all sources, yielding the final estimation of the delay matrix \mathbf{T} , that together with the estimation of attenuation matrix \mathbf{B} allows to recover the whole linear mixing model with delays. This procedure can be easily extended to the case of $m > 2$ channels, by estimating the differential delays between each pair of channels.

The third and last step of the algorithm is source estimation, needed because, when $m < n$, the equation system (7), for all i 's is overcomplete, also knowing the mixing parameters β_{ij} and τ_{ij} . A Maximum Likelihood (ML) estimation of sources is then performed, employing the previously inferred linear mixing parameters. Under the assumption of independent and laplacian distributed sources, source estimation turns into a special formulation of a second order cone programming problem

$$\min_s \sum_{j=1}^n \text{Mag}(S_j) \text{ under the constraint } \mathbf{Z} = \mathbf{W}\mathbf{S}, \quad (8)$$

where $\mathbf{W} = \mathbf{B} * \mathbf{e}^{-\mathbf{j}2\pi\mathbf{k}\mathbf{T}/\mathbf{K}}$ and $*$ denotes the element wise matrix product operator. Different techniques can be found in the literature to solve (8) (e.g. [11], [10]).

4 Experimental Results

In order to show the validity of the proposed method several test simulations have been performed by using synthetic mixtures of real speech signals and different sets of mixing parameters and nonlinearities. Since, up to authors' knowledge, no other algorithm has been proposed in the literature to handle the PNL-uBSS with delays, no direct comparisons can be reported. In the following, we shall focus on two test scenarios, differing for the type of component-wise nonlinearities involved. Two male and one female speech signals (37k samples at 11025Hz sampling frequency) are used and mixed-up to yield two observable sequences. For objective evaluation of performances the Signal-to-Noise Ratio (SNR) and the Itakura-Saito (IS) index have been employed. The SNR is defined as follows

$$\text{SNR} = 10 \log \left\{ \frac{\|s_k\|^2}{\|s_k - y_k\|^2} \right\}, \quad (9)$$

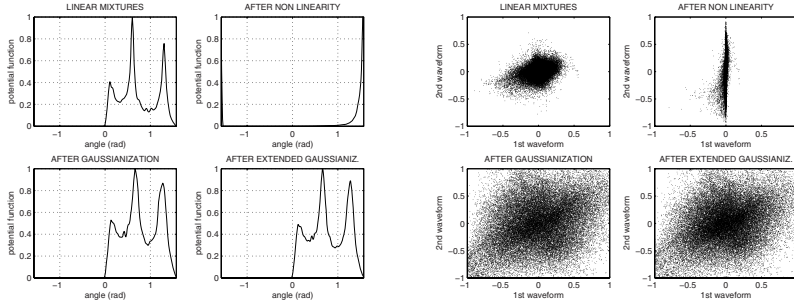


Fig. 2. a) Potential functions. b) scatterplot of PNL-uBSS mixtures with delays for the first case study (strong nonlinearity).

where s_k and y_k are the original and the recovered k -th signals respectively. IS calculates the spectral distance between the AR coefficients relative to related signals and obtained by means of a 16-th order linear predictor. The IS measure is very sensitive to the spectral mismatch in formant locations and is less affected by the mismatch in spectral valleys. Such characteristics are closely related to those of the human auditory system, that is why the IS index is widely used also for subjective evaluation of the speech quality. As previously mentioned, the two addressed experimental test sessions share the same linear mixing parameterization. In particular, the attenuation matrix is:

$$\mathbf{B} = \begin{bmatrix} \cos(\theta_1) & \cos(\theta_2) & \cos(\theta_3) \\ \sin(\theta_1) & \sin(\theta_2) & \sin(\theta_3) \end{bmatrix}, \quad \text{with } \theta_1 = 0.1, \theta_2 = 0.6, \theta_3 = 1, 3(\text{rad}), \tag{10}$$

whereas the delay matrix (as known written in terms of differential delays) is:

$$\mathbf{D} = \begin{bmatrix} 0 & 0 & 0 \\ -10 & 4 & 8 \end{bmatrix}. \tag{11}$$

The first test considers the presence of strong nonlinearities having a relevant distortion impact on the mixtures, by using the following functions h_1 and h_2 :

$$\begin{aligned} x_1 &= h_1(u_1) = u_1^3, \\ x_2 &= h_2(u_2) = \tanh(2 * u_2). \end{aligned} \tag{12}$$

In Fig. 2a the potential functions for parameter estimation are depicted; they have been calculated in the following case studies: linear, non-linear without nonlinearity compensation, non-linear with compensation accomplished through Gaussianization and non-linear with compensation accomplished through Extended Gaussianization. As we can observe, in the case of nonlinear mixtures it is not possible to proceed with the mixing parameter and delay estimation and so with the fulfilment of the separation procedure. In contrast, this can be

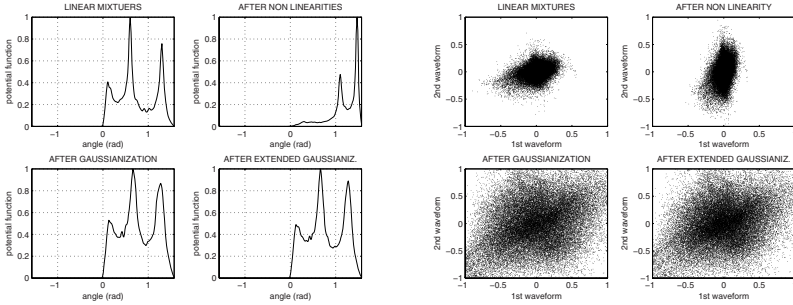


Fig. 3. a) Potential functions. b) scatterplot of PNL-uBSS mixtures with delays for the second case study (weak nonlinearity).

carried out if Gaussianization is applied. This can be also confirmed if we look at the scatterplots in Fig. 2b.

Table 1 reports the values of the estimated parameters: it must be observed that the attenuation matrix is perfectly recovered in the linear case study and that the extended Gaussianization allows achieving a significant improvement compared to the classic Gaussianization. In all situations the differential delays are estimated correctly. In Table 2 the SNR and IS index values are reported, confirming and the beneficial impact of the improved nonlinearity compensation through the C-F expansion based method.

In the second test weak nonlinearity acting on the second mixture has been used. Therefore the mixture equations become:

$$\begin{aligned} x_1 &= h_1(u_1) = u_1, \\ x_2 &= h_2(u_2) = \tanh(2 * u_2) + u_2. \end{aligned} \tag{13}$$

Looking at the plots in Fig. 3a, always related to the potential functions in the four different case studies already considered above, it can be noted that now it is possible to recover the unknown mixing parameters even neglecting the nonlinearity compensation stage. In such a situation, the detection of potential function peaks is not as straightforward as in the cases of Gaussianization based approaches, indeed the achieved results are sensitive to the algorithm adopted on purpose (the ones reported in table 2 are the best attained). This is also confirmed by the fact that the first recovered source in the "nonlinear" case present unsatisfying SNR and IS values, whereas the performances are likely uniform when Gaussianization is employed. Again, also in this test session, the estimation of differential delays is not critic and the Extended Gaussianization is preferable compared to the classic one.

Looking at achieved results from a global perspective, it can be highlighted that in presence of weak nonlinearities the Gaussianization method is not as much effective as in the presence of strong nonlinearities. Therefore its beneficial impact on the source recovery procedure is particularly recommended when mixtures are highly distorted by post-nonlinear functions. This observation

confirms the overall efficiency of the practical Gaussianization based approach here proposed, even though the source separability properties of the method still need to be theoretically analyzed: this surely represents an interesting issue for future research.

Table 1. Attenuation coefficients estimated for the performed computer simulations. Two are the case studies addressed, referred to (12) and (13) respectively.

	$\theta_1 = 0.10$	$\theta_1 = 0.60$	$\theta_1 = 1.30$
Linear	0.1	0.6	1.3
Case Study 1			
After Gaussianization	0.13	0.66	1.27
After Extended Gaussianiz	0.12	0.66	1.28
Case Study 2			
Non Linear	0.25	1.1	1.47
After Gaussianization	0.12	0.66	1.27
After Extended Gaussianiz	0.11	0.66	1.28

Table 2. SNRs and IS index of recovered sources for the performed computer simulations. Two are the case studies addressed, referred to (12) and (13) respectively.

	SNR(dB)			IS INDEXES		
	S1	S2	S3	S1	S2	S3
Linear	9.3	11.65	13.54	1,7E-3	1.10E-02	5.50E-03
Case Study 1						
Non linear	-	-	-	-	-	-
Gaussianization	1.84	3.83	3.51	6.90E-02	2.30E-01	7.70E-01
Ext. Gaussianiz.	2.08	4.09	3.84	6.20E-02	2.00E-01	6.50E-01
Case Study 2						
Non linear	3.84	9,26	10,94	1.96E-02	4.90E-02	2.10E-02
Gaussianization	6.57	8.62	9.89	1.20E-02	3.70E-02	1.30E-01
Ext. Gaussianiz.	7.1	9.15	10.65	9.60E-03	2.70E-02	1.00E-01

5 Conclusions

In this work the validity of a practical approach for speech source recovery for the PNL-uBSS mixing with delays in the linear part of the mixing has been proved. As previously noticed the Gaussianization and Extended Gaussianization techniques do not allow to get a perfect recovery of nonlinearities, as amplitude distributions of a linear mixtures are approximated with a Gaussian or close-to-Gaussian distribution. Nevertheless experimental results given in Section 4 have shown that the inverse nonlinearity recovery obtained with the previously introduced Gaussianization based techniques leads to an improved quality of recovered signals, specially in presence of strong distortions. For these kinds of

distortion the mixing parameters could not be recovered without any preprocessing on the nonlinearities, so that the separation is not achievable.

Current efforts are on the development of a better estimation of the unknown C-F parameters for further improvement of separation performances, whereas future works will consider the presence of convolutive mixing with non linear distortion in order to achieve good performances also in real world data situations.

Acknowledgment

This work was supported by the European Commission as sponsor of the hArtes Project, number 035143.

References

1. Cichocky, A., Amari, S.: Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications. John Wiley and Sons Chichester Ed. UK (2001)
2. Hyvriinen, A., Karhunen, J., Oja, E.: Independent Component Analysis. Wiley & Sons, New York (2001)
3. Taleb, A., Jutten, C.: Source Separation in Post-Nonlinear Mixtures. *IEEE Trans. on Signal Processing* **47** (10)(1999) 2807-2820
4. Squartini, S., Bastari, A., Piazza, F.: A Practical Approach Based on Gaussianization for Post-Nonlinear Underdetermined BSS. *Proc. ICCAS, Guilin, China* (2006)
5. Bofill, P.: Underdetermined Blind Separation of Delayed Sound Sources in the Frequency Domain. *Neurocomputing, Special Issue ICA and BSS* (2001)
6. Zhang, K., Chan, L.W.: Extended Gaussianization Method for Blind Separation of Post-Nonlinear Mixtures. *Neural Computation* **17** (2005) 425-452
7. Chen, S.S., Gonipath, R.A.: Gaussianization. *Proc. NIPS, Denver, USA* (2000)
8. Ziehe, A., Kawanabe, M., Harmeling, S., Mueller, K.R.: Blind Separation of Post-Nonlinear Mixtures Using Gaussianizing Transformations and Temporal Decorrelation. *Proc. ICA2003* (2003) 269-274
9. Papoulis, A., Pillai, S.U.: Probability, Random Variables and Stochastic Process. 4th Edition, McGrawHill (2002)
10. Bastari, A., Squartini, S., Piazza, F.: Underdetermined Blind Separation of Speech Signals with Delays in Different Time-Frequency Domains. G. Chollet et al. (Eds.): *Nonlinear Speech Modeling, LNAI 3445* (2005) 136-163
11. Lobo, M.S., Vandenberghe, L., Boyd, S., Lebret, H.: Applications of Second-order Cone Programming. In *Linear Algebra and Its Applications* (1998)

Regularized Alternating Least Squares Algorithms for Non-negative Matrix/Tensor Factorization

Andrzej Cichocki* and Rafal Zdunek**

Laboratory for Advanced Brain Signal Processing,
RIKEN BSI, Wako-shi, Saitama 351-0198, Japan

Abstract. Nonnegative Matrix and Tensor Factorization (NMF/NTF) and Sparse Component Analysis (SCA) have already found many potential applications, especially in multi-way Blind Source Separation (BSS), multi-dimensional data analysis, model reduction and sparse signal/image representations. In this paper we propose a family of the modified Regularized Alternating Least Squares (RALS) algorithms for NMF/NTF. By incorporating regularization and penalty terms into the weighted Frobenius norm we are able to achieve sparse and/or smooth representations of the desired solution, and to alleviate the problem of getting stuck in local minima. We implemented the RALS algorithms in our NMFLAB/NTFLAB Matlab Toolboxes, and compared them with standard NMF algorithms. The proposed algorithms are characterized by improved efficiency and convergence properties, especially for large-scale problems.

1 Introduction and Problem Formulation

Nonnegative Matrix Factorization (NMF) and its multi-way extensions: Non-negative Tensor Factorization (NTF) and Parallel Factor analysis (PARAFAC) models with sparsity and/or non-negativity constraints have been recently proposed as promising sparse and quite efficient representations of signals, images, or general data [1,2,3,4,5,6,7,8,9,10]. From a viewpoint of data analysis, NMF/NTF provide nonnegative and usually sparse common factors or hidden (latent) components with physiological meaning and interpretation [4,7,11]. NMF, NTF and SCA are used in a variety of applications, ranging from neuroscience and psychometrics to chemometrics [1,2,7,8,9,10,11,12,13,14].

In this paper we impose nonnegativity and sparsity constraints, and possibly other natural constraints, such as smoothness for the following linear model:

$$Y = AX + E, \quad (1)$$

* On leave from Warsaw University of Technology, Dept. of EE, Warsaw, Poland.

** On leave from Institute of Telecommunications, Teleinformatics and Acoustics, Wrocław University of Technology, Poland.

where $\mathbf{Y} \in \mathbb{R}^{I \times T}$ is a matrix of the observed data or signals, $\mathbf{A} \in \mathbb{R}_+^{I \times R}$ is a mixing or basis matrix, $\mathbf{X} \in \mathbb{R}_+^{R \times T}$ represents unknown sources or hidden (nonnegative and sparse) components, and $\mathbf{E} \in \mathbb{R}^{I \times T}$ represents a noise or error (residuum) matrix. Usually, in BSS applications: $T \gg I \geq R$, and R is known or can be estimated using SVD. Our objective is to estimate the mixing (basis) matrix \mathbf{A} and the sources \mathbf{X} , subject to nonnegativity and sparsity constraints.

The above model can be extended to the 3D PARAFAC2 or NTF2 model in which a given tensor $\underline{\mathbf{Y}} \in \mathbb{R}^{I \times T \times K}$ is decomposed to a set of matrices \mathbf{X} , \mathbf{D} and $\{\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_K\}$ with nonnegative entries [9,15,16,17]. The NTF2 model can be described as

$$\mathbf{Y}_k = \mathbf{A}_k \mathbf{D}_k \mathbf{X} + \mathbf{E}_k, \quad (k = 1, 2, \dots, K) \quad (2)$$

where $\mathbf{Y}_k = \mathbf{Y}_{::,k} = [y_{itk}]_{I \times T} \in \mathbb{R}^{I \times T}$ are frontal slices of $\underline{\mathbf{Y}} \in \mathbb{R}^{I \times T \times K}$, K is a number of frontal slices, $\mathbf{A}_k = [a_{irk}]_{I \times R} \in \mathbb{R}_+^{I \times R}$ are the basis (mixing matrices), $\mathbf{D}_k \in \mathbb{R}_+^{R \times R}$ is a diagonal matrix that holds the k -th row of the $\mathbf{D} \in \mathbb{R}_+^{K \times R}$ in its main diagonal, and $\mathbf{X} = [x_{rt}]_{R \times T} \in \mathbb{R}_+^{R \times T}$ is a matrix representing the sources (or hidden nonnegative components or common factors), and $\mathbf{E}_k = \mathbf{E}_{::,k} \in \mathbb{R}^{I \times T}$ is the k -th frontal slice of the tensor $\underline{\mathbf{E}} \in \mathbb{R}^{I \times T \times K}$ representing error or noise depending upon the application. The objective is to estimate the set of nonnegative matrices $\{\mathbf{A}_k\}, (k, \dots, K)$, \mathbf{D} and \mathbf{X} , subject to some non-negativity constraints and other possible natural constraints such as sparseness and/or smoothness. Since the diagonal matrices \mathbf{D}_k are scaling matrices they can be usually absorbed by the matrices \mathbf{A}_k by introducing the column-normalized matrices $\mathbf{A}_k := \mathbf{A}_k \mathbf{D}_k$, so usually in BSS applications the matrix \mathbf{X} and the set of scaled matrices $\mathbf{A}_1, \dots, \mathbf{A}_K$ need only to be estimated. It should be noted that the 3D PARAFAC2 and the corresponding NTF2 models [1] can be easily transformed to a 2D non-negative matrix factorization problem by unfolding (matricizing) tensors. Such 2D models are equivalent to a standard NMF model. In fact, the 3D PARAFAC2 model can be represented as column-wise unfolding. The unfolded system can be described by a single system of the matrix equation: $\mathbf{Y} = \mathbf{A}\mathbf{X} + \mathbf{E}$, where $\mathbf{Y} = [\mathbf{Y}_1; \mathbf{Y}_2; \dots; \mathbf{Y}_K] \in \mathbb{R}^{IK \times T}$ is a column-wise (vertical) unfolded matrix of all the frontal slices \mathbf{Y}_k , $\mathbf{A} = [\mathbf{A}_1; \mathbf{A}_2; \dots; \mathbf{A}_K] \in \mathbb{R}_+^{IK \times R}$ is a column-wise unfolded matrix of the slices \mathbf{A}_k representing (the frontal slices), and $\mathbf{E} = [\mathbf{E}_1; \mathbf{E}_2; \dots; \mathbf{E}_K] \in \mathbb{R}^{IK \times T}$ is a column-wise unfolded matrix of errors.

Solutions of NMF algorithms may not be unique, therefore it is often required to impose additional data-driven natural constraints, such as sparsity or smoothness. Moreover, many existing algorithms for NMF are prohibitively slow and inefficient, especially for very large-scale problems. For large-scale problems a promising approach is to apply the Alternating Least Squares (ALS) algorithm [18]. Unfortunately, the standard ALS algorithm and its simple modifications suffer from unstable convergence properties, giving often not optimum solution,

¹ Analogously, NTF1 model described by a set of the matrix equations $\mathbf{Y}_k = \mathbf{A} \mathbf{D}_k \mathbf{X}_k + \mathbf{E}_k$, $k = 1, 2, \dots, K$, can be transformed to the standard NMF problem by row-wise unfolding.

and they are characterized by high sensitivity to near-collinear data [14,8,10]. The main objective of this paper is to develop efficient and robust regularized ALS (RALS) algorithms. For this purpose, we exploit several approaches from constrained optimization and regularization theory, and propose additionally several heuristic algorithms.

2 Regularized ALS Algorithms

The most of known and used adaptive algorithms for NMF are based on alternating minimization of the squared Euclidean distance expressed by the Frobenius norm: $D_F(\mathbf{Y}||\mathbf{A}, \mathbf{X}) = \frac{1}{2}\|\mathbf{Y} - \mathbf{A}\mathbf{X}\|_F^2$, subject to nonnegativity constraints of all the elements in \mathbf{A} and \mathbf{X} . Such a cost function is optimal for a Gaussian distributed noise [2,11].

In this paper we consider minimization of more general and flexible cost function that is a regularized weighted least-squares function with sparsity penalties:

$$D_F^{(\alpha)}(\mathbf{Y}||\mathbf{A}\mathbf{X}) = \frac{1}{2}\|\mathbf{W}^{-1/2}(\mathbf{Y} - \mathbf{A}\mathbf{X})\|_F^2 + \alpha_{A_s} \|\mathbf{A}\|_{L_1} + \alpha_{X_s} \|\mathbf{X}\|_{L_1} + \frac{\alpha_{A_r}}{2} \|\mathbf{W}^{-1/2}\mathbf{A}\mathbf{L}_A\|_F^2 + \frac{\alpha_{X_r}}{2} \|\mathbf{L}_X\mathbf{X}\|_F^2, \tag{3}$$

(usually subject to additional constraints such as nonnegativity constraints) where $\mathbf{W} \in \mathbb{R}^{I \times I}$ is symmetric positive definite weighting matrix², $\alpha_{A_s} \geq 0$ and $\alpha_{X_s} \geq 0$ are parameters controlling a sparsity level of the matrices, and $\alpha_{A_r} \geq 0$, $\alpha_{X_r} \geq 0$ are regularization coefficients. The penalty terms $\|\mathbf{A}\|_{L_1} = \sum_{ir} |a_{ir}|$ and $\|\mathbf{X}\|_{L_1} = \sum_{rt} |x_{rt}|$ enforce sparsification in \mathbf{A} and \mathbf{X} , respectively, and sparseness can be adjusted by α_{A_s} and α_{X_s} . The regularization matrices \mathbf{L}_A and \mathbf{L}_X are used to enforce a certain application-dependent characteristics of the solution. These matrices are typically unit diagonal matrices or discrete approximations to some derivative operator. Another option is to use the following setting: $\alpha_{X_r}\mathbf{L}_X^T\mathbf{L}_X = \mathbf{A}^T(\mathbf{I} - \mathbf{A}_S\mathbf{A}_S^T)\mathbf{A}$ where \mathbf{A}_S contains the R first principal eigenvectors of the data covariance matrix $\mathbf{R}_Y = (\mathbf{Y}^T\mathbf{Y})/I = \mathbf{U}\Sigma\mathbf{U}^T$ associated with the R largest singular values [2]. It is worth noting that both matrices $\mathbf{L}_X^T\mathbf{L}_X \in \mathbb{R}^{R \times R}$ and $\mathbf{L}_A\mathbf{L}_A^T \in \mathbb{R}^{R \times R}$ are in general symmetric and positive definite matrices.

The gradients of the cost function (3) with respect to the unknown matrices \mathbf{A} and \mathbf{X} are expressed by

$$\frac{\partial D_F^{(\alpha)}(\mathbf{Y}||\mathbf{A}\mathbf{X})}{\partial \mathbf{A}} = \mathbf{W}^{-1}(\mathbf{A}\mathbf{X} - \mathbf{Y})\mathbf{X}^T + \alpha_{A_s} \mathbf{S}_A + \alpha_{A_r} \mathbf{W}^{-1}\mathbf{A}\mathbf{L}_A\mathbf{L}_A^T, \tag{4}$$

$$\frac{\partial D_F^{(\alpha)}(\mathbf{Y}||\mathbf{A}\mathbf{X})}{\partial \mathbf{X}} = \mathbf{A}^T\mathbf{W}^{-1}(\mathbf{A}\mathbf{X} - \mathbf{Y}) + \alpha_{X_s} \mathbf{S}_X + \alpha_{X_r} \mathbf{L}_X^T\mathbf{L}_X \mathbf{X}, \tag{5}$$

² $\mathbf{W}^{-1/2} = \mathbf{V}\mathbf{\Lambda}^{-1/2}\mathbf{V}^T$ means in Matlab notation $\mathbf{W}^{-1/2} = \text{inv}(\text{sqrtn}(\mathbf{W}))$ and $\|\mathbf{W}^{-1/2}(\mathbf{Y} - \mathbf{A}\mathbf{X})\|_F^2 = \text{tr}\{(\mathbf{Y} - \mathbf{A}\mathbf{X})^T\mathbf{W}^{-1}(\mathbf{Y} - \mathbf{A}\mathbf{X})\}$.

where $\mathbf{S}_A = \text{sign}(\mathbf{A})$ and $\mathbf{S}_X = \text{sign}(\mathbf{X})$ ³. In the particular case for a NMF problem the matrices $\mathbf{S}_A, \mathbf{S}_X$ will be transformed to the matrices $\bar{\mathbf{E}}_A$ and $\bar{\mathbf{E}}_X$ of the same dimension with all the entries equal to ones.

By equalizing the gradients (4)-(5) to zero, we obtain the following fixed-point regularized ALS algorithm

$$\mathbf{A} \leftarrow (\mathbf{Y}\mathbf{X}^T - \alpha_{As}\mathbf{W}\mathbf{S}_A)(\mathbf{X}\mathbf{X}^T + \alpha_{Ar}\mathbf{L}_A\mathbf{L}_A^T)^{-1} \tag{6}$$

$$\mathbf{X} \leftarrow (\mathbf{A}^T\mathbf{W}^{-1}\mathbf{A} + \alpha_{Xr}\mathbf{L}_X^T\mathbf{L}_X)^{-1}(\mathbf{A}^T\mathbf{W}^{-1}\mathbf{Y} - \alpha_{Xs}\mathbf{S}_X). \tag{7}$$

In order to achieve high performance, the regularization parameters $\alpha_{Ar} \geq 0$ and $\alpha_{Xr} \geq 0$ are usually not necessarily fixed but rather should be dynamically changed in time, depending how far we are from the desired solution. For example, we may gradually decrease exponentially the regularization coefficients during a convergence process. We found by computer experiments that quite a good performance for small-scale problems can be achieved by choosing $\alpha_{Ar}(k) = \alpha_{Xr}(k) = \alpha_0 \exp(-k/\tau)$ with typical values $\alpha_0 = 20$ and $\tau = 50$ and $\mathbf{L}_X^T\mathbf{L}_X \approx \bar{\mathbf{E}}_X$ where $\bar{\mathbf{E}}$ means a matrix with all ones entries⁴. For large-scale problems α_0 should be higher.

An alternative approach is to keep the regularization parameters fixed and try to compensate (reduce) their influence by additional terms as the algorithm converges to the desired solution. For this purpose let us consider the following simple approach [10]. It is easy to note that the equation (7) can be re-written in the equivalent form as

$$(\mathbf{A}^T\mathbf{W}^{-1}\mathbf{A} + \alpha_{Xr}\mathbf{L}_X^T\mathbf{L}_X)\mathbf{X}_{new} = \mathbf{A}^T\mathbf{W}^{-1}\mathbf{Y} - \alpha_{Xs}\mathbf{S}_X \tag{8}$$

In order to compensate the regularization term $\alpha_{Xr}\mathbf{L}_X^T\mathbf{L}_X\mathbf{X}_{new}$ we can add to the right-hand side the similar term $\alpha_{Xr}\mathbf{L}_X^T\mathbf{L}_X\mathbf{X}_{old}$ which gradually compensates the regularization term when $\mathbf{X} \rightarrow \mathbf{X}^*$, i.e.

$$(\mathbf{A}^T\mathbf{W}^{-1}\mathbf{A} + \alpha_{Xr}\mathbf{L}_X^T\mathbf{L}_X)\mathbf{X}_{new} = \mathbf{A}^T\mathbf{W}^{-1}\mathbf{Y} - \alpha_{Xs}\mathbf{S}_X + \alpha_{Xr}\mathbf{L}_X^T\mathbf{L}_X\mathbf{X}_{old}$$

The magnitude of the bias (or influence of the regularization term) is a function of the difference between \mathbf{X}_{old} and \mathbf{X}_{new} . As the algorithm tends to converge to the desired solution \mathbf{X}^* , this difference is gradually decreasing, and the effect of regularization and bias is smaller and smaller.

Hence, after simple mathematical manipulations our RALS algorithm can take the following general and flexible form:

$$\mathbf{A} \leftarrow (\mathbf{Y}\mathbf{X}^T - \alpha_{As}\mathbf{W}\mathbf{S}_A + \alpha_{Ar}\mathbf{A}\mathbf{L}_A\mathbf{L}_A^T)(\mathbf{X}\mathbf{X}^T + \alpha_{Ar}\mathbf{L}_A\mathbf{L}_A^T)^+, \tag{9}$$

$$\mathbf{X} \leftarrow (\mathbf{A}^T\mathbf{W}^{-1}\mathbf{A} + \alpha_{Xr}\mathbf{L}_X^T\mathbf{L}_X)^+(\mathbf{A}^T\mathbf{W}^{-1}\mathbf{Y} - \alpha_{Xs}\mathbf{S}_X + \alpha_{Xr}\mathbf{L}_X^T\mathbf{L}_X\mathbf{X}), \tag{10}$$

³ $\text{sign}(\mathbf{X})$ means a componentwise sign operation (or its robust approximation) for each element in \mathbf{X} .

⁴ In this case, to drive the RALS algorithm rigorously, we have used the following modified regularized cost functions: $0.5\|\mathbf{Y} - \mathbf{A}\mathbf{X}\|_F^2 + \alpha_X\|\mathbf{X}\|_{L_1} + 0.5\alpha_{Xr}\text{tr}\{\mathbf{X}^T\bar{\mathbf{E}}\mathbf{X}\} + \alpha_A\|\mathbf{A}\|_{L_1} + 0.5\alpha_{Ar}\text{tr}\{\mathbf{A}\bar{\mathbf{E}}\mathbf{A}^T\}$.

where \mathbf{A}^+ means Moore-Penrose pseudo-inverse of \mathbf{A} . It should be noted that the proposed algorithm for $\mathbf{W} = \mathbf{I}$ and for all the regularization coefficients setting to zero ($\alpha_{As} = \alpha_{Ar} = \alpha_{Xs} = \alpha_{Xr} = 0$) simplifies to the standard ALS. On the other hand, if we take all the regularization parameters equal to zero and $\mathbf{W} = \mathbf{R}_E = (\mathbf{E}\mathbf{E}^T)/I$, where the error matrix $\mathbf{E} = \mathbf{Y} - \mathbf{A}\mathbf{X}$ is evaluated in each iteration step, we obtain the extended BLUE (Best Linear Unbiased Estimated) ALS algorithm. Finally, in the special case when $\mathbf{W} = \mathbf{I}$ and matrices $\mathbf{L}_A\mathbf{L}_A^T$ and $\mathbf{L}_X^T\mathbf{L}_X$ are diagonal our algorithm is similar to the MALS (modified ALS) proposed by Wang et al in [10], and Hancewicz and Wang in [4].

3 Implementation of RALS Algorithms for NMF

On the basis of the above consideration we have developed and implemented in MATLAB the following RALS algorithm for the NMF, especially suitable for large-scale problems:

Outline of the RALS algorithm for NMF

- 1a. Set the initial values of matrices $\mathbf{W}, \mathbf{L}_A, \mathbf{L}_X$ and parameters $\alpha_{As}, \alpha_{Xs}, \alpha_{Ar}, \alpha_{Xr}$,
- 1b. Set the initial values of \mathbf{A} , (e.g., multi-start random initialization, or eigenvalue decomposition (SVD), ICA, or dissimilarity criteria [15,48],
- 2a. Calculate the new estimate of \mathbf{X}_{new} from \mathbf{Y} and \mathbf{A}_{old} using iterative formula (10), (set $\mathbf{S}_X = \bar{\mathbf{E}}_X$),
- 2b. $\mathbf{X}_{new} = \max\{\mathbf{X}_{new}, 0\}$ (set negative values to zero or alternatively to a small positive value, typically, $\varepsilon = 10^{-16}$). Impose additional optional natural constraints on rows of \mathbf{X} such as low-pass filtering or smoothness,
- 3a. Calculate the new estimate of \mathbf{A}_{new} from (9), (set $\mathbf{S}_A = \bar{\mathbf{E}}_A$),
- 3b. $\mathbf{A}_{new} = \max\{\mathbf{A}_{new}, 0\}$ (set negative values to zero or to a small positive value ε). Impose some additional finite constraints such as clustering or smoothness,
- 3c. Normalize each column of \mathbf{A}_{new} to unit length l_1 -norm,
- 4. Repeat the steps (2) and (3) until convergence criterion is reached.

The above algorithm with a suboptimal set of the default parameters have been implemented in our NMFLAB and NTFLAB [15].

Further improvement of the RALS algorithm has been achieved by applying a hierarchical multi-layer system with multi-start initialization [13,15] which can be implemented as follows: In the first step, we perform the basic decomposition (factorization) $\mathbf{Y} = \mathbf{A}_1\mathbf{X}_1$ using the RALS algorithm. In the second stage, the results obtained from the first stage are used to perform the similar decomposition: $\mathbf{X}_1 = \mathbf{A}_2\mathbf{X}_2$ using the same or different set of parameters, and so on. We continue our factorization taking into account only the last achieved components. The process can be repeated arbitrarily many times until some stopping criteria are satisfied. In each step, we usually obtain gradual improvements of the performance. Thus, our model has the form: $\mathbf{Y} = \mathbf{A}_1\mathbf{A}_2 \cdots \mathbf{A}_L\mathbf{X}_L$, with the

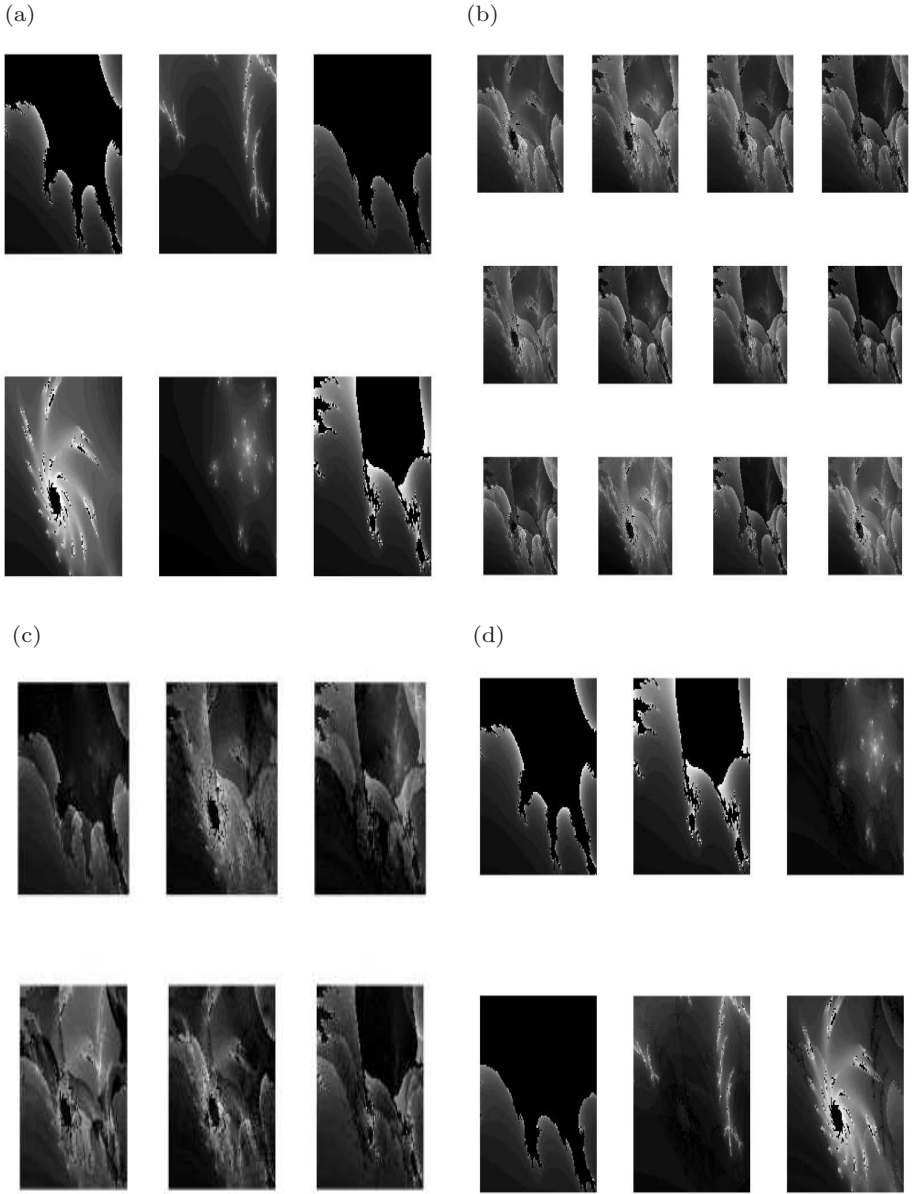


Fig. 1. Example 1: (a) Original 6 sources of Mandelbrot fractal images; (b) Observed 12 mixed images (uniformly distributed random mixing matrix); (c) Estimated sources using the standard Lee-Seung algorithm with Kullback-Leibler divergence (SIR = 6.4, 7.6, 0.2, 3.7, -0.2, 7.3 [dB], respectively); (d) Estimated source images using RALS algorithm (SIR = 50.61, 128.1, 17.6, 41, 16.6, 13.1 [dB], respectively) given by (6)–(7) with parameters: $\mathbf{W} = \mathbf{I}$ (identity), $\alpha_{Ar} = \alpha_{As} = \alpha_{Xs} = 0$, $\mathbf{L}_X^T \mathbf{L}_X = \bar{\mathbf{E}}$ and α_{Xr} given by the exponential rule with $\alpha_{Xr} = 20 \exp(-k/50)$.

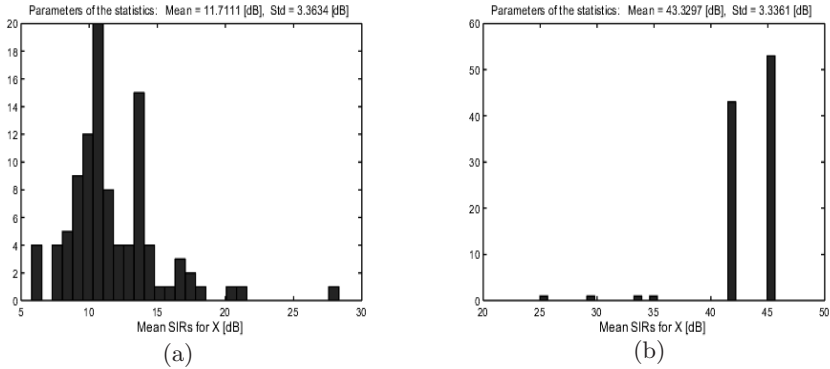


Fig. 2. Example 1: Histograms of 100 mean-SIR samples from Monte Carlo analysis performed with the algorithms: (a) standard ALS; (b) RALS with the same parameters as in Fig. 1.

basis nonnegative matrix defined as $\mathbf{A} = \mathbf{A}_1 \mathbf{A}_2 \cdots \mathbf{A}_L$. An open theoretical issue is to prove mathematically or explain more rigorously why the multilayer distributed NMF/NTF system results in considerable improvement in performance and reduces the risk of getting stuck at local minima. An intuitive explanation is as follows: the multilayer system provides a sparse distributed representation of basis matrix \mathbf{A} , which in general can be a dense matrix. So even a true basis matrix \mathbf{A} is not sparse it can be represented by a product of sparse factors. In each layer we force (or encourage) a sparse representation. On the other hand, we found by extensive experiments that if the basis matrix is very sparse, most NTF/NMF algorithms have improved performance (see next section). However, not all real data provides sufficiently sparse representations, so the main idea is to model any data by a distributed sparse hierarchical multilayer system. It is also interesting to note that such multilayer systems are biologically motivated and plausible.

4 Simulation Results

In order to confirm validity and high performance of the proposed algorithm we extensively tested it for various sets of free parameters and compared them with standard NMF algorithms. We illustrate the performance by giving only two examples. In the first example (see Fig. 1), we used 6 images which were mixed by a uniformly distributed randomly generated mixing matrix $\mathbf{A} \in \mathbb{R}^{12 \times 6}$. We found by the Monte Carlo analysis performed for 100 runs that our RALS algorithm (with the exponentially decaying regularization term for the matrix \mathbf{X}) significantly outperforms the standard ALS (see Fig. 2).

In the second example, the 9 sparse nonnegative signals (representing synthetic spectra) have been mixed by the mixing matrix $\mathbf{A} \in \mathbb{R}^{18 \times 9}$. Additive Gaussian noise with SNR = 20 dB has been added. In this case the standard

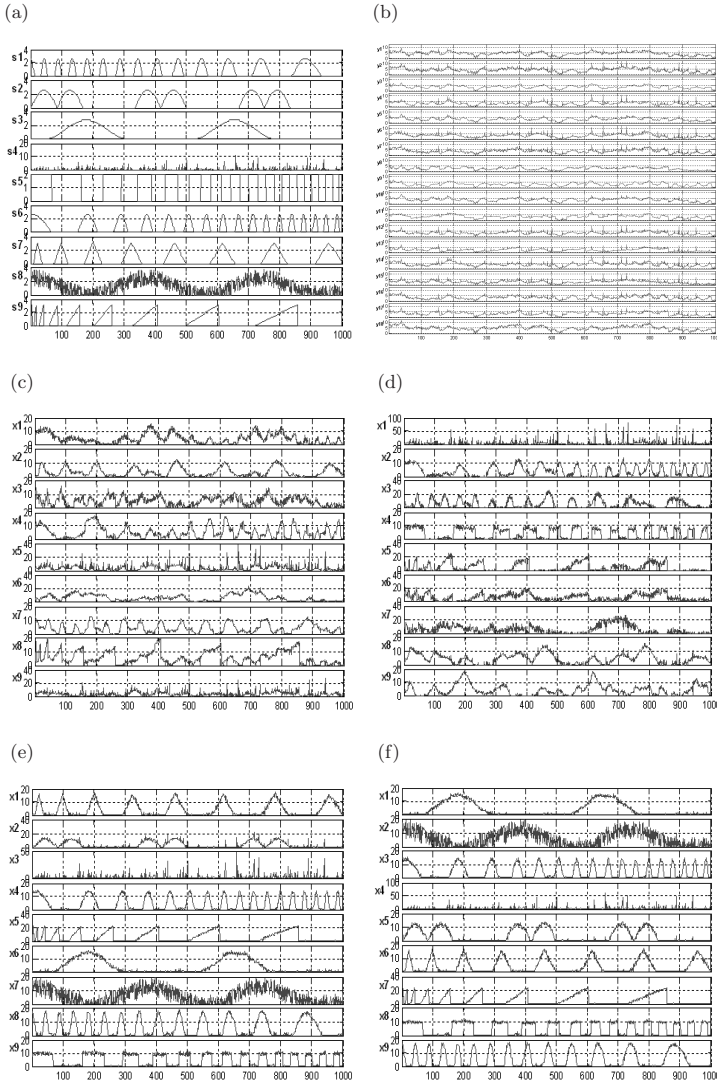


Fig. 3. Example 2: (a) Original 9 source signals; (b) Observed 18 mixed signals (uniformly distributed random mixing matrix) with SNR = 20 dB; (c) Estimated sources using the standard Lee-Seung algorithm with Kullback-Leibler divergence (SIR = 3.7, 8.2, 3.6, 6.1, 4.5, 2.9, 5.2, 5.8, 2.2 [dB], respectively) with 1 layer; (d) Estimated sources using the standard Lee-Seung algorithm with Kullback-Leibler divergence (SIR = 6.9, 6.6, 6.7, 18.2, 14, 8.7, 7.6, 5.8, 15.9 [dB], respectively) with 3 layers; (e) Estimated source images using the RALS algorithm (SIR = 18.2, 12.2, 21.1, 20.7, 22.5, 19.1, 21.3, 19.9 [dB], respectively) given by (9)–(10) with 1 layer and for the following parameters: $\mathbf{W} = \mathbf{R}_E$, $\alpha_{Ar} = \alpha_{As} = 0$, $\alpha_{Xs} = \alpha_{Xr} = 0.1$, $\mathbf{L}_X^T \mathbf{L}_X = \mathbf{I}$; (f) Estimated source images using the same RALS algorithm (SIR = 19.4, 17.4, 21.5, 22.6, 17.9, 18.5, 22.2, 21.6, 22.2 [dB], respectively) with 3 layers.

NMF algorithms completely failed to estimate the original sources while the RALS algorithm successfully estimates all the sources. We observed a considerable improvement in performance applying the multilayer procedure with 10 initializations in each layer.

We also performed the test on large-scale problems, increasing the number of observations in the second example to 1000. The mixing matrix $\mathbf{A} \in \mathbb{R}^{1000 \times 9}$ was randomly chosen. For such a case we got the elapsed times and mean-SIRs given in Table 1. It should be noted that the ISRA and EMLL algorithms (which are the basic Lee-Seung multiplicative algorithms that minimize the Frobenius norm and Kullback-Leibler divergence, respectively) failed to estimate the original components.

Table 1. Performance of the NMF algorithms for a large-scale problem with 1000 observations and 9 nonnegative components

Algorithm	Elapsed time [s]	Mean-SIRs [dB]
RALS	16.6	$SIR > 43$
ISRA	36	$SIR < 10$
EMLL	81	$SIR < 16$

5 Conclusions and Discussion

In this paper we proposed the generalized and flexible cost function (controlled by sparsity penalty and flexible multiple regularization terms) that allows us to derive a family of robust and efficient alternating least squares algorithms for NMF and NTF. We proposed the method which allows us to automatically self-regulate or self-compensate the regularization terms. This is a unique modification of the standard ALS algorithm and to the authors' best knowledge, the first time this type of constraints has been combined together with the ALS algorithm for applications to NMF and NTF. The performance of the proposed algorithm is compared with the ordinary ALS algorithm for NMF. The proposed algorithm is shown to be superior in terms of performance, component resolution ability, speed and convergence properties, and ability to be used for large-scale problems. The proposed RALS algorithm may be also promising for other applications, such as Sparse Component Analysis and EM Factor Analysis because it overcomes the problems associated with ALS, i.e. the solution of RALS tends not to get trapped in local minima and will generally converges to the global desired solution.

References

1. Berry, M., Browne, M., Langville, A., Pauca, P., Plemmons, R.: Algorithms and applications for approximate nonnegative matrix factorization. Computational Statistics and Data Analysis (2006) submitted.
2. Cichocki, A., Amari, S.: Adaptive Blind Signal And Image Processing (New revised and improved edition). John Wiley, New York (2003)

3. Dhillon, I., Sra, S.: Generalized nonnegative matrix approximations with Bregman divergences. In: Neural Information Proc. Systems, Vancouver, Canada (2005)
4. Hancewicz, T.M., Wang, J.H.: Discriminant image resolution: a novel multivariate image analysis method utilizing a spatial classification constraint in addition to bilinear nonnegativity. *Chemometrics and Intelligent Laboratory Systems* **77** (2005) 18–31
5. Heiler, M., Schnoerr, C.: Controlling sparseness in non-negative tensor factorization. Springer LNCS **3951** (2006) 56–67
6. Hoyer, P.: Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research* **5** (2004) 1457–1469
7. Morup, M., Hansen, L.K., Herrmann, C.S., Parnas, J., Arnfred, S.M.: Parallel factor analysis as an exploratory tool for wavelet transformed event-related EEG. *NeuroImage* **29** (2006) 938–947
8. Albright, R., Cox, J., Duling, D., Langville, A.N., Meyer, C.D.: Algorithms, initializations, and convergence for the nonnegative matrix factorization. Technical report, NCSU Technical Report Math 81706 (2006) submitted.
9. Smilde, A., Bro, R., Geladi, P.: *Multi-way Analysis: Applications in the Chemical Sciences*. John Wiley and Sons, New York (2004)
10. Wang, J.H., Hopke, P.K., Hancewicz, T.M., Zhang, S.L.: Application of modified alternating least squares regression to spectroscopic image analysis. *Analytica Chimica Acta* **476** (2003) 93–109
11. Lee, D.D., Seung, H.S.: Learning the parts of objects by nonnegative matrix factorization. *Nature* **401** (1999) 788–791
12. Cichocki, A., Zdunek, R., Amari, S.: Csiszar’s divergences for non-negative matrix factorization: Family of new algorithms. Springer LNCS **3889** (2006) 32–39
13. Cichocki, A., Amari, S., Zdunek, R., Kompass, R., Hori, G., He, Z.: Extended SMART algorithms for non-negative matrix factorization. Springer LNAI **4029** (2006) 548–562
14. Kim, M., Choi, S.: Monaural music source separation: Nonnegativity, sparseness, and shift-invariance. Springer LNCS **3889** (2006) 617–624
15. Cichocki, A., Zdunek, R.: NTFLAB for Signal Processing. Technical report, Laboratory for Advanced Brain Signal Processing, BSI, RIKEN, Saitama, Japan (2006)
16. Cichocki, Zdunek, R., Choi, S., Plemmons, R., Amari, S.: Novel multi-layer non-negative tensor factorization with sparsity constraints. In: Proc. 8-th International Conference on Adaptive and Natural Computing Algorithms, Warsaw, Poland (2007)
17. Cichocki, Zdunek, R., Choi, S., Plemmons, R., Amari, S.: Nonnegative tensor factorization using alpha and beta divergencies. In: Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP07), Honolulu, Hawaii, USA (2007)

Underdetermined Blind Source Separation Using SVM

Yang Zuyuan¹, Luo Shiguang², and Chen Caiyun¹

¹School of Electronic & Information Engineering, South China University of Technology,
Guangzhou 510641, Guangdong, China

²Department of Applied Mathematics, GuangDong University of
Finance,
Guangzhou 510521, Guangdong, China
yangzuyuan99811@eyou.com, yangzuyuan@yahoo.com.cn,
wangyongle811@eyou.com

Abstract. A novel sparse measure of signal is proposed and the efficient number of sources is estimated by the best confidence limit in this work. The observations are classified by SVM trained through samples which are constructed by direction angle of sources. And columns of the mixing matrix corresponding to clustering centers of each class are obtained based on the sum of samples belong to the same class with different weights which are adjusted adaptively. It gets out of the trap of the initial values which interfere k-mean clustering quite a lot. Furthermore, the online algorithm for estimating basis matrix is proposed for large scale samples. The shortest path method is used to recover the source signals after estimating the mixing matrix. The favorable simulations show the stability and robustness of the algorithms.

1 Introduction

Nowadays, blind source separation (BSS) has been attached more and more importance because of its widely application in signal processing, including space signal, EEG, ECG, speech signal, geophysical and chaos signals and so on. Most of the algorithms separated the signals successfully under the consumption that the mixing matrix is invertible, such as classical ICA [8], algorithms based on signal temporal predictability [10]. However, they failed when the matrix was ill-conditioned (e.g., the sources are more than sensors) [2], [3]. It found that the separation quality seemed to improve with the higher sparsity of the sources for the underdetermined case [6], and many source signals were sparse [2], [3], [4], [7] in the time-domain, furthermore, many natural signals could have sparser representations in other transform domain, such as the Fourier Transform, the Wavelet Transform and the Modified Discrete Cosine Transform (MDCT) [2]. Based on the sparse quality of sources, several methods had been developed for BSS. For instance, the mixing matrix and sources were estimated using overcomplete representation [1], the maximum posterior approach and maximum-likelihood approach [11]. And a two-stage method (TSM) that was clustering-then- l^1 -optimization approach was proposed by Bofill, in which six sources were separated [2]. The necessary and sufficient condition of separability using TSM was introduced in [4], [9], K-mean clustering [3]

and K-EVD [5] were used to estimate the mixing matrix. TSM was an attractive method to recover the sources for underdetermined case in stead of ICA which asked for independence and stationarity of the sources, however, the error from the estimation of the basis matrix would affect precision of separation inevitably. There existed several problems using methods above to obtain basis matrix (mixing matrix): 1) the number of the sources was unknown, so that it was hard to get the amount of the clustering centers, but the method with k-means clustering depended on it quite a lot; 2) clustering algorithms above were sensitive to initialization which was random in reality; 3) the high sparsity of sources could guarantee a high probability that the sources could be recovered [6], however, there was still lack of an efficient measure for the sparsity of sources.

In this work, a novel measure for sparsity is proposed, including the efficient estimation of the amount of the sources with the similar method in [15], after that, the observations are classified using support vector machine (SVM) [13], [14]. The clustering center of each class is obtained by sum of samples with different weights such that the mixing matrix is estimated at last. After that, the shortest path method is used to recover the sources by minimizing the 1-norms of the source vectors under some constraints.

The typical model for BSS with m sources and n sensors is as follows [11]:

$$X(t) = A \cdot S(t) + v(t), \tag{1}$$

where $X(t)$ are observations, $A \in R^{n \times m}$ is a mixing matrix, $S(t)$ are sources and $v(t)$ are additive noises. If $m > n$, it is the underdetermined model, and equation (1) can be wrote as following neglecting noise:

$$\begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix} = \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{n1} \end{bmatrix} s_1(t) + \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{n2} \end{bmatrix} s_2(t) \cdots + \begin{bmatrix} a_{1m} \\ a_{2m} \\ \vdots \\ a_{nm} \end{bmatrix} s_m(t). \tag{2}$$

Based on the sparsity of the sources, m basis vectors with n dimension are obtained to construct the mixing matrix and the shortest path way is used to recover the sources.

The paper is organized as follows: the learning theory of SVM is introduced in Section 2, together with the training method of sequential minimal optimization (SMO). In Section 3, the algorithms for estimating the basis matrix are described in detail in both batch mode and online mode, including recovering the sources using TSM. The experimental results are shown in Section 4. Finally, a conclusion is given in Section 5.

2 The Learning Theory of SVM

In 1960s, the statistic learning theory was introduced by Vapnik and Cervonenkis, and as a new learning machine algorithm, SVM was proposed under the standard of

structure risk minimum (SRM). It is adopted widely because of its firm theory basis, good generalizing quality and implementing algorithms.

The process of training the SVM equals to solving a quadratic programming (QP) problem [13]:

For a group of training samples as follows:

$$\{(\mathbf{x}_i, y_i) | i = 1, \dots, T; \mathbf{x}_i \in R^n, y_i \in \{-1, +1\}\}. \tag{3}$$

The aim is to get the decision-making function below:

$$f(\mathbf{x}) = \text{sign} \left\{ \sum_{i=1}^T \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) - b \right\}, \tag{4}$$

where $k(\mathbf{x}_i, \mathbf{x}_j)$ is the kernel function, α is the Lagrange multiplier, b is the threshold and $\{\alpha_i\}_{i=1}^T$ are results of the following QP problem [13]:

$$\begin{cases} \max_{\alpha} W(\alpha) = \sum_{i=1}^T \alpha_i - \frac{1}{2} \sum_{i,j=1}^T \alpha_i Q(i, j) \alpha_j \\ S.T. \quad \sum_{j=1}^T \alpha_i y_j = 0, 0 \leq \alpha_i \leq C, i = 1, \dots, T \end{cases} \tag{5}$$

where $Q(i, j) = y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \in R^{T \times T}$.

For large scale samples, the traditional method such as Newton’s way can not be used to solve (5) because of the limitation of the PC memory, and sequential minimal optimization (SMO) will be adopted to improve it in this work. SMO is a famous approach for training SVM [13], and it is a reduced algorithm which gets out of the limitation of the scale of the samples.

3 BSS with TSM

The traditional TSM will be used for BSS for underdetermined case [2], that is estimating mixing matrix firstly and then recovering sources using the shortest path method [15]. Furthermore, online algorithm for estimating mixing matrix is proposed for large scale samples.

In this work, SVM is used to estimate the mixing matrix for its excellent quality. The best confidence limit will be introduced firstly, based on which, a novel measure for sparsity (Spa) is proposed, including the efficient estimation of the amount of the sources [15]. In the case of $n = 2$, the direction angle of the sources is as follows:

$$\theta(t) = a \tan \frac{x_2(t)}{x_1(t)}, \tag{6}$$

where $t = 1, 2, \dots, T$.

Obviously, $\theta(t) \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, let $\theta(t) \in U = [\min_ \theta, \max_ \theta] \subseteq [-\frac{\pi}{2}, \frac{\pi}{2}]$.

Definition 1. let confidence interval sets $\{U_i\}, i = 1, 2, \dots, N$ cover U , and the length d of which is the same, then, the interval covering the most samples is the best confidence interval and covering the least samples is the worst one;

Definition 2. let $U_i, i \in \{1, 2, \dots, N\}$ cover M_i samples, and then the confidence limit of U_i is as following [15]:

$$D_i = \frac{M_i}{T} \tag{7}$$

Based on the sparsity of the sources, the distribution of direction angles can be used to estimate its efficient number through the best confidence limit;

Definition 3. For interval $B = [b, b + \varepsilon]$, if $\varepsilon < 0$, then $B = \emptyset$;

Definition 4. For $B_i = [b_i, b_i + \varepsilon], (i = 1, 2)$, $b_1 < b_2, B_1, B_2 \neq \emptyset$, $A = [a, a + \varepsilon]$, if $a \in (b_1, b_2)$, then $A \in (B_1, B_2)$, that is A lies between B_1 and B_2 ;

Definition 5. let $U_{j_1}, U_{j_2}, U_{j_3}$ are three conjoint best confidence intervals, the confidence limit of U_{j_2} is D_{j_2} , for interval sets constructed by $U_{s_k} \in (U_{j_1}, U_{j_2}) \cup (U_{j_2}, U_{j_3})$, its average confidence limit is \bar{D}_s , then the sparsity of the source signal corresponding to U_{j_2} is defined as following:

$$Spa = \frac{D_{j_2}}{D_{j_2} + \bar{D}_s}, \tag{8}$$

where \bar{D}_s is also called the rest average confidence limit. If $U_{j_1} = \emptyset$, then $U_{j_1} = [\min_ \theta, \min_ \theta + d]$, and if $U_{j_3} = \emptyset$, then $U_{j_3} = [\max_ \theta - d, \max_ \theta]$.

From equation (8), we can see that bigger Spa corresponding to higher sparsity of sources and better precision of the estimation for mixing matrix.

Two methods will be introduced for estimating mixing matrix as following:

3.1 Batch Algorithm

1) Initialization, if observations are not sparse in time area, then we can transform them into sparse area, using the Fourier Transform, the Wavelet Transform, the Modified Discrete Cosine Transform (MDCT) and so on. Signals with sparse quality are flagged as $x(t), t = 1, 2, \dots, T$;

2) Generalize $x(t)$ into $x'(t)$, that is

$$x'(t) = \frac{x(t) - E[x(t)]}{Var[x(t)]} \tag{9}$$

Where $E[\cdot]$ means expectation and $Var[\cdot]$ means variance;

- 3) Filter $x'(t)$ into $\tilde{x}(t), t = 1, 2, \dots, \tilde{T}$, based on the direction angles from equation (6), set several intervals with the same length, and calculate the corresponding confidence limit using equation (7). The number m of efficient sources is the number of intervals with the first largest confidence limit and the samples in the intervals with the last least confidence limit will be deleted.
- 4) Observations in the intervals with the first largest confidence limit are chosen as the training samples, based on which, super-planes are constructed using multi-class SVM [14], and $\tilde{x}(t), t = 1, 2, \dots, \tilde{T}$ will be classified into m classes;
- 5) Reconstruct intervals for each class, and obtain m clustering centers through sums with weights which got from confidence limit of intervals in each class. These centers are corresponding to columns of the mixing matrix.

3.2 Online Algorithm

When observations are sufficient, online algorithm can be adopted for estimating mixing matrix. For the existing q observations with sparsity, calculating the mean μ and variance σ , the first best confidence intervals $U_i, i = 1, 2, \dots, m$, the last worst confidence $V_i, i = 1, 2, \dots, p$ and the corresponding mixing matrix are obtained using method above. The online algorithm is as following:

Begin: for the $q+1$ sample x

- 1) generalize x into x' :

$$x' = \frac{x - \mu}{\sigma} \tag{10}$$

- 2) calculate the direction angles θ' of x' using equation (6),

if $\theta' \in \bigcup_{i=1}^p V_i$

delete x' , and the mixing matrix remains the same, goto "Begin";

else if $\theta' \in \bigcup_{i=1}^m U_i$

reconstruct training samples, renew clustering centers such that renew mixing matrix;

else

let super-plane remain the same, and renew the clustering center of the class to which x' belongs, so as to renew mixing matrix;

- 3) renew μ, σ

$$\begin{cases} \mu^{new} = (1 - \alpha)\mu + \alpha x' \\ \sigma^{new} = \frac{q}{q+1}\sigma + \frac{1}{q+1}x' \odot x' \end{cases} \tag{11}$$

Where $z = x \odot y$ means that $z_i = x_i \cdot y_i$

- 4) goto "Begin";

4 Simulations and Discussions

The following simulations will be introduced to display the advantages of the algorithms in this work using two kinds of indices as follows:

Angle Error of a, b :
$$AE(a, b) = \frac{180}{\pi} \arccos\left(\frac{a^T b}{\|a\|_2 \cdot \|b\|_2}\right), \tag{12}$$

where a, b are column vectors with the same dimension.

Signal-Noise Ratio:
$$SNR_i = 10 \log \frac{E[s_i(k)]^2}{E[(y_i(k) - s_i(k))^2]}. \tag{13}$$

Obviously, better estimation of mixing matrix and sources are corresponding to lower AE and higher SNR .

In this experiment, three speech signals are used (see Fig. 1). They are mixed by a random matrix and 16000 samples are chosen for each signal for online learning.

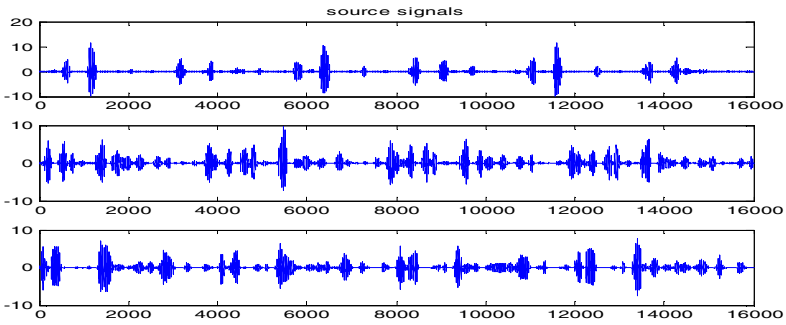


Fig. 1. Three speech source signals

Mixing matrix is as follows:

Mixing Matrix:
$$A = [a_1, a_2, a_3] = \begin{bmatrix} 0.8275 & -0.6674 & 0.1465 \\ 0.5614 & 0.7447 & -0.9892 \end{bmatrix}. \tag{14}$$

4.1 Experiments for Batch Algorithm

Based on the method above, the mixing matrix is estimated as following:

Estimating Matrix:
$$\tilde{A} = [\tilde{a}_1, \tilde{a}_2, \tilde{a}_3] = \begin{bmatrix} 0.8315 & -0.6481 & -0.1780 \\ 0.5564 & 0.7621 & 0.9852 \end{bmatrix}. \tag{15}$$

And the result based on k-means clustering is:

Estimating Matrix: $\hat{A} = [\hat{a}_1, \hat{a}_2, \hat{a}_3] = \begin{bmatrix} 0.8431 & -0.6393 & -0.1992 \\ 0.5378 & 0.7689 & 0.9800 \end{bmatrix}$ (16)

The distribution of direction angles of the observations (Fig. 2) and the sparsity (Spa), SNR and Angle Errors corresponding to sources (Table 1) with two methods above are as following:

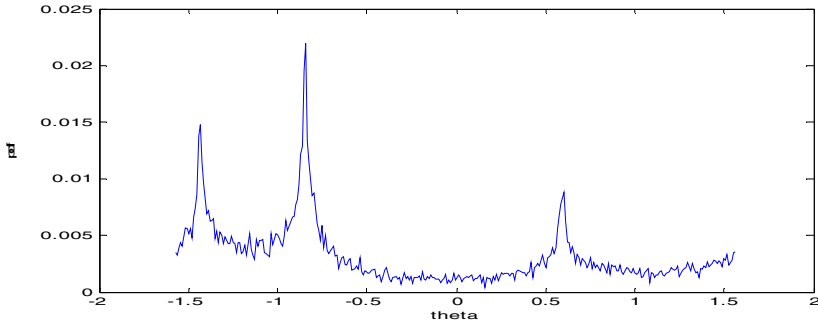


Fig. 2. Distribution of direction angles

Table 1. Spa, SNR, and AE from the methods above

sources	Spa	SNR	AE(a_i, \tilde{a}_i)	AE(a_i, \hat{a}_i)
s_1	0.8020	29.3061	0.3798	1.7460
s_2	0.6537	19.6893	1.4884	2.1242
s_3	0.6023	17.8069	1.8171	3.0658

Using SVM in this work, the distribution of the direction angles of samples after filtering (Fig. 3) and the recovering sources (Fig. 4) are as following:

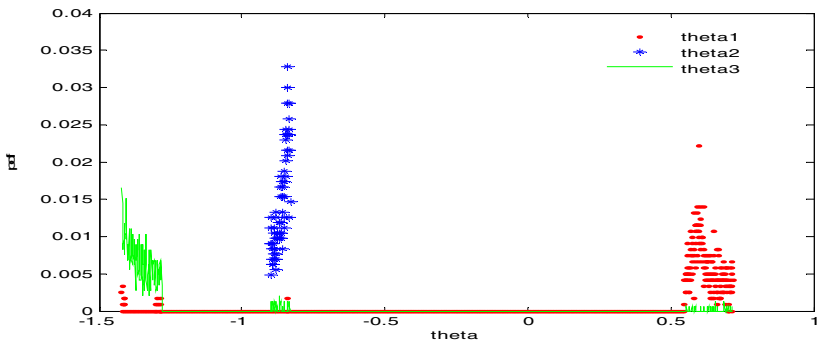


Fig. 3. Distribution of direction angles from SVM

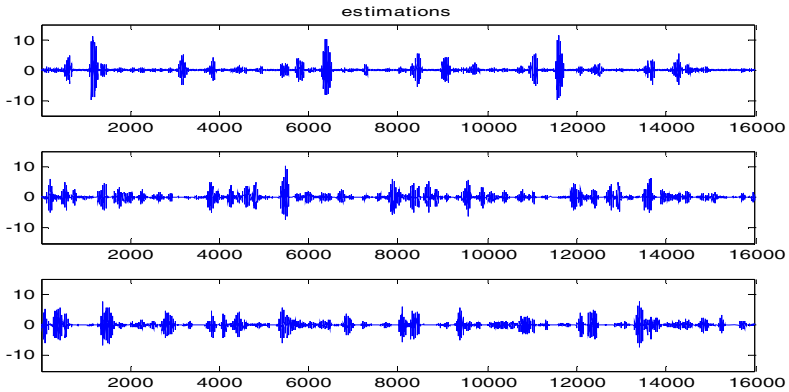


Fig. 4. Recovery signals of sources

4.2 Experiments for Online Algorithms

When samples are sufficient, the mixing matrix can be estimated with online algorithm. The direction angle errors of sources are as following (Fig. 5)

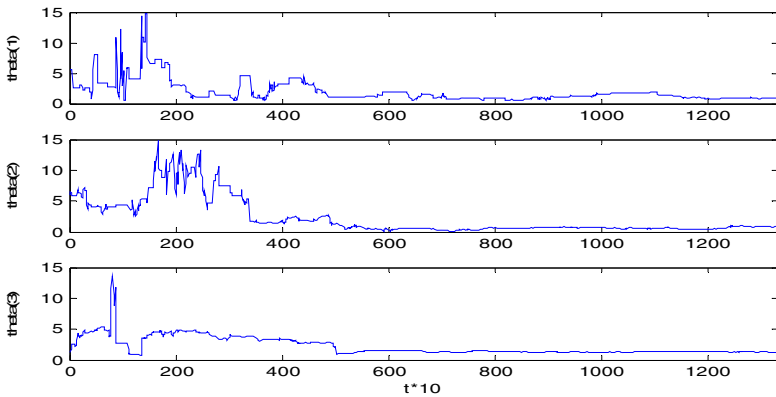


Fig. 5. Direction angle error using online algorithm

5 Conclusions

In this work, TSM is used for underdetermined BSS. In the first step, mixing matrix is estimated using SVM, together with the number of the source signals and the corresponding sparsity measure is introduced. The result is independent on the initial condition and there are no assumptions for the stationarity and independence of sources using shortest method to recover sources. Both batch and online methods are proposed for estimating mixing matrix and good simulations show that our algorithms work quite well.

Acknowledgement

The work is supported by National Natural Science Foundation of China for Excellent Youth (Grant 60325310), Guangdong Province Science Foundation for Program of Research Team (Grant 04205783), Specialized Prophasic Basic Research Projects of Ministry of Science and Technology, China (Grant 2005CCA04100), the National Natural Science Foundation of China (Grant 60505005, 60674033), the Natural Science Fund of Guangdong Province, China (Grant 05103553, 05006508), Key Program of National Natural Science Foundation of China (Grant U0635001).

References

1. Lewicki, M.S., Sejnowski, T.J.: Learning Overcomplete Representations. *Neural Computation* **12** (2000) 337-365
2. Bofill, P., Zibulevsky, M.: Underdetermined Blind Source Separation Using Sparse Representations. *Signal Processing* **81** (2001) 2353-2362
3. Li, Y., Cichocki, A., Amari, S.: Analysis of Sparse Representation and Blind Source Separation. *Neural Computation* **16** (2004) 1193-1234
4. Li, Y., Amari, S., Cichocki, A., Daniel, W.C. Ho, Xie, S.L.: Underdetermined Blind Source Separation Based on Sparse Representations. *IEEE Trans on Signal Processing* **54** (2006) 423-437
5. He, Z., Cichocki, A.: K-EVD Clustering and Its Applications to Sparse Component Analysis. *LNCS* **3889** (2006) 90-97
6. Li, Y., Amari, S., Cichocki, A., Cuntai, G.: Probability Estimation for Recoverability Analysis of Blind Source Separation Based on Sparse Representation. *IEEE Trans on Information Theory* **52** (2006) 3139-3152
7. He, Z., Xie, S., Fu, Y.: Sparsity Analysis of Signals, *Progress in Natural Science* **16** (2006) 879-884
8. Comon, P.: Independent Component Analysis, A New Concept? *Signal Processing*, **36** (1994) 287-314
9. Xiao, M., Xie, S. L., Fu, Y.L.: A Novel Approach for Underdetermined Blind Sources Separation in Frequency Domain. *ISNN* **3497** (2005) 484-489
10. Xie, S., He, Z., and Fu, Y.: A Note on Stone's Conjecture of Blind Signal Separation. *Neural Computation* **17** (2005) 321-330
11. Zibulevsky, M., Pearlmutter, B.A.: Blind Source Separation by Sparse Decomposition. *Neural Computation* **13** (2001) 863-882
12. Xie, S.L., He, Z.S., Gao, Y.: *Adaptive Theory of Signal Processing*. 1st ed. Chinese Science Press, Beijing (2006) 103-129
13. Boser, E.B., Guyon, M., Vapnik, V.: A Training Algorithm for Optimal Margin Classifiers. *Proceedings of the 5th Annual ACM Workshop on COLT* (1992) 144-152
14. Ana, C. L., Carvalho, C.P.L.F.: Comparing Techniques for Multiclass Classification Using Binary SVM Predictors. *LNAI* **2972** (2004) 272-281
15. Tan, B., Li, X.: Estimation of Source Signals Number and Underdetermined Blind Separation Based on Sparse Representation. *ICCIS* (2006) 1730-1733

Predicting Time Series Using Incremental Lagrangian Support Vector Regression

Hua Duan^{1,2}, Weizhen Hou², Guoping He², and Qingtian Zeng²

¹ Department of Mathematics, Shanghai Jiaotong University, Shanghai 200240, P.R. China

² College of Information Science and Engineering, Shandong University of Science and Technology, Qingdao 266510, China
{hduan, hegp, qtzeng}@sdust.edu.cn

Abstract. A novel Support Vector Regression(SVR) algorithm has been proposed recently by us. This approach, called Lagrangian Support Vector Regression(LSVR), is an reformulation on the standard linear support vector regression, which leads to the minimization problem of an unconstrained differentiable convex function. During the process of computing, the inversion of matrix after incremented is solved based on the previous results, therefore it is not necessary to relearn the whole training set to reduce the computation process. In this paper, we implemented the LSVR and tested it on Mackey-Glass time series to compare the performances of different algorithms. According to the experiment results, we achieve a high-quality prediction about time series.

1 Introduction

Time is a phenomenon which is both very complex and very important in many real world problems. Its importance comes from the fact that almost every kind of data contains time-dependent information. Time series is the representation of time. In the modelling of time series, two of the key problems are noise and non-stationarity. The noisy characteristic refers to the unavailability of complete information from the past behaviors of the time series to catch the dependency between the future and the past. The information that is not included in the model is considered as noise. The non-stationarity characteristic implies that the distribution of time series is changing over time. This will lead to gradual changes in the dependency between the input and output variables.

Support vector machines(SVM)^{[1][2]} is a very specific type of learning algorithms characterized by the capacity control of the decision function, the use of kernel functions and the sparsity of the solution. SVMs have been applied successfully to classification and regression with the introduction of ε -insensitive loss function. We also call ε -support vector regression(SVR). SVR^[13] is a powerful technique for predictive data analysis with many applications to varied areas of study. For example, SVR has been used in drug discovery^[15], civil engineering^[16], and time series prediction^{[10][11][12][17]}.

Based on the fast and efficient properties of Lagrange Support Vector Regression (LSVR), we propose the online incremental regression algorithms for LSVR. In the algorithms proposed, the inversion of matrix is solved by iteration when the new samples are added, therefore it is no necessary to relearn the whole training set to reduce the computation process. In this paper, we implemented the LSVR and tested it on Mackey-Glass time series to compare the performances of different algorithms. According to the experimental results, the algorithms proposed in this paper are much efficient in time series. The paper is organized as follows: section briefly sketches the algorithm, Section 3 gives experiments on Mackey-Glass time series, and Section concludes the whole paper.

2 Lagrange Support Vector Regression

Lagrange Support Vector Machine (LSVM) is transformed from a standard SVM in [3], and Lagrange Support Vector Regression (LSVR) is defined in [4]. Similarly to the classification problem, the training set of the regression problem, $T = \{(x_i, y_i) | x_i \in R^n, y_i \in R\}, i = 1, \dots, m$, where x_i is a sample of an n -dimensional space, and $y_i \in R$ is the target value of the corresponding x_i , which is different from the class label in the classification problem. The regression problem is to construct a predicted function $f(x)$ based on the training set, in which the decision function can be represented by fewer support vectors (That is to say the sparsity (e.g.[7]) of the classifier.) While the approach is extended to the regression problem, it is also expected to construct the SVR algorithm satisfying the sparsity. Since the sparsity of the SVR algorithm is related to the loss function, Vapnik^[1] introduced the ε -insensitive loss function:

$$c(x, y, f(x)) = |y - f(x)|_\varepsilon,$$

such that

$$|y - f(x)|_\varepsilon = \max\{0, |y - f(x)| - \varepsilon\},$$

where ε is a predefined positive number.

This defines an ε tube so that the loss is zero if the predicted value is within the tube. If the predicted point is outside the tube, the loss is the magnitude of the difference between the predicted value and the radius ε of the tube.

LSVR is a reformulation of the standard regression problem, in which the $\xi^{(*)}$ in the objective function is changed from 1-norm to the square of 2-norm, so the non-negative constraint of $\xi^{(*)}$ is omitted. The maximal margin of the LSVR is also changed from the n -dimensional to $(n + 1)$ -dimensional space, i.e., (w, b) . The regression problem after reformulation is strong convex and its solution is almost identical to that of the standard SVR problem^[3].

The linear LSVR problem is:

$$\begin{aligned} & \min \frac{1}{2}(\|w\|^2 + b^2) + \frac{C}{2}(\xi^{*T}\xi + \xi^T\xi) \\ & s.t. \begin{pmatrix} A & e_m \\ -A & -e_m \end{pmatrix} \begin{pmatrix} w \\ b \end{pmatrix} - \begin{pmatrix} y \\ -y \end{pmatrix} \leq \varepsilon e_{2m} + \begin{pmatrix} \xi \\ \xi^* \end{pmatrix}. \end{aligned} \tag{1}$$

The dual model is:

$$\begin{aligned} \min & \frac{1}{2}(\alpha^* - \alpha)^T(AA^T + e_m e_m^T)(\alpha^* - \alpha) - y^T(\alpha^* - \alpha) + \\ & \varepsilon e_m^T(\alpha^* + \alpha) + \frac{1}{2C}(\alpha^{*T}\alpha^* + \alpha^T\alpha) \\ \text{s.t. } & \alpha^{(*)} \geq 0. \end{aligned} \tag{2}$$

According to the KKT condition, w and b can be solved based on the solution of the dual problem:

$$w = A^T(\alpha^* - \alpha), \quad b = e^T(\alpha^* - \alpha).$$

Therefore, the optimal regression function is,

$$f(x) = (\alpha^* - \alpha)^T Ax + e^T(\alpha^* - \alpha).$$

Let $\hat{\alpha}^T = (\alpha^{*T} \ \alpha^T)$, then the dual problem shown in (2) can be transformed into:

$$\min_{\hat{\alpha} \geq 0} \frac{1}{2} \hat{\alpha}^T \left(\frac{I}{C} + \begin{pmatrix} AA^T + e_m e_m^T & -(AA^T + e_m e_m^T) \\ -(AA^T + e_m e_m^T) & AA^T + e_m e_m^T \end{pmatrix} \right) \hat{\alpha} - ((y - \varepsilon e_m)^T, -(y + \varepsilon e_m)^T) \hat{\alpha}. \tag{3}$$

Let $H = \begin{pmatrix} A & e_m \\ -A & -e_m \end{pmatrix}_{2m \times (n+1)}$, $Q = \frac{I}{C} + HH^T$, and $l^T = ((y - \varepsilon e_m)^T, -(y + \varepsilon e_m)^T)$.

Equation (3) can be transformed into:

$$\min_{\hat{\alpha} \geq 0} \frac{1}{2} \hat{\alpha}^T Q \hat{\alpha} - l^T \hat{\alpha}, \tag{4}$$

therefore the sufficient and necessary KTT condition of problem (4) is:

$$0 \leq \hat{\alpha} \perp Q\hat{\alpha} - l \geq 0. \tag{5}$$

LSVR algorithm is to solve the above KTT condition to obtain the solution of the problem.

For each two real numbers (or vectors) a and b ,

$$0 \leq a \perp b \geq 0 \Leftrightarrow a = (a - \lambda b)_+, \lambda > 0,$$

where $x_+ = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$. Thus, the KKT condition in (5) can be represented as:

$$Q\hat{\alpha} - l = ((Q\hat{\alpha} - l) - \lambda\hat{\alpha})_+, \quad \lambda > 0. \tag{6}$$

Therefore, the following simple iteration formula for the LSVR algorithm can be obtained based on the above optimality condition,

$$\hat{\alpha}^{i+1} = Q^{-1} (l + ((Q\hat{\alpha}^i - l) - \lambda\hat{\alpha}^i)_+), \quad \lambda > 0, \quad i = 0, 1, 2, \dots \tag{7}$$

While $0 < \lambda < \frac{2}{C}$, the algorithm is the global linear convergence from any starting point [3]. The inversion of m matrix Q changes to the inversion of $n +$

$1(n \ll m)$ matrix by using SMW identity. This leads to process large data sets feasibly, and the computation time is reduced.

The SMW identity is:

$$\left(\frac{I}{C} + HH^T\right)^{-1} = C \left(I - H\left(\frac{I}{C} + H^T H\right)^{-1} H^T\right),$$

where $C > 0$ and H is an $m \times n$ matrix.

When the LSVR algorithm is extended from the linear case to the nonlinear classification case, $Q = \frac{I}{C} + K(H, H^T)$, and the optimal regression function is

$$f(x) = (\alpha^* - \alpha)^T K(A, x) + e^T (\alpha^* - \alpha).$$

Other properties are same to those of the linear case.

Let us consider the case that there is one new sample added into the training set. Assume that there are m samples in the original training set T represented by A , and the sample added newly is x_{m+1} .

The corresponding dual problem after adding x_{m+1} is:

$$\begin{aligned} \min \frac{1}{2}(\alpha^{*T} \ \alpha_{m+1}^* \ \alpha^T \ \alpha_{m+1}) Q_{new} \begin{pmatrix} \alpha^* \\ \alpha_{m+1}^* \\ \alpha \\ \alpha_{m+1} \end{pmatrix} - \hat{\gamma}^T \begin{pmatrix} \alpha^* \\ \alpha_{m+1}^* \\ \alpha \\ \alpha_{m+1} \end{pmatrix}, \quad (8) \\ (\alpha^{*T} \ \alpha_{m+1}^* \ \alpha^T \ \alpha_{m+1})^T \geq 0, \end{aligned}$$

where $\hat{\gamma}^T = ((y - \varepsilon e_m)^T, y_{m+1} - \varepsilon, -(y + \varepsilon e_m)^T, -(y_{m+1} + \varepsilon))$, and $\hat{\alpha}_{new}^T = (\alpha^{*T}, \alpha_{m+1}^*, \alpha^T, \alpha_{m+1})$.

Let $h = (x_{m+1}^T, 1)$, and $H_1 = (A \ e)$, therefore $H = \begin{pmatrix} H_1 \\ -H_1 \end{pmatrix}$, and $H_{new}^T = (H_1^T \ h^T \ -H_1^T \ -h^T)$.

The solution of the new problem can be computed using Equation (7). According to Equation (7), the key problem of the iteration is to solve Q_{new}^{-1} . In order to solve Q_{new}^{-1} , the following lemmas are given first.

Lemma 1^[8]. If $T = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$ and the matrix D is inverse, then

$$T^{-1} = \begin{pmatrix} (A - BD^{-1}C)^{-1} & -(A - BD^{-1}C)^{-1}BD^{-1} \\ -D^{-1}C(A - BD^{-1}C)^{-1} & D^{-1}C(A - BD^{-1}C)^{-1}BD^{-1} + D^{-1} \end{pmatrix}.$$

Lemma 2^[9]. If $1 + v^T A^{-1}u \neq 0$, then $(A + uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^T A^{-1}}{1 + v^T A^{-1}u}$. ($u, v \in R^n$, and A is nonsingular).

Following the above lemmas, we present the following strategies for online incremental regression, which includes two cases: linear case and nonlinear case.

(1) Linear Case

In the linear case, $Q = \frac{I_{2m}}{C} + HH^T$, and $Q_{new} = \frac{I_{2m+2}}{C} + H_{new}H_{new}^T$. Based on the SMW identity,

$$\begin{aligned} Q_{new}^{-1} &= C \left(I_{2m+2} - H_{new} \left(\frac{I_{n+1}}{C} + H_{new}^T H_{new} \right)^{-1} H_{new} \right) \\ &= C \left(I_{2m+2} - H_{new} \left(\frac{I_{n+1}}{C} + H^T H + 2h^T h \right)^{-1} H_{new} \right). \end{aligned}$$

Let $B = \frac{I_{n+1}}{C} + H^T H$, and B^{-1} be the computation value of the previous step. According to Lemma 2, $(B + 2h^T h)^{-1} = \left(B^{-1} - \frac{2B^{-1}h^T h B^{-1}}{1 + 2hB^{-1}h^T} \right)$, therefore

$$Q_{new}^{-1} = C \left(I_{2m+2} - H_{new} \left(B^{-1} - \frac{2B^{-1}h^T h B^{-1}}{1 + 2hB^{-1}h^T} \right) H_{new}^T \right). \quad (9)$$

(2) Nonlinear Case

In the nonlinear case, $Q = \frac{I_{2m}}{C} + K(H, H^T)$, $Q_{new} = \frac{I_{2m+2}}{C} + K(H_{new}, H_{new}^T)$, therefore SMW identity is not available. We have elementary operation on $H_{new}H_{new}^T$ first.

In the nonlinear case, $Q = \frac{I_{2m}}{C} + K(H, H^T)$. We have elementary operation on $H_{new}H_{new}^T$ first.

$$P_2 P_1 H_{new} H_{new}^T Q_1 Q_2 = \begin{pmatrix} HH^T & Hh^T & \mathbf{0}_{2m \times 1} \\ hH^T & hh^T & 0 \\ \mathbf{0}_{1 \times 2m} & 0 & 0 \end{pmatrix} = \overline{H},$$

where

$$\begin{aligned} P_1 &= \begin{pmatrix} I_m & \mathbf{0}_{m \times 1} & \mathbf{0}_{m \times m} & \mathbf{0}_{m \times 1} \\ \mathbf{0}_{1 \times m} & 1 & \mathbf{0}_{1 \times m} & 0 \\ \mathbf{0}_{m \times m} & \mathbf{0}_{m \times 1} & I_m & \mathbf{0}_{m \times 1} \\ \mathbf{0}_{1 \times m} & 1 & \mathbf{0}_{1 \times m} & 1 \end{pmatrix}, \quad P_2 = \begin{pmatrix} I_m & \mathbf{0}_{m \times 1} & \mathbf{0}_{m \times m} & \mathbf{0}_{m \times 1} \\ \mathbf{0}_{m \times m} & \mathbf{0}_{m \times 1} & I_m & \mathbf{0}_{m \times 1} \\ \mathbf{0}_{1 \times m} & 1 & \mathbf{0}_{1 \times m} & 0 \\ \mathbf{0}_{1 \times m} & 0 & \mathbf{0}_{1 \times m} & 1 \end{pmatrix}, \\ Q_1 &= \begin{pmatrix} I_m & \mathbf{0}_{m \times 1} & \mathbf{0}_{m \times m} & \mathbf{0}_{m \times 1} \\ \mathbf{0}_{1 \times m} & 1 & \mathbf{0}_{1 \times m} & 1 \\ \mathbf{0}_{m \times m} & \mathbf{0}_{m \times 1} & I_m & \mathbf{0}_{m \times 1} \\ \mathbf{0}_{1 \times m} & 0 & \mathbf{0}_{m \times 1} & 1 \end{pmatrix}, \quad Q_2 = \begin{pmatrix} I_m & \mathbf{0}_{m \times m} & \mathbf{0}_{m \times 1} & \mathbf{0}_{m \times 1} \\ \mathbf{0}_{1 \times m} & \mathbf{0}_{1 \times m} & 1 & 0 \\ \mathbf{0}_{m \times m} & I_m & \mathbf{0}_{m \times 1} & \mathbf{0}_{m \times 1} \\ \mathbf{0}_{1 \times m} & \mathbf{0}_{1 \times m} & 0 & 1 \end{pmatrix}, \end{aligned}$$

$$H_{new}H_{new}^T = P_1^{-1} P_2^{-1} \overline{H} Q_2^{-1} Q_1^{-1},$$

$$\begin{aligned} Q_{new} &= \frac{I_{2m+2}}{C} + K(H_{new}, H_{new}^T) \\ &= P_1^{-1} P_2^{-1} \left(\frac{P_2 P_1 Q_1 Q_2}{C} + \begin{pmatrix} K(H, H^T) & K(H, h^T) & \mathbf{0}_{2m \times 1} \\ K(h, H^T) & K(h, h^T) & 0 \\ \mathbf{0}_{1 \times 2m} & 0 & 0 \end{pmatrix} \right) Q_2^{-1} Q_1^{-1} \\ &= P_1^{-1} P_2^{-1} \left(\frac{I_{2m+2}}{C} + K(H, H^T) \quad K(H, h^T) \quad \mathbf{0}_{2m \times 1} \right. \\ &\quad \left. \begin{matrix} K(h, H^T) & \frac{1}{C} + K(h, h^T) & \frac{1}{C} \\ \mathbf{0}_{1 \times 2m} & \frac{1}{C} & \frac{1}{C} \end{matrix} \right) Q_2^{-1} Q_1^{-1}. \end{aligned}$$

$$\text{Let } B = (K(H, h^T) \quad \mathbf{0}_{2m \times 1}), \quad B^T = \begin{pmatrix} K(h, H^T) \\ \mathbf{0}_{1 \times 2m} \end{pmatrix}, \quad D = \begin{pmatrix} \frac{1}{C} + K(h, h^T) & \frac{1}{C} \\ \frac{1}{C} & \frac{1}{C} \end{pmatrix},$$

$$D^{-1} = \frac{C^2}{1 + 2CK(h, h^T)} \begin{pmatrix} \frac{2}{C} & -\frac{1}{C} \\ -\frac{1}{C} & \frac{1}{C} + K(h, h^T) \end{pmatrix}.$$

According to Lemma 1,

$$\begin{pmatrix} Q & B \\ B^T & D \end{pmatrix}^{-1} = \begin{pmatrix} (Q - BD^{-1}B^T)^{-1} & -(Q - BD^{-1}B^T)^{-1}BD^{-1} \\ -D^{-1}B^T(Q - BD^{-1}B^T)^{-1} & D^{-1}B^T(Q - BD^{-1}B^T)^{-1}BD^{-1} + D^{-1} \end{pmatrix}.$$

According to Lemma 2,

$$\begin{aligned} (Q - BD^{-1}B^T)^{-1} &= \left(Q - \frac{2CK(H, h^T)K(H, h^T)^T}{1 + 2CK(h, h^T)} \right)^{-1} \\ &= Q^{-1} + \frac{2CQ^{-1}K(H, h^T)K(H, h^T)^TQ^{-1}}{1 + 2CK(h, h^T) - K(H, h^T)Q^{-1}K(H, h^T)} = A, \\ Q_{new}^{-1} &= Q_1Q_2 \begin{pmatrix} A & -ABD^{-1} \\ -D^{-1}B^T A & D^{-1}B^T ABD^{-1} + D^{-1} \end{pmatrix} P_2P_1. \end{aligned} \tag{10}$$


Therefore, we can compute the new solution by using equation 7.

3 Experimental Results

In order to test the performance of the algorithms presented in this paper, the experiments are made based on time series data set. In all experiments, 10-fold cross-validation was used to get an estimation of the mean squared error(MSE). The experiments are implemented by Matlab 7.0, and they run on PC environment. The main configurations of the PC are: (1) CPU: Pentium IV 2.0G, (2) Memory: 256M, and (3) OS: Windows XP.

The data considered in this experiment are a high dimensional chaotic system generated by the Mackey-Glass delay differential equation, which are originally introduced as a model of blood cell regulation^[6] and become quite common as artificial forecasting benchmark. The equation is :

$$\frac{dx(t)}{dt} = -0.1x(t) + \frac{0.2x(t - \Delta)}{1 + x(t - \Delta)^{10}}$$

with parameter $\Delta = 17$. In order to keep consistent with the approaches in [4], the initial condition for the above equation is $x(t) = 0.9$ for $0 \leq t \leq \Delta$. The series is generated by numerical integration using a fourth order Runge-Kutta method. Figure.  presents the first 1000 points of the time series. The dataset is divided as follows: (1) the first 400 points are used for training set; (2) the points in the range from 401th to 500th provide a validation set to select parameters; and (3) the prediction error is measured on the test set in the range from 501th to 1000th. The kernel for LSVR is a Gaussian Radial Basis Kernel $K(x, y) = \exp(-\|x - y\|^2/2\sigma^2)$.

First some preliminary experiments based on validation set ensure that the free parameters σ , C , and ε work fairly well. Through experiments, the values of the free parameters are determined, which are $\sigma = 1$, $C = 100$, and $\varepsilon = 0.1$.

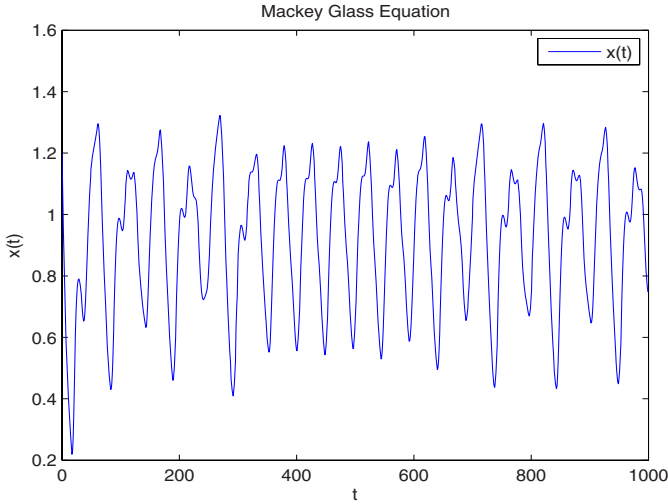


Fig. 1. 1000 points of the Mackey-Glass time series MG_{17}

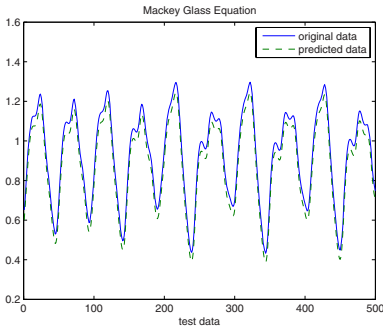


Fig. 2. The predicted results of test set of MG_{17}

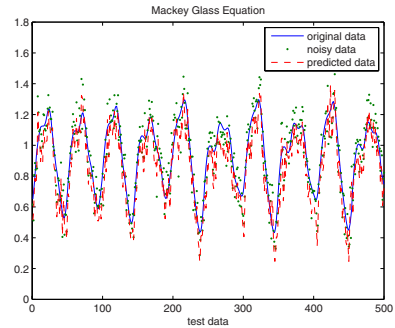


Fig. 3. The predicted results of test set of MG_{17} with noise, $\theta = 0.1$

Table 1. The comparison results of three methods for Mackey-Glass time series

Dataset	Methods	(σ, C, ϵ)	MSE
Mackey-Glass	SVM^{light}	(1,1000,0.1)	0.0049
	SOR-SVR	(1,1000,0.1)	0.0047
	LSVR	(1,100,0.1)	0.0021

In order to test the prediction performance of the algorithms proposed in this paper, we have experiments on the test set. In order to test the robustness of our algorithm, we add normal distribution noise model Ω with mean 0 and $E\Omega = \theta^2$. The experimental results are shown in Figure. 2 and Figure. 3, respectively.

We compare our algorithm with other approaches such as SVM^{light} and SOR-SVR(Successive OverRelaxation)^[14]. The experimental results are shown in Table 1.

4 Conclusions

This paper presents the online incremental learning algorithms for LSVR that can be used for time series analysis. The main contributions of our algorithm are that it is not necessary to re-learn the whole data set while a new sample is added. The experimental results show that our algorithm performs very well on time series prediction.

Acknowledgement

The work presented in this paper is fully supported by national science foundation of China (10571109 and 60603090).

References

1. Vapnik, V.: The Nature of Statistical Learning Theory. Springer-Verlag, New-York (1995)
2. Deng, N.Y., Tian, Y.J.: New Methods in Data Mining—Support Vector Machine. Science Press, Beijing (2004)
3. Mangasarian, O.L., Musicant, D.R.: Lagrangian Support Vector Machines. Journal of Machine Learning Research, **1** (2001) 161-177
4. Shao, X.J., He, G.P.: Lagrange Support Vector Regression. Intelligent Information Management Systems and Technologies **1** (3) (2005) 434-440
5. Casdagli, M.: Nonlinear Prediction of Chaotic Time-series. Physica D **35** (1989) 335-356
6. Mackey, M.C., Glass, L.: Oscillation and Chaos in Physiological Control Systems. Science **197** 287-294
7. Cortes, C., Vapnik, V.: Support Vector Networks. Machine learning **20** (1995) 273-297
8. Peking University. Advanced algebra. China High Education Press, Beijing (1987)
9. Yuan, Y.X., Sun, W.Y.: Optimization Theories and Tethods. Science Press, Beijing (1997)
10. Mukherjee, S., Osuna, E., Girosi, F.: Nonlinear Prediction of Chaotic Time Series using Support VectorMachines. Proc IEEE NNSP97, Amelia Island, FL (1997)
11. Muller, K.R., Smola, A.J., Ratsch, G., Scholkopf, B., Kohlmorgen, J.: Using Support Vector Machines for Time Series Prediction. in: B. Scholkopf, C.J.C. Burges, A.J. Smola (Eds.), Advances in Kernel MethodsSupport Vector Learning, MIT Press, Cambridge, MA (1999) 243-254
12. Muller, K.R., Smola, J.A., Ratsch, G., Scholkopf, B., Kohlmorgen, J. , Vapnik, V.N.: Predicting Time Series with Support Vector Machines. in: ICANN97: Proceedings of the seventh International Conference on Arti3cial Neural Networks, Lausanne, Switzerland (1997) 999-1004

13. Smola, A.J., Schölkopf, B.: A Tutorial on Support Vector Regression. NeuroCOLT Technical Report NC-TR-98-030, Royal Holloway College, University of London, UK (1998)
14. Quan, Y., Yang, J., Yao, L.X., Ye, C.Z.: Successive Over Relaxation for Support Vector Regression. *Journal of Software* **15** (2) (2004) 200-206
15. Demiriz, A., Bennett, K., Breneman, C., Embrechts, M.: Support Vector Machine Regression in Chemometrics. *Computing Science and Statistics* (2001)
16. Dibike, Y.B., Velickov, S., Solomatine, D.: Support Vector Machines: Review and Applications in Civil Engineering. In *AI Methods in Civil Engineering Applications*, O.Schleider and A.Zijderveld, Eds., Cottbus (2000) 45-58
17. Collobert, R., Bengio, S.: SVM Torch: Support Vector Machines for Large-scale Regression Problems. *Journal of Machine Learning Research* **1** (1) (2001) 143-160

A New Approach for Image Restoration Based on CNN Processor

Jianye Zhao, Quansheng Ren, Jian Wang, and Hongling Meng

Department of Electronics, Peking University,
100871 Beijing, China
phdzjy@263.net

Abstract. A new approach for maximum posterior probability (MAP) image restoration based on cellular neural network (CNN) is proposed in this paper, and hardware realization is also discussed. According to analysis of MAP image restoration, a new template is proposed for CNN image restoration. The computer simulation result proves the approach is reasonable, then a hardware system based on CNN processor is setup for the restoration algorithm, and the effectiveness of the CNN processor is also confirmed in this system.

1 Introduction

Cellular Neural Network (CNN) is a parallel processing system [1,2] that has been utilized widely in the field of image processing [3]. It is attractive that CNN could be easily realized with electronic circuit. Though the structure of CNN is simpler than that of Hopfield Neural Network, such advantage as parallel processing ability is still kept. There is the definition of energy function in the field of CNN, so such algorithm based on stochastic image model could be utilized when the template is designed.

Because each cell of CNN is only connected with its neighbor, CNN is more easily realized than Hopfield Neural Network. The connection relationship is similar and definite, so it is easily realized with circuit or hardware system. According to the similarities between stochastic image model [4] and CNN, the CNN template for MAP image restoration is designed in this paper. After the template of CNN is found, the design method for integrate circuit is employed, then a neural network processor based on FPGA is easily set up.

This paper is organized as follows, the similarities of Gibbs image model (GIM) and CNN is shown in section 2. Then physics meaning of MAP restoration algorithm based on GIM is analyzed in next section,. According to the analysis, the CNN template for MAP binary image restoration is proposed, and some simulation results are also shown in section 3. Section 4 describes how to set up a CNN processor for image restoration based on FPGA. At last the conclusion is given in Section 5.

2 Similarity Between Gibbs Image Model and CNN

At first some basic definitions in Gibbs image are introduced. Think about random fields defined over a rectangular $N_1 \times N_2$ lattice of points:

$L = \{(i, j) | 1 \leq i \leq N_1, 1 \leq j \leq N_2\}$, (i, j) is a pixel, to the image which is processed in this paper, $N_1 = N_2 = 512$. A very important concept in Gibbs image model is *clique*, a clique C is a subset of L , and following condition should be satisfied: (1) C consists of a single pixel, or (2) for $(i, j) \neq (k, l)$, $(i, j) \in C$ and $(k, l) \in C$ implies that (i, j) in the neighborhood system of (k, l) .

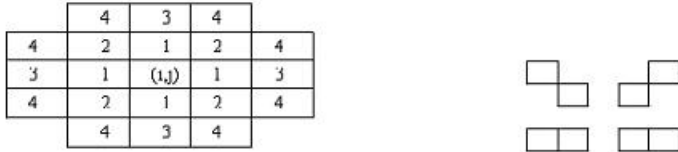


Fig. 1. Neighborhood system of pixel (i, j) and Clique which contains two pixels

The Arabic numerals 1,2 mean that the correspond pixel is in the *m*th order neighborhood system.

Another important concept is potential function, the appearing probability of image $X = \{X_{ij}\}$ is decided by $U(x)$: the potential function of the image. The clique’s potential is denoted by $V_c(x)$, and $U(x)$ is the sum of each clique’s potential. This relationship between appearing probability and $U(x)$ is described as following equations:

$$P(X = x) = \frac{e^{-U(x)}}{Z}, \tag{1}$$

$$U(x) = \sum_{c \in C} V_c(x), \tag{2}$$

$$Z = \sum_x e^{-U(x)}. \tag{3}$$

When Gibbs image model is utilized for image restoration question, the initial normalized image $Y = \{Y_{ij}\}$ is noisy, and we assume the noise is white gauss noise which its mean value is 0, then the relationship between Y and X is defined as follows:

$$Y_{ij} = F(X_{ij}) + W_{ij}, \tag{4}$$

Y_{ij} is the gray-level value of pixel (i, j) . X_{ij} is region type of pixel (i, j) , the random field X consist of M discrete values. X_{ij} is 0 or 1 to binary image. $F(\cdot)$ is a function which maps the region type to the corresponding gray-level. We can make MAP estimation to the segmentation question:

$$P(X = x|Y = y) = \frac{P(Y = y|X = x)P(X = x)}{P(Y = y)}. \tag{5}$$

In order to get MAP (maximum a posterior) restoration result, a NP complete question need to be solved [4]. The cost of computation is very high. The approach based on CNN is easily realized with hardware system and the result could be gotten rapidly, so it is valuable to realize the statistical restoration algorithm with CNN. Now we note the similarity between Gibbs image model and CNN.

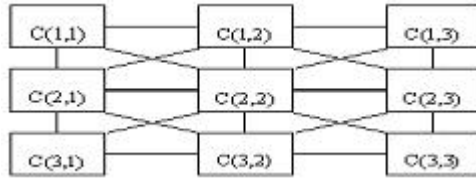


Fig. 2. Neighborhood system of pixel (i,j) and Clique which contains two pixels

The structure of network is presented in Fig.2. The cell C(i,j) is only linked with its *r-neighborhood* [4], only *1-neighborhood* of the cell C(i,j) is shown in Fig.2. Obviously, it is similar with neighborhood system η^d of Gibbs image model when $d=1, 2$. If these characteristics are utilized, more image problems could be processed by CNN. Equation (6) shows the state equation of CNN, and (7) shows the energy function of CNN. When CNN become stable, the energy function gets it local minimum or global minimum.

$$C \frac{dV_{Xij}(t)}{dt} = -\frac{1}{R_x} V_{Xij}(t) + \sum_{(k,l) \in N_r(i,j)} A_{ijkl} V_{Ykl}(t) + \sum_{(k,l) \in N_r(i,j)} B_{ijkl} V_{Ukl}(t) + I_{ij}, \tag{6}$$

$$E(t) = -\frac{1}{2} \sum_i \sum_j A(i, j; k, l) V_{Yij} V_{Ykl} - \sum_{i,j} IV_{Yij} + \frac{1}{2R_x} \sum_{i,j} V_{Yij}^2 - \sum_i \sum_j B(i, j; k, l) V_{Yij} V_{Ukl} \tag{7}$$

To make use of Gibbs image model to process an image question, we also need to minimize a cost function. The cost function gets to its global minimum, then the right solution is gotten. So the similarity is not lies in the structure, but also in the processing approach.

Only the clique that contains two pixels is used. The clique which contain one pixel is nonsense to the noisy image because statistic characteristic of the image can not be utilized [5]. To the clique which contains two pixels, its potential can be described as:

$$V_C(x) = -W(i, j; k, l) V_{ij} V_{kl}, \tag{8}$$

V_{ij} and V_{kl} is the gray value of the pixel, $W(i,j;k,l)$ is the connection parameter (coefficient of template A or B).

To make MAP estimation, formula (5) should be maximized. Since the noisy image is given, $P(Y=y)$ is also definite. So maximization of the numerator is equal to maximization of (9).

$$\ln P(Y = y|X = x) = -\frac{N_1 N_2}{2} \ln(2\pi\sigma^2) - \sum_{m=1}^M \sum_{i,j} \frac{(y_{ij} - q_m)^2}{2\sigma^2}, \tag{9}$$

$$\ln P(X = x) = -\ln Z - \sum_{c \in C} V_C(x). \tag{10}$$

Calculate logarithm of (9) and ignore the constant, the next equation should be minimized:

$$-\sum_{i,j} \sum_{k,l} W(i, j; k, l) V_{ij} V_{k,l} - \frac{1}{\sigma^2} \sum_{i,j} I_{ij} V_{ij} + \frac{1}{2\sigma^2} \sum_{ij} V_{ij}^2. \tag{11}$$

From (9),(11), if we minimize function (11) we can make MAP estimation

$$\min(-\sum_{i,j} \sum_{k,l} W(i, j; k, l) V_{ij} V_{k,l} - \frac{1}{\sigma^2} \sum_{i,j} I_{ij} V_{ij} + \frac{1}{2\sigma^2} \sum_{ij} V_{ij}^2). \tag{12}$$

Equation (12) is similar to CNN energy function (7), so CNN can be used to get minimum of (12), but the form of $W(i,j;k,l)$ is still unknown, in next section we will analyze the physics meaning of MAP restoration, and show the answer.

3 Physics Meaning of MAP Image Restoration

In this section we note the relationship between MAP restoration and special maximum entropy (ME) restoration [6], it’s helpful to analyze the meaning of MAP restoration. The discussion is focused on the linear image model. The corrupt image model is:

$$g = Hf + n_0. \tag{9}$$

In equation (13), g is the corrupt image; f is the original image, H the correlative matrix between g and f .

To the following ME estimation approach:

$$\max(-f^T \ln f). \tag{10}$$

Reference [6] prove that ME restoration is a kind of MAP restoration. The result of ME image restoration is similar with that of MAP estimation. We can get the CNN template from the physics meaning of ME image restoration [7]. The physics meaning of ME is very definite, H_f is the sum of the entropy, E is the sum of the gray-level, because the mean value of Gaussian noise is zero, E is a constant. So the gray-level of each pixel should be close when the entropy get its maximum value, so the meaning

of ME image restoration is smoothness. Based on the analysis and equation (9)(12),

the CNN template should be: $A = \begin{bmatrix} 1.0 & 1.0 & 1.0 \\ 1.0 & 1.05 & 1.0 \\ 1.0 & 1.0 & 1.0 \end{bmatrix}$, $B=0$, $C=1$, $R_x = 1$. Because CNN

should become stable, the medium factor should be greater than other. So we adopt the medium factor is 1.05 because of these two requirements.

4 Hardware Realization

After the parameter is definite, the CNN could be realized with CNN. We choose a chip that is named as Altera cyclone EPIC-20, there are 20060 logic units (LE) and 295K RAM in the chip [8]. Each pixel of the image corresponds to one LE. The most difficult is not the speed of calculation, it is the speed of data exchanging between CNN processor and other devices. To deal with this question, dual-access RAM is utilized. The following image is 128×128 pixel, so the size of dual-access RAM is. $130 \times 130 = 16.9\text{Kbytes}$, because the boundary pixel should be calculated specially, and the enlarged pixel is equal to the original boundary pixel. When the calculation is carried out, lots of processing data is created. The last result is rely on these data, so 128×128 Bytes dual-access RAM is defined for these data. FIFO (first in first out) RAM is defined for CNN template, and the size is 2Kbytes. Now the buffer is enough, so we can read the noisy image data, then calculate in FPGA.

In our experiment, the CNN processor is connected with DSP (TMS 320C5416) [9]. The MCBSP protocol (multi-channel synchronization port) is employed, and the speed of data exchange is 50Mbits/s. It's enough to process the small image. After DSP get the result, it transfers the data to the computer, and the results are shown at last.

The work procedure is described as follows:

- (1) Initial RAM, read the noisy data from DSP.
- (2) Multiply and add, calculate according to equation (6), the step is 0.002.
- (3) Write the result to the buffer, and if the calculation times are not enough, then calculate again. If the condition satisfied, transfer the data to DSP through MCBSP.
- (4)The CNN processor goes back to the waiting status, if a processing instruction is sent, the CNN processor will initial RAM again.



Fig. 3. Lenna



Fig. 4. Noisy image (SNR=0dB)



Fig. 5. Restoration

We test the system with a 128×128 image. Fig.3 is original image “LENA”. Fig.4 is the noisy image which is corrupted by Gauss noise, and the variance of noise is 1.0, and SNR=0dB. Fig.5 is the result of CNN processor. The result is good, and SNR=27.5dB. Though we discuss image restoration in this paper, the CNN processor is useful for other CNN templates, such as segmentation. The speed of the processor is 30 frames /second in our test.

We also test the case of a 512×512 image with the CNN processor. The image is divided into 16 blocks to process. Fig.6 is original image “LENA”. Fig.7 is the noisy image which is corrupted by Gauss noise, and the variance of noise is 1.0, and SNR=0dB. Fig.8 is result of median filter. Fig.9 is the result of CNN processor, and SNR (signal/noise) is 37.6dB. Obviously the result of CNN is better than that of median filter. The speed of this case is 1 frames /second in our test.



Fig. 6. Lenna



Fig. 7. Noisy image (SNR=0dB)



Fig. 8. Restoration with median filter



Fig. 9. Restoration with CNN

5 Conclusion

In this paper a new approach for MAP image restoration based on CNN is proposed, and a CNN processor for image restoration is realized with FPGA. According to

analysis of MAP image restoration, a new template is proposed for CNN image restoration. The computer simulation result confirms that we can get very good result with this novel CNN. Experimental hardware system based on this approach is setup, and the effectiveness of CNN processor is also confirmed in this system.

References

1. Chua, L.O., Yang, L.: Cellular Neural Network: Theory. IEEE Trans. on CAS **35** (1988) 1257-1272
2. Chua, L.O., Yang, L.: Cellular Neural Network: Applications. IEEE Trans. on CAS **35** (1988) 1273-1290
3. International Journal of Circuit: Theory and Applications. Special Issue on CNN **24** (1996)
4. Derin, H., Elliot, H.: Modeling and Segmentation of Noisy Textured Image Using GRF. IEEE Trans. on PAMI **9** (1987) 39-55
5. Tan, H.L., etc.: A Comparative Cost Function Approach to Edge Detection. IEEE Tran. on SMC **9** (1989) 1337-1349
6. Trussel, H.: The Relationship between Image Restoration by The MAP Method and A ME Method. IEEE Trans. on ASSP **28** (1980) 114-117
7. Zhao, J.Y., Yu, D.: A New Approach for ME Image Restoration Based on CNN. International Journal of Circuit: Theory and Applications **27** (1999) 339-346
8. Altera Corp.: Cyclone EP1C-20 handbook. (2004)
9. Texas Instrument Corp.: TMS320C5416 handbook. (2003)

Using Alpha-Beta Associative Memories to Learn and Recall RGB Images

Cornelio Yáñez-Márquez, María Elena Cruz-Meza,
Flavio Arturo Sánchez-Garfias, and Itzamá López-Yáñez

Centro de Investigación en Computación, Instituto Politécnico Nacional,
Laboratorio de Inteligencia Artificial, Av. Juan de Dios Bátiz s/n,
México, D.F., 07738, México
cyanez@cic.ipn.mx, mcruzam@ipn.mx,
{fgarfias, ilopezb05}@sagitario.cic.ipn.mx

Abstract. In this paper, an algorithm which enables Alpha-Beta associative memories to learn and recall color images is presented. The latter is done even though these memories were originally designed by Yáñez-Márquez [1] to work only with binary patterns. Also, an experimental study on the proposed algorithm is presented, showing the efficiency of the new memories.

1 Introduction

Basic concepts about associative memories were established three decades ago in [2-4], nonetheless here we use the concepts, results and notation introduced in the Yáñez-Márquez's PhD Thesis [1]. An associative memory \mathbf{M} is a system that relates input patterns, and outputs patterns, as follows: $\mathbf{x} \rightarrow \mathbf{M} \rightarrow \mathbf{y}$, whose k -th association is denoted as (x^k, y^k) . Associative memory \mathbf{M} is represented by a matrix whose ij -th component is m_{ij} , which is generated from an *a priori* finite set of known associations, called the fundamental set of associations. If μ is an index, the fundamental set is represented as: $\{(x^\mu, y^\mu) \mid \mu = 1, 2, \dots, p\}$ with p the cardinality of the set. The patterns that form the fundamental set are called fundamental patterns. If it holds that $x^\mu = y^\mu, \forall \mu \in \{1, 2, \dots, p\}$, \mathbf{M} is auto-associative, otherwise it is heteroassociative. A distorted version of a pattern x^k to be recalled will be denoted as \tilde{x}^k . If when feeding a distorted version of x^ϖ with $\varpi = \{1, 2, \dots, p\}$ to an associative memory \mathbf{M} , it happens that the output corresponds exactly to the associated pattern y^ϖ , we say that recall is correct. Among the variety of associative memory models described in the scientific literature, there are two models that, because of their relevance, it is important to emphasize: morphological associative memories which were introduced by Ritter *et al.* [5], and Alpha-Beta associative memories [1].

In this paper we propose an extension of the binary operators Alpha and Beta, foundation for the Alpha-Beta associative memories [1], which allows memorizing and then recalling k -valued input and output patterns. Sufficient conditions for perfect recalling and examples are provided.

2 Alfa-Beta Associative Memories

$\alpha\beta$ associative memories are of two kinds and are able to operate in two different modes. The operator α is useful at the learning phase, and the operator β is the basis for the pattern recall phase. The heart of the mathematical tools used in the Alpha-Beta model, are two binary operators designed specifically for these memories. These operators are defined as follows: First, we define the sets $A=\{0,1\}$ and $B=\{00,01,10\}$, then the operators α and β are defined in tabular form:

$$\alpha : A \times A \rightarrow B$$

x	y	$\alpha(x,y)$
0	0	01
0	1	00
1	0	10
1	1	01

$$\beta : B \times A \rightarrow A$$

x	y	$\beta(x,y)$
00	0	0
00	1	0
01	0	0
01	1	1
10	0	1
10	1	1

The ij -th entry of the matrix $y \oplus x^t$ is: $[y \oplus x^t]_{ij} = \alpha(y_i, x_j)$. If we consider the fundamental set of patterns: $\{(x^\mu, y^\mu) | \mu = 1, 2, \dots, p\}$ where $x^\mu \in A^n$ and $y^\mu \in A^m$, then the ij -th entry of the matrix $y^\mu \oplus (x^\mu)^t$ is: $[y^\mu \oplus (x^\mu)^t]_{ij} = \alpha(y_i^\mu, x_j^\mu)$.

3 The New Model

In this section we show how binary $\alpha\beta$ memories can be used to operate with RGB images. Without lose of generality, let us just analyze the case of the Alpha-Beta autoassociative memories of kind \mathbf{V} .

First, we need to define four operators and prove four propositions derived from them, which will be useful for both phases of the model: learning and recalling. Due to reasons of space, the full proofs of the propositions are omitted here.

Definition 1. Let r be a non-negative integer number. The *minimum binary string operator* $k(r)$ is defined as follows: $k(r)$ has r as input argument and its output is the minimum of the members of the set $\{x | x = \log_2 2^k, \text{ where } k \in \mathbb{Z}^+ \text{ and } 2^k > r\}$.

Proposition 1. If x is an integer number such that $0 \leq x \leq 255$, then $k(x) \leq 8$.

Definition 2. Let r be a non-negative integer number and k a positive integer number, which make the expression $k \geq k(r)$ true. The *k -binary expansion operator* $\mathcal{E}(r, k)$ is defined as follows: $\mathcal{E}(r, k)$ has r and k as input arguments and its output is a binary k -dimensional column vector whose components correspond to the k bits binary expansion of r , with the least significant bit in the lower side.

Proposition 2. If x is an integer number such that $0 \leq x \leq 255$, then it is possible to obtain the 8-binary expansion operator $\mathcal{E}(x,8)$.

Definition 3. Let \mathbf{b} be a binary column vector of dimension n , and k an integer positive number such that $k \geq n$. The k -binary inverse expansion operator $\mathcal{E}_k^{-1}(\mathbf{b})$ as follows: $\mathcal{E}_k^{-1}(\mathbf{b})$ has as input argument a binary k -dimensional column vector, whose first $k-n$ components are 0's, and the last n components coincide one to one with the components of vector \mathbf{b} , having the least significant bit in the lower side. The output of $\mathcal{E}_k^{-1}(\mathbf{b})$ is a non-negative integer number r which is computed through the

$$\text{expression: } \sum_{i=1}^k b_i \cdot 2^{k-i} .$$

Proposition 3. If \mathbf{b} is a binary column vector of dimension $n \leq 8$, it is possible to obtain the 8-binary inverse expansion operator $\mathcal{E}_8^{-1}(\mathbf{b})$, whose output is calculated as:

$$\sum_{i=1}^8 b_i \cdot 2^{8-i} .$$

Definition 4. Let m be a positive integer number and r_m non-negative integer numbers r_1, r_2, \dots, r_m . Additionally, let k be a positive integer number whose value is compatible with Definition 2 for the computation of the m k -binary expansion operators $\mathcal{E}(r_1, k), \mathcal{E}(r_2, k), \dots, \mathcal{E}(r_m, k)$. The ordered concatenation \mathcal{C} of $\mathcal{E}(r_1, k), \mathcal{E}(r_2, k), \dots, \mathcal{E}(r_m, k)$ is defined as a binary column vector of dimension m , made up of the binary strings $\mathcal{E}(r_1, k), \mathcal{E}(r_2, k), \dots, \mathcal{E}(r_m, k)$ put in order from top to bottom. This ordered concatenations is denoted by:

$$\mathcal{C}[\mathcal{E}(r_1,k), \mathcal{E}(r_2,k), \dots, \mathcal{E}(r_m,k)] = \begin{pmatrix} \mathcal{E}(r_1,k) \\ \mathcal{E}(r_2,k) \\ \vdots \\ \mathcal{E}(r_m,k) \end{pmatrix} .$$

Proposition 4. If x, y, z are three integer numbers which make true these inequalities: $0 \leq x \leq 255, 0 \leq y \leq 255$ and $0 \leq z \leq 255$, then the ordered concatenation $\mathcal{C}[\mathcal{E}(x,8), \mathcal{E}(y,8), \dots, \mathcal{E}(z,8)]$ is a binary column vector of 24 bits.

The fundamental set for the new model is made up by p color images in RGB format, where p is a positive integer number. The A set for the new model is formed by RGB triplets, and is denoted as: $A = \{ x \mid x \text{ is an RGB triplet} \}$.

If \mathbf{I}^μ represents the μ -th image, the fundamental set is represented as: $\{(\mathbf{I}^\mu, \mathbf{I}^\mu) \mid \mu = 1, 2, \dots, p\}$.

Let us call $n=hv$ to the total number of pixels in each \mathbf{I}^μ , where h is the number of horizontal pixels and v is the number of vertical pixels. That is, \mathbf{I}^μ is made up by n RGB pixels. Also, $\mathbf{I}^\mu \in A^n, \forall \mu \in \{1, 2, \dots, p\}$ and $I_i^\mu \in A, \forall i \in \{1, 2, \dots, n\}$.

According to the alter paragraph, for each $\mu \in \{1, 2, \dots, p\}$ and each $i = 1, 2, \dots, n$, component I_i^μ has three parts, corresponding to the R, G, and B of that RGB triplet. These three parts will be denoted as R_i^μ , G_i^μ , and B_i^μ , respectively.

Just as in the original model of Alpha-Beta associative memories, the B set is $B = \{00, 01, 10\}$.

LEARNING PHASE

For each $\mu=1, 2, \dots, p$ and each $i=1, 2, \dots, n$, obtain $\mathcal{E}(R_i^\mu, 8)$, $\mathcal{E}(G_i^\mu, 8)$, $\mathcal{E}(B_i^\mu, 8)$, and $\mathcal{C}[\mathcal{E}(R_i^\mu, 8), \mathcal{E}(G_i^\mu, 8), \mathcal{E}(B_i^\mu, 8)]$.

For each $\mu=1, 2, \dots, p$ obtain $\mathbf{x}^\mu = \mathcal{C}[C_1^\mu, C_2^\mu, \dots, C_m^\mu]$.

For each $\mu=1, 2, \dots, p$ and each association $(\mathbf{x}^\mu, \mathbf{x}^\mu)$ build $\left[\mathbf{x}^\mu \otimes (\mathbf{x}^\mu)^\top \right]_{m \times n}$.

Apply the binary operator \vee to the former matrices in order to obtain

$$\mathbf{V} = \bigvee_{\mu=1}^p \left[\mathbf{x}^\mu \otimes (\mathbf{x}^\mu)^\top \right]_{m \times n}.$$

RECALLING PHASE

CASE 1: Recall of a fundamental pattern $\mathbf{I}^\omega \in A^n$ with $\omega \in \{1, 2, \dots, p\}$.

For each $i=1, 2, \dots, n$ obtain $\mathcal{E}(R_i^\omega, 8)$, $\mathcal{E}(G_i^\omega, 8)$, $\mathcal{E}(B_i^\omega, 8)$, and $C_i^\omega = \mathcal{C}[\mathcal{E}(R_i^\omega, 8), \mathcal{E}(G_i^\omega, 8), \mathcal{E}(B_i^\omega, 8)]$.

Obtain $\mathbf{x}^\omega = \mathcal{C}[C_1^\omega, C_2^\omega, \dots, C_n^\omega]$.

Do operation $\mathbf{V} \Delta_\beta \mathbf{x}^\omega$.

The result is a binary column vector of dimension $m = 24n$, with its i -th component given by:

$$\begin{aligned} (\mathbf{V} \Delta_\beta \mathbf{x}^\omega)_i &= \bigwedge_{j=1}^m \beta(v_{ij}, x_j^\omega), \\ (\mathbf{V} \Delta_\beta \mathbf{x}^\omega)_i &= \bigwedge_{j=1}^m \beta \left\{ \left[\bigvee_{\mu=1}^p \alpha(x_i^\mu, x_j^\mu) \right], x_j^\omega \right\}. \end{aligned}$$

For each $i=1, 2, \dots, n$:

Form a binary column vector \mathbf{b} of dimension 8 such that:

$$b_j = (\mathbf{V} \Delta_\beta \mathbf{x}^\omega)_{24(i-1)+j}, \text{ for } 0 < j \leq 8.$$

Calculate $R_i^\omega = \sum_{j=1}^8 b_j \cdot 2^{8-j}$.

Form a binary column vector \mathbf{b} of dimension 8 such that:

$$b_j = (\mathbf{V} \Delta_\beta \mathbf{x}^\omega)_{24(i-1)+j-8}, \text{ for } 8 < j \leq 16.$$

Calculate $G_i^\omega = \sum_{j=1}^8 b_j \cdot 2^{8-j}$.

Form a binary column vector \mathbf{b} of dimension 8 such that:

$$b_j = \left(\mathbf{V} \Delta_\beta \mathbf{x}^\omega \right)_{24(i-1)+j-16}, \text{ for } 16 < j \leq 24.$$

Calculate $B_i^\omega = \sum_{j=1}^8 b_j \cdot 2^{8-j}$.

Create the RGB triplet and assign it to the i -th component of I_i^μ .

The recalled pattern is the fundamental pattern $I^\mu \in A^n$.

CASE 2: Recall of a pattern $\tilde{\mathbf{I}}$ which is a version of some fundamental pattern $I^\omega \in A^n$, $\omega \in \{1, 2, \dots, p\}$, altered with additive, subtractive or mixed noise. The steps of the algorithm are similar to those of the Case 1, using $\tilde{\mathbf{I}}$ instead of I^μ .

4 Experiments with RGB Images

In this section the new Alpha-Beta associative memories are tested with ten color images. The images, shown in the Figure 1, are 100 by 75 pixels and 24 bits of depth per pixel, RGB. Only the new Alpha Beta autoassociative memories type \mathbf{V} were tested.

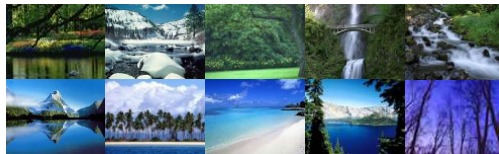


Fig. 1. Images of the ten objects used to test the new $\alpha\beta$ associative memories

LEARNING PHASE

Each one of all the ten images was presented to the new Alpha Beta autoassociative memory type \mathbf{V} , following the learning phase described in the latter section.

RECALLING PHASE

All the ten patterns in the fundamental set were perfectly recalled.

To perform the experiments with altered versions of the fundamental patterns, the images in the fundamental set were corrupted with additive noise. Four groups of images were generated: The first one with very weak additive noise (1%), the second one with weak additive noise (5%), the third one with medium additive noise (20%), and the fourth one with severe additive noise (50%), a huge amount of noise. Forty corrupted images were obtained changing randomly some pixel values. In all the cases the desired image was correctly recalled. Notice how despite the level of noise introduced in the fourth column is too severe (in any system, 50% of noise is a huge amount), all the images are still correctly recalled!

5 Conclusion and Future Work

We have shown how it is possible to use binary Alpha-Beta associative memories, to efficiently recall patterns made with color images, in particular using the RGB format. This is possible because an RGB image can be decomposed into binary patterns. It is worth to mention that the proposed technique can be adapted to any kind of binary associative memories while their input patterns can be obtained from the binary expansions of the original patterns. Currently, we are investigating how to use the proposed approach in the presence of mixed noise and other variants. We are also working toward the proposal of new associative memories based on others mathematical results.

Acknowledgements

The authors would like to thank the Instituto Politécnico Nacional (Secretaría Académica, COFAA, SIP, CIC and ESCOM), the CONACyT, and SNI for their economical support to develop this work.

References

1. Yáñez-Márquez, C.: Associative Memories Based on Order Relations and Binary Operators (In Spanish). PhD Thesis. Center for Computing Research, México (2002)
2. Kohonen, T.: Correlation Matrix Memories. *IEEE Transactions on Computers* **21** (4) (1972) 353-359
3. Kohonen, T.: *Self-Organization and Associative Memory*. Springer-Verlag, Berlin Heidelberg New York (1989)
4. Hassoun, M. H.: *Associative Neural Memories*. Oxford University Press, New York (1993)
5. Ritter, G.X., Sussner, P., Diaz-de-Leon, J.L.: Morphological Associative Memories. *IEEE Transactions on Neural Networks* **9** (1998) 281-293

A Method for Enlargement of Digital Images Based on Neural Network

Zhao JiuFen¹, Zhang Xurong², and Li Qingzhen¹

¹ The Second Artillery Engineering Institute
710025 Xi'an, China
zhaojf@tom.com

² Beijing Institute of Technology
100081 Beijing, China

Abstract. The enlargement of the digital image implies the improvement of the image resolution, where the high frequency components lost in sampling must be estimated. In this paper, an image enlargement method using a high resolution neural network is proposed, corresponding to the region with rapid change (high local variance) and the region requiring a smooth interpolation (low local variance). It is shown that the high resolution NN has high potential ability.

1 Introduction

The enlargement of the digital image implies the improvement of the image resolution, where the high frequency components lost in sampling must be estimated [2-5]. As a means to cope with the requirement, the authors have proposed a method using the high-resolution neural network [3]. The method is an extension and generalization of the method proposed by Greenspan [2]. The high resolution NN is a three-layered NN, and is trained by error back-propagation. The high resolution obtained by training is not much affected by the kind of training image or its resolution, and the performance is improved more remarkably than by Greenspan's method.

The method [3] has the following two problems:

(1) As already pointed out in Ref. 3, when the image is enlarged using the high-resolution NN, artifacts may be produced on the enlarged image, if the magnification ratio is 4 or more.

(2) Although a high-resolution NN can be composed with a remarkably better performance than by Greenspan's method, independently of the training image, the performance may vary by approximately 10%, in terms of the mean square error (MSE), due to the different properties of the training image and the image to be enlarged.

The artifacts on the enlarged image in problem (1) are especially remarkable in the smoothly changing region of the image. This is due to the fact that the high-resolution NN is trained to improve the resolution in the region with a rapid signal change as in the case of the edge signal, and produces unnecessary changing components in the region with the smooth signal. In order to remedy this point, there is an approach where two NN divide the processing for the region of the image with smooth changes and the region with rapid changes.

As seen from the above discussion, an image contains the region where the resolution should be improved, and the region where the signal should simply be interpolated. It is expected that the high-resolution NN will be constructed with different properties, depending on the ratio of those regions in the image. This is the reason for problem (2). It is again expected that this problem can be solved by preparing separate NN for those requirements (i.e., the requirement to improve the resolution and the requirement for simple interpolation).

With such a view as the background, this paper proposes the use of high-resolution multi-neural networks (MNN) for enlarging digital images (high resolution). In the proposed method, the local variance is used as the criterion for discriminating the regions for resolution improvement and the regions for smooth interpolation. It is logical to expect that the signal region with a high local variance will require resolution improvement, and the signal region with a low local variance will require smooth interpolation.

From such a viewpoint, the high-resolution MNN based on the local variance is composed of two NN, and the result of enlargement (or interpolation) is given as the weighted sum of the two NN outputs. The weights are determined as functions of the local variance. By learning of the high-resolution MNN, the coupling weights in the two NN, as well as the weights for the two outputs are determined in parallel. The two constructed NN are defined as the NN for low local variance and the NN for high local variance, respectively.

2 High Resolution Multi-neural Networks Based on Local Variance

It is expected that the problems of the high-resolution NN pointed out in Fig. 1 will be remedied by constructing and operating separately the network to interpolate the region near the edge, and the network to interpolate the region with smooth signal changes. This section newly proposes a high-resolution MNN based on the local variance along the above idea, and presents its training procedure.

The signal region with rapid changes that requires the resolution improvement and the signal region that requires the smooth interpolation can be discriminated using the local variance. From such a viewpoint, the proposed high resolution NN is constructed using the NN for high variance (NN_H) and the NN for low variance (NN_L), and the result of enlargement (interpolation) is defined by the weighted sum of the two NN outputs. The weights are determined as functions of the local variance. Then, the two NN satisfying the above requirements can be derived by training.

Formulating the above idea, the output O of the high resolution MNN is given as follows, using the outputs O_H and O_L of NN_H and NN_L , respectively:

$$O = f(v)O_H + g(v)O_L \quad (1)$$

In the above, $f(v)$ and $g(v)$ are functions of the local variance v . They are defined as follows:

$$f(v) = \begin{cases} 1 & v_H \leq v \\ \frac{v - v_L}{v_H - v_L} & v_L < v < v_H \\ 0 & v \leq v_L \end{cases} \tag{2}$$

$$g(v) = \begin{cases} 0 & v_H \leq v \\ \frac{v - v_L}{v_H - v_L} & v_L < v < v_H \\ 1 & v \leq v_L \end{cases} \tag{3}$$

Thus, the high resolution MNN proposed in this paper has the structure shown in Fig. 1. v_H and v_L are the thresholds for the local variance. The functions $f(v)$ and $g(v)$ are represented as in Fig. 2. Those two threshold values are also adjusted by training.

As in the case of the high resolution NN in [3], MNN is trained by minimizing the mean square error between the supervisor’s signal L_{n-1} and the MNN output O over all training signals. More precisely, letting the coupling weights for NN_H and NN_L in the high resolution MNN be ω_H and ω_L , respectively, the following optimization problem is solved:

Minimize:

$$J(\omega_H, \omega_L, v_H, v_L) = E[(L_{n-1} - O)^2] \tag{4}$$

Let the instantaneous error J be

$$J = 1/2(L_{n-1} - O)^2 \tag{5}$$

The coefficients are updated as follows:

$$\begin{aligned} v_* &= v_* - \alpha \frac{\partial J}{\partial v_*} \\ &= v_* + \alpha(L_{n-1} - O)(O_H \frac{\partial f(v)}{\partial v_*} + O_L \frac{\partial g(v)}{\partial v_*}) \end{aligned} \tag{6}$$

$$\begin{aligned} \omega_H &= \omega_H - \beta \frac{\partial J}{\partial \omega_H} \\ &= \omega_H + \beta(L_{n-1} - O)f(v) \frac{\partial O_H}{\partial \omega_H} \end{aligned} \tag{7}$$

$$\begin{aligned} \omega_L &= \omega_L - \beta \frac{\partial J}{\partial \omega_L} \\ &= \omega_L + \beta(L_{n-1} - O)g(v) \frac{\partial O_L}{\partial \omega_L} \end{aligned} \tag{8}$$

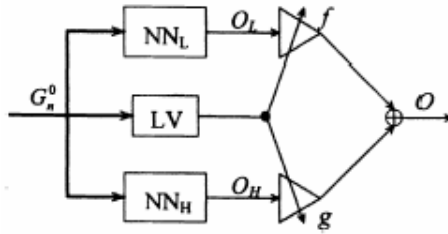


Fig. 1. The structure of MNN

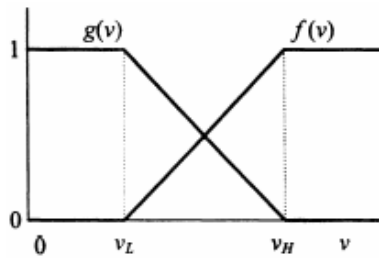


Fig. 2. Nonlinear functions $f(v)$, $g(v)$

By the above training of MNN based on the local variance, the threshold value v_H, v_L as well as the weight coefficients ω_H, ω_L of the networks are optimized.

3 Simulation

This section examines the characteristics of the high resolution MNN, and also examines whether or not the two problems in the high-resolution NN are dissolved, through an application example of the high resolution MNN to the natural image signal. As the numerical indices for evaluation, the MSE between the ideal enlarged image and the resultant image of the enlargement processing, the MSE in the flat region (the region where the local variance in 3×3 region of the ideal enlarged image is less than 200) of the image (called background MSE, BMSE), and the MSE in the detailed region (the region where the local variance in 3×3 region of the ideal enlarged image is 200 or more) of the image (called detailed region MSE, DMSE) are used.

As the test images in this study, the original images of Lena and Lighthouse are contracted to one-half by down sampling after Gaussian filtering. The contracted image is used as the input to the high-resolution NN and the high resolution MNN. The corresponding original image is used as the supervisor signal in the training. Five networks are constructed [NN-Le, MNN-Le (5×5), (7×7), NN-Lh, MNN-Lh (5×5)]. As the region in which the local variance is calculated, two regions- 5×5 and 7×7 are set in the input image in the input layer, especially for Lena. The one-half-contracted

images (Lena, Boat, Lighthouse) are input to the five networks, and the resolution is improved (enlarged). Table 1 shows the numerical evaluations between the enlarged results and the ideal images.

In this study, it is assumed that the two NN are switched based on the local variance (soft switching). The size of the region for which the local variance should be calculated is discussed in the following. In the input image to the high-resolution MNN (the image obtained by inserting 0 values in the image to be enlarged), 5×5 and 7×7 regions are specified as the regions for calculating the local variance, and two MNN-Le are constructed. It should be noted that 5×5 is the minimum region for which the local variance can be calculated.

As seen from Table 1, the result of MNN-Le (5×5) is better than that of MNN-Le (7×7). Since the two NN in MNN are the networks for satisfying, respectively, the contradictory requirements, it may seem better that the networks should have different properties. When the local region is widened, however, there may arise regions which can not be decided as the edge or the flat region. This will make discrimination of the two NN ambiguous, and degrades the performance. From such a viewpoint, the 5×5 region seems adequate as the region to calculate the local variance in the high resolution MNN.

Table 1. Enlargement results (MSE/BMSE/DMSE)

Image	Lenna	Boat	Lighthouse
Greenspan	106.4 (35/361.2)	76.5 (30.5/256.4)	254.5 (40.0/669.4)
NN-Le	75.9 (22.4/273.7)	52.9 (20.8/178.5)	217.8 (31.6/577.8)
NN-Lh	83.0 (27.3/276.8)	58.6 (25.6/187.9)	213.5 (34.6/559.5)
MNN-Le (5×5)	70.3 (20.5/248.2)	53.2 (18.6/188.9)	215.3 (30.2/573.4)
MNN-Le (7×7)	73.1 (21.3/258.3)	53.2 (19.4/184.9)	220.5 (31.6/585.9)
MNN-Lh	76.4 (24.2/263.2)	56.1 (23.4/184.0)	205.3 (21.7/541.1)

4 Conclusions

This paper has pointed out the problems in digital image enlargement by the high-resolution NN, which was previously proposed by the authors. In order to solve those problems, a high-resolution MNN is proposed, which is composed of two NN. It is shown that the high resolution MNN has high potential ability and can dissolve the problems in the high-resolution NN.

In this paper, the processing is divided between the two NN, based on the local variance information. It is of course conceivable that the number of NN is increased to 3, 4, . . . So that the local variance is considered in more detail. As an example, the processing is divided among three NN based on the local variance information. The result for Lena, however, shows only an improvement of MSE from 70.3 to 70.1. The situation is similar for Lighthouse and other images. Thus, it is verified experimentally that the division of processing is sufficient if two NN are used. In order to improve further the enlargement performance by MNN, new local information should be introduced, which is left for future study.

References

1. Burt, P.J., Adelson, E.H.: The Laplacian Pyramid as a Compact Image Code. *IEEE Trans Commun COM* **31** (2003) 532-540.
2. Greenspan, H., Anderson, C.H.: Image Enhancement by Non-Linear Extrapolation in Frequency Space. *SPIE Image and Video Processing-II* **2182** (1994) 2-13.
3. Sekiwa, D., Taguchi, A., Murata, Y.: Enlargement of Digital Image by Laplacian Pyramid using Neural Network. *Trans IEICE J* **80-A** (2002) 1499-1508.
4. Shinbori, E., Takagi, M.: A High-Quality Image Enlargement using Gerchberg Papoulis Iteration by DCT. *Trans IEICE J* **76-D-II** (2003) 1932-2002.
5. Tanaka, A., Imai, H., Miyakoshi, M., Date, J.: Enlargement of Digital Image by Multi-resolution Analysis. *Trans IEICE J* **79-D-II** (2001) 819-825.

Neural Network Based Correction Scheme for Image Interpolation

Liyong Ma, Yi Shen, and Jiachen Ma

School of Information Science and Engineering,
Harbin Institute of Technology at Weihai, Weihai 264209, P.R. China
hitmaly@yahoo.com.cn, shen@hit.edu.cn, hitmjc@sohu.com

Abstract. A generalized regression neural network based error correction scheme for linear image interpolation approach is proposed. A middle image with the same size of source image is obtained by interpolating a down-sampled image from the source image. Then neural network is established with employing the interpolation error between the source image and the middle image. Finally interpolation correction is applied to the linear interpolation result of source image using neural network estimation to obtain more accuracy result image. Experimental results of the proposed approach demonstrate the effectiveness of the scheme.

1 Introduction

Image interpolation has a wide range of applications in remote sense, medical diagnoses, multimedia communication and other image process fields. The well-known approaches to image interpolation are linear interpolation and cubic interpolation [1]. However these methods blur images particularly in edge regions [2]. Other algorithms have been extensively studied to solve the problem of blurring, such as adaptive interpolation methods [3], [4]. Some other learning based interpolation approaches have also been proposed, such as support vector machines based image interpolation approach that was suggested in [5].

Most of these interpolation approaches did not use interpolation error correction scheme. However error correction scheme is usually efficient to improve interpolation accuracy of result images. An error correction scheme employing support vector machines is provided in [6]. And it is shown that error correction scheme is benefit to improve the quality of result images. In this paper a more efficient error correction scheme using neural network for linear interpolation approach is proposed. And experimental results show that the quality of the result images produced with the proposed scheme is better than the well-known interpolation approaches.

2 Generalized Regression Neural Network Estimation

Line regression and logistic regression are the standard statistical approaches to establish mathematical relationship between the independent variables and the

final decision. They often require prior knowledge about the relationship patterns. Neural network is efficient for patterns detection where the relationships can not be easily expressed in a mathematical form. Neurons in neural network are interconnected to receive signals in input layer and produce output in output layer. Firstly the network structure is decided. Then a learning algorithm is employed for the network training with sample vectors. In this training process network parameters are modified to minimum the error between the actual output and the desired output. Finally the trained neural network can be used to estimate the output for any vector from the input space.

Radial basis function (RBF) neural network is an important neural network architecture with many important applications. The RBF neural network has the universal approximation ability, so it can be used for the interpolation problem [7]. A Gaussian radial basis function is highly nonlinear and powerful for learning complex input-output mapping. A typical RBF neural network includes an input layer, a single hidden layer that is called radial basis layer for non-linear processing, and a output linear layer [7]. The output of RBF neural network is

$$f(x) = \sum_{i=1}^n w_i \phi_i(\|\mathbf{x} - \mathbf{c}_i\|), \tag{1}$$

where \mathbf{x} is input vector, $\phi_i(\cdot)$ denotes the processing function of the i -th node in the hidden layer, $\|\cdot\|$ denotes the Euclidean norm, w_i are weights between i -th node in the hidden layer and output node, n is the total number of neurons in the hidden layer, and \mathbf{c}_i are the RBF centers in the input vector space. Euclidean distances that are the distances between the input vector and the input weight matrix for each neuron in the hidden layer are calculated, and a nonlinear function of the distance is obtained in the hidden layer outputs. The output of the neural network is a weighted sum of the hidden layer outputs. Gaussian function is often employed as the radial basis process function.

Constant spread S is employed in the radial basis layer to set each bias of this layer to $0.8326/S$. The spread S can determine the area width for each neuron responds in the input space. So the spread S needs to be selected correctly to overlap the regions of the input space.

Generalized regression neural network (GRNN) which is similar to the RBF is often used for function approximation [8]. The number of the neurons in the radial basis layer of GRNN is just the number of input target vectors. And the radial basis layer of GRNN is similar to that of RBF. GRNN has a special linear output layer that is different from RBF. The neurons number in the output layer of GRNN is also equal to the number of input target vectors. And in this layer the dot product of the weights and the input vector is calculated and normalized at first, then the result is employed as input of linear neurons.

It has been shown that GRNN can approximate a continuous function to an arbitrary accuracy when a sufficient number of hidden neurons are given. GRNN can be established quickly for no training is needed. In this paper generalized regression neural network is employed to estimate interpolation error distribution.

3 Previous Work of Interpolation

3.1 Linear and Cubic Interpolation

Let x denote the coordinate value to be interpolated. Assume that x_{k-1} , x_k , x_{k+1} and x_{k+2} are the nearest available neighbors of x , where $x_k \leq x < x_{k+1}$. Let $f(x_{k-1})$, $f(x_k)$, $f(x_{k+1})$ and $f(x_{k+2})$ denote the available gray value of x_{k-1} , x_k , x_{k+1} and x_{k+2} , respectively. Then the distance between x and neighbors can be defined as

$$s = x - x_k, \quad 1 - s = x_{k+1} - x \quad (0 \leq s \leq 1). \tag{2}$$

We have one-dimensional linear interpolation of x

$$\hat{f}(x) = (1 - s)f(x_k) + sf(x_{k+1}). \tag{3}$$

Similarly, we have one-dimensional cubic interpolation of x

$$\begin{aligned} \hat{f}(x) = [& f(x_{k-1})((3 + s)^3 - 4(2 + s)^3 + 6(1 + s)^3 - 4s^3) \\ & + f(x_k)((2 + s)^3 - 4(1 + s)^3 + 6s^3) \\ & + f(x_{k+1})((1 + s)^3 - 4s^3) + f(x_{k+2})s^3] / 6. \end{aligned} \tag{4}$$

For an image with two dimensions, we can apply one-dimensional linear interpolation equation (3) or cubic interpolation equation (4) to the image along the rows firstly, then the interpolation is applied to the image along the columns. And this two-dimensional interpolation algorithm is called bilinear interpolation or bicubic interpolation [1] [3].

3.2 Warped Distance Based Adaptive Interpolation

Recently an adaptive linear space variant approach based on the evaluation of warped distance was proposed in [3]. To sharpen edge regions the concept of warped distance was introduced to evaluate image local activities properties. To adjust the distance s in (3) and (4) an asymmetry operator was denoted by

$$A = \frac{|f(x_{k+1}) - f(x_{k-1})| - |f(x_{k+2}) - f(x_k)|}{L - 1}. \tag{5}$$

For 8-bit gray images, $L=256$ and $A \in [-1, 1]$. Adaptive interpolation expressions of (3) and (4) can be modified by replacing distance s with warped distance. Then we have adaptive bilinear interpolation function

$$\hat{f}(x) = (1 - s)cf(x_k) + sdf(x_{k+1}), \tag{6}$$

and adaptive bicubic interpolation function

$$\begin{aligned} \hat{f}(x) = [& cf(x_{k-1})((3 + s)^3 - 4(2 + s)^3 \\ & + 6(1 + s)^3 - 4s^3) \\ & + cf(x_k)((2 + s)^3 - 4(1 + s)^3 + 6s^3) \\ & + df(x_{k+1})((1 + s)^3 - 4s^3) + df(x_{k+2})s^3] / 6, \end{aligned} \tag{7}$$

where $c = 1 - mA$, $d = 1 + mA$, and m denotes a constant.

4 Proposed Interpolation Correction Scheme

The critical task of image interpolation is to reduce estimation error of the pixel gray value to be interpolated, especially in edges and detail regions where the interpolation error is usually greater than in other regions. Usually the edges and detail regions in the interpolated result images of the source image is similar to that of the down-sampled images. So the interpolation error of the source images can be estimated by using the interpolated result image of the down-sampled image. GRNN can be employed to estimate the interpolation error distribution.

Suppose we need calculate the $2\times$ interpolation result image \mathbf{T} with size $4a \times 4b$ from source image \mathbf{P} with size $2a \times 2b$, our proposed error correction scheme can be described as follows:

Step 1. The source image \mathbf{P} is down-sampled to get image \mathbf{Q} with size $a \times b$. And \mathbf{Q} is interpolated to establish middle image \mathbf{Q}_1 with size $2a \times 2b$ by employing linear interpolation approach.

Step 2. An interpolation residual image \mathbf{R} can be calculated as

$$\mathbf{R} = \mathbf{P} - \mathbf{Q}_1. \quad (8)$$

Some pixels in \mathbf{Q}_1 those have corresponding residue gray values in \mathbf{R} are obtained with interpolation calculation, that is say they are not in the down-sampled image \mathbf{Q} . Their corresponding residue gray values are collected as the sample vectors of GRNN. The patterns in the input vector of these samples are their relative coordinates, and the output is the corresponding residual image gray values. After the GRNN is established, it can be employed to estimate interpolation error.

Step 3. Linear interpolation result image \mathbf{T}_1 with size $4a \times 4b$ is calculated by applying the linear interpolation algorithm to the source image \mathbf{P} .

Step 4. The interpolation result image \mathbf{T}_1 is corrected with GRNN error estimation. The input vector of GRNN is the relative coordinates of the pixels those are in image \mathbf{T}_1 and not in image \mathbf{P} , and output is the error estimation of gray values. The final corrected result image \mathbf{T} can be calculated as

$$\mathbf{T} = \mathbf{T}_1 + \mathbf{R}/v, \quad (9)$$

where v is the correction parameter to control correction degree.

In this correction scheme for linear interpolation, spread S and v can be selected by optimal search. In the search process image \mathbf{Q} can be regarded as source image, then different S and v can be applied to the proposed scheme for interpolation to decide the optimal parameter values for S and v . In this procedure all the interpolation results images are known, so we can use these known result images to compare interpolation results and judge the optimal parameters. After these parameters are selected, these optimal parameters can be used in the proposed scheme to perform interpolation and correction for source image \mathbf{P} to get final unknown interpolation result \mathbf{T} .

This scheme can be easily generalized to those interpolations where the magnification times is integer.

5 Experimental Results

Some standard images that have been widely used in other literature were tested in our experiments. We obtained similar results when these standard images are tested with linear, cubic, warped distance adaptive interpolation algorithms and our proposed scheme. Experiments are performed in Matlab. Results with different interpolation approaches are employed to the test image peppers and airplane are compared in Table 1 and Table 2 respectively. Where the double row and column expanded is performed to the source images with different approaches. All the parameters m for warped distance based adaptive interpolation approaches and S, v for proposed correction scheme are selected with optimal search procedure. In Table 1, $m = 0.6$ for adaptive linear interpolation algorithm, $m = 0.7$ for adaptive cubic interpolation algorithm, where $S = 2.5$ and $v = 2$ for proposed scheme. In Table 2, $m = 0.1$ for both adaptive interpolation algorithms, where $S = 2.0$ and $v = 3$ for proposed scheme. It is shown from the tables that the proposed correction scheme reduces mean square error (MSE) and improves peak signal to noise ratio (PSNR) of result images. It is also shown that these experimental results are superior than the support vector machines based error correction scheme in [6].

Table 1. Different Interpolation Approaches Applied to Image Peppers

Algorithm	MSE	PSNR
Linear	115.96	27.488
Adaptive Linear	112.45	27.621
Cubic	120.58	27.318
Adaptive Cubic	116.98	27.450
Proposed	103.65	27.975

Table 2. Different Interpolation Approaches Applied to Image Airplane

Algorithm	MSE	PSNR
Linear	49.33	31.199
Adaptive Linear	49.30	31.202
Cubic	50.85	31.068
Adaptive Cubic	50.87	31.066
Proposed	43.57	31.739

6 Conclusion

A novel linear interpolation error correction scheme based on neural network has been proposed. This scheme can be employed to interpolation applications, such as enlarge image where the magnification times is integer. We can also notice

that applying the error correction scheme to the linear interpolation approach, we can get better result images than employing more complex approaches, such as cubic approach and warped distance based adaptive algorithms. It shows that the interpolation compensation scheme can be applied to simple interpolation algorithm to get better result images than some complex interpolation algorithms.

References

1. Russ, J.: The Image Processing Handbook. 4th edn. CRC Press, Boca Raton (2002)
2. Thevenaz, P., Blu, T., Unser, M.: Interpolation Revisited. *IEEE Trans. Medical Imaging*. **19** (2000) 739-758
3. Ramponi, G.: Warped Distance for Space-variant Linear Image Interpolation. *IEEE Trans. Image Processing*. **8** (1999) 629-639
4. Ma, L.Y., Ma, J.C., Shen, Y.: Local Activity Levels Guided Adaptive Scan Conversion Algorithm. In: Proceedings of the 27th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. (2005) 6718-6720
5. Ma, L.Y., Shen, Y., Ma, J.C.: Local Spatial Properties Based Image Interpolation Scheme Using SVMs. *Journal of Systems Engineering and Electronics*. (In Press)
6. Ma, L.Y., Ma, J.C., Shen, Y.: Support Vector Machines Based Image Interpolation Correction Scheme. In: Wang, G., Peters, J., Skowron, A., Yao, Y. (eds.): *Rough Sets and Knowledge Technology. Lecture Notes in Artificial Intelligence*. Springer-Verlag, Berlin Heidelberg New York **4062** (2006) 679-684
7. Gupta, M., Jin, L., Homma, N.: *Static and Dynamic Neural Networks*. (John Wiley, New Jersey 2003)
8. Specht, D.F.: A General Regression Neural Network. *IEEE Trans. Neural Networks*. **2** (1991) 568-576

Edge Enhancement Post-processing Using Hopfield Neural Net

Zhaoyu Pian, Liqun Gao, Kun Wang, Li Guo, and Jianhua Wu

College of Information Science & Engineering, Northeastern University
110004, Shenyang, China

{Zhaoyu.PIAN, bgfhdragon}@126.com

<http://www.springer.com/lncs>

Abstract. A novel edge enhancement based on Hopfield neural net is presented in this paper, which is a post-processing complement for a pre-existing edge detector. This term is added to the output of the edge detector. Firstly, the energy function which is used to find the final stable edges is provided in the Hopfield neural net, and then, based on the window iteration, it improves the performance of the edge detector by recovering missing edges and eliminating false edges. In experiments conducted on various images, we demonstrate the performance of the algorithm on them.

1 Introduction

Edges are of primary importance in visual quality perception[1], because object boundaries are crucial information to the human visual system (HVS). In many image analysis and computer vision applications edges are used as primary features[2], such as in case of image enhancement, object classification, object detection and tracking. Therefore, the edge enhancement is an essential step for obtaining the excellent edges.

Many edge enhancement methods have been researched in last decades, apart from the dedicated algorithms, it is desirable to incorporate appropriate contrast enhancement on edges in various image processing tasks, such as image reconstruction or demosaicing, post-processing for decompressed images/video. Edge enhancement can be evaluated via estimating local contrast[3], kurtosis[4] and width/amplitude of lines and edges[5], based on subband decomposition. Although most edge detectors do a reasonable job of locating better edges, most of enhancement methods for edges was treated as a positive factor towards visual quality, and excessive edge sharpness and the influence of surrounding pixels were not considered.

In this paper, we propose a novel enhancement algorithm using Hopfield Neural Net (HNN) for edge enhancement, which are only added to a pre-existing edge detector as a post-processing complement for an exquisite edge. Firstly, an energy function corresponding to HNN is to be revealed, whose minimum should correspond to the possible stable situations in which the edges are a pixel width

and continuous. And based on it, the structure of a HNN is designed. This efficiently captures the topological and structural properties of the edge data obtained from pre-existing edge detector. Finally, stable structures for edges are obtained by establishing proper interconnections among neurons and updating the neural computation toward the right solution. To verify the proposed method, we compare it with the common methods in the term of images which have no noise and noise respectively. The experimental results show that the effect of the proposed method is superior to them.

The remainder of this paper is organized as follows: Section 2 contains a review of HNN technique. Based on the analysis of such technique, the motivation and contribution of the proposed method is explained. In section 3, the customized HNN process is described where the energy function to be minimized is derived. Section 4 exhibits and discusses the experimental results. Lastly, the conclusions of this paper are provided in section 5.

2 The Review of HNN

The HNN paradigm initially proposed by Hopfield has been widely used for solving optimization problems [6]. This implies fixing two characteristics: Its activation dynamics and an associated energy function which decreases as the network evolves.

The HNN is a recurrent network containing feedback paths from the outputs of the nodes back into their inputs so that the response of such a network is dynamic. This means that after applying a new input, the output is calculated and fed back to modify the input. The output is then recalculated, and the process is repeated again and again. Successive iterations produce smaller and smaller output changes, until eventually the outputs become constant, i.e., at this moment the network achieves an acceptable stability. The connection weights between the nodes in the network may be considered to form a matrix T . The classical approach shown that a recurrent network is stable if the matrix is symmetrical with zeros on its diagonal [7], that is, if $T_{ij} = T_{ji}$ for all i and j and for $T_{ii} = 0$ all i neurons.

There are two kinds of Hopfield networks: 1) Analog, in which the states of the neurons are allowed to vary continuously in an interval, such as $[-1, 1]$; and 2) discrete, in which these states are restricted to the binary values -1 and $+1$. The drawback of these binary networks is that they oscillate between different binary states and settle down into one of many locally stable states. Hopfield has shown that analog networks perform better since they have the ability to smooth the surface of the energy function which prevents the system from being stuck in minor local minima. For analog Hopfield networks, the total input into a node is converted into an output value by a sigmoid monotonic activation function instead of the thresholding operation for discrete Hopfield networks. The dynamic of a node is defined by

$$\frac{du_i}{dt} = -\frac{u_i}{R_i} + \sum_{j \neq i} T_{ij}V_j + \Theta_i \tag{1}$$

Where the V_j value represents the output of the j th node; T_{ij} is the weight of the connection between nodes; R_i is a time constant that can be set to one for simplicity. The quantity describing the state of the network, called energy function, is defined as follows:

$$E = -\frac{1}{2} \sum_i \sum_{j \neq i} T_{ij}V_iV_j - \sum_i \Theta_iV_i + \beta \sum_i \int_0^{V_i} g^{-1}(V)dV \tag{2}$$

Where $g(V)$ is the sigmoid activation function, β is a scale coefficient. The continuous Hopfield model described by the system of nonlinear first-order differential equation (1) represents a trajectory in phase space, which seeks out the minima of the energy function in (2).

3 Edge Enhancement by HNN

Our purpose is to enhance edges to obtain an edge map with one pixel width based on existent edges by the pre-existing edge detector, remove noise and recover missing edges. Several factors such as what kind of edge structures are stable, and how to update the edge measurement to approach the final result should be considered. For existent edge pixels, we define four edge orientations, such as fig.1 shown, which are the expected outputs of HNN. And the number of pixel in every orientation is counted for a 5×5 window. The maximum decide central pixel belong to which orientation. If the number is equal for all four orientations, the orientation is decided by previous pixel. In our research, the end pixels and abrupt points of edges are ignored. The key problem of HNN is to find a configuration for minimizing an energy function which describes the stable situations of edges. The neurons inside the window are fully connected to each other. The correlation between the central element and the element outside the window can be ignored without effecting the final result.

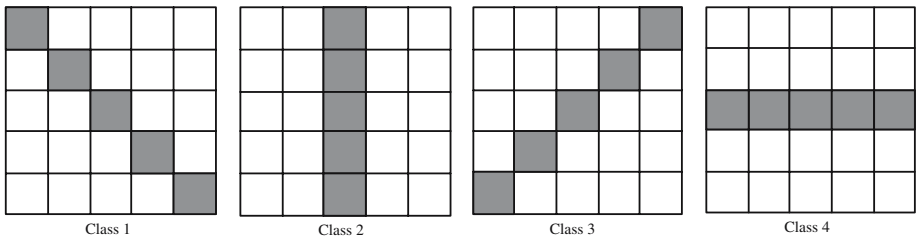


Fig. 1. The four edge orientations

The possible stable edge structures are analyzed according to the expected effects which are single pixel width, continuous. Summarize the conditions in which computation will be finished if window reaches one of them:

1. An edge with the orientation under consideration through the window,
2. No edge in the window.

Here, let the value "-1" be an edge credit for a non-edge, and the value "+1" denote an edge pixel. Therefore, the energy function for the Class 2 (others are similar) should be constructed such that it favors states that:

1. Have in each row only one +1 and other all -1,
2. Have all -1 in all rows, and
3. The total number of +1 is n (for n×n window).

The energy function which satisfies the above states is:

$$\begin{aligned}
 E = & A \sum_{x=1}^5 \sum_{i=1}^5 \sum_{j=1, \neq i}^5 (V_{x,i} + 1)(V_{x,j} + 1) + B \sum_{x=1}^5 \sum_{i=1}^5 [(V_{x,i} + 1) - 2]^2 \\
 & \times \sum_{j=1, \neq i}^5 (V_{x,j} + 1) + C \sum_{x=1}^5 (|\sum_{i=1}^5 V_{x,i}| - 5)^2 \tag{3}
 \end{aligned}$$

Where, V represents the output of the neural cell with values ranging from -1 to +1, and A, B, C, D , and are parameters. Each item in the energy function has minimum value 0 since the output V is from -1 to 1. The first two items are designed to satisfy the first state. When having all ?1 in all rows, or only one +1 and other all -1, $(V_{x,i} + 1)(V_{x,j} + 1)$ is zero. So the first two items have minimum value zero. For the second item, as there is only one +1 at each row, $\sum_{i=1}^5 [(V_{x,i} + 1) - 2]^2$ will be 2, which makes the first term equal zero. Therefore, the first two items reach their minimum zero when the edge structure in a window is stable at state 1. The last three items are designed to satisfy the second state. When having all -1 or +1 in a column, $|\sum_{i=1}^5 V_{x,i}|$ be 5, and $(|\sum_{i=1}^5 V_{x,i}| - 5)^2$ will be zero, which makes the whole item equal zero. The same result can be obtained for the last two items.

From the energy function, the time derivation of the central unit in a window can be derived as:

$$\begin{aligned}
 \frac{dy_{3,3}}{dt} = & -y_{3,3} - A \sum_{j=1, i \neq 3}^5 (V_{3,j} + 1) - B \sum_{i=1}^5 [(V_{3,i} + 1) - 2]^2 - 2B \sum_{j=1, \neq i}^5 (V_{3,j} + 1) \\
 & \times \sum_{j=1}^5 (V_{3,j} + 1) \sum_{i=1}^5 [(V_{3,i} + 1) - 2] - 2C \sum_{x=1}^5 V_{i,3} (|\sum_{i=1}^5 V_{x,i}| - 5) \tag{4}
 \end{aligned}$$

The terms with coefficients A and B are for interactions between the central unit and the other units on the row. The last three terms determine the interactions between the central unit and the other units on the column. All parameters in the above equation are positive. When the system does not reach a stable state,

the first parts of the last three terms decide whether to inhibit or excite the central unit, and how strong the interaction is. When it is close to a stable state, the second parts approach 0 so that the exciting or inhibiting is very small.

The enhancement procedure will be terminated when neural computation enters into a situation such that most of the windows are in a stable state with minimum energy. After processing maps in each of four orientations, we assemble four resultant maps and special edges into one edge map.

4 Experiments and Discussion

All of our experimental results were obtained by detecting every pixel in the image using a 5×5 neighborhood window, and the enhancement is applied after labeling edges by a pre-existing edge detector. To test the performance of different algorithms, we compare the results we obtained from other algorithms with those from our algorithm.

Fig 2(a) shows a standard class1 edge, and (b) shows a interrupted edge. Based on Matlab7.04 simulating, the interrupted edge is convergent to the standard class1 edge by 11 iterations, the 10 errors are -1.0484, -0.3894, 0.0669, 0.3829, 0.6018, 0.7533, 0.8582, 0.9309, 0.6332, and 0.1080 respectively.

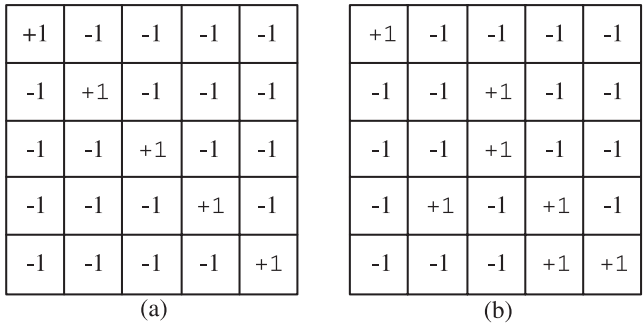


Fig. 2. A standard and interrupted edge for class1 orientation

Fig 3 displays the results by different enhancement algorithms. Fig 3(b) is the results by Canny operator. the enhanced result in literature [8] is shown in Fig 3(c), and Fig 3(d) present enhanced edge of the proposed algorithm corresponded to them. From the results, it could be seen that our operator obtains a single pixel width and continuous edge, while the Canny operator obtains a disconnected edge, and the enhanced method of literature [8] get a more wide edge. That proves that the HNN can thin edge efficiently. But the proposed algorithm does not perform well for edges at corners comparing to other algorithms, this is our next work that we want to amend.

Fig 4 displays the enhancement result of edges with noise. Fig 4(b) is the results by Canny operator. the enhanced result in literature [8] is shown in Fig 4(c),



Fig. 3. The different enhancement results of edge without noise

and Fig 4(d) present enhanced edge by our method. As shown, although the original images have been corrupted by noise, the proposed algorithm could get the better edge images which outlines are still clearly legible. It has little effect on recognizing the image contour and many details. But, there are many false edges in the results detected by Canny and literature [8] operators which are brought by the noises. That makes details so bad that it could not be recognized what they are. From a mass of experiments, we find that the more noise contained, the worse the results are. And contrastive effects between the proposed algorithm and classical edge operators are more evident.

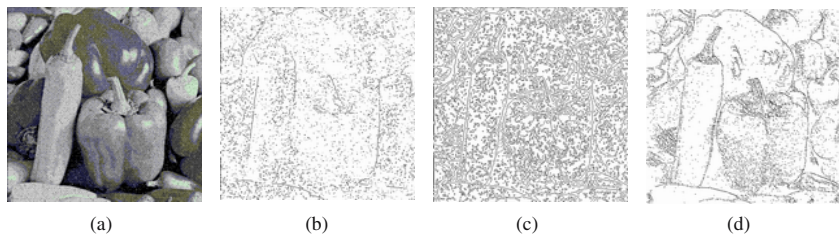


Fig. 4. The different enhancement results of edge with noise

5 Conclusions

In this paper, we have proposed a novel post-processing edge enhancement algorithm. Taking advantage of pre-exist edge detector, we obtain the outlines of edge maps. The edge enhancement is performed on the four edge orientations respectively, and the maps are updated after iteration by the HNN. It efficiently thinned and repaired the edge elements with any orientation. This approach enables the utilization of global edge information as the information is "propagated" to surrounding elements in the edge maps. Therefore, the edges can be enhanced by recovering missing edges, and eliminating false edges. From the experimental results, it can be seen that the proposed algorithm in this paper is superior to other methods, this advantage is more prominent under the noisy condition and the robusticity is better.

In future work, we will modify our algorithm at where end pixels and abrupt pixels are, and make our method more available. We will also explore the use of weak edges, and asymmetric illumination, to add more information into the edge map.

References

1. Ran, X., Farvardin, N.: A Perceptually Motivated Three-component Image Model. Part I: Description of the model. *IEEE Trans. Image Process* **4** (1995) 401-415
2. Polesel, A., Ramponi, G., Mathews, V.J.: Image Enhancement via Adaptive Unsharp Masking, *IEEE Trans. Image Process*, **3**(2000)505-510
3. Winkler, S.: Visual fidelity, Perceived Quality: Towards Comprehensive Metrics. *Proc. SPIE*, **4299**(2001)114-125
4. Caviedes, J., Gurbuz, S.: No-reference Sharpness Metric Based on Local Edge Kurtosis. *Proc. IEEE Int. Conf. Image Processing (ICIP)*, **3**(2002)53-56
5. Dijk, J., van Grinkel, M., van Asselt, R.J., van Vliet, L.J., Verbeek, P.W.: A New Sharpness Measure Based on Gaussian Lines and Edges. *Proc. Int. Conf. Computer Analysis on Images and Patterns (CAIP)*, *Lecture Notes in Computer Science*, **2756**(2003) 149-156
6. Hopfield, J., Tank, D.W. : Neural Computation of Decisions in Optimization Problems. *Biol. Cybern.* **52** (1985)141-152
7. Cohen, M.A., Grossberg, S.G.: Absolute Stability of Global Pattern Formation and Parallel Memory Storage by Competitive Neural Networks. *IEEE Trans. Syst., Man, Cybern.* **13** (1983) 815-826
8. Lily, R., Liang, C., Looney, G.: Competitive Fuzzy Edge Detection. *Applied Soft Computing* **3** (2003) 123-137

Image Quality Assessment Based on Wavelet Coefficients Using Neural Network

Dongxue Yue, Xinsheng Huang, and Hongli Tan

College of Mechatronic Engineering and Automation,
National University of Defense Technology, Changsha, P.R. China
ydx315@126.com

Abstract. A novel image quality metric based on the characteristics of wavelet coefficients of images is proposed in this paper. An image is decomposed into several levels by means of wavelet transform. The standard deviations of the diagonal details (HH coefficients) at each level increase with the noise standard deviation increasing and decrease with the blurring radius increasing. According to that, an image quality can be measured by analyzing the characteristics of its wavelet coefficients. Neural network is used to realize the algorithm of image quality assessment. The results of experiments demonstrate that the image quality metric is reasonable and the algorithm realization using neural network is feasible and performs well.

1 Introduction

In vision-aided terminal guidance system using inertial navigation system (INS), a reliability metric of target recognition is necessary for the system to decide the vision information can be used or not. The performance of the guidance system can be improved by the real target or distorted by the error target, which provided by the vision system. The precision of the vision target relies on both good image and appropriate algorithm of image processing, so that image quality measure is needed.

There are many kinds of image quality measures, which vary with the application of image [1,2,3]. For example, in multimedia application, since a human observer is the end user an image quality measure that is based on a human vision model seems to be more appropriate for predicting user acceptance and for system optimization. However for computer vision task, prediction of the algorithmic performance in terms of imaging distortions is of great significance.

The wavelet transform produces a hierarchical decomposition of functions. According to Mallat [4] a function is described by means of a low-resolution function plus a series of details from low to high resolution. Wavelets provide means of frequency and space analysis. Neural networks [5,6] are very sophisticated modeling techniques capable of modeling extremely complex functions. For their huge power and easing to use, neural networks have been successfully applied across an extraordinary range of problem domains.

Noise reduces the number of edges those can be detected. Blurring reduces the standard deviation of the HH coefficients at each level of wavelet decomposition. A novel image quality metric based on the loss of edges and the reduction of standard deviation of HH coefficients is proposed. Wavelet transform is used to analysis the distortion of images in various frequencies. Neural network is used to model the function about the image quality metric and the wavelets coefficient of images.

The rest of the paper is organized as follows: section 2 gives the wavelet coefficients property of the image. In section 3 we describe the algorithm of image quality metric. Section 4 presents the results of experiments. Section 5 reports some conclusions and future works.

2 Characteristics of Wavelet Coefficients

2.1 Wavelet Transform

A wavelet transform decomposes a function in different Levels-Of-Detail [7]. It provides frequency and space analysis.

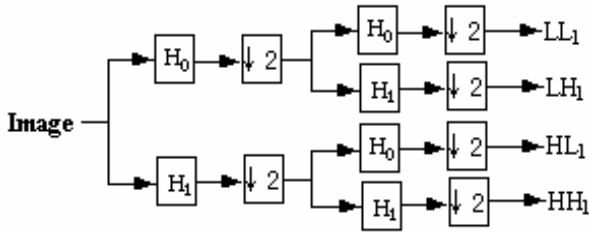


Fig. 1. Wavelet decomposition of an image

The original points are grouped and related to the wavelet coefficients according to their position and level. Each set of 2×2 -neighboring points in the original image is related to four corresponding coefficients in LL, LH, HL and HH bands at level 1 of the wavelet transform, as shown in figure 1. These coefficients are extracted from the original image by using four kinds of filters: row high pass (horizontal details H_1), column high pass (vertical details H_1), row low pass (horizontal average H_0) and column low pass (vertical average H_0). The filtered data from the row high pass filter and the column high pass filter is stored as the HH coefficients, which are the diagonal edge details of image. The filtered data from the row low pass filter and the column high pass filter is the LH coefficients. The filtered data from the row high pass filter and the column low pass filter is the HL coefficients. And the filtered data from the row low pass filter and the column low pass filter is the LL coefficient, which is the approximate signal of original image. The LL coefficient is now treated as the original data and the low and high pass filters are then applied to this data again. This process repeats until there only one point left in the LL coefficient.

2.2 Characteristic of Wavelet Coefficients in Distorted Images

Edge points have high gradient in contrast with others in images. The edge contrast decrease due to the distortion, for example, blurring, noise, compression and sensor inadequacy. In this paper we mainly discuss noise and blurring distortion. Noise usually has higher contrast than real edge, and blurring smoothes the edge. In noise image, the noise points as their high gradient may be detected as edge points, and the real edge can't be detected, thus high error rate will occur. In blurred image, the edge contrast is smoothed and the real edge is replaced by several points, in this case, multiple responses to a single edge will happen. According to Canny's criteria of optimal edge detector measure [8]: low error rate, well localized and one response to signal edge; noise and blurring brake the law of optimal detector, and the target position is not exactly in noise or blurred image. Hence, in vision guidance system, the distortion degree of images is needed to measure the reliability of the target detected from images.

As mentioned in section 2.1, the HH coefficients reflect the diagonal edge details of image, noise may be detected as the details. That is, the HH coefficients are connected with the edge of object; however, in noise image those may be noise signal. The HH coefficients also can be distorted by blurring for its smoothing. Experiments are designed to show the distortion of noise and blurring (Fig 2).

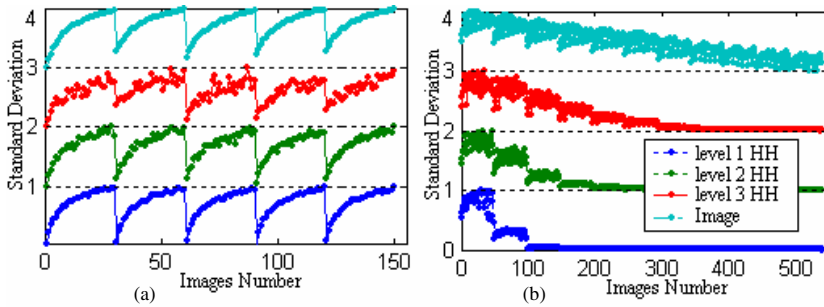


Fig. 2. Standard deviations of three levels wavelets coefficients in images. (a) Coefficients standard deviation (Std) in noise images. (b) Coefficients Std in blurred images.

The original image in figure 2 is Lena image. In Fig 2a we select 15 (36×36 pixels) images at random in Lena image. The noise image is generated by adding standard Gaussian white noise to each image. The standard deviations of Gaussian noise σ^2 are given by $\sigma^2 = 0.01i \{i = 0, 1, \dots, 29\}$. There are total 150 (15×30) noise images. In Fig 2b the blurred images are generated by increase the blurring radius, r , which are given by $r = 2i \{i = 0, 1, \dots, 5\}$ and $r = 4i - 10 \{i = 6, 7, \dots, 10\}$. There are 49 (36×36 pixels) original images selected

from Lena image. Hence, there are totally 539(49×11) images. Each of noise or blurred images is decomposed into three levels using Daubechies (db2) wavelets.

To discriminate the lines, we respectively plot the standard deviation of HH coefficients at three levels and that of original images on the baseline of zero, 1, 2 and 3. From figure 2 we can see that the standard deviations of HH coefficients and original images are increase as the noise increase, and decrease as the radius of blurring increase. Therefore, wavelet coefficients can reflect the distortion degree of images.

3 Algorithm

3.1 Image Quality Metric

The number of edge points that can be detected is smaller in noise images than in 'good' image. The number decreases gradually as the noise standard deviation increases (Fig 3a). Therefore we can use the loss of edge points as the image quality metric to reflect the noise distortion degree. In noise free image, the edge loss is zero; we set image quality measure of the 'good' image as 1, on the contrary if the object is masked by noise, we set the measure as zero. We give the image quality metric, d , as follows:

$$d = 1 - \frac{e - e_1}{e} = \frac{e_1}{e} \tag{1}$$

where d varies from zero to 1, which reflects the distortion of the image; e and e_1 are the numbers edge points in 'good' and noise image respectively. The image quality measures of noise images used in Fig 2a are shown in Fig 3a.

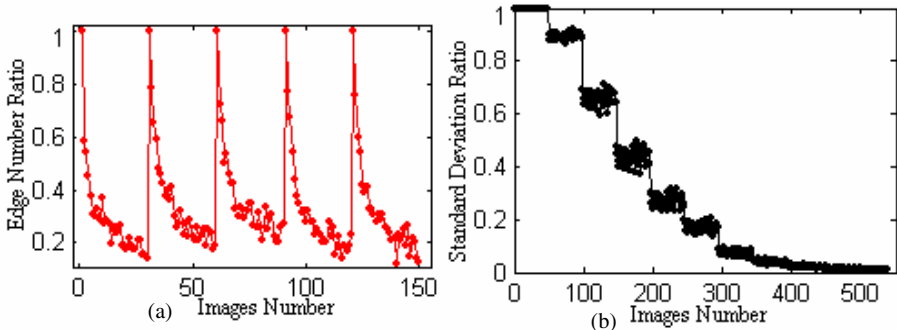


Fig. 3. The image quality metric. (a) Edge number ratios of 'good' and noise images. (b) Std ratios of the HH at level 3.

As shown in Fig 2b, the standard deviations of HH coefficients at every level of wavelet decomposition decrease with the blurring radius growing. Here we use the ratio of the HH coefficients standard deviation of blurred images to that

of 'good' images, as the metric to reflect the blurring distortion in images. To omit noise influence, the HH coefficient at level 3 is selected. The image quality measures of blurred images used in Fig.2.b are shown in Fig.3.b.

3.2 Algorithm Using Neural Network

As discussed in section 2.2, the standard deviation of the HH coefficients at three levels and that of original image can reflect the distortion including noise and blurring. On the other hand, the image quality metric proposed in section 3.1 also can reflect the distortion. There is some relationship between the image quality metric and the standard deviation of wavelets coefficients. Neural networks are sophisticated modeling extremely function. Therefore, neural network is used to model the relation between the image quality metric and the standard deviations of wavelets coefficients. Back propagation (BP) network is used in this work.

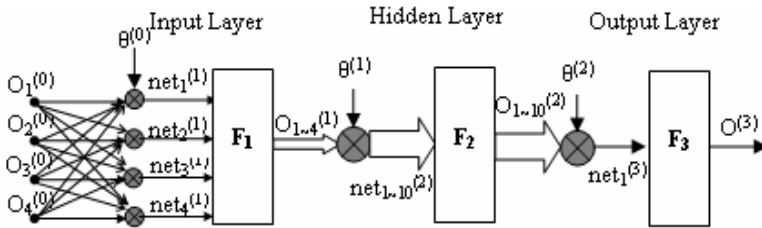


Fig. 4. Architectures of feed-forward BP neural network

The parameters of the network architectures used in this work are shown in figure 4. The feed-forward back-propagation network has three layers. There are 4 TANSIG neurons in the input layer (first layer) and 10 TANSIG neurons in the hidden layer (second layer) and one PURELIN neuron in the output layer (third layer). $O_i^{(L)}$, $net_i^{(L)}$ and $\theta_i^{(L-1)}$ are the i th output, input and threshold of the L th layer respectively.

The neural network has four inputs, $O_{1-4}^{(0)}$, and one output, $O^{(3)}$. The HH coefficients standard deviations at levels 1 to 3 are the three of the inputs of neural network. The standard deviation of original image is the fourth input. The output of neural network is the image quality metric, which varies from zero to 1. If the metric is 1, that means the image is not distorted (that is 'good' image). On the contrary, if the metric is small (for example, smaller than 0.2), that means the image can't be used for its severe distortion.

4 Simulation Results

Experiments are designed to show the results of the algorithm described above. There are 64(36×36 pixels) original images selected from Lena image(128×128 pixels). Different noise and blurring distortions added to the original images to

generate 1452 sample images, those are used to train the neural network. The sample images include 'good' images, noise images, blurred images and blurred-noise images. Using Daubechies (db2) wavelet, each image is decomposed into three levels. The standard deviations of the three levels HH coefficients and those of sample images are the inputs of neural network. The output of neural network is the image quality metric, which is computed by the method proposed in section 3.1. In noise images the metric are the ratios of edge number and in blurred image the metric are the ratios of standard deviations of the HH coefficients at level 3. The BP neuron network described in section 3.2 is used in the experiments.

Two groups of test images are presented. The original images of test 1 are selected from Lena image, which are different pixels of training samples images. The original images of test 2 are selected from an airport image (Fig.5c), which is cut from a satellite image of Orlando, Florida's airport (orlando airport.jpg in [9]). The noise and blurred test images are generated in the same way as training sample images.

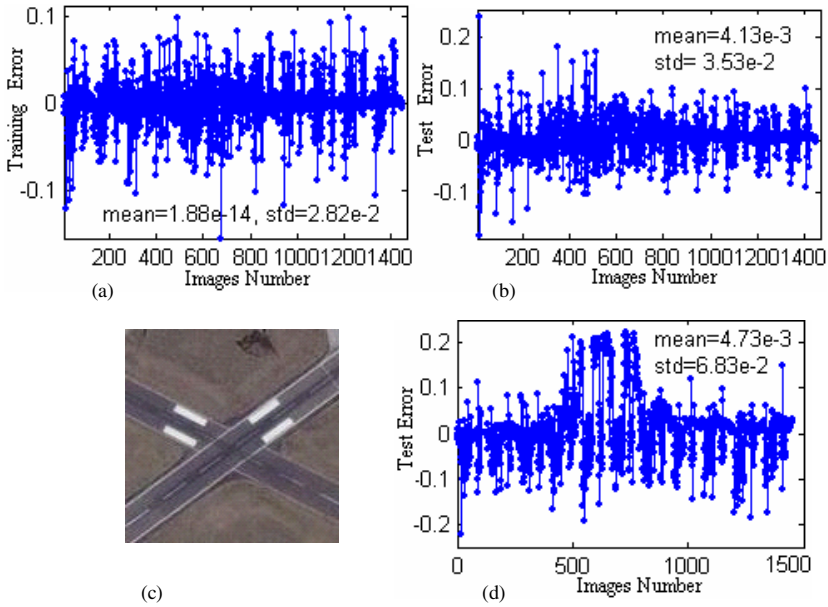


Fig. 5. The results of the neural network of experiments. (a) The neural network training error. (b) The image quality metric error of test 1. (c) Airport image. (d) The image quality metric error of test 2.

From the errors of image quality measure shown in figure 5, we can see that the neural network performance well to the similar images and different images. That means the image quality metric is reasonable and the algorithm realization using neural network is feasible and performs well.

5 Conclusions and Future Works

The image quality measure is needed in some applications. Through theoretical analysis and experiment demonstration, we found that the standard deviations of wavelet coefficients are varying with the distortion. Noise reduces the number of edges those can be detected. Blurring reduces the standard deviation of the HH coefficients at each level of wavelet decomposition. An image quality metric based on the loss of edges and the reduction of standard deviation of HH coefficients is proposed. Neural network is used to realize the algorithm of image quality assessment. The results of experiments show that the image quality metric is reasonable and the algorithm realization using neural network is feasible and performs well.

The image quality metric proposed in this paper make full use of the property of multiresolution analysis of wavelet transform to analyze image in frequency domain. The metric do not need any standard images to compare with and can be computed fast. Those are very important for real-time guidance system, where no picture store and little time can be spent.

In future works, the images quality metric will be used as a factor to assess the reliability of images object in vision aided guidance system and other vision applications.

References

1. Ismail, A., Bulent, S., Khalid, S.: Statistical Evaluation of Image Quality Measures. *Journal of Electronic Imaging* **11**(2) (2002) 206-223
2. Kanugo, T., Haralick, R.M: A Methodology for Quantitative Performance Evolution of Detection Algorithms. *IEEE Trans. Image Process* **4**(12) (1995) 1667-1673
3. Babu, R.V., Perki, A.: An HVS-Based No-Reference Perceptual Quality Assessment of JPEG Coded Images Using Neural Networks. *IEEE International Conference on Image Processing* **1** (2005) 433-436
4. Mallat, S.: A Theory for Multiresolution Signal Decomposition: The Wavelet Decomposition. *IEEE Trans. Pattern Analysis and Machine Intelligence* **11**(7) (1989) 674-693
5. Stergiou, C., Siganos, D.: Neural networks.
<http://www.statsoft.com/textbook/stneunet.html>.
6. <http://www.doc.ic.ac.uk/~nd/surprise96/journal/vol4/cs11/report.html>.
7. Abasolo, M.J., Perales, F.J.: Wavelet Analysis for a New Multiresolution Model for Large-Scale Textured Terrains. *Journal of WSCG* **11**(1) (2003)
8. Canny, J.: A Computational Approach to Edge Detection. *IEEE Trans. Pattern Analysis and Machine Intelligence* **PAMI-8**(6) (1986) 679-698
9. <http://www.spaceimage.com/gallery/orlando/airport.jpg>

Neural Network Approach for Designing One- and Two-Dimensional Quasi-Equiripple FIR Digital Filters

Xiaohua Wang^{1,2}, Yigang He², and Yulou Peng¹

¹ Department of Electrical and Information Engineering,
Changsha University of Science & Technology, Changsha 410077, P.R. China

² Department of Electrical and Information Engineering,
Hunan University, Changsha 410082, P.R. China
cslgwxh@163.com

Abstract. A quasi-equiripple one- and two-dimensional linear-phase FIR digital filters design approach is proposed based on a novel neural network optimization technique. Its goal is to minimize the weighted square-error function in the frequency domain. The design solution is presented as a parallel algorithm to approximate the desired frequency response specification, and the weight coefficients are updated according to the error function. Thus, the proposed approximation method can avoid the overshoot phenomenon which may happen near the pass-band and stop-band edges of the designed filter, and may make a fast calculation of the filter's coefficients possible. Several optimal design examples are given to illustrate the effectiveness of the proposed approach.

1 Introduction

With the rapid development of digital devices having increased speed and storage capabilities, the design and applications of one-dimensional (1-D) and two-dimensional (2-D) digital filters have received considerable attention in recent years. Many techniques [1]–[8] have been established for the design of 1-D and 2-D FIR filters. The window function method [1] is one of the earliest and simplest techniques for FIR filters design, but this method gives a filter design which is not optimal in any sense. McClellan transformation technique [2] yields a good approximation for some frequency responses. However, this method cannot be used to closely approximate all magnitude responses. The optimization techniques approximate a desired behavior by minimizing an error function that is formulated using the L_p or the L_∞ norm. The weight least-square (WLS) technique [3]–[4] using L_2 norm is seen as an efficient optimal design method. Although it can be reduce to the solution of a system of linear equations, the time-consuming iterative procedure or matrix inversion operation is often needed. The analytical WLS method [5]–[6] avoids matrix inversion operation, but this method may result in an unstable filter. Semi-definite programming [7] can design optimal filters, but it suffers from a heavy computation burden.

In this paper, we proposed a new weighted neural network algorithm (NNA) for the design of linear-phase quasi-equiripple 1-D and 2-D FIR filters. The design problem is formulated based on the approximation of a magnitude response. The focus is placed on developing the neural network design method.

2 Neural Network Optimal Approach of 1-D FIR Filters

The frequency response of a 1-D FIR filter with length N can be expressed as

$$H_d(e^{j\omega}) = \sum_{i=0}^{N-1} h(i)e^{-ji\omega}, \tag{1}$$

where the impulse response $h(i)$ may be complex numbers or real ones. If N is odd integer, and $h(n) = h(N-1-i)$, then this is a linear-phase FIR filter and its magnitude response is

$$H_d(\omega) = \sum_{i=0}^n a(i) \cos(i\omega), \tag{2}$$

where, $n = (N-1)/2$, and $a(0) = h(n)$, $a(i) = 2h(n-i)$, $i = 1, 2, \dots, n$.

Therefore, the design problem of a linear-phase FIR filter is turned to find the coefficients $a(i)$ in (2) such that the magnitude response $\{H_d(\omega)\}$ is close to a given magnitude response in some optimal sense. In this section, a neural networks algorithm is used for solving the filter design problem.

2.1 Neural Network Algorithm

First, sample uniformly $H_d(\omega)$ in $\omega \in [0, \omega_p] \cup [\omega_s, \pi]$ to get its discrete values, then (2) can be expressed as

$$H_d(m) = \sum_{i=0}^n a(i) \cos[i\omega(m)], \quad m = 1, 2, \dots, M. \tag{3}$$

Let

$$\mathbf{c} = \begin{bmatrix} 1 & \cos[\omega(1)] & \dots & \cos[n\omega(1)] \\ 1 & \cos[\omega(2)] & \dots & \cos[n\omega(2)] \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \cos[\omega(M)] & \dots & \cos[n\omega(M)] \end{bmatrix}^T \tag{4}$$

$$\mathbf{a} = [a_0, a_1, \dots, a_n]^T \tag{5}$$

then (3) can be rewritten as

$$\mathbf{H}_d = \mathbf{c}^T \mathbf{a}. \tag{6}$$

Now define error function as

$$\mathbf{e} = \mathbf{D}\mathbf{B}(\mathbf{M}_d - \mathbf{H}_d) \tag{7}$$

where, \mathbf{M}_d is desired magnitude response vector, and

$$\mathbf{D} = \text{diag}([W(1) \ W(2) \ \dots \ W(M)]) \tag{8}$$

$$\mathbf{B} = \text{diag}([b(1) \ b(2) \ \dots \ b(M)]) \tag{9}$$

here, $W(m)$, $W(m)$ is one in the pass-band and δ_p / δ_s in the stop-band, is a weighting coefficient, and $b(m)$, $b(m) > 0$, is a weighting function. Then define performance index J as

$$J(\mathbf{e}) = \frac{1}{2} \sum_{m=1}^M e^2(m) = \frac{1}{2} \|\mathbf{e}\|_2^2. \tag{10}$$

To minimize J , a parallel back-propagation neural network model is chosen in Fig. 1. Then \mathbf{a} is recursively calculated as

$$\mathbf{a}_{k+1} = \mathbf{a}_k - \eta \frac{\partial J}{\partial \mathbf{a}_k} = \mathbf{a}_k + \eta \mathbf{cBD} \mathbf{e}_k \tag{11}$$

where η , $\eta > 0$, is learning rate.

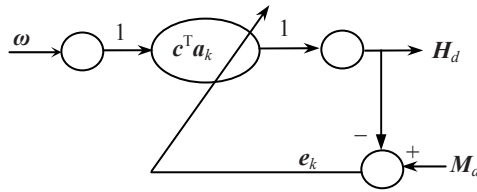


Fig. 1. Neural network model

In order to ensure convergence of the neural network, it is important to select proper learning rate η . Next, we present and prove the convergence theorem of neural networks algorithm.

Theorem 1. If learning rate satisfies $\eta < 2 / \|(\mathbf{cBD})^T \mathbf{cBD}\|_2$, the neural network algorithm is stable and convergent asymptotically to its global minimum.

Proof: Define (10) as a Lyapunov function. Since

$$\Delta \mathbf{e}_k = \left(\frac{\partial \mathbf{e}_k}{\partial \mathbf{a}_k} \right)^T \Delta \mathbf{a}_k = -\eta (\mathbf{cBD})^T \mathbf{cBD} \mathbf{e}_k \tag{12}$$

thus

$$\begin{aligned} \Delta J(\mathbf{e}_k) &= \frac{1}{2} \|\mathbf{e}_k - \eta (\mathbf{cBD})^T \mathbf{cBD} \mathbf{e}_k\|_2^2 - \frac{1}{2} \|\mathbf{e}_k\|_2^2 \\ &\leq \frac{1}{2} \|\mathbf{I} - \eta (\mathbf{cBD})^T \mathbf{cBD}\|_2^2 \|\mathbf{e}_k\|_2^2 - \frac{1}{2} \|\mathbf{e}_k\|_2^2 \\ &= \frac{1}{2} \eta \|\mathbf{e}_k\|_2^2 \left[\|(\mathbf{cBD})^T \mathbf{cBD}\|_2 (\eta \|(\mathbf{cBD})^T \mathbf{cBD}\|_2 - 2) \right] \end{aligned} \tag{13}$$

It is easy to see from (13) that if

$$\eta < 2 / \left\| (cBD)^T cBD \right\|_2 \tag{14}$$

then $\Delta J(\mathbf{e}_k) \leq 0$. Since

$$\left\| (cBD)^T cBD \right\|_2 > 0 \tag{15}$$

thus if $\Delta J(\mathbf{e}_k) = 0$, we have

$$\mathbf{e}_k = 0, \tag{16}$$

$$\Delta \mathbf{a}_k = 0, \tag{17}$$

$$J(\mathbf{e}_k) = 0. \tag{18}$$

Therefore, if $\eta < 2 / \left\| (cBD)^T cBD \right\|_2$, the neural network algorithm is stable and convergent asymptotically to its global minimum. The theorem is proved completely.

To design equiripple FIR filters, the weighting function $b(m)$ is to be updated as follow (see [3]):

$$b_{k+1}(m) = b_k(m) \frac{|W(m)(M_d(m) - H_d(m))|}{\sum_m |e_k(m)|}. \tag{19}$$

In order to avoid $b_k(m)$ getting too close to zero, a lower bound, r , is necessary to be specified. Our extensive numerical experience indicates that, for numerical stability, r should be close to 0.001.

Due to illustrate the design procedure, some guidelines are listed below.

- 1) Initial values: Specify the frequencies sample point length M , weighting coefficients $W(m)$ and $b(m)$. Then define an arbitrary small positive real number ε , and select learning rate η . Produce an initial random weight vector \mathbf{a} .
- 2) Produce new predicted output H_d of neural networks using (6), and calculate \mathbf{e} and J via (7) and (10).
- 3) Update the weighting coefficient vector \mathbf{b} according to (19).
- 4) Update the weighting vector \mathbf{a} according to (11).
- 5) If $J > \varepsilon$, go to step 2), otherwise, close the training of the neural network.

The above guidelines will be followed in the design examples given in the next subsection.

2.2 Design Examples

In this section, two design examples are considered to illustrate the effectiveness of the proposed design algorithm.

Example 1. Consider a multi-band FIR filter design example given in [2], i.e., $N = 65$, and

$$H(\omega) = \begin{cases} 1, & \text{for } 0 \leq \omega \leq 0.1\pi, \quad 0.5\pi \leq \omega \leq 0.66\pi, \\ 0, & \text{for } 0.16\pi \leq \omega \leq 0.44\pi, \quad 0.7\pi \leq \omega \leq \pi. \end{cases} \tag{20}$$

Let $\eta=1.792$, $M=100$, $\delta_p / \delta_s = 1.4$. It takes the proposed algorithm 3000 times training to converge to a multi-band FIR filter with the magnitude response shown in Fig. 2. The maximum ripple is 0.1955 dB in the first pass-band and 0.1999 dB in the second pass-band, and the minimum attenuation is 35.5212 dB in the first stop-band and 35.0320 dB in the second stop-band. By comparison, a Chebyshev design algorithm was used to design this filter in [2], the maximum ripples in the first and second pass-band are all more than 0.5 dB, while the minimum attenuation is less than 35 dB in the first stop-band and 32 dB in the second stop-band.

Example 2. Considering the example 2 in [8], design a FIR notch filter specified by $\omega^* = 0.84\pi$.

Let $N = 135$, $\eta = 1.5$, $M = 1000$, $\delta_p / \delta_s = 5$. After 2500 times training, the algorithm converged to a linear phase FIR notch filter with the magnitude response shown in Figure 3. The actual parameters are $\omega^* = 0.8402\pi$ and $\Delta\omega = 0.0812\pi$ for $a = -3.0103\text{dB}$, the maximum ripple 0.0198 dB in the two pass-bands, and the minimum attenuation 81.55dB on notch point. Comparing with the design example 2 in [8], the actual 135-order filter parameters are $\omega^* = 0.8428\pi$ and $\Delta\omega = 0.1206\pi$ for $a = -3.0103\text{dB}$, and the minimum attenuation 60 dB on notch point, based on an analytical design algorithm.

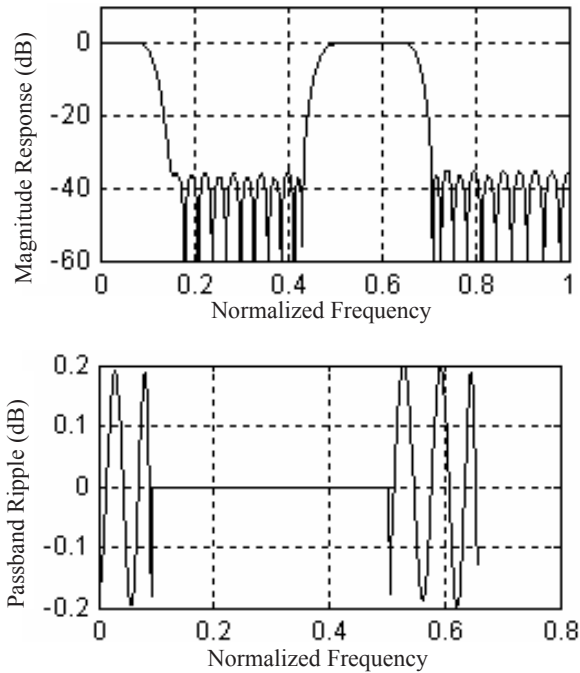


Fig. 2. Magnitude response of the designed multi-band FIR filter, see Example 1

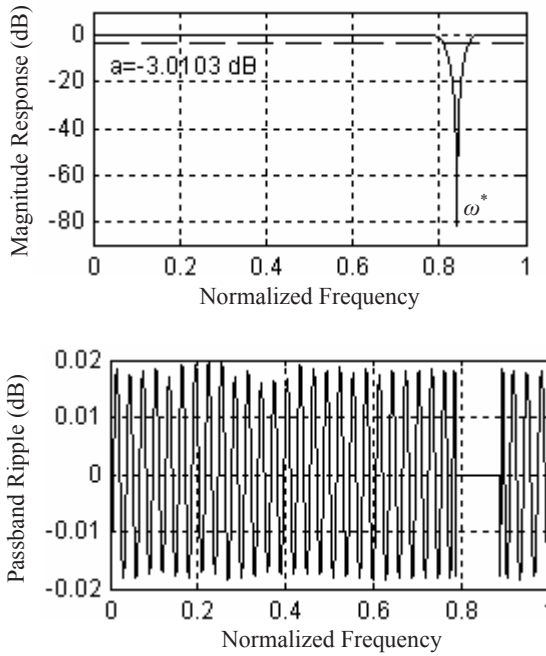


Fig. 3. Magnitude response of the designed notch Filter, see Example 2

3 2-D FIR Filter Design

For a class of 2-D linear-phase filter with quadrantal symmetry, the coefficient matrix \hat{H} of the $N_1 \times N_2$ th-order filter can be partition as (see [7])

$$\hat{H} = \begin{bmatrix} \mathbf{H}_{11} & \mathbf{h}_{12} & \mathbf{H}_{13} \\ \mathbf{h}_{21}^T & h_{22} & \mathbf{h}_{23}^T \\ \mathbf{H}_{31} & \mathbf{h}_{32} & \mathbf{H}_{33} \end{bmatrix} \tag{21}$$

where, $\hat{H} \in R^{N_1 \times N_2}$, N_1 and N_2 are odd integers, $\mathbf{H}_{11}, \mathbf{H}_{13}, \mathbf{H}_{31}, \mathbf{H}_{33} \in R^{n_1 \times n_2}$, $\mathbf{h}_{12}, \mathbf{h}_{32} \in R^{n_1 \times 1}$, $\mathbf{h}_{21}, \mathbf{h}_{23} \in R^{n_2 \times 1}$, $h_{22} \in R$, $n_1 = (N_1 - 1) / 2$, $n_2 = (N_2 - 1) / 2$, and

$$\begin{cases} \mathbf{H}_{11} = \text{flipud}(\text{fliplr}(\mathbf{H}_{33})) \\ \mathbf{H}_{13} = \text{flipud}(\mathbf{H}_{33}) \\ \mathbf{H}_{31} = \text{fliplr}(\mathbf{H}_{33}) \\ \mathbf{h}_{12} = \text{flipud}(\mathbf{h}_{32}) \\ \mathbf{h}_{21}^T = \text{fliplr}(\mathbf{h}_{23}^T) \end{cases} \tag{22}$$

here, *flipud* and *fliplr* represent the operation of flipping a matrix upside down and from left to right, respectively. Therefore, the magnitude response of the filter is given by

$$H_d(\omega_1, \omega_2) = \mathbf{c}_1^T(\omega_1) \mathbf{H} \mathbf{c}_2(\omega_2) \tag{23}$$

where $\mathbf{c}_i(\omega_i) = [1, \cos \omega_i, \cos 2\omega_i, \dots, \cos n_i \omega_i]^T$ for $i = 1, \text{ and } 2$, and

$$\mathbf{H} = \begin{bmatrix} h_{22} & 2\mathbf{h}_{23}^T \\ 2\mathbf{h}_{32} & 4\mathbf{H}_{33} \end{bmatrix}. \tag{24}$$

Consequently, the task of designing a 2-D linear-phase FIR filter reduces to the following problem: find filter coefficient matrix \mathbf{H} such that $H_d(\omega_1, \omega_2)$ approximates the desired magnitude response $M_d(\omega_1, \omega_2)$.

3.1 Approximation of $M_d(\omega_1, \omega_2)$ Using NNA

Now we denote the sampled magnitude response matrix by $M_d^{M \times M}$, and let

$$\mathbf{D}_i = [\mathbf{c}_i(\omega_i(1)) \quad \mathbf{c}_i(\omega_i(2)) \quad \dots \quad \mathbf{c}_i(\omega_i(M))], \quad i = 1, 2 \tag{25}$$

then (23) can be rewritten as

$$\mathbf{H}_d = \mathbf{D}_1^T \mathbf{H} \mathbf{D}_2. \tag{26}$$

Now we define error function as

$$\mathbf{e} = \mathbf{W} .* \mathbf{B} .* (\mathbf{M}_d - \mathbf{H}_d) \tag{27}$$

where, “.*” is matrix element group multiplication operation in MATLAB, i.e., $\mathbf{W} .* \mathbf{B} = [W_{ij} B_{ij}]_{M \times M}$, $W(m_1, m_2)$, $B(m_1, m_2)$ is one in the pass-band and δ_p / δ_s in the stop-band, is a weighting coefficient, and $B(m_1, m_2) > 0$ in the pass-band and stop-band and $B(m_1, m_2) = 0$ in the transition band, is a weighting function. Now define performance index J as

$$J(\mathbf{e}) = \frac{1}{2} \sum_{m_1=1}^M \sum_{m_2=1}^M e^2(m_1, m_2) \tag{28}$$

then the steepest descent algorithm for the approximate mean square error is

$$\mathbf{H}_{k+1} = \mathbf{H}_k - \eta \frac{\partial J}{\partial \mathbf{H}_k} = \mathbf{H}_k + \eta \mathbf{D}_1 (\mathbf{W} .* \mathbf{B} .* \mathbf{e}_k) \mathbf{D}_2^T. \tag{29}$$

To design equiripple 2-D FIR filters, the weighting function $B(m_1, m_2)$ is to be updated as follow:

$$B_{k+1}(m_1, m_2) = B_k(m_1, m_2) \frac{|W(m_1, m_2)(M_d(m_1, m_2) - H_d(m_1, m_2))|}{\sum_{m_1} \sum_{m_2} |e_k(m_1, m_2)|} \tag{30}$$

Obviously, H can be obtained from formula (30), and the coefficient matrix \hat{H} of 2-D linear phase FIR filter can also be obtained. So we can design a 2-D linear phase FIR filter by training the magnitude response of a desired 2-D FIR filter based on the NNA.

3.2 Design Example

Consider a circularly symmetric low-pass filter in [7], the designed 2-D magnitude response $M_d(\omega_1, \omega_2)$ is

$$M_d(\omega_1, \omega_2) = \begin{cases} 1 & , \quad 0 \leq \omega_g \leq \omega_p, \\ (\omega_a - \omega_g)/(\omega_a - \omega_p), & \omega_p \leq \omega_g \leq \omega_a, \\ 0 & , \quad \text{others,} \end{cases} \tag{31}$$

where $\omega_g = \sqrt{\omega_1^2 + \omega_2^2}$, $\omega_p = 0.425\pi$, $\omega_a = 0.575\pi$.

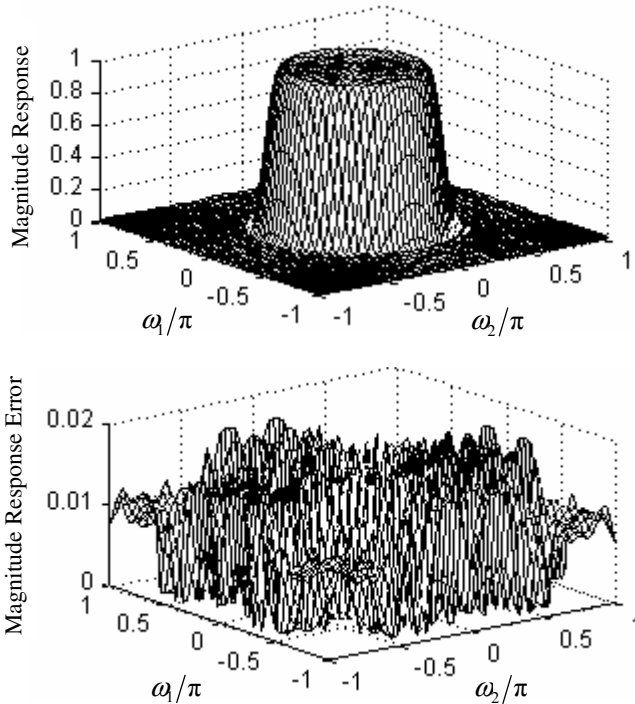


Fig. 4. Magnitude response of the designed circularly symmetric low-pass filter

Sample uniformly the desired amplitude response $M_d(\omega_1, \omega_2)$ to obtain 100×100 training samples in $\omega_1, \omega_2 \in [0, \pi]$, and let $(N_1, N_2) = (23, 23)$. First specify $\delta_p / \delta_s = 1$, and $\eta = 40$. After 1000 times training, the magnitude response of the design circularly symmetric FIR low-pass filter is shown in Fig. 4, and the actual maximum ripples of the pass-band and stop-band are 0.0168 and 0.0167, respectively. Comparing with the designed example 1 in [7], the maximum ripple is 0.0397 in the pass-band and 0.0578 in the stop-band based on semi-definite programming.

4 Conclusion

In this paper, we propose a method to design linear-phase quasi-equiripple FIR filters using neural network. A novel weight back-propagation neural networks algorithm is developed to control the overshoot phenomenon that may happen near the pass-band and stop-band edge of the designed filter. The design equations are given together with some guidelines. Several design examples are also given and the results show that the neural network method can easily achieve higher design accuracy than some conventional methods.

Acknowledgement

This work was supported by the National Natural Foundation of China under grant No. 50677014, the Doctoral Special Fund of Education Ministry of China under grant No. 20060532006, the Program for New Century Excellent Talents in University of China (NCET-04-0767), and the Foundation of Hunan Provincial Science and Technology of China under grant No. 06JJ2024, 03GKY3115, 04FJ2003, and 05GK2005.

References

1. Speake, T.C., Mersereau, R. M.: Note on the Use of Windows for Two-Dimensional Filter Design. *IEEE Tans. ASSP* **29** (1981) 125-127
2. Lai, X.P.: Constrained Chebyshev Design of FIR Filters. *IEEE Trans. Circuits Syst. II* **51** (2004) 143-146
3. Lee, W.R., Rehboch, V., Teo, K.L., Caccetta, L.: A Weight Least Square-Based Approach to FIR Filter Design Using the Frequency-Response Masking Technique. *IEEE Signal Processing Letters* **11** (2004) 593-596
4. Hsieh, C.H., Kuo, C.M., Jou, Y.D., Han, Y.L.: Design of Two-Dimensional FIR Digital Filters by A Two-Dimensional WLS Technique. *IEEE Trans. Circuits Syst. II* **44** (1997) 348-412
5. Zhu, W.P., Ahmad, M.O., Swamy, M.N.S.: A Least-Square Design Approach for 2-D FIR Filters with Arbitrary Frequency Response. *IEEE Trans. Circuits Syst. II* **46** (1999) 1027-1036
6. Zhu, W.P., Ahmad, M.O., Swamy, M.N.S.: Weighted Least-Square Design of FIR Filters Using A Fast Iterative Matrix Inversion Algorithm. *IEEE Trans. Circuits Syst. I* **49** (2002) 143-146
7. Lu, W. S.: A Unified Approach for the Design of 2-D Digital Filters via Semidefinite Programming. *IEEE Trans. Circuits Syst. I* **49** (2002) 814-825
8. Zahradnik, P., Vlček, M.: Fast Analytical Design Algorithms for FIR Notch Filters. *IEEE Trans. Circuits Syst. I* **51** (2004) 608-623

Texture Image Segmentation Based on Improved Wavelet Neural Network

Deng-Chao Feng^{1,2}, Zhao-Xuan Yang¹, and Xiao-Jun Qiao²

¹ Institute of Electronic & Information Engineering, Tianjin University,
Tianjin, 300072 P.R. China

² National Engineering Research Center for Information Technology in Agriculture, Beijing,
100089 P.R. China
Tyfdc001@163.com

Abstract. In this paper, a texture image segmentation algorithm based on improved wavelet neural network is proposed. This algorithm can overcome shortcomings of traditional threshold segmentation technologies. By using texture features of images, a series of fractal texture feature parameters which will be taken as input layer factors of wavelet network are created by this algorithm. Then, the wavelet neural network is trained with self-adaptive pheromone volatilization mechanism and dynamic heuristic search strategy of improved ant colony algorithm. Finally, the trained wavelet neural network is taken as the classifier of image pixel to realize segmentation of texture images. Simulation experiment shows that, improved algorithm could realize self-adaptive segmentation based on different texture features of images and it is robust. However, further researches on methods of improving convergence speed of this algorithm and objective criteria for assessing whether texture images have been segmented successfully or not are needed.

1 Introduction

As a key technology of image processing, the aim of image segmentation is to segment closed areas which are relatively complicated and abstract parts needed. Common segmentation methods mainly adopt threshold technology, which can achieve better effect for images with uniformly distributed gray, but often generates holes within and noise outside of the segmented area for most complicated images due to blurry boundary and severe gray overlapping between image background and the target area. Therefore, this results in great error.

By combining wavelet transform principle and artificial neural network theory, wavelet neural network constructs a new neural network model, which has strong approximation capability and fault-tolerant capability by virtue of good time-frequency localization characteristics of wavelet transform and self-learning function of neural network. As a new branch of neural network research, wavelet neural network has been applied in signal processing, data compression, pattern recognition, fault diagnosis and other research areas. In training process of wavelet neural, stochastic gradient method has relatively slow convergence speed and may result in

local minimum. In orthogonalization process with orthogonal least square method, overlarge weight of network could be resulted because of orthogonal vector with quite small amplitude that may appears [1]. Ant colony algorithm proposed by Italian researchers M.Dorigo, V.Maniezzo and A.Colorini is a kind of heuristic search algorithm based on bionics. It has features of positive feedback and parallelism and has been widely applied in combinatorial optimization [2]. On the basis of basic ant colony algorithm, the pheromone updating strategy, heuristic function and pheromone volatilization mechanism are revised and applied to parameter training of wavelet network in this paper. Meanwhile, fractal dimension eigenvector of image is constructed with fractal theory. Finally, image segmentation is achieved through using both of improved wavelet neural network and the fractal dimension eigenvector of texture image. Simulation experiments have proved the efficiency of this algorithm.

2 Design of Wavelet Neural Network

WNN (Wavelet Neural Network) combines wavelet analysis theory with artificial neural network theory. In this paper, we use compact wavelet neural structure. Hidden layer function in common neural network is replaced with wavelet function, and corresponding weight values from input layer to hidden layer and hidden threshold are replaced with scale parameter and translation parameter of wavelet function, respectively.

Researches on existence and structure of compactly supported orthonormal wavelet carried out by Daubechies and multiresolution analysis theory proposed by Mallat strongly guarantees the feasibility of orthonormal basis wavelet network of multiresolution analysis. Hidden layer nodes of wavelet network are composed of wavelet function node ψ and scale function node φ . The analysis process is shown as follows:

In wavelet network constructed by discrete wavelet function, if $f(x) \in L^2(R)$, to approximate $f(x)$ with resolution m can be described as

$$f_m(x) = C_m f(x) = c_{m,n} \varphi_{m,n}(x), \tag{1}$$

where, C_m is the orthogonal projection operator in space V_m , and $c_{m,n}$ is the weight of scale cell of corresponding network that is the projection of scale function $\varphi_{m,n}(x)$. Owing to the multiresolution analysis theory, W_m denotes the orthonormal complement space of V_m on V_{m-1} , that is, $V_{m-1} = V_m \oplus W_m$, $V_m \perp W_m$, where, D_m is the orthogonal projection operator in W_m . The approximation of $f(x)$ at $m-1$ scale is

$$f_{m-1} = [C_m \oplus D_m] f(x) = C_m f(x) \oplus D_m f(x), \tag{2}$$

$$D_m f(x) = \sum_{k=-\infty}^{+\infty} d_{m,n} \psi_{m,n}(x), \tag{3}$$

where, $d_{m,n}$, the projection of wavelet function $\psi_{m,n}(x)$ in corresponding network is called the detail on this resolution.

The approximation of $f(x)$ on resolution $m-1$ will be given by

$$f_{m-1}(x) = f_m(x) + \sum_{n=-\infty}^{+\infty} d_{m,n} \psi_{m,n}(x) = \sum_{n=-\infty}^{+\infty} c_{m,n} \varphi_{m,n}(x) + \sum_{n=-\infty}^{+\infty} d_{m,n} \psi_{m,n}(x). \quad (4)$$

When wavelet network to be constructed is a single hidden layer, the weight value from input layer to hidden layer is set to be 1. Scale function $\varphi_{m,n}(x)$ in hidden layer constructs the approximation to signal on the most rough resolution and wavelet function $\psi_{m,n}(x)$ constructs the gradual detail approximation to signal.

In this paper, we take Daubechies wavelet as the activation function of wavelet network. Scale function $\varphi_{m,n}(x)$ and wavelet function $\psi_{m,n}(x)$ are described as follows:

$$\varphi(t) = \sqrt{2} \sum_{k=0}^{N-1} h_n \varphi(2t - k), \quad (5)$$

$$\psi(t) = \sqrt{2} \sum_{k=0}^{N-1} g_n \psi(2t - k), \quad (6)$$

where, h_k and g_k are filter coefficients of corresponding orders and 4th order vanishing moments is taken where $N=8$.

In wavelet network learning algorithms, the stochastic gradient may easily make a network with local optimization and the least square method may result in parameter drift of the network. In this paper, by using characteristics of positive feedback, distributive computation and heuristic convergence of ant colony algorithm, we adopt improved ant colony algorithm to implement parameter training of the wavelet network.

3 Coupling Between Improved Ant Colony Algorithm and WNN

Ant colony algorithm is a kind of heuristic optimal algorithm with distributed parallel computing mechanism, and it has been applied in neural network optimization to some extent [4]. In this paper, problems of local minimum of wavelet neural network are effectively avoided through improving basic ant colony algorithm and training wavelet neural network with self-adaptive pheromone dynamic update strategy and heuristic learning algorithm [5].

Suppose there are n parameters in wavelet neural network including all scale function parameters and translation parameters of wavelet function, which are called weight value and threshold value respectively. Firstly, wavelet network parameters p_i ($1 \leq i \leq n$) are initialized to N random non-zero values to form the aggregate $I = \{p_i\}$. Each ant in ant colony selects one weight value in aggregate I so that a set of weight values of wavelet neural network are made in all aggregates. The number of ants is m and τ_j denotes the information amount of the j^{th} element p_i in aggregate I . Elements chosen by different ants are independent from each other during the process of ants search. Each ant starts from aggregate I and select one element according to information amount and state transition probability of each aggregate respectively. The ant could not arrive at food source until it completes selections in all aggregates. When the ant arrives at food source, information amounts of elements in aggregate I

are adjusted. The process is repeated till evolution trend is not apparent or the given iteration time is reached.

The procedure of learning algorithm of the improved wavelet network is described as follows. In ant colony initialization stage, both time t and circle time N_c are set to be zero, and the maximum of circle time N_{cmax} is decided. The information amount of each element in every aggregate is made where $\tau_j = C$ and $\Delta\tau_j = 0$. When all ants are in formicary, start up all ants. The state transition probability of ant k ($k=1,2,\dots,n$) in aggregate I could be obtained according to Eq.(7) and Eq.(8)[6]:

$$s = \arg \max_{u \in allowed_k} (\{[\tau(r,u)]^\alpha [\eta(r,u)]^\beta\}, \quad q \leq q_0, \tag{7}$$

$$p_{ij}^k(t) = \begin{cases} [\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta / \sum_{s \in allowed_k} [\tau_{is}(t)]^\alpha \cdot [\eta_{is}(t)]^\beta & j \in allowed_k, \\ 0 & otherwise. \end{cases} \quad q \geq q_0, \tag{8}$$

where, q and q_0 are uniformly distributed random number and the parameter within the range of $[0,1]$ respectively, p_{ij}^k presents the probability of ant at node i to move to node j , $allowed_k$ denotes the allowed node which ant k can select at next step, and η_{ij} and τ_{ij} are visibility factor and pheromone amount remained at position (i,j) , respectively.

When all ants arrive at food source, we replace t and N_c with $t+n$ and N_c+1 , respectively. According to weight values chosen by each ant, the output value and error of wavelet network are gained, and the current optimal solution is obtained. After n time units, ants reach food source from the formicary, and the information amount on every route is updated according to following equations:

$$\tau(r,s) \leftarrow (1 - \rho(t)) \cdot \tau(r,s) + \rho(t) \cdot (\Delta\tau(r,s) + r \cdot \max_{u \in allowed_k} \tau(s,u)), \tag{9}$$

$$\Delta\tau_{ij} = \sum_{k=1}^m \Delta\tau_{ij}^k(t) = \sum_{k=1}^m Q / e_k = \sum_{k=1}^m Q / |O - O_F|_k, \tag{10}$$

where, e_k presents the output error gained by taking a set of weight values chosen by the k^{th} ant as the weight value of wavelet network, and O and O_F denote the actual output and the expected output respectively. For pheromone volatilization factor ρ , when the optimal value obtained by ant colony algorithm is not obviously improved in N cycles, we adopt $\rho(t)$ to achieve self-adaptive adjustment, where, ρ_{min} is the minimum of ρ , which could avoid decreasing the convergence speed of algorithm resulted from a too small value of ρ .

$$\rho(t) = \begin{cases} 0.95\rho(t-1), & \text{when } 0.95\rho(t-1) \geq \rho_{min}, \\ \rho_{min} & \text{otherwise.} \end{cases} \tag{11}$$

When all ants converge to the same route or cycle time N_c is not less than N_{cmax} , the cycle is finished and the computation result is outputted, or else, the computation of state transition probability is continued.

To avoid the local optimization, we limit the maximum and minimum of the information amount at every route stage, that is, $\forall \tau_{ij}(t), \exists$

$\tau_{\min} \leq \tau_{ij} \leq \tau_{\max}, (i, j \in [1, \dots, n])$. Moreover, in order to improve the global search ability of ant colony and increase its search speed, tabu table is adopted to record the optimal route matrix, from which the weight matrix and coefficient matrix of wavelet neural network are taken subsequently. The optimal value within one cycle is gained and conserved. And in the next cycle, the optimization is continued according to features of the conserved optimal solution. Therefore, during the evolution process, as a result of diffusion of pheromone, this algorithm has the ability of continuously obtaining the optimal solution.

4 Segmentation Algorithm Based on Improved WNN

In the texture image segmentation process, feature abstraction and segmentation algorithm are two most important parts of the research. As fractal dimension can be taken as an image roughness measurement which is insensitive to scale [7], and it could combine spatial information with gray information of image simply and organically, therefore, in this paper, according to self-similarity of image texture, we adopt box-counting method to compute fractal dimension [8] and construct a set of fractal texture eigenvalues of image. Then self-adaptive segmentation of image is realized through taking improved wavelet neural network as the classifier of image pixel.

Let F be a randomly nonempty bounded subset of R^n . $N_\delta(F)$ is the least number of subsets, whose biggest diameter is δ and which could cover F . The box dimension is described as

$$Dim(F) = \lim_{\delta \rightarrow 0} \log N_\delta(F) / (-\log \delta). \quad (12)$$

Different textures could probably correspond to the same fractal dimension because fractal dimensions of images usually exist within the range of [2.0, 3.0]. To reach a better segmentation result, on the basis of fractal dimension of origin image, we define a set of fractal dimensions $Dim(F1)$, $Dim(F2)$, $Dim(F3)$, $Dim(F4)$ and $Dim(F5)$ based on transforming image and combine them into one eigenvector to segment image, where, fractal dimension (F1) of the original image is selected as the first feature and fractal dimensions of gradient images ($0^\circ, 45^\circ, 90^\circ, 135^\circ$) of texture image in four directions are selected as the last four features ($Dim(F2)$, $Dim(F3)$, $Dim(F4)$ and $Dim(F5)$).

Owing to directivity of texture image, direction gradient is helpful to distinguish textures in different directions. In this paper, Kirsch template is adopted. By convolution operation of template and the original image, the corresponding gradient image is formed, and each fractal dimension is obtained with box-counting method.

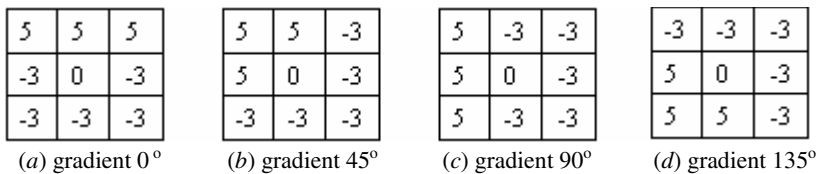


Fig. 1. Kirsch template

For a $N \times N$ image, gradient images in four directions are formed with Kirsch template in Fig. 1. From the start point of original image, the $M \times M$ window slides in horizontal and vertical direction, respectively. Fractal dimension of image within the window is computed and taken as the fractal eigenvalue of image at the center of the window. On selection of the size of sliding window, if the window is too large, the computation speed and precision of straight line fitting would be affected. If it is too small, statistic characteristics of images could not be accurately reflected. In this experiment, the size of the window is 9×9 . By using of the above algorithm, the local fractal dimension of each point in the inputted grey image is obtained, and the fractal dimension distribution image is gained.

In the segmentation process of texture image based on wavelet neural network, the network has 3 layers. The input layer is divided into 5 units, which correspond to a set of fractal dimensions ($\text{Dim}(F1)$, $\text{Dim}(F2)$, $\text{Dim}(F3)$, $\text{Dim}(F4)$ and $\text{Dim}(F5)$), respectively. The output value of output layer with 1 unit is 1 or 0, which corresponds to the background and the target, respectively. The activation function of wavelet network adopts Daubechies wavelet ($N=8$), which is 4th order vanishing moments. When computing fractal dimension, line-by-line scanning of image is done by sliding window and fractal eigenvalue is taken as input of the wavelet neural network. The segmentation algorithm is divided into two parts of training stage and testing stage. Learning sample is composed of pixels randomly abstracted from sample image, and the eigenvalue of each sample is computed inputted into wavelet neural network and trained with improved ant colony algorithm. In testing stage, image pixels are classified according to optimized network parameters. Finally, the segmented image is obtained by specifying a kind of color for each class.

5 Result and Analysis of Simulation Experiment

In simulation experiment, we take standard $512 \times 512 \times 3$ image with gray level of $[0, 255]$ as the testing image shown in Fig. 2.(a_0) under the circumstance of MATLAB7.0. To verify the robustness of this algorithm, we add Gauss white noise with 0 mean and 0.001 variance into the original image like shown in Fig. 2.(b_0).

In the improved wavelet network, parameters of ant colony algorithm are set as $\alpha=1, \beta=2, q_0=0.6, N=20, N_{\max}=2000, \rho_{\min}=0.1, \tau_{\min}=0.001$ and $\tau_{\max}=0.001$. Daubechies wavelet function DB8 is taken as the activation function of network and the size of the sliding window is 9×9 for computation of fractal dimension of image at the input end of the network. After 2380 iterations, error less than 0.001 is obtained. To prove the efficiency of this algorithm, wavelet networks with gradient descend learning algorithm and *OSTU* algorithm are taken to compare with this proposed algorithm, where, in wavelet network formed by gradient descend learning algorithm, learning rate and momentum coefficient are set as 0.1 and 0.08 respectively, and in *OSTU* algorithm, the graythresh function is adopted to calculate the global threshold. Simulation results are shown in Fig. 2, where, a_0 is the original image; b_0 is the image with noise; (a_i) and (b_i) ($i=1,2,3$) are segmentation results of a_0 and b_0 generated by *OSTU* algorithm, gradient descend learning algorithm and the improved WNN, respectively.

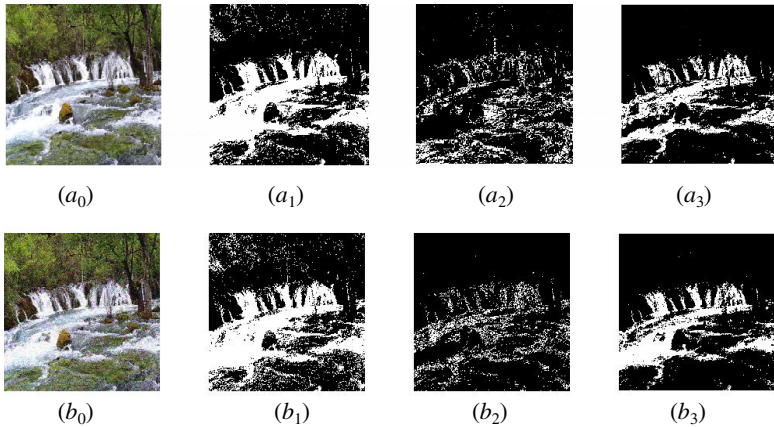


Fig. 2. Comparison of Image Segmentation Results

In the experiment, the eigenvector at network input end is the fractal dimension generated when the window sliding over the original image and its four gradient images, where, the fractal dimension of corresponding target area and that of other areas are shown in Tab. 1, in which D_1 is the window image with target area and D_2 and D_3 are fractal dimensions of two window images in other areas.

Table 1. Fractal Dimensions of Some Areas in the Original Image

Image	Dim(F1)	Dim(F2)	Dim(F3)	Dim(F4)	Dim(F5)
D_1	2.0552	1.2107	1.2115	1.1124	1.1258
D_2	2.5861	1.4123	1.4352	1.5124	1.4258
D_3	2.6034	1.5429	1.5817	1.6124	1.4121

On visual expression, some discrete spots appear within the even area when using *OSTU* algorithm to segment image; and with gradient descend learning algorithm, the even area is continuous but more misjudgements would exist for edges of areas. However, this algorithm put forwarded in this paper could not only achieve the continuousness with in the even area but also increase the judgment accuracy for points near the edge of the area.

On evaluation criterion for image segmentation, error rate is taken to evaluate performance of this algorithm. By comparing results in Fig. 2 (a_0) with that generated by all methods we have taken, error rates are obtained and shown in Tab. 2.

Table 2. Comparison of Error Rates

Method	OSTU	WNN	Improved WNN
Err(%)	15.127	11.023	6.271
Err(%) (with noise)	18.214	12.926	7.512

According to comparison of these three methods, we can see that image segmentation done by the algorithm proposed in this paper is more accurate, which decreases misjudgement of inner points through smoothing inner part of the area and increases judgment accuracy for points near the edge of the area by greatly reserving edge of the area.

Evaluation for image segmentation is very important for improving performance and segmentation quality of the existing algorithm. Different evaluation criteria often reflect different aspects (usually one aspect) of segmentation algorithms. Therefore, several evaluation criteria are usually used in combination for a comprehensive evaluation. Thus, we will carry out further research on how to combine all kinds of evaluation criteria together and realize objective evaluation for image segmentation performance.

6 Conclusion

Image segmentation is a key step from image processing to image analysis. It is the basis of object expression and has great influence on feature measurement. According to self-comparability of image texture, we adopt the fractal theory to obtain fractal dimensions of segmented areas, and segment the image by coupling strategy of the improved ant colony and wavelet neural network afterwards. Simulation experiment results show that this method is more reliable and accurate, and it could greatly segment texture images. However, due to computation of fractal dimensions, segmentation procedure based on texture information takes much longer time. Therefore, further researches on how to improve the selection of input parameter eigenvalues of network, increase convergence speed of network and makeup criteria for objective evaluation on segmentation results are needed.

References

1. Zhang, H., Wu, B.: Research and Prospects of Wavelet Neural Networks. *Journal of Southwest China Institute of Technology* **17** (1) (2002) 10–12
2. Dorigo, M., Maniezzo, V., Coloni, A.: The Ant system: Optimization by A Colony of Cooperating Agents. *IEEE Trans On System, Man and Cybernetics-Part B* **26** (1) (1996) 1-13
3. Feng, D.-C., Yang, Z.-X., Qiao, X.-J.: The Application of Wavelet Neural Network with Orthonormal Bases in Digital Image Denoising. In: Wang, J., et al.(eds.): *ISNN2006. Lecture Notes in Computer Science*, **3972**. Springer-Verlag, Berlin Heidelberg New York (2006) 539-544
4. Hong, B.-R., Jin, F.-H., Gao, Q.-J.: Multi-layer Feedforward Neural Network Based on Ant Colony System. *Journal of Harbin Institute of Technology* **35** (7) (2003) 823-825
5. Liu, J.-C., Wang, Z.-O.: Fast Learning Algorithm of Wavelet Neural Networks and Its Application. *Journal of Tianjin University* **34** (4) (2001) 455-457
6. Duan, H.-B.: *Ant Colony Algorithms: Theory and Applications*. 1st edn. Science Publishing House (2005)
7. He, Z.-Y., Bao, K., Dong, H., He, S.-C.: Texture Image Segmentation Based on the Fractal Dimension. *Journal of Data Acquisition & Processing* **11** (3) (1996)163-165
8. Sun, X., Wu, Z.-Q., Huang, Yun.: *Fractal Theory and Application*. University of Science and Technology of China Publishing House (2003)

Local Spatial Properties Based Image Interpolation Using Neural Network

Liyong Ma, Yi Shen, and Jiachen Ma

School of Information Science and Engineering,
Harbin Institute of Technology at Weihai, Weihai 264209, P.R. China
hitmaly@yahoo.com.cn, shen@hit.edu.cn, hitmjc@sohu.com

Abstract. A neural network based interpolation scheme using the local spatial properties of the source image for image enlargement is proposed. The local spatial properties that are used for neural network training include the neighbor pixels gray values, the average value and the gray value variations between neighbor pixels in the selected region. Gaussian radial basis function neural network is used for image local spatial properties pattern learning and regression estimation for image interpolation. The trained neural network is used to estimate the gray values of unknown pixels using the known neighbor pixels and local spatial properties information. Some interpolation experiments demonstrate that the proposed approach is superior to linear, cubic and other neural network and support vector machines based interpolation approaches.

1 Introduction

In recent years there has been considerable interest in image interpolation. Image interpolation has a wide range of applications in image processing fields. The well-known approaches to image interpolation are linear interpolation and cubic interpolation. However these methods blur images particularly in edge regions [1] [2]. Some learning based image interpolation approaches have been developed to get better result images recently. Neural network is powerful to solve nonlinear mappings problems and has been used for image interpolation in [3] where a multi-layer perceptron neural network based interpolation approach is proposed. As powerful machine learning tools, support vector machines (SVMs) are also employed for image interpolation. A support vector machines based image interpolation approach was presented in [4] where the gray value of the pixel to be interpolated was estimated with SVMs that were trained with the gray values of neighbor pixels. Error correction schemes employing learning based approaches for image interpolation were also reported recently. A support vector machines based error correction scheme was developed in [5], and a more efficient error correction algorithm using neural network for linear interpolation is proposed in [6].

Most of these learning based approaches employ only neighbor pixels gray values and their corresponding coordinates. And other local spatial properties of the source image are not used for the interpolation. In this paper a neural network based image interpolation approach in which more local spatial properties are

used as input vector for neural network is proposed. The experimental results showed that the proposed approach can produce higher quality result images than linear interpolation, cubic interpolation approach and other learning based interpolation approaches.

2 Gaussian RBF Neural Network

Line regression and logistic regression are the standard statistical approaches to establish mathematical relationship between the independent variables and the final decision. They often require prior knowledge about the relationship patterns. Neural network is efficient for patterns detection where the relationships can not be easily expressed in a mathematical form. Neurons in neural network are interconnected to receive signals in input layer and produce output in output layer. Firstly the network structure is decided. Then a learning algorithm is employed for the network training with sample vectors. In this training process network parameters are modified to minimum the error between the actual output and the desired output. Finally the trained neural network can be used to estimate the output for any vector from the input space.

Radial basis function (RBF) neural network is an important neural network architecture with many important applications. The RBF neural network has the universal approximation ability, so it can be used for the interpolation problem [7]. A Gaussian radial basis function is highly nonlinear and powerful for learning complex input-output mapping. A typical RBF neural network includes an input layer, a single hidden layer that is called radial basis layer for non-linear processing, and a output linear layer [7]. The output of RBF neural network is

$$f(x) = \sum_{i=1}^n w_i \phi_i(\|\mathbf{x} - \mathbf{c}_i\|), \quad (1)$$

where \mathbf{x} is input vector, $\phi_i(\cdot)$ denotes the processing function of the i -th node in the hidden layer, $\|\cdot\|$ denotes the Euclidean norm, w_i are weights between i -th node in the hidden layer and output node, n is the total number of neurons in the hidden layer, and \mathbf{c}_i are the RBF centers in the input vector space. Euclidean distances that are the distances between the input vector and the input weight matrix for each neuron in the hidden layer are calculated, and a nonlinear function of the distance is obtained in the hidden layer outputs. The output of the neural network is a weighted sum of the hidden layer outputs.

Gaussian function is often employed as the radial basis process function. Gaussian radial basis function neural network has the process function given as

$$\phi(r) = \exp\left(\frac{-r^2}{S^2}\right). \quad (2)$$

Spread parameter S is used in the radial basis layer to set each bias of this layer to $0.8326/S$. The spread parameter S can determine the area width for each neuron responds in the input space. So the spread S needs to be selected

correctly to overlap the regions of the input space. Gaussian RBF network is established as a three-stage network, that is input stage, intermediate stage of Gaussian units and output stage of conventional summation units. Gaussian RBF network is illustrated in Fig 1. In this paper Gaussian RBF network is used for local spatial properties of the source image learning and interpolation estimation.

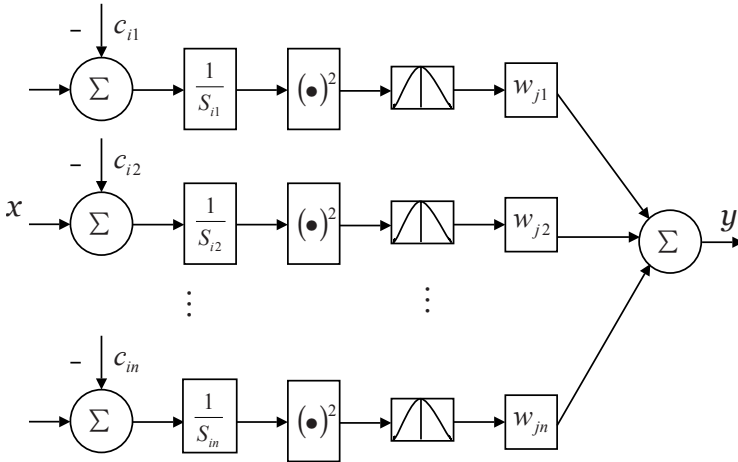


Fig. 1. Gaussian RBF network

3 Interpolation

3.1 Linear and Cubic Interpolation

Let x and $f(x_k)$ denote the coordinate value to be interpolated and available data respectively. Assume that x_k and x_{k+1} are nearest available neighbors of x . Then the distance between x and neighbors can be defined as

$$s = x - x_k, \quad 1 - s = x_{k+1} - x \quad (0 \leq s \leq 1). \tag{3}$$

We have one-dimensional linear interpolation of x

$$\hat{f}(x) = (1 - s)f(x_k) + sf(x_{k+1}). \tag{4}$$

Similarly, we have one-dimensional cubic interpolation of x

$$\begin{aligned} \hat{f}(x) = [& f(x_{k-1})((3 + s)^3 - 4(2 + s)^3 + 6(1 + s)^3 - 4s^3) \\ & + f(x_k)((2 + s)^3 - 4(1 + s)^3 + 6s^3) \\ & + f(x_{k+1})((1 + s)^3 - 4s^3) + f(x_{k+2})s^3] / 6. \end{aligned} \tag{5}$$

Applying above two equators to image along the rows then columns we can calculate two-dimensional bilinear or bicubic interpolation.

3.2 Learning Based Interpolation

We can regard the position and the gray value of a pixel in a digital image that is two dimensions as the coordinates value and the corresponding curve surface value. So we can perform interpolation by data fitting approach employing SVMs as proposed in [4]. Let q denote the pixel to be interpolated. We can select the nearest known pixels around q and the number of these neighbor pixels is denoted as $m \times m$. Then SVMs are trained by employing relative coordinates value of the neighbor pixels around q as input pattern and the gray value of the neighbor pixels as output pattern. The neighbor area size can be selected as 4×4 . After training the pixel gray value of pixel q can be estimated with trained SVMs. During the estimation period the input pattern of SVMs is the relative coordinates value of q and the output pattern the estimated interpolation value. We call this approach as SVMs based data fitting (SVMDF) interpolation approach.

Another learning based interpolation approach is multi-layer perception neural network based neighbor pixels (MLPNP) approach [3]. Every pixel in the known source image is used as sample for neural network training. And the gray value of every pixel and the gray values of the neighbor pixels are employed as output vector and input vector of samples during the training process. In this approach the coordinates of pixels are not used at all. Then a multi-layer perception network is employed for image patterns training. After the training, interpolation result image can be obtained with every pixel to be interpolated is estimated by trained neural network. During the estimation process the gray values of neighbor pixels around the pixel to be interpolated are employed as an input vector. And the output of the neural network is the gray value of the pixel to be estimated.

4 Proposed Interpolation Approach

Little information about the local spatial properties of the pixel to be interpolated is used in the above learning based approaches. The interpolation results are related to some critical local spatial properties of the pixel to be interpolated, such as the smooth properties of the region and the directional information of the edges around the pixel. We expect to obtain high quality interpolation result image by using more local spatial information of the source image for neural network training and estimation. Our proposed interpolation scheme employs the average value in the region around the pixel to be interpolated as one of the properties in the input vector. And the gray value differences of various direction around this pixel are also used as properties of input vector.

Row expansion and column expansion of a source image are often performed for image enlargement. Now we consider the situation that the source image is requested to perform row double expansion. Other generalized interpolation with the magnification times is integer can be performed as row expansion firstly and then column expansion. And it is easier to apply the proposed approach to source image for other expansion magnification times is greater than 2. Our proposed interpolation scheme is described as follows.

Firstly a region around the pixel to be interpolated needs to be decided before the interpolation calculation. The regions selected for double row expanding is illustrated in Fig. 2. As illustrated in the figure, the gray values of six known pixels in the selected region are denoted as $p(i)$, where $i = 1, 2, \dots, 6$.

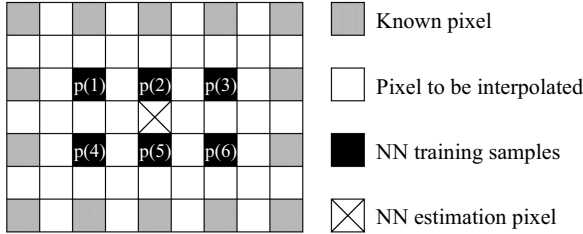


Fig. 2. Region for proposed interpolation approach

Secondly the neural network training is performed. Gaussian RBF network is used for training and estimation. Two methods can be used for network architecture decision. One is to initialize the neurons in the RBF layer as zero, then repeat the neurons increase steps until the mean squared error of the network is under the given value or the maximum neurons is reached. The steps for neurons increase is to find the input vector with the greatest error during the simulation firstly, then a neuron is added with its weights equal to the input vector, and linear layer is redesigned to minimize error. Another method is to establish a zero error RBF network quickly. This can be achieved with setting the first layer weights according to input vectors, and the first layer biases are all set to $0.8326/S$. The latter method is used in our proposed interpolation approach.

During the training period every pixel in the source image is employed as the samples of neural network. The input vector include the pixel gray values of the neighbor known pixels in the selected region. Other properties of the input vector include average gray value of neighbor known pixels and some other local spatial properties in the selected region. The average gray value of neighbor known pixels in the selected region is calculated as

$$\bar{v} = \sum_{i=1}^6 p(i). \tag{6}$$

Other local spatial properties employed as input are calculated as

$$\begin{aligned} v_1 &= p(1) - p(3) \\ v_2 &= p(4) - p(6) \\ v_3 &= p(1) - p(4) \\ v_4 &= p(2) - p(5) \\ v_5 &= p(3) - p(6) \\ v_6 &= p(1) - p(6) \\ v_7 &= p(3) - p(4) \end{aligned} \tag{7}$$

Then a 14 dimensions vector is employed for neural network input. And the output is the gray value of the central pixel to be estimated. After the training, the trained neural network can be employed to estimate unknown gray values of the pixels to be interpolated.

In this interpolation approach, spread S can be selected by optimal search. In the search process the source image is reduced to get a middle image with half size of the source image width and height. Then this middle image can be regarded as a new source image, then different S can be applied to the proposed approach for interpolation to decide the optimal parameter values. In this procedure all the interpolation result images are known, so we can use these known result images to compare interpolation results and judge the optimal parameters. After these parameters are selected, these optimal parameters can be used in the proposed approach to perform interpolation to get final unknown interpolation result.

This approach can be easily generalized to other interpolations where the magnification times is integer.

5 Experimental Results

Row and column expansion [3] is used for interpolation experiments. Some standard test images that have been widely used in other literatures are used in our experiments. These test images include Cameraman, Lena, etc. The test images are downsampled with row reduction, column reduction and row and column reduction with different scales. Then these reduced images are enlarged to obtain result images with various interpolation approaches to perform row expanding, column expanding and row and column expanding. Result images are compared with the original images to calculate mean square error (MSE) and peak signal to noise ratio (PSNR).

Table 1. Interpolation Results of Image Cameraman

Interpolation	MSE	PSNR
Linear	313.99	23.162
Cubic	223.66	24.635
MLPNP	334.06	22.893
SV MDF	333.92	22.894
Proposed	193.70	25.259

Table 2. Interpolation Results of Image Lena

Interpolation	MSE	PSNR
Linear	195.46	25.220
Cubic	156.02	26.199
MLPNP	291.98	23.477
SV MDF	269.05	23.832
Proposed	122.50	27.250

The interpolation approaches tested in our experiments are linear interpolation, cubic interpolation, MLPNP, SVMDF and our proposed local spatial properties based neural network interpolation. The experiments are performed in Matlab. All the neural network training is performed in the reduced image that is obtained by reducing the original source image.

Some result data of the 2 times row expanding to image Cameraman and Lena are listed in Table 1 and Table 2. The best spread parameter S is searched for optimal result images, and the result is that $S = 23$ for image Cameraman, $S = 35$ for image Lena. It is obvious that our proposed local spatial properties based neural network interpolation scheme gets the least MSE value and the highest PSNR value. It means that the proposed scheme obtains the best result image.

6 Conclusion

A novel local spatial properties based image interpolation approach using neural network is proposed. The proposed approach can be used to interpolation applications, such as enlarge image where the magnification times is integer. And some experiments indicate the effectiveness of the scheme. Experimental results also showed that the proposed approach is superior to linear interpolation, cubic interpolation, SVMs based interpolation and multi-layer perceptron neural network interpolation approaches.

References

1. Thevenaz, P., Blu, T., Unser, M.: Interpolation Revisited. *IEEE Trans. Medical Imaging*. **19** (2000) 739-758
2. Ma, L.Y., Ma, J.C., Shen, Y.: Local Activity Levels Guided Adaptive Scan Conversion Algorithm. In: *Proceedings of the 27th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. (2005) 6718-6720
3. Plaziac, N: Image Interpolation Using Neural Networks. *IEEE Trans. Image Process*. **8** (1999) 1647-1651.
4. Wang, J., Ji, L.: Image Interpolation and Error Concealment Scheme Based on Support Vector Machine. *Journal of Image and Graphics*. **7** (2002) 558-564
5. Ma, L.Y., Ma, J.C., Shen, Y.: Support Vector Machines Based Image Interpolation Correction Scheme. In: Wang, G., Peters, J., Skowron, A., Yao, Y. (eds.): *Rough Sets and Knowledge Technology. Lecture Notes in Artificial Intelligence*, Springer-Verlag, Berlin Heidelberg New York **4062**(2006) 679-684
6. Ma, L.Y., Shen, Y., Ma, J.C.: Neural Network Based Correction Scheme for Image Interpolation. In: Liu, D., Fei, S., Hou, Z.-G., Zhang, H. (eds.): *Advanced in Neural Networks - ISSN 2007. Lecture Notes in Computer Science*. Springer-Verlag, Berlin Heidelberg New York (2007) (In Press)
7. Gupta, M., Jin, L., Homma, N.: *Static and Dynamic Neural Networks*. (John Wiley, New Jersey 2003)

Image Processing Applications with a PCNN

Mario I. Chacon M., Alejandro Zimmerman S., and Pablo Rivas P.

DSP & Vision Lab, Chihuahua Institute of Technology, Mexico
mchacon@itchihuahua.edu.mx,
pablorp80@ieee.org

Abstract. This paper illustrates the potentials of the PCNN for image processing. A description of three schemes for image processing using the PCNN is presented in this paper. The first scheme is related to image segmentation, the second to automatic target location, ATL, and the third to face recognition. The first scheme was developed in order to obtain an insight of the behavior of the PCNN as a preprocessor element, the second one is an application to test the performance of the PCNN in an ATL problem. The third is a feature extraction method for face recognition. The segmentation scheme showed great potentials to perform pixel grouping. The second scheme turned into a system with an ATL performance as good as other systems reported in the literature. And the third scheme seems to improve the performance of a face recognition system.

1 Introduction

The Pulse Coupled Neural Network, PCNN, is a relative new ANN model with a great potential in the area of image processing. A PCNN, is a model derived from a neural mammal model [1]-[4]. Current works with PCNN document how the PCNN can be used to perform important image processing tasks; edge detection, segmentation, feature extraction, and image filtering [2]-[10]. Because of this kind of performance the PCNN is considered a good preprocessing element.

The basic model of a neuron element of a PCNN has three main modules: the dendrite tree, the linking and the pulse generator [1]. The dendrite tree includes two special regions of the neuron element, the linking and the feeding. Neighborhood information is incorporated through the linking. The input signal information is obtained through the feeding. The pulse generator module compares the internal activity, linking plus feeding activity, with a dynamic threshold to decide if the neuron element fires or not. Fig.1 illustrates the basic model of the PCNN. A PCNN mathematical definition is given by (1) to (5). Equation (1) corresponds to the feeding region of the neural element, where G_{Feed} is the feed gain, S is the input image, $\alpha_{F\Delta t}$ is the time constant of the leakage filter of the feeding region, $Y(t)$ is the neuron output at time t , and W is the feeding kernel. The outputs $Y(t)$ of the PCNN can be observed as output images called pulsed images of the PCNN. Equation (2) describes the linking activity. Here G_{Link} is the linking gain, $\alpha_{L\Delta t}$ is the time constant of the leakage filter of the linking region, and M is the linking kernel. Equation (3) corresponds to the

internal activity of the neuron element. The internal activity depends on the linking and feeding activity. In (3) β is the linking coefficient. β defines the amount of modulation of the feeding due to the linking activity. The dynamic threshold is implemented by (4), where α_θ is the time constant of the leakage filter of the threshold and V is the threshold gain. Finally the output of the neuron is defined by (5). In the case of an image processing task, each pixel is related to a neural element. For more information on how a PCNN works consult [1].

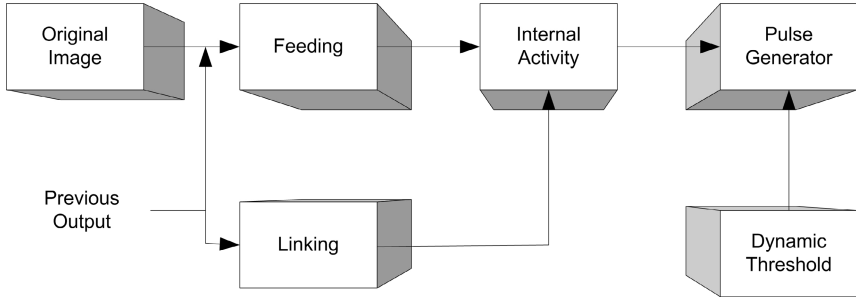


Fig. 1. Basic model of the PCNN

$$F(t) = G_{Feed} e^{-\alpha_F \Delta t} F(t-1) + S + Y(t-1) * W, \tag{1}$$

$$L(t) = G_{Link} e^{-\alpha_L \Delta t} L(t-1) + Y(t-1) * M, \tag{2}$$

$$U(t) = F(t) [1 + \beta L(t)], \tag{3}$$

$$\theta(t) = e^{-\frac{1}{\alpha_\theta} \Delta t} \theta(t-1) + VY(t), \tag{4}$$

$$Y(t) = \begin{cases} 1, & \text{if } U(t) > \theta(t), \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

2 Segmentation Experiment with the PCNN

One important characteristic of the PCNN is the grouping property. Since the fire of a neuron element depends on the input information and the neighbor information, it is possible to make that a set of neurons fire at the same time. This situation occurs when the neurons are related to image pixels corresponding to a uniform region. Taking advantage of this property we use the PCNN to perform segmentation. Fig 2 illustrates the pulsed images of the PCNN. In this application a noisy image, composed of

a black square and noise, was processed by a PCNN using low pass filters as the feeding and linking kernels and using the basic PCNN model. In this case low pass filter kernels are used since the purpose is to emphasize the ability to generate uniform regions, as observed in pulse 4. It can be observed in pulsation 4 that the object is practically defined.

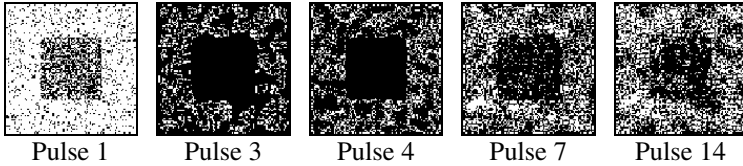


Fig. 2. Noisy image processed by the PCNN for segmentation purpose

3 PCNN Automatic License Plate Location Scheme

This scheme shows the application of the PCNN in automatic target location, ATL. The problem to solve is to locate license plates on digital images using a scheme based on a dynamic PCNN network. The images tested with the proposed ATL system are taken without special conditions, which is a difference with other works reported in the literature [11]-[13]. Not special conditions means; images acquired with a video camera, no special illumination equipment, and there is not restriction about the position of the license plate. The purpose of not restrict the acquisition stage is to evaluate the PCNN system with similar conditions where a person will succeed.

The dynamic PCNN ATL model that we proposed is an iterative model based on the PCNN, Fig. 3. The original image is preprocessed by the PCNN. The PCNN pulses and generates an output image. The PCNN is designed to yield regions [6], such that they may contain candidate regions of the license plate. A whole license plate segmentation is guaranteed because of the region constitution of the license plate. These regions are then analyzed to obtain labeled regions, $r_n(x, y)$. These candidate regions are analyzed using area, and the minimum rectangle features to keep only regions that are good candidates, $r_{ng}(x, y)$. After this process, statistics of the Fourier transform of the candidate regions are computed. These statistics are then used to decide if the region contains or not a license plate. If any of the regions contains the license plate then the process iterates to the point where the PCNN pulses again. In this new iteration the parameters of the PCNN are redefined, making a dynamic process. The objective of the change of parameters is to force the PCNN to generate smaller regions than the ones generated in the previous step. In the first iteration the parameters are adjusted to generate big regions. The parameters are adjusted after the second iteration to yield smaller regions. The initial parameters of the PCNN and the updated parameters are shown in Table 1. The kernels W and M are 3x3 average kernels, because average kernels reinforce grouping. At the time of the realization of this work it was not known an analytic procedure to determine the parameter changes, thus the parameters were estimated by experimentation using a PCNN processor software [9], and based on the knowledge of the PCNN behavior.

Table 1. PCNN parameters

First iteration	Next iterations
$\beta = 1.0$	$\beta = 1.0$
$G_{Feed} = 0.2$	$G_{Feed} = 0.7$
$\alpha_F = -0.1$	$\alpha_F = -0.1$
$G_{Link} = 0.1$	$G_{Link} = 0.6$
$\alpha_L = -0.1$	$\alpha_L = -0.1$
$\alpha_\theta = 0.75$	$\alpha_\theta = 0.75$
$V = 10$	$V = 10$

A first discrimination process of regions is used to discard regions like point, lines, etc. Discrimination of regions is performed by (6).

$$f(A, M_r) : r_n(x, y) \rightarrow r_{ng}(x, y), \tag{6}$$

where A is the area of the region and M_r is the minimum rectangle. Equation (6) analyzes the possibility of a $r_n(x, y)$ to be a $r_{ng}(x, y)$ based on the features area and minimum rectangle. Each region $r_{ng}(x, y)$ is then represented by a feature vector consisting of statistics of the Fourier transform of the region as follows

$$R_{ng}(u, v) \rightarrow \mathbf{T} = \begin{bmatrix} \mu \\ \sigma \end{bmatrix}, \tag{7}$$

where μ , and σ , are the mean and standard deviation, of $R_{ng}(u, v)$. The representative vectors for each class are, region with plate, \mathbf{T}_{wp} , region without plate, \mathbf{T}_{np} . \mathbf{T}_{wp} and \mathbf{T}_{np} were determined by statistical analysis. Under this scheme if the condition $\|\mathbf{T}_{wp} - \mathbf{T}_i\| > \|\mathbf{T}_{np} - \mathbf{T}_i\|$ is true for all regions $R_{ng_i}(u, v)$, then the pulsed image does not have a region with the license plate and the process iterates to the point where the PCNN pulses again. Fig 4 shows an example of a car image and the license plate found.

3.1 Results for License Plate Location

The segmentation scheme yield adequate results when the PCNN is used with low pass filter kernels.

High pass filters kernels were also implemented to achieve segmentation by edges; however, results were not satisfactory. In the case of the ATL scheme, the proposed system was tested with a database of 60 images acquired with a video camera without

special conditions of illumination. Results yield an 85% of correct license plate location. Other systems to locate license plate report results from 80% to 95%, of correct location under special conditions [14]-[16]. Based on these results it can be said that the PCNN architecture may provide important advantages in the preprocessing stage of images.

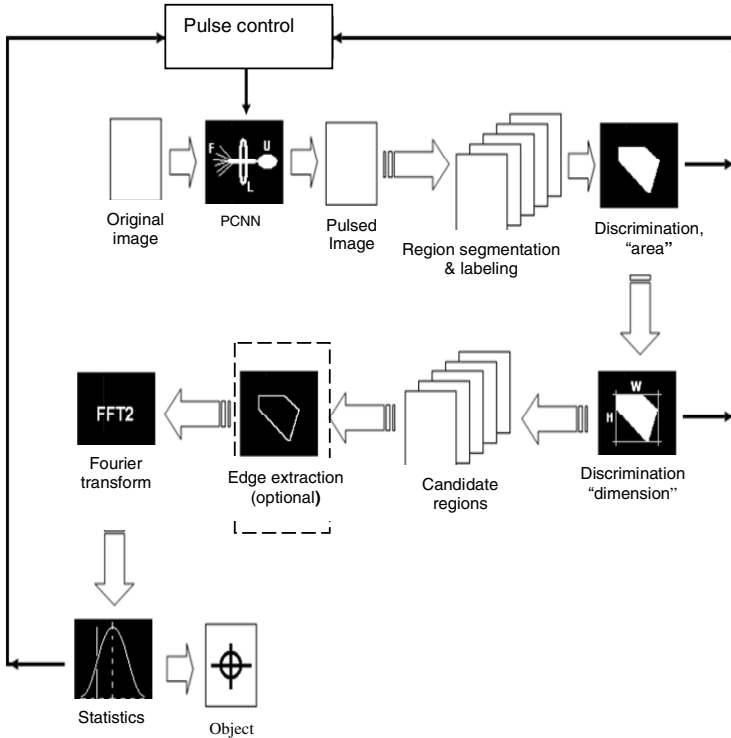


Fig. 3. Dynamic PCNN ATL model



Fig. 4. Original image and the plate located in the image

4 PCNN in the Face Recognition Problem

The problem related to face recognition has been investigated from different points of view. Artificial Neural Networks, ANN, is not the exception. In this section we present a novel approach for face recognition based on three features, first the Pulse Coupled Neural Network, PCNN; second, the Hough transform, HT; third, the Karhunen-Loeve transformation. The facial features are introduced in a neurofuzzy system based on the fuzzyfication of the inputs on an RBF neural network with a variable architecture on the first layer. The system performs well with ORL and YALE face databases, reaching recognition rates comparable with current face recognition systems.

The automatic face recognition systems have 5 main stages: Input, Motion Detection/Face Detection, Feature Extraction, Face Recognition, and the Output/Identification, as shown in Fig 5. Here we use the PCNN as feature extraction method for a face recognition system.

4.1 PCNN Initial Parameters

For this particular project, the initial parameters of the PCNN are shown in Table 2. The kernels W and M are 3×3 average kernels, because average kernels reinforce grouping.

4.2 PCNN Facial Feature Extraction

In previous sections we have defined what a pulsed image is, as a result of applying the PCNN to a given image. Now, if a gray-scale image is given to the input of the PCNN, we obtain pulsed images similar to the ones in Fig. 6 where it is shown that the pulsations of the faces changes across time. The selected pulses are the 36 to 40. These pulsations are selected because their content of facial information is useful to construct a feature vector. As in Fig. 6 the pulses 36 to 40 have more content rather than the first 10 pulsations.

The original image $I(x, y)$ is preprocessed by the PCNN. The PCNN pulses and generates an output image $I_p(x, y)$. The canonical form of a pulsed image $I_p(x, y)$ is $\mathbf{i}_{p_{40}}$ (row vector). The pulsed images, 36 to 40, from the PCNN are collected to generate a feature vector,

$$\mathbf{T}_{pcnn} = \left[\mathbf{i}_{p_{36}} \quad \mathbf{i}_{p_{37}} \quad \dots \quad \mathbf{i}_{p_{40}} \right]. \quad (8)$$

The original image face is reduced to the static size of 20×18 pixels no matter what the original size is. This amount of reduction is performed only for this particular part of the feature vector. Now we have that the size of \mathbf{T}_{pcnn} will be 1800 because $xy \times 5 = 20 \times 18 \times 5 = 1800$.

Table 2. PCNN initial parameters

For 50 iterations
$\beta = 1.0$
$G_{Feed} = 0.1$
$\alpha_F = 0.1$
$G_{Link} = 1$
$\alpha_L = 0.1$
$\alpha_\theta = 5$
$V = 5$

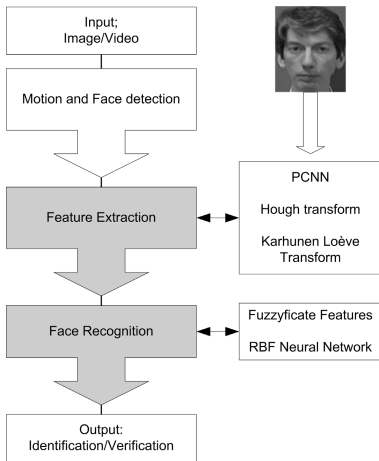


Fig. 5. Proposed face recognition system using PCNN as feature extractor

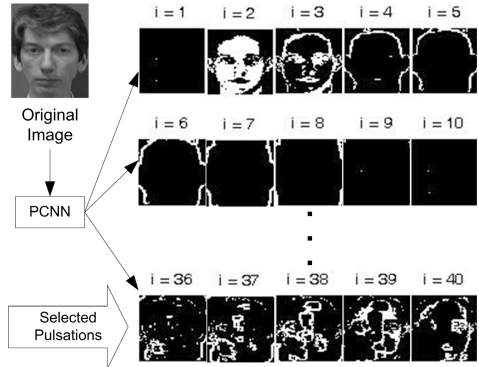


Fig. 6. Images pulsed by a PCNN when a face image is the input. Pulsations from 36 to 40 are selected as features.

4.3 Final Face Feature Vector

The final face feature vector includes three main components, face feature lines FFL \mathbf{z}_i , Karhunen-Loève feature transformation [14], KLT $\hat{\mathbf{I}}_{xy}$ (a variation of the PCA), and the features extracted with the PCNN \mathbf{T}_{pcnn} . Face feature lines are four prominent lines that can be extracted with the Hough transform from low resolution image faces, and are important features documented in newborn face recognition studies [15]. The KLT features are defined by

$$\hat{\mathbf{i}}_{xyn} = \mathbf{W}_{KLT}^T \mathbf{i}_{xyn}, \quad (9)$$

where \mathbf{i}_{xyn} is the whole training set composed by n \mathbf{i}_{xy} vectors, \mathbf{i}_{xy} is the canonical form of the original $I(x, y)$, and \mathbf{W}_{KLT}^T is the transformation matrix composed by the eigenvectors of the covariance matrix of \mathbf{i}_{xyn} .

The final feature vector can be now defined as

$$\mathbf{d}_{i+xy} = [\mathbf{z}_i \quad \hat{\mathbf{i}}_{xy} \quad \mathbf{T}_{pcm}]. \quad (10)$$

4.4 Neuro-Fuzzy Network

The design of the network is based on a probabilistic RBF neural network with two layers and fuzzy inputs. The number of neurons on the first layer is the same as the number of the training samples. In this case we will be using different number of samples from 10 (one sample per class) to 80 (8 samples per class). Therefore we have from 10 to 80 neurons. The activation functions of the first layer are Gaussian. The number of neurons of the last layer is equal to the number of classes, in this work we will be recognizing ten people, therefore we have 10 classes, consequently 10 neurons. The way to fuzzificate the inputs of the network is achieved by membership functions for each component of the feature vector. These membership functions are created according to the distribution of each component for every single person. The input vectors \mathbf{z}_i and $\hat{\mathbf{i}}_{xyn}$ are fuzzyfied with the membership functions just created.

4.5 Experiments and Results

We have experimented with two options to see the performance with and without PCNN. This experiment is one of the main contributions of this paper, to see how a PCNN improves or makes poor the performance of a face recognition system. The general scheme for the face recognition system when the PCNN is added to the system is shown in Fig 7. The experiments consist on changing the number of samples for training, selecting from 1 to 9 out of 10 samples per subject, and randomly selection of the samples. The experiments were designed to recognize 10 individuals. These experiments were realized over the OLR and YALE face databases. The testing results on the ORL without the PCNN have a maximum performance of 98%. For YALE without the PCNN the highest performance obtained reaches 78%. The testing results on ORL with PCNN show an improvement on 2 training samples (TS) for the subject #8 (S8) reaching 95%. The performance for YALE increases using the PCNN to 81%. Fig. 8 illustrates the performance of the experiments on the two databases, ORL and YALE with/without the PCNN. It is shown that the algorithm performs better on the ORL database because of less variation of the face samples regarding lighting conditions.

4.6 Conclusions of the Face Recognition Scheme

We have presented a neurofuzzy scheme for face recognition based on the PCNN feature extraction in a combination with the FFL and KLT features. The neurofuzzy algorithm was constructed extracting facial features via the PCNN, fuzzyfying the FFL and KLT features. These features are the inputs of an RBF neural network classifier which reaches 98% of recognition rate. This result is comparable to other systems previously developed [14]-[18]. As can be shown in Fig. 7 the algorithm performs well on ORL and it also performs better using PCNN over the YALE database even its severe lighting condition variations.

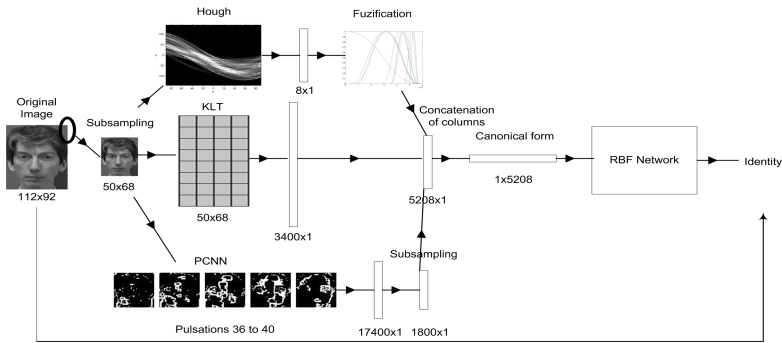


Fig. 7. General architecture for the PCNN neurofuzzy Hough-KLT face recognition system

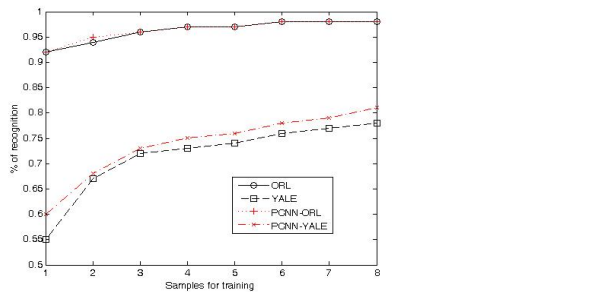


Fig. 8. Comparison of the experiments performed on the ORL and the YALE face databases with/without PCNN

5 Final Conclusions

This paper presented three applications of the PCNN architecture in image processing tasks. The first scheme is related to image segmentation, the second to automatic target location, and the third to face recognition. Results achieved in these three applications suggest that the PCNN architecture may be considered as a good preprocessor element to increase the performance of vision systems. The findings have demon-

strated the potential of the PCNN to generate useful information especially in segmentation and feature extraction tasks. The two last applications have shown how the PCNN architecture can be incorporated in vision systems to achieve complex tasks.

Acknowledgment

The authors appreciate the support of COSNET, and SEP-DGEST for the support of this research under Grant 445.05-P. Also CONACyT under Grant #193324.

References

1. Lindbland, T., Kinser J.: Image Processing Using Pulse-Coupled Neural Networks. Springer (1998)
2. Johnson, J.L., Padgett, M.L.: PCNN Models and Applications. *IEEE Transactions on Neural Networks* **10** (3) (1999) 480-498
3. Johnson, J.: Pulse-Coupled Neural Nets: Translation, Rotation, Scale, Distortion, and Intensity Signal Invariance for Images. *Applied Optics* **30** (26) (1994) 6239-6253
4. Kuntimad, G., Ranganath, H.: Perfect Image Segmentation Using Pulse Coupled Neural Networks. *IEEE Transactions on Neural Networks* **10** (3) (1999) 591-598
5. Keller, P., Johnson, J.: Pulse-Coupled Neural Network for Medical Analysis. *Proceedings of SPIE Applications and science of Computation Intelligence II* (1999) 444- 451
6. Ranganath, H.S., Kuntimad G.: Object Detection Using Pulse Coupled Neural Networks. *IEEE Transactions on Neural Networks* **10** (3) (1999) 615-620
7. Rughooputh, H., Rughooputh S.: Spectral Recognition Using a Modified Eckhorn Neural Network Model. *Image and Vision Computing* (2000) 1101-1103
8. Ranganath, H.S., Kuntimad G.: Image Segmentation Using Pulse Coupled Neural Networks. *IEEE World Congress on Computational Intelligence* (1994) 1285-1290
9. Chacón, M.I., Zimmeman, A.: PCNNP: A Pulse-Coupled Neural Network Processor. *Proceedings of the IEEE IJCNN* (2002) 1581-1585
10. Chacón, M.I., Zimmeman A.: Image Processing Using the PCNN Time Matrix as a Selective Filter. *Proceedings of the IEEE ICIP* (2003) 1195-1199
11. Parisi, R., Di Claudio, E.D., Lucarelli, G., Orlandi, G.: Car Plate Recognition by Neural Networks and Image Processing. *IEEE Proceedings of ISCAS-98* **3** (1998) 195-197
12. Yan, Y., Jin, G., Wu, M., He Q.: Optoelectronic Hybrid License Plate Recognition System. *SPIE Conference on Applications of Digital Image Processing XXII* **3808** (1999) 685-692
13. Zimic, N., Ficzek, J., Mraz, M., Virant J.: The Fuzzy Logic Approach to The Car Number Plate Locating Problem. *IEEE Proceedings Intelligent Information Systems* (1997) 227-230
14. Chacón, M.I.M., Rivas, P., Ramirez, G.: A Fuzzy Clustering Approach for Face Recognition Based on Face Feature Lines and Eigenvectors. *IEEE International Seminar on Computational Intelligence* (2006)
15. Rivas, P., Chacón, M.I.: Face Recognition Using Hough-KLT and a FeedForward Back-Propagation Neural Network. *International Conference on Electronics Engineering, ELECTRO 2006* (2006)
16. Li, S.Z., Jain, A.K.: *Handbook of face recognition*. Springer (2004)
17. Li, W., Zhou, C.: Face recognition using feature Combination and Improved LDA. *IEEE Transactions on Systems, Man, and Cybernetics* **2** (2004) 1292-1296
18. Ekenel, H.K., Sankur, B.: Multiresolution Face Recognition. *Image and Vision Computing* **23** (2005) 469-477

Image Segmentation Based on Cluster Ensemble

Zhiwen Yu, Shaohong Zhang, Hau-San Wong, and Jiqi Zhang

Department of Computer Science,
City University of Hong Kong
{cshswong, yuzhiwen}@cs.cityu.edu.hk

Abstract. Image segmentation is a classical problem in the area of image processing, multimedia, medical image, and so on. Although there exist a lot of approaches to perform image segmentation, few of them study the image segmentation by the cluster ensemble approach. In this paper, we propose a new algorithm called the cluster ensemble algorithm (CEA) for image segmentation. Specifically, CEA first obtains two set of segmented regions which are partitioned by EM according to the color feature and the texture feature respectively. Then, it integrates these regions to k segmented regions based on the similarity measure and the fuzzy membership function. Finally, CEA performs the denoise algorithm on the segmented regions to remove the noise. The experiments show that CEA works well during the process of image segmentation.

1 Introduction

Image segmentation is a hot topic in many areas [1]-[4], such as medical analysis, image processing, pattern recognition, multimedia, and so on, due to its broad applications. Given an image I , the image segmentation algorithm subdivides the image I into k regions. The pixels in each region have common properties, such as similar color, similar texture, similar contour, and so on.

The color feature and the texture feature are the most popular features which are used by the image segmentation algorithm. We focus on the image segmentation algorithms which consider the color feature and the texture feature at the same time. There exist three approaches to combine the color feature and the texture feature: (i) the approach which combines the color feature and the texture feature directly. Since the value range of the color feature is greater than that of the texture feature, the approach will pay more attention to the color feature. (ii) the approach which combines the color feature and the texture feature after normalizing within the range $[0, 1]$. It will emphasize the texture feature, since the number of dimensions of the texture feature are much larger than that of the color feature in most cases, especially when the number of dimensions of the color feature is 3. (iii) the approach which adds the weighs to the variables of the color and the texture. Although it can balance the importance of the color and the texture, the weighs are difficult to determine. In order to combining the color feature and the texture feature more effectively, we explore a new approach based on cluster ensemble which is used to combine the property of the color and the texture.

The remainder of the paper is organized as follows. Section 2 describes the framework of the cluster ensemble algorithm. Section 3 evaluates the effectiveness of our approach through the experiments. Section 4 concludes the paper and describes possible future works.

2 Overview of the Algorithm

Figure 1 illustrates the framework of the cluster ensemble algorithm (CEA) for image segmentation. It first extracts the color feature and the texture feature from the original image. Then, the expectation maximization algorithm is applied to perform image segmentation based on the color feature and the texture feature respectively. In the third step, CEA performs cluster ensemble which merges the similar regions by considering the position, color and texture. Finally, CEA obtains the final segmented regions.

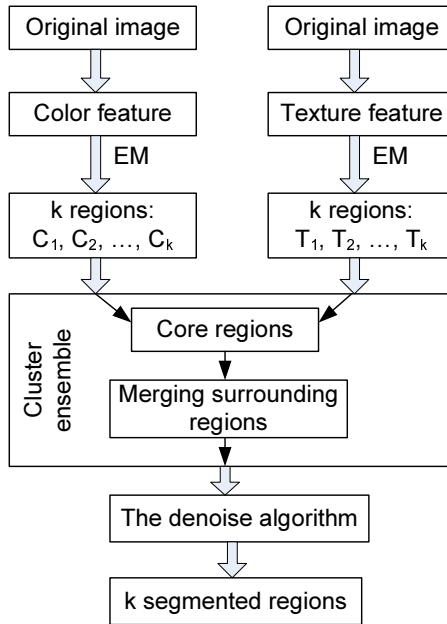


Fig. 1. The framework of cluster ensemble

2.1 Feature Extraction

The color feature is extracted from the CIE L*a*b* color space for each pixel. We adopt the CIE lab color feature here since this color space is one of the most widely adopted color models for describing colors visible to the human eye. Color

and color contrast are important features for humans to identify the objects in an image, and our approach is based on this observation. A 3-dimensional color feature vector is obtained for each pixel of the image.

Gabor filter [5] [6] is applied to extract the texture feature, since the filter not only can achieve the required selectivity in the preferred orientation and the preferred spatial frequency, but also possesses optimal joint localization properties in both spatial and frequency domains. we apply the family of two dimensional Gabor functions (GF) to extract the local image texture features :

$$GF_{l,\theta,\phi}(x, y) = e^{-\frac{(x'^2+r^2y'^2)}{2\sigma^2}} \cos(2\pi\frac{x'}{l} + \phi) \tag{1}$$

$$x' = x\cos\theta + y\sin\theta \quad y' = -x\sin\theta + y\cos\theta \tag{2}$$

where l represents spatial frequency bandwidth ($l \in \{20, 30, 40\}$), θ denotes the eight different equidistant preferred orientations ($\theta \in \{0, \frac{\pi}{8}, \frac{2\pi}{8}, \dots, \frac{7\pi}{8}\}$), ϕ is the initial phase ($\phi \in \{0, -\frac{\pi}{2}\}$), σ is the standard deviation of the Gaussian component ($\sigma = 0.56l$), and r is the spatial aspect ratio of the x- and y-axis of the Gaussian ellipse ($r = 0.5$).

Then, the Gabor feature image $GFI(x, y)$ is obtained by convolving the input image $I(x, y)$ with the Gabor function $GF(x, y)$:

$$GFI(x, y) = \int \int_S I(x, y)GF(u - x, v - y)dx dy \tag{3}$$

where $(x, y) \in S$, and S denotes the set of points in the image domain.

We further merge the outputs of the symmetric Gabor kernel filter and the antisymmetric Gabor kernel filter by the following Gabor energy equation (GE):

$$GE_{l,\theta}(x, y) = \sqrt{GF_{l,\theta,0}^2(x, y) + GF_{l,\theta,-\frac{\pi}{2}}^2(x, y)} \tag{4}$$

where $GF_{l,\theta,0}(x, y)$ and $GF_{l,\theta,-\frac{\pi}{2}}(x, y)$ are the responses of the symmetric and the antisymmetric Gabor kernel filters respectively. By the Gabor energy equation, we obtain a 24-dimensional texture feature vector for each image pixel.

2.2 Expectation Maximization Algorithm

In the second, the expectation maximization algorithm (EM [7]) is applied to subdivide the image into k regions based on the color feature and the texture feature respectively. We assume (i) $X = \{x_1, x_2, \dots, x_n\}$ denotes a set of feature vectors (color vectors or texture vectors); (ii) the distribution of the feature vectors can be approximated by Gaussian mixture models with k components c_j ($j \in [1, k]$); and (iii) the objective of EM is to maximize the log-likelihood $L(\theta|X)$ as follows:

$$\theta^* = argmax_{\theta \in \Theta} L(\theta|P)$$

$$= \operatorname{argmax}_{\theta \in \Theta} \sum_{x \in X} \log \left(\sum_{j=1}^k \pi_j \cdot p(x|c_j) \right) \tag{5}$$

$$p(x_i|c_j) = \frac{1}{\sqrt{(2\pi)^d |\Sigma_j|}} e^{[-\frac{1}{2}(x_i - \mu_j)^T (\Sigma_j)^{-1} (x_i - \mu_j)]} \tag{6}$$

where $\theta = \{\pi_j, \mu_j, \Sigma_j\}_{j=1}^k$, π_j is the mixing proportion of the component c_j , μ_j is the mean vector of the component c_j , and Σ_j is the covariance matrix of the component c_j .

After randomly selecting the parameters, EM first performs E-step, which estimates the probability that the data point x_i belongs to the component c_j in the t th iteration by the Bayes rule:

$$p^{(t)}(c_j|x_i) = \frac{p(x_i|c_j)}{\sum_{l=1}^k p(x_i|c_l)} \tag{7}$$

$$s_j^{(t)} = \sum_{x \in X} \frac{p(x_i|c_j)}{\sum_{l=1}^k p(x_i|c_l)} \tag{8}$$

Where $s_j^{(t)}$ is the sum of the probabilities.

Then, it updates the parameters of GMM as follows:

$$\pi_j^{(t+1)} = \frac{s_j^{(t)}}{n} \tag{9}$$

$$\mu_j^{(t+1)} = \frac{1}{s_j^{(t)}} \sum_{i=1}^n x_i \cdot p^{(t)}(c_j|x_i) \tag{10}$$

$$\Sigma_j^{(t+1)} = \frac{1}{s_j^{(t)}} \sum_{i=1}^n (p^{(t)}(c_j|x_i))(x_i - \mu_j^{(t+1)})(x_i - \mu_j^{(t+1)})^T \tag{11}$$

E-step and M-step are performed recursively until the the log-likelihood $L(\theta|X)$ is maximized. Finally, the feature vectors are assigned to the corresponding components.

2.3 Cluster Ensemble

The output of EM is two sets of segmented regions: the segmented regions based on color ($C = \{C_1, C_2, \dots, C_k\}$) and the segmented regions based on texture ($T = \{T_1, T_2, \dots, T_k\}$). The common regions (R), in which all the pixels have similar color and similar texture, is defined as follows:

$$R_h = C_i \cap T_j, \text{ for } \forall i, j \in [1, k] \tag{12}$$

where $h \in [1, k^2]$.

The core regions are defined as the first k common regions which have the largest number of pixels. Then, the common regions can be subdivided into two parts: the core regions (CR) and the non-core regions (NCR). The core regions are the basic blocks for the final segmented results. After obtaining the core regions, the cluster ensemble algorithm (CEA) merges non-core regions with the core regions based on the similarity measure and the fuzzy membership function.

For each region, CEA extracts a 29-dimensional feature vector ($\vec{f} = \{c_1, c_2, c_3, t_1, \dots, t_{24}, p_1, p_2\}$) which not only considers the color (c_1, c_2, c_3) and the texture (t_1, \dots, t_{24}), but also considers the position (p_1, p_2). Since the position is as important as the color and the texture, we adopt a new representation of the feature vector \vec{f} :

$$\vec{f} = \{\omega_{c_1} \cdot c_1, \dots, \omega_{t_1} \cdot t_1, \dots, \omega_{t_{24}} \cdot t_{24}, \omega_{p_1} \cdot p_1, \omega_{p_2} \cdot p_2\} \tag{13}$$

$$\sum_{j=1}^3 \omega_{c_j} = 1, \sum_{j=1}^{24} \omega_{t_j} = 1, \sum_{j=1}^2 \omega_{p_j} = 1 \tag{14}$$

where $\omega_{c_1} = \omega_{c_2} = \omega_{c_3}$, $\omega_{t_1} = \omega_{t_2} = \dots = \omega_{t_{24}}$ and $\omega_{p_1} = \omega_{p_2}$.

The similarity $Sim(\vec{f}_i, \vec{f}_j)$ between two regions (R_i and R_j) is defined as the Euclidean distance between two feature vectors (\vec{f}_i and \vec{f}_j):

$$Sim(\vec{f}_i, \vec{f}_j) = d(\vec{f}_i, \vec{f}_j) = (\vec{f}_i - \vec{f}_j)^2 \tag{15}$$

The fuzzy membership function $m(\vec{f}_{NCR_i}, \vec{f}_{CR_h})$ between the non-core region NCR_i ($NCR_i \in NCR$) and the core region CR_h ($CR_h \in CR$) is defined as follows:

$$m(\vec{f}_{NCR_i}, \vec{f}_{CR_h}) = \frac{1}{\sum_{j=1}^k \left(\frac{d(\vec{f}_{NCR_i}, \vec{f}_{CR_h})}{d(\vec{f}_{NCR_i}, \vec{f}_{CR_j})} \right)^{\frac{2}{q-1}}} \tag{16}$$

where \vec{f}_{NCR_i} denotes the feature vector of the i -th non-core region NCR_i , \vec{f}_{CR_j} and \vec{f}_{CR_h} denotes the feature vector of the j -th and h -th core regions CR_h and CR_j respectively, and $\frac{2}{q-1}$ is the fuzziness exponent and we set $q = 2$.

The membership function $m(\vec{f}_{NCR_i}, \vec{f}_{CR_h})$ can be converted to the following form:

$$m(\vec{f}_{NCR_i}, \vec{f}_{CR_h}) = \frac{\frac{1}{d(\vec{f}_{NCR_i}, \vec{f}_{CR_h})^{\frac{2}{q-1}}}}{\sum_{j=1}^k \frac{1}{d(\vec{f}_{NCR_i}, \vec{f}_{CR_j})^{\frac{2}{q-1}}}} \tag{17}$$

CEA will merge the non-core region NCR_i with the core region CR_{h^*} which has the largest value for the membership function:

$$h^* = argmin_{h \in [1, k]} m(\vec{f}_{NCR_i}, \vec{f}_{CR_h}) \tag{18}$$

After merging all the non-core regions, we obtain k segmented regions of the original image.

2.4 The Denoise Algorithm

CEA uses morphological operations [8] [9] to eliminate the very small disconnected regions and the noise from the segmented regions after merging. Our denoising algorithm is motivated by the observed properties of the noise and outliers, which are essentially a set of discontinuous and distributed pixels with small areas. The denoising algorithm considers the segmented regions one by one, and applies morphological operations to eliminate all connected components whose areas are smaller than a threshold, which is set at 1% of the total image area. Figure 2 shows examples of applying this algorithm to the images.

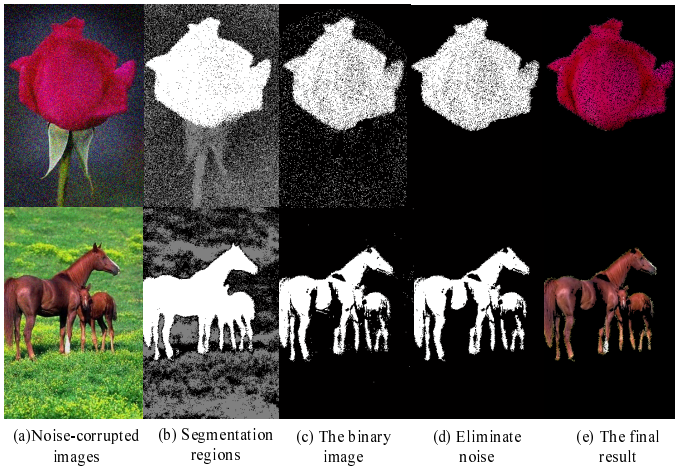


Fig. 2. Examples of applying the denoising algorithm

2.5 The Example

Figure 3 illustrates an example of the clustering ensemble algorithm (CEA). CEA first partitions the original image by the EM algorithm based on the color feature and the texture feature respectively. The partition results are shown in the second row of Figure 3. Then, CEA generates a set of the common regions. In the third step, it selects k core regions from the common regions as shown in the second red rectangle of Figure 3. The corresponding non-core regions will be merged with the similar core regions based on the similarity measure and the fuzzy membership function. Finally, we obtain the final segmented regions as shown in the last row of Figure 3, after eliminating the very small regions and the noise.

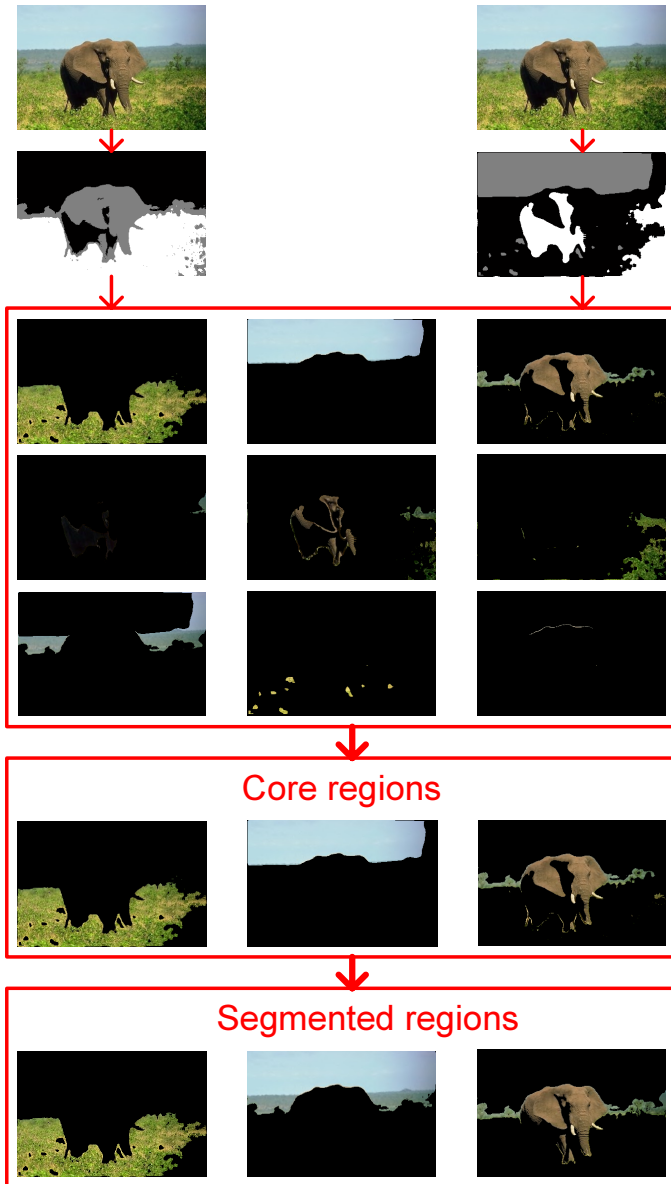


Fig. 3. The example of cluster ensemble

3 Experiment

Our database consists of 2000 images. Part of the images come from the COREL image library, and other images are mutually collected from the web.



Fig. 4. Comparison of different approaches

In the following experiment, we compare our method (CEA) with two approaches: (i) the approach (AD) which combines the color and the texture as the feature vector directly, and (ii) the approach (AN) which combines the color and the texture after normalizing the values within $[0, 1]$.

We randomly select four images from the image database to illustrate the performance of three approaches. Figure 4 compares the segmented results by different approaches. CEA can separate the mountain completely from the sky and the beach in the first image of Figure 4, while AD and AN only identify part of the mountain. In the second image of Figure 4, CEA combines the houses with the mountains, since the positions of the houses are closest to that of the mountains. But AD and AN separates the houses from the mountain, which break the connection between the houses and the mountain. CEA subdivides the original image into three segmented regions (the sky, the elephant and the grass) in the fourth image of Figure 4, while AD and AN confuse part of the elephant and the grass.

4 Conclusion and Future Work

In this paper, we investigate the problem of image segmentation based on cluster ensemble. Although there exist a lot of approaches for image segmentation, few of them explore the possibility of image segmentation by the cluster ensemble approach. Our major contribution is a new algorithm (CEA) for image segmentation according to cluster ensemble. Specifically, CEA applies EM to separate the original image into k regions by the color feature and the texture feature respectively. Then, it obtains k^2 common regions by integrating the regions based on color feature and texture feature. In the third step, CEA partitions the common regions into the core regions and the non-core regions. Finally, the segmented results of the original image are obtained by merging the non-core regions to the closest core regions. In the future, we will improve the cluster ensemble approach and explore the application in the image processing area further.

Acknowledgments. The work described in this paper was partially supported by a grant from the Research Grants Council of Hong Kong Special Administrative Region, China [Project No. CityU 1160/04E], and a grant from the City University of Hong Kong [Project No. 7001766].

References

1. Chen, J., Pappas, T.N., Mojsilovic, A., Rogowitz, B.E.: Adaptive Perceptual Color-texture Image Segmentation. *IEEE Transactions on Image Processing* **14**(10) (2005) 1524 -1536
2. Ko, B., Byun, H.: FRIP: A Region-based Image Retrieval Tool using Automatic Image Segmentation and Stepwise Boolean and Matching. *IEEE Transactions on Multimedia* **7**(1) (2005) 105-113

3. Neumann, A.: Graphical Gaussian Shape Models and Their Application to Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **25**(3) (2003) 316-329
4. Liu, J., Yang, Y.: Multiresolution Color Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **16**(7) (1994) 689 -700
5. Grigorescu, S., Petkov, N., Kruzinga, P.: Comparison of Texture Features Based on Babor Filters. *IEEE Trans on Image Processing* **11**(10) (2002) 1160-1167
6. Bovik, A. C.: Analysis of Multichannel Narrowband Filters for Image Texture Segmentation. *IEEE Trans. Signal Processing* **39** (1991) 2025-2043
7. Yu, Z., Wong, H.: FEMA: A Fast Expectation Maximization Algorithm based on Grid and PCA. *IEEE international conference on multimedia & expo* (2006) 1913-1916
8. Henk J.A., Heijmans, M.: Connected Morphological Operators for Binary Images. *Computer Vision and Image Understanding* **73**(1) (1999) 99-120
9. Henk J.A., Heijmans, M.: *Morphological Image Operators*. Academic Press, Boston, 1994

Decision-Based Hybrid Image Watermarking in Wavelet Domain Using HVS and Neural Networks

Hung-Hsu Tsai

Dept. of Information Management, National Formosa University,
Huwei, Yulin, 63208 Taiwan, R.O.C.
tjh@nfu.edu.tw

Abstract. This paper presents a Decision-based Hybrid Image Watermarking (DHIW) technique, based on the Human Visible System (HVS) and an Artificial Neural Network (ANN), for image copyright protection in wavelet domain. In [1], an image watermarking technique, called the IWNN technique, utilizes an ANN to extract watermarks without original images. However, the IWNN technique performs poorly for highly complicated image textures because the generalization capability of neural networks is powerfully effective in dealing with smooth image textures. Therefore, the PAIW method is proposed to enhance the IWNN technique, which uses the spatial information associated with wavelet-transformed images. The DHIW technique takes advantages of these two techniques by using a decision preprocessor. Experimental results prove that the DHIW technique remarkably outperforms other existing schemes.

1 Introduction

Nowadays, digital watermarking, which allows for the imperceptibly embedding watermarks (or digital signatures) in an original multimedia data, has emerged as a broadly approved way of performing copyright protection and ownership identification [1]–[11].

Many image watermarking schemes have been developed in frequency domain, which invisibly insert a watermark into an original image by modifying the coefficients of the transformed image [5]–[9]. Some watermark-embedding algorithms apply the Discrete Cosine Transform (DCT) to an image, and then hide the ownership information in DCT coefficients of an image [5] – [6]. Recently, another frequency transformation, the Discrete Wavelet Transform (DWT), is involved in the design of the wavelet-based watermarking techniques that modify DWT coefficients of the image upon embedding owner signatures [7] – [9]. Moreover, both the DCT and the DWT are also employed to devise robust image watermarking [9]. An advantage of the frequency-domain methods is that, in general, they are often robust to malicious attacks. Unfortunately, less embedding capacity is a major drawback of them. Consequently, a class of watermarking methods has been proposed by taking the advantages of the spatial-domain and the frequency-domain methods [10] – [11]. However, above methods still have three disadvantages, requiring original images during watermark extraction, mere performing watermark detection, and not exploiting the HVS to

enhance their transparency. Hence the DHIW technique, which integrates the IWNN and the PAIW techniques, is proposed here to overcome disadvantages mentioned above.

The IWNN technique employs an ANN to memorize the relationship between the original DWT coefficients and their modified versions. That is, the relationship can be realized by a trained ANN due to that a trained ANN can perform a universal mapping [12]. The IWNN technique utilizes the trained ANN to retrieve a watermark without the original image. Additionally, a wavelet-transformed image still preserves its spatial information. The PAIW technique adaptively modifies two DWT coefficients in a detailed sub band (for example, the vertical-line information), such as HL2 [13]. During watermark embedding, these two methods refer the Just Noticeable Difference (JND) values of two DWT coefficients so as to improve the imperceptibility of a watermarked image. Note that the JND is a crucial characteristic of the HVS, and is applied in diverse real-world applications of image processing [1], [3], [11]. Most methods, which employ the HVS to enhance their imperceptibility, almost perform watermark detection instead of watermark extraction. The methods, which merely perform watermark detection, suffer from the problem, not getting the clues of the piracy. Although the method proposed in [1] can perform watermark extraction, however, the trained ANN fails to retrieve watermarks from the sort of images with highly complex textures, for instance, Baboon image. Consequently, the paper presents the DHIW technique which employs a decision algorithm to alternatively apply one of the two methods upon watermark embedding and watermark extraction. Fig. 1 depicts the conceptual design of the DHIW technique. The decision procedure plays an important role that distributes non-complex-texture image blocks to the IWNN technique or dispatches the remainder of image blocks to the PAIW technique.

The rest of this paper is organized as follows. Section 2 presents image denotations and the DWT. Then, Section 3 describes the DHIW technique. Next, Section 4 shows experimental results. Finally, Section 5 gives conclusions.

2 Image Denotations and DWT

A gray-level image, X , with size $L \times K$ can be denoted by $X = [x_\rho]_{L \times K}$, where $x_\rho \in \{0, 1, \dots, 255\}$. That is, x_ρ represents the gray value of a pixel located at position $\rho = (i, j)$ over X , where $i \in \{0, 1, \dots, L-1\}$ and $j \in \{0, 1, \dots, K-1\}$. Fig. 2(a) shows that X is parti-

tioned into $\lfloor \frac{L}{8} \rfloor \times \lfloor \frac{K}{8} \rfloor$ non-overlapped blocks of size 8×8 . For example, b_{21} stands for a block that the center pixel in the block is located at position (2, 1) on X . Let $b_{21}(r, c)$ stands for the gray level of a pixel at the position (r, c) in the block b_{21} . As a result, a set Φ of the non-overlapped blocks of X can be represented by

$$\Phi = \left\{ b_{ij} \mid i=1, \dots, \lfloor \frac{L}{8} \rfloor, j=1, \dots, \lfloor \frac{K}{8} \rfloor \right\}, \tag{1}$$

where the size of each block b_{ij} is 8×8 .

In the experiment of the paper, a watermark, W , is denoted as a 2D binary image, which can be transformed to be a binary sequence in a row-major fashion. Thus, the watermark can be expressed as

$$W = (w_1, w_2, \dots, w_k, \dots, w_m) \tag{2}$$

where m stands for the size of W and $w_k \in \{-1, 1\}$.

Fig. 2(b) illustrates that each non-overlapped block b_{ij} in Φ for an image X is decomposed through the DWT. Let B_{ij} represent the corresponding wavelet block consisting of 64 coefficients. Moreover, $B_{ij}(r, c)$ stands for the coefficient at the position (r, c) on the block B_{ij} . Fig. 2(c) shows b_{ij} is transformed through DWT with two levels. First, each wavelet block B_{ij} at each level consists of four bands (components): low-low band (LL1), low-high band (LH1), high-low band (HL1), and high-high band (HH1). LL1 band can be further divided into four subbands: low-low subband (LL2), low-high subband (LH2), high-low subband (HL2), and high-high subband (HH2). Generally, HL2, LH2, HH2, HL1, and LH1 are called middle-frequency components (or detailed subbands).

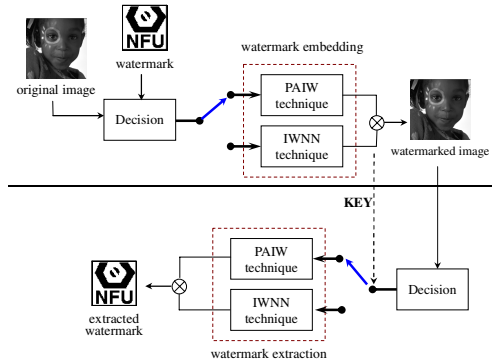


Fig. 1. The conceptual design of the DHIW technique

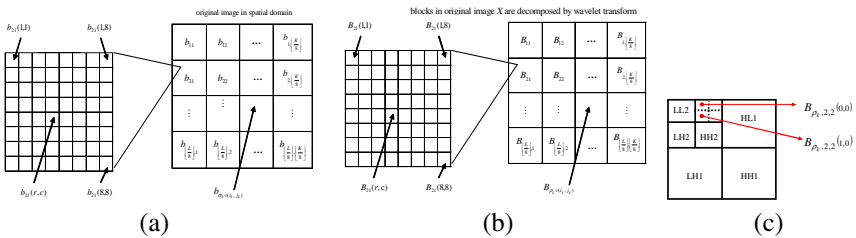


Fig. 2. (a) X is segmented into $\left\lfloor \frac{L}{8} \right\rfloor \times \left\lfloor \frac{K}{8} \right\rfloor$ non-overlapped blocks with size 8×8 . (b) b_{ij} is transformed through DWT. (c) Components are constituted of a wavelet image block with size 8×8 at level 2.

3 The DHIW Technique

3.1 The Embedding Algorithm of the DHIW Technique

Fig. 3(a) exhibits the diagram of the DHIW technique. First, the pseudo-random number generator (PRNG) component is realized by the scheme in [14]. While feeding the PRNG component with two seeds (s_1, s_2), a sequence \mathbf{P} of random positions can be generated and denoted as

$$\mathbf{P} = \{\rho_1, \rho_2, \dots, \rho_k \dots, \rho_m\}, \tag{3}$$

where $\rho_k = (i_k, j_k)$. The DHIW technique constructs a set, Ψ , comprising m blocks which are randomly selected from Φ by using the PRNG component. The set Ψ can be expressed as a form

$$\Psi = \{b_{\rho_k=(i_k, j_k)} \mid \rho_k \in \mathbf{P}\}, \tag{4}$$

where $k = 1, 2, \dots, m$.

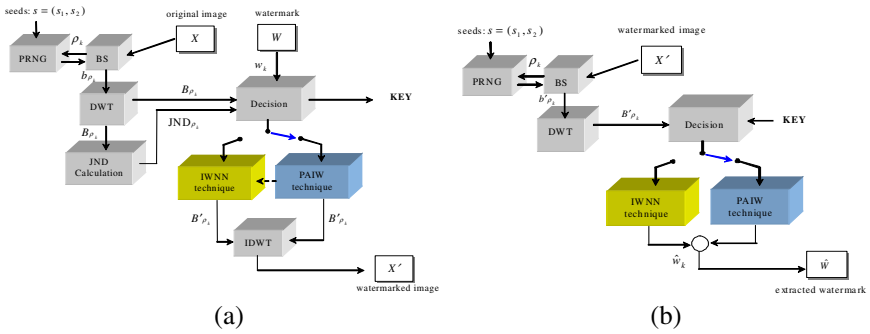


Fig. 3. (a)The diagram of the watermark embedding of the DHIW technique. (b) The structure of watermark extraction of the DHIW technique.

Next, Observing experimental results, the difference between the absolute values of these two coefficients $B_{\rho_k,2,2}(0,0)$ and $B_{\rho_k,2,2}(1,0)$ in subband HL2, is employed in the design of the decision procedure as described in the following.

Algorithm Decision ($B_{\rho_k=(i_k, j_k)}$).

Step 1. Input $B_{\rho_k=(i_k, j_k)}$.

Step 2. If $(|B_{\rho_k,2,2}(0,0)| - |B_{\rho_k,2,2}(1,0)|) < TH$

Step 3. then using the PAIW technique and $key_k = 0$.

Step 4. else using the IWNN technique and $key_k = 1$.

The PAIW technique embeds a watermark bit w_k via using the formula in (5). Note that $\text{sgn}(\cdot)$ and $|\cdot|$ represent the sign function and the absolute value operator, respectively.

$$\begin{aligned}
 B'_{\rho_k,2,2}(0,0) &\leftarrow \text{sgn}(B_{\rho_k,2,2}(0,0)) \times \left(\begin{array}{c} |B_{\rho_k,2,2}(0,0)| + \\ w_k (JND_{\rho_k,2,2}(0,0) + \tau) \end{array} \right) \\
 B'_{\rho_k,2,2}(1,0) &\leftarrow \text{sgn}(B_{\rho_k,2,2}(1,0)) \times \left(\begin{array}{c} |B_{\rho_k,2,2}(1,0)| - \\ w_k (JND_{\rho_k,2,2}(1,0) + \tau) \end{array} \right)
 \end{aligned}
 \tag{5}$$

Specifically, in (5), the technique employs the superimposition operation to two coefficients $B_{\rho_k,2,2}(0,0)$ and $B_{\rho_k,2,2}(1,0)$ in subband HL2. An attempt of the technique, like the patchwork algorithm in [2], is to enlarge the distance between these two coefficients in order to enhance its robustness. The PAIW technique is mainly developed to compensate for the incorrect watermark estimates with the IWNN technique for non-smooth image textures. The intention of the embedding procedure of the PAIW technique is to hide information in the non-smooth image textures. Generally, the visual results to modify non-smooth image textures are more imperceptible than smooth image textures.

The IWNN technique employs the modification formula in (6) for embedding w_k .

$$B'_{\rho_k,2,2}(0,0) \leftarrow B_{\rho_k,2,2}(0,0) + w_k (JND_{\rho_k,2,2}(0,0) + \tau)
 \tag{6}$$

Additionally, a positive constant τ is involved in (6) to increase the strength of the modification in order to enhance its robustness. The embedding algorithm of the technique differs from the PAIW technique in only modifying a coefficient, $B_{\rho_k,2,2}(0,0)$. The reason is that, ideally, the DHIW technique assigns smooth-textures image blocks to the IWNN technique. In order to avoid annoying artifacts on watermarked images, it just modifies one coefficient instead of two coefficients. Let the key vector **KEY** be expressed as a form in (7).

$$\mathbf{KEY} = (\text{key}_1, \text{key}_2, \dots, \text{key}_m)
 \tag{7}$$

where a bit, key_k , indicates which embedding algorithm is involved for w_k . The embedding algorithm of the DHIW technique is summarized in the following.

- Step 1. Input X , W , (s_1, s_2) , τ , and a threshold TH.
- Step 2. Segment X into a set Φ of non-overlapped image blocks with size 8×8 .
- Step 3. Present the PRNG with two seeds (s_1, s_2) to generate \mathbf{P} in (3).
- Step 4. Use \mathbf{P} to get Ψ in (4).
- Step 5. For each $b_{\rho_k=(i_k, j_k)}$.
- Step 6. $B_{\rho_k=(i_k, j_k)} = \text{DWT}(b_{\rho_k=(i_k, j_k)})$.
- Step 7. Calculate JND_{ρ_k} .
- Step 8. Call Algorithm Decision ($B_{\rho_k=(i_k, j_k)}$).
- Step 9. If ($\text{key}_k = 0$) then embed w_k using (5).
- Step 10. If ($\text{key}_k = 1$) then embed w_k using (6).
- Step 11. $b'_{\rho_k=(i_k, j_k)} = \text{IDWT}(B'_{\rho_k=(i_k, j_k)})$.
- Step 12. Until all w_k are embedded.
- Step 13. Output an watermarked image X' and a key vector **KEY**.

Note that the IDWT stands for inverse DWT. The calculation of JND_{ρ_k} can be found in [1]. Each coefficient in JND_{ρ_k} is associated with a range of error visibility. That is, a modified coefficient still retains imperceptions if the magnitude of the modification to the coefficient is in the range of error visibility. Subsequently, two seeds (s_1, s_2) and the **KEY** should be secured from pirates.

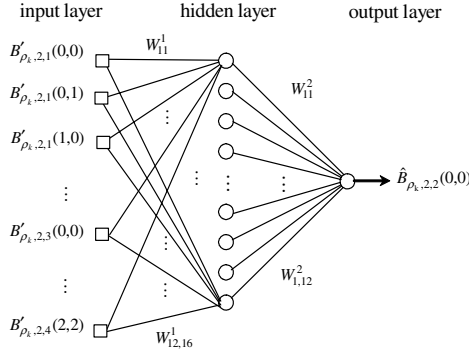


Fig. 4. The architecture of the ANN used in the DHIW technique

In the IWNN method, an ANN is used for memorizing the relationship between original coefficients and their watermarked version. The architecture of the ANN is displayed in Fig. 4. The ANN comprises three layers: the input layer with sixteen neurons, the hidden layer with twelve neurons, and the output layer with a single neuron 11. The ANN is so-called a 16-12-1 Multilayer Perceptrons (MLP). A back-propagation learning algorithm is applied to train the ANN by correcting its weights. These weights are adjusted to decrease the errors between the inputs and their corresponding target outputs. The set Υ of weights, associated with the ANN, can be expressed as a form in (8),

$$\Upsilon = \{W^1_{ij} | i = 1, 2, \dots, 12, j = 1, 2, \dots, 16\} \cup \{W^2_{ij} | i = 1, j = 1, 2, \dots, 12\}. \tag{8}$$

A set \mathbf{T} of training patterns is denoted as

$$\mathbf{T} = \left\{ (\mathbf{B}'_k, B_{\rho_k, 2,2}(0,0)) \mid k = 1, 2, \dots, m_1 \right\} \tag{9}$$

and

$$\mathbf{B}'_k = (B'_{\rho_k, 2,1}(0,0), \dots, B'_{\rho_k, 2,2}(0,0), \dots, B'_{\rho_k, 2,4}(0,0), \dots, B'_{\rho_k, 2,4}(1,1)). \tag{10}$$

Here \mathbf{B}'_k and $B_{\rho_k, 2,2}(0,0)$ represent an input vector and its corresponding desired output, respectively. The cardinality of the set \mathbf{T} in (9) is m_1 where m_1 is less than or equal to m and stands for the number of image blocks which is assigned to the IWNN technique by the Decision component in the DHIW technique. In other words, the

number of 1's in the **KEY** is equal to m_1 . Let $\hat{B}_{\rho_k,2,2}(0,0)$ denote the corresponding estimated output of the trained ANN while presenting the trained ANN with an input vector \mathbf{B}'_k in (10).

3.2 The Extraction Algorithm of the DHIW Technique

Fig. 3(b) depicts the structure of watermark extraction of the DHIW technique. Two seeds (s_1, s_2) and the **KEY** are required at the receiver site before watermark extraction. The position sequence \mathbf{P} can be obtained after feeding the PRNG with (s_1, s_2) . Next, a set Ψ' of watermarked blocks can be constructed using \mathbf{P} and a watermark image X' . Here two seeds (s_1, s_2) , the **KEY**, and the position sequence \mathbf{P} are the same as those utilized in the watermark embedding of the DHIW technique. The estimated watermark \hat{w}_k is computed by

$$\hat{w}_k = \begin{cases} 1, & \text{if } (\text{key}_k = 0 \ \& \ |B'_{\rho_k,2,2}(0,0) - \hat{B}_{\rho_k,2,2}(1,0)| \geq 0) \\ 1, & \text{if } (\text{key}_k = 1 \ \& \ B'_{\rho_k,2,2}(0,0) - \hat{B}_{\rho_k,2,2}(0,0) \geq 0) \\ -1, & \text{else.} \end{cases} \tag{11}$$

The watermark-extraction algorithm of the DHIW technique is described as follows.

- Step 1. Input X' , the key vector **KEY**, (s_1, s_2) , τ , and a threshold TH.
- Step 2. Segment X' into a set Φ' of 8×8 non-overlapped image blocks.
- Step 3. Present the PRNG with two seeds (s_1, s_2) to generate \mathbf{P} in (3).
- Step 4. Use \mathbf{P} to get Ψ' in (4).
- Step 5. For each $b'_{\rho_k=(i_k, j_k)}$.
- Step 6. $B'_{\rho_k=(i_k, j_k)} = \text{DWT}(b'_{\rho_k=(i_k, j_k)})$.
- Step 7. Call Algorithm Decision ($B'_{\rho_k=(i_k, j_k)}$).
- Step 8. Retrieve \hat{w}_k using (11).
- Step 9. Until all \hat{w}_k are retrieved.
- Step 10. Output an estimated watermark \hat{W} .

4 Experimental Results

In this experiment, a binary image, as shown in Fig. 5(a), is taken as the watermark (digital signature) of the copyright owner of an image. The quantitative index, PSNR (Peak Signal to Noise Ratio), is employed to measure the imperceptible quality of the watermarked images, and is defined by

$$\text{PSNR}(X, X') = 10 \log \left(\frac{255^2}{\text{MSE}(X, X')} \right), \tag{12}$$

where

$$\text{MSE}(X, X') = \frac{1}{L \times K} \sum_{i=0}^{L-1} \sum_{j=0}^{K-1} (x_{ij} - x'_{ij})^2. \quad (13)$$

Higher PSNR value reveals that the watermarked image X more resembles its original version X' . Another quantitative index, BCR (Bit Correction Rate), is used for evaluating the quality of the estimated watermarks, and is specified by

$$\text{BCR}(W, \hat{W}) = 1 - \frac{\sum_{k=1}^m |w_k - \hat{w}_k|}{2m}. \quad (14)$$

Note that the $\text{BCR}(W, \hat{W}) \in [0, 1]$. The higher BCR value conveys that \hat{W} is more similar to W .

Fig. 5(b) - (g) exhibit six 512×512 original images, Baboon, Babara, Peppers, Lena, Couple, and Goldhill images, respectively. Their watermarked images are displayed in Fig. 5(h) - (m). The imperceptibility of the DHIW technique is compared with that of existing methods [5], [6], [8], [9]. Fig. 6 shows the comparison results in terms of the PSNR values of the six test images for these six methods. The transparency of Shih's method is the best among them because it adopts least-significant-bit (LSB) embedding algorithm. However, it is very susceptible to common attacks. According, its robustness is poor. Fig. 7 displays its results.

In the attack-free case, the watermark-extraction ability of the DHIW technique is investigated here and also compared with that of these methods under consideration. Fig. 7 illustrates the comparison results in terms of the average BCR value of the six test images examined by exploiting these six methods. In addition, Fig. 7 displays the comparison results for the robustness of above methods under 15 unintentional attacks which are simulated by image-processing manipulations. Fig. 8 exhibits the visual results of extracted watermarks using these six methods for testing Baboon image. Observing Fig. 7, in most cases, the robustness of the DHIW technique is significantly better than that of other existing methods being considered here.



Fig. 5. (a) the original watermark. (b) - (g) are original images, Baboon, Babara, Peppers, Lena, Couple, and Goldhill images, respectively. (h) - (m) are their watermarked version.

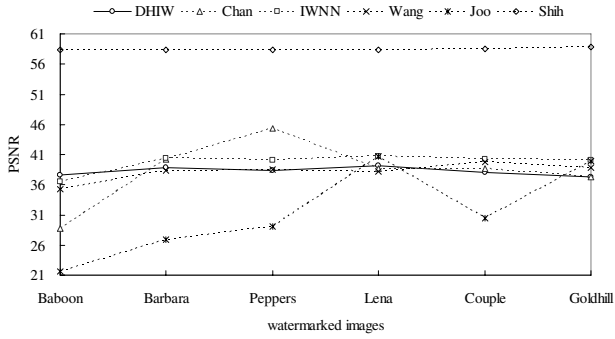


Fig. 6. The comparison results in terms of the PSNR value of the six test images

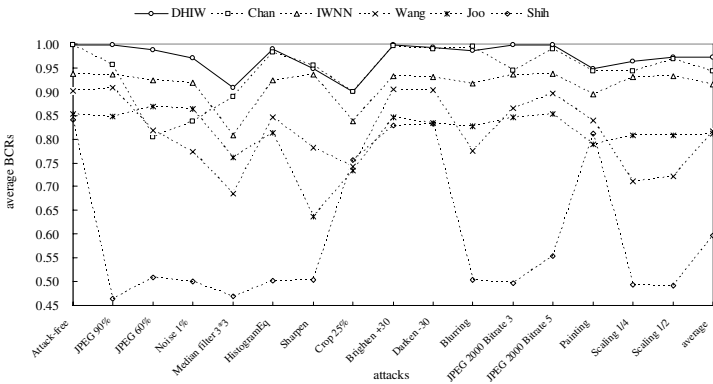


Fig. 7. The comparison results of watermark retrieval for six methods where 15 attacks, simulated by image-processing manipulations, are involved to investigate the robustness of six methods

Boboon	Attack-free	JPEG 60%	JPEG 90%	Noising 1%	Median filtering 3x3	Histogram equalization	Sharpening	Cropping 25%	Brightening +30	Darkening +30	Blurring	JPEG 2000 bitrate 3	JPEG 2000 bitrate 5	Painting	Scaling 1/4	Scaling 1/2
DHIW	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU
Chan	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU
IWNN	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU
Joo	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU
Wang	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU
Shih	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU	NFU

Fig. 8. The comparison results of extracted watermarks using six methods for 15 attacks while testing Baboon image

5 Conclusions

In this paper, a decision-based hybrid image watermarking technique, called the DHIW technique, has been proposed to improve the performance of the existing methods. The DHIW technique employs a decision algorithm so as to take the advantages of the IWNN and the PAIW methods. Experimental results demonstrate that the DHIW technique can achieve acceptable performance of both imperceptibility and robustness, and show that the technique is superior to other existing schemes under consideration.

Acknowledgments. The author would like to thank the National Science Council of Taiwan, R.O.C., for financially supporting this research under Contract No. NSC 95-2221-E-150-084. My thanks go to Mr. Chi-Chih Liu who provided the codes for the computer simulations.

References

1. Tsai, H.H., Liu, C.C.: Wavelet-Based Image Watermarking with Visibility Range Estimation Based on HVS and Neural Networks," submitted to *Computer Standards & Interfaces*
2. Bender, W., Gruhl, D., Morimoto, N., Lu, A.: Techniques for Data Hiding. *IBM Systems Journal* **35** (1996) 313-336
3. De Vleeschouwer, C., Delaigle, J.F., Macq, B.: Invisibility and Application Functionalities in Perceptual Watermarking-An Overview. *Proceedings of the IEEE* **90** (2002) 64-77
4. Tsai, H.H., Sun, D.W.: Color Image Watermarking Based on Support Vector Machines. *Information Sciences* **177** (2007) 550-569
5. Hsu, C.T., Wu, J.L.: Hidden Digital Watermarks in Images. *IEEE Trans. on Image Processing* **8** (1999) 58-68
6. Wang, Y., Pearmain, A.: Blind Image Data Hiding Based on Self Reference. *Pattern Recognition Letters* **25** (2004) 1681-1689
7. Joo, S., Suh, Y., Shin, J., Kikuchi, H., Cho, S.J.: A New Robust Watermark Embedding into Wavelet DC Components. *ETRI Journal* **24** (2002) 401-404
8. Yua, G.J., Lub, C.S., M. Liao, H.Y.: A Message-Based Cocktail Watermarking System. *Pattern Recognition* **36** (2003) 957-968
9. Chan, P.W., Lyu, M.R., Chin, R.T.: A Novel Scheme for Hybrid Digital Video Watermarking: Approach, Evaluation and Experimentation. *IEEE Trans. on Circuits and Systems for Video Technology* **15** (2005) 1638-1649
10. Shih, F.Y., Wu, Y.T.: Combinational Image Watermarking in the Spatial and Frequency Domains. *Pattern Recognition* **36** (2003) 969-975
11. Zhang, X.D., Feng, J., Lo, K.T.: Image Watermarking Using Tree-Based Spatial-Frequency Feature of Wavelet Transform. *Journal Visual and Communication Image Representation* **14** (2003) 474-491
12. Haykin, S.: *Neural Networks: A Comprehensive Foundation*, 2th edn. Macmillan College Publishing Company, New York, (1999)
13. Li, Z.-N., Drew, M.S.: *Fundamentals of Multimedia*. Prentice Hall. Upper Saddle River NJ (2004)
14. Blum, M., Micali, S.: How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits. *SIM Journal of Computing* **13** (1984) 850-864

MR Image Registration Based on Pulse-Coupled Neural Networks

Zhiyong Qiu, Jiyang Dong, and Zhong Chen

Department of Physics, Fujian Engineering Research Center for Solid-State Lighting,
Xiamen University, Xiamen, 361005, P.R. China
jydong@xmu.edu.cn

Abstract. A new algorithm for magnet resonance (MR) image registration is proposed based on a modified Pulse-Coupled Neural Networks (PCNN's). The transformed image and reference image are applied as inputs to two modified networks with the same parameters respectively. Taking advantage of translation, rotation, and distortion invariant characteristics of PCNN's, fired neuron groups of the two networks are acquired correspondingly, then the barycenters of those groups are extracted as characteristic points to attain the registration parameters. Experiment results showed that the proposed algorithm for MR image registration is fast and effective.

1 Introduction

Over the past decades, the brain research has been an active area of research and has made much important progress, which should be mainly owed to the development of medical imaging technology to some extent. Magnet resonance imaging (MRI) is one of the powerful imaging technologies which would be able to detect the structure and function of the brain woundlessly. People always extract the information of brain function by preprocessing and analyzing the time-series images acquired by MRI. At present, many modified processing algorithms have played an important role in the preprocessing course, such as image enhancement and registration etc.[1]. In this paper, we propose a new algorithm about the registration of magnet resonance image based on the modified Pulse-Coupled Neural Networks (PCNN's).

The medical image registration, using appropriate optimization method to achieve the maximal similarity of pixel intensity, is always used to implement the process of registration of entire image information. In general, most of the optimization methods can be classified as two classes, the local search such as Powell method and the global search such as simulated annealing method [2]. The local search approach in process is likely to run into a local extremum, resulting in a false registration. But the global search approach is more complicated and demands a great deal of computational time and space. Therefore, appropriate optimization method must be chosen to achieve a compromise between the accuracy and the speed. Because of the small computational load and strong anti-interference in small translation and rotation bias condition, the image registration algorithm based on projective method has attracted the attention of

many researches. However, when the contrast of translation and rotation between the transformed image and reference image is obvious, the extent of search will be largened, and the computational load will be increased rapidly as well. So this paper makes an effort to look for a new registration algorithm based on characters including the corner, point, line, boundary and surface etc. to outcome this problem.

According to the principle of the pulse synchronization behavior in the visual cortices of cats[3], a new simplified model, PCNN has been proposed and developed in 1990's, which primarily based on the grouping characteristic has proven to be highly effective when use in a diverse set of application such as image processing, communication and optimization [4] etc.. Taking advantage of the translation, rotation, scale, distortion and intensity signal invariance [5] of PCNN, this paper proposes a new registration algorithm, which considers the coordinate barycenter of neuron group as the characteristic point. This algorithm has been applied to the MR image registration successfully and has strong anti-noise performance. In addition, those factors, such as image rotation bias and translation bias etc., have no effect on its computational load. A great number of experiment results have shown the validity of this new algorithm.

In Section 2 of this paper, the modified model of PCNN is described simply. The image registration algorithm based on PCNN is proposed in Section 3. In Section 4, the experiment result is presented. Finally, conclusions and recommendations are given in Section 5.

2 PCNN Model and Its Characteristics

PCNN is a new simplified model [6] based on pulse synchronization behavior in the visual cortices of cats [7]. Now it has been successfully applied to various image processing, including image segmentation [8], image smoothing [9], image thinning [10], and target extraction [11] etc.

In PCNN model, there are three leaky integrators. Each leaky integrator has three parameters, the amplification factor, decay time constant and weighing factor, which will interact with each other. In reality, the mathematical analysis of the operation of the network is a difficult task. And the determination of values for all parameters in the model to effectively control the network operation is not a trivial task. Therefore this paper modifies the PCNN model to overcome the above shortcoming as follows:

$$F_{ij}(n) = S_{ij}, \quad (1)$$

$$L_{ij}(n) = \sum_{k \in N_{ij}} Y_k(n-1), \quad (2)$$

$$U_{ij}(n) = F_{ij}(n)(1 + \beta L_{ij}(n)), \quad (3)$$

$$Y_{ij}(n) = \text{step}(U_{ij}(n) - \theta_{ij}(n-1)), \quad (4)$$

$$\theta_{ij}(n) = \theta_{ij}(n-1)e^{-\alpha} + V_{\theta} Y_{ij}(n-1), \quad (5)$$

where F_{ij} , S_{ij} , L_{ij} , U_{ij} , θ_{ij} and Y_{ij} are the feeding input, external stimulus, linking input, internal activity, threshold and output of the neuron ij ; N_{ij} the set of adjacent neurons; V_θ the threshold amplitude constant; β linking strength; α time decay coefficient; n iteration. Furthermore, each neuron can pulse only once during a pulsing cycle in the modified model.

As well known, PCNN has a feature that the fire of one neuron can capture its adjacent neuron to fire via the linking stimulus due to their spatial proximity and intensity similarity. These neurons stimulate each other to compose of a unique group. The firing time and relative location of each group is fixed even if the network is translated, rotated or zoomed. So in the image registration processing, when the transformed image and reference image are applied as input to the PCNN's with the same parameters, one-to-one correspondence exists between the groups of the images, and the firing time and relative location of the corresponding grouping neurons are the same. Assumption that the characteristic points of the corresponding groups can be picked up, they will surely inherit the invariant characteristics of translation, rotation and a little distortion. That is why this paper expects to use the PCNN to achieve the image registration.

3 Image Registration Algorithm Using PCNN's

The main idea of the new algorithm is described as follows:

- (1) Find out all the firing groups of the transformed image and reference image based on the grouping characteristic of PCNN's.
- (2) Then use their invariant characteristics of translation, rotation and a little distortion to match all the groups so that the image registration will be implemented. But the number of grouping neurons in every set is usually too large to achieve the registration process fast. Considering that the barycenter in physics also has the invariant characteristics of translation and rotation, the barycenter of every firing group is picked up as the character point to complete the registration process.

The algorithm is composed of three steps. Firstly, obtain the object region by a special PCNN segmentation algorithm. Secondly, pick up barycenters of all firing groups as characteristic points. This step is the most important one in the image registration. Finally, compute the registration parameters on the basis of the data of barycenters and then finish the image registration process.

3.1 Background Segmentation of MR Image

The background pixels consist of those ones that often distribute in the image edge and the intensity is close to 0. To reduce their influence on the registration precision, the segmentation algorithm of MR image is implemented to obtain the object region. Considering the distributing character of the intensity of background pixels, the pixels belonging to the boundary of the image were fired first. For the internal pixels, they will be captured unless their intensity are close to 0 and receive linking inputs from

three neighbors. In this paper, their intensity are assumed less than 15. The segmentation algorithm using PCNN's is described as follows:

Step 1: Initialize the network parameters. Let $L_{ij} = U_{ij} = Y_{ij} = 0$, $\theta_{ij} = 0.4$, $\beta = 1$, $\alpha = 0$ for each neuron ij . The F element corresponding to the boundary pixels is adjusted to 1, others 0.1.

Step 2: Run the network by using Eq. (1)-(5) and capture the appropriate neurons that haven't fired with an intensity of less than 15.

Step 3: If there is not any neuron fires in this iteration, go back to **Step 2**, otherwise algorithm is completed.

The pixels corresponding to the fired neurons in the processing are called the background pixels. Others are object pixels. In image registration processing, only the object pixels should be considered in the next step.

3.2 Pick Up Characteristic Points — Barycenters of Firing Groups

Because of the singular intensity distribution of MR image, most of the neurons may be captured in some firing groups and the number of some groups may be too small. In order to get enough characteristic points to improve the registration, this algorithm adopts the process below. The object region is applied as input to the PCNN, and then all the pixels will be divided into 8 groups according to their firing time order, each group has the same number of pixels. As a result, the pixels in any group fire prior to those in the next group, but later than the former group. After that, compute the barycenters of all the groups by using barycenter formula and get two barycenter sets corresponding to the transformed image and reference image respectively.

The algorithm using PCNN is described as follows:

Step 1: Initialize the network parameters. Let $L_{ij} = U_{ij} = Y_{ij} = 0$, $\theta_{ij} = 255$, $F_{ij} = S_{ij}$ for each neuron ij .

Step 2: Let $\beta = \frac{1}{\sqrt{\theta_{ij}}}$, $\alpha = \ln\left(\frac{\theta_{ij}}{\theta_{ij} - 1}\right)$ and run the network using Eq.(1)-(5).

Then divide all the neurons into 8 groups in order.

Step 3: If not any neuron fires in this iteration, go to **Step 2**.

Step 4: Compute the barycenters of all the groups by using barycenter formula and get two barycenter sets.

It appears that the implementation of PCNN could be greatly simplified if threshold signals are allowed to decay linearly rather than exponentially. So this paper

let $\alpha = \ln\left(\frac{\theta_{ij}}{\theta_{ij} - 1}\right)$. At the same time, supposing that the intensity of most of the

pixels are low, let $\beta = \frac{1}{\sqrt{\theta_{ij}}}$ so that the number of firing neuron in every firing process

will be more homogeneous and the linking strength can be adjusted automatically.

By the time, the barycenters have been acquired. The next step is to use the result to finish the image registration process.

3.3 Compute the Registration Parameters

Consider the 8 barycenters as the characteristic points arranging in order for transformed image and reference image. Firstly, compute the barycenter of the above 8 barycenters that has been defined as the center of the image before. The difference between the centers of the two images is consistent with the registration translation bias $(\Delta x, \Delta y)$. Secondly, connect all the barycenters to the center of every image and calculate the angles relative to the horizontal axis. Then add the differences between one-to-one corresponding angles and the average of the sum is equal to the rotation parameter $\Delta\alpha$ relative to center of images. An example is shown in Fig.1, where the red crosses denote the centers of these images. Finally translate and rotate the transformed image with the parameters $(\Delta x, \Delta y, \Delta\alpha)$ and complete the registration process.

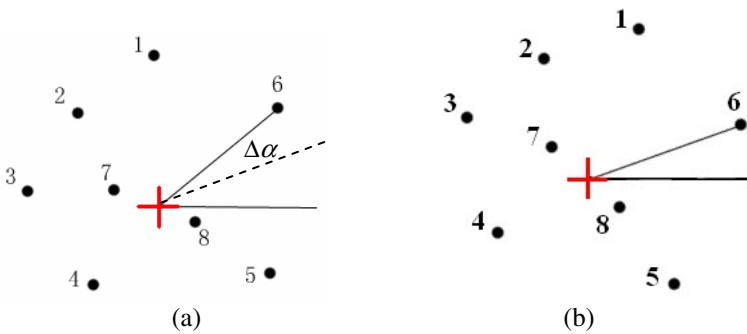


Fig. 1. The distribution of the barycenters of reference image is shown in (a). The distribution of the barycenters of transformed image is shown in (b).

One considerable problem of the algorithm is that the rotation angle $\Delta\alpha$ is relative to the center which is not actually the rotation center. Strictly speaking, the center of the rotation in the MRI experiment is random. Hence, the translation bias calculated from the algorithm derives from two factors: one is the singular translation; the other is the rotation. And translation bias will increase along with increment of the rotation angle. However, these phenomena will not affect the registration result. After taking the translation and rotation operation, the transformed image will match the reference image well.

3.4 Evaluation of Image Registration

There are many evaluation methods for the image registration. The sum of square error [12] and mutual information are in common use. Considering that these above

approaches would be influenced by other factors such as the number of image pixels and intensity distribution, this paper presents a new evaluation parameter σ as follows:

$$\sigma = \sqrt{\frac{1}{N} \sum_{i,j} (S_{ij} - S'_{ij})^2} \quad (6)$$

It represents the average difference of intensity between the transformed image and reference image. And the definition of the evaluation parameter σ would be more intuitionistic to evaluate the registration result.

4 Simulation and Analysis

The image registration algorithm described in Section 3 was coded in Visual C++6.0 and applied to large number of medical MR images. Take the 512×512 fMRI image shown in Fig. 2(a) for example. Fig.2 (b) shows the result of background segmentation of Fig.2 (a). Fig.2 (b) illustrates that the segmentation is effective and satisfying.

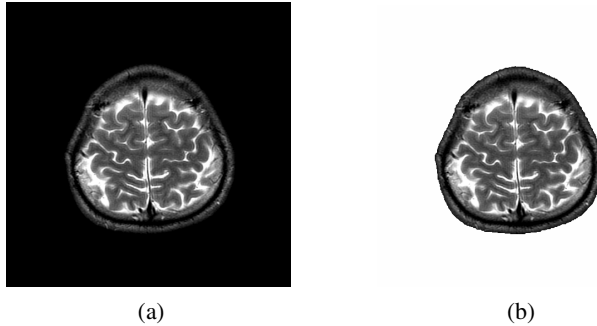


Fig. 2. The reference image is shown in (a). The result of background segmentation is shown in (b).

The next step is the registration algorithm of object region based on PCNN. Fig. 3 and Fig.4 show the firing groups of the reference image and transformed image rotated 45 degree at odd iterations. The bright pixels denote the neurons firing in the corresponding iteration computation, whose barycenter is labeled by a red cross. The figures clearly verify the grouping characteristic and invariant characteristics of translation and rotation of PCNN.

As to the case that the image is only translated by a constant, this paper assumes two cases. In one case small translation bias is provided, the other large. The registration result is shown in the Table 1, where the angle unit is degree, and translation and rotation parameters are denoted as $(\Delta x, \Delta y, \Delta \alpha)$.

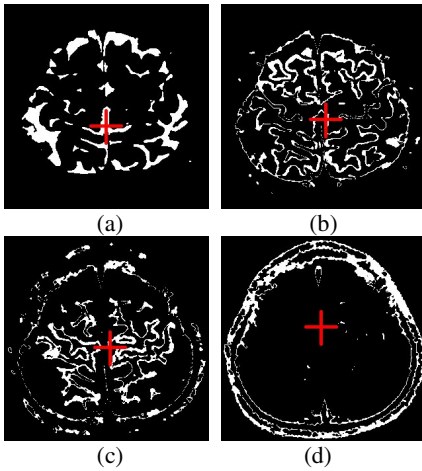


Fig. 3. The first, third, fifth and seventh firing groups of the reference image are shown in (a), (b), (c) and (d) respectively

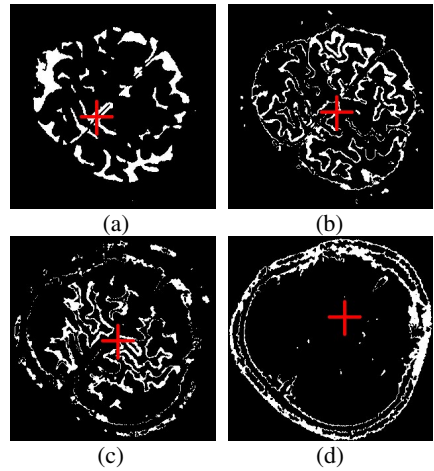


Fig. 4. The first, third, fifth and seventh firing groups of the transformed image are shown in (a), (b), (c) and (d) respectively

Table 1. The registration result considering the case the image is only translated by a constant

Translation bias	Standard parameters	Registration parameters	Evaluation parameter σ
large	(75.0, 26.0, 0.0)	(75.0,26.0,0.97 $\times 10^{-4}$)	0
small	(4.0, 4.0, 0.0)	(4.0, 4.0, 9.46 $\times 10^{-4}$)	0

The evaluation parameter σ is equal to 0, which illustrates that the registration algorithm does well in this case.

Table 2. The registratio result with translation and rotation bias

Standard parameters	Registration result	Evaluation parameter σ
(4.0, 7.0, 1.0)	(4.3, 6.7, 0.86)	1.9384
(4.0, 7.0, 2.0)	(4.1,6.8,1.88)	1.5771
(4.0, 7.0, 4.0)	(3.6,7.2,4.14)	1.8453

Moreover, the translation and rotation always appear synchronously, so this paper assumes that the transformed image has a small translation bias and rotation bias compared to the reference image. Considering that the image resolution is 512 \times 512,

the translation bias is given to be (4.0, 7.0). The rotation center of the image is adopted randomly near by the center of object region. The Table 2 shows the typical result of registration with certain rotation bias, where the angle unit is degree.

The results indicate that:

(1) Regardless of the rotation bias, the rotation parameters of registration are similar to the standard parameters and the angle differences are always less than 0.2 degree, even 0.1 degree. A great number of experiments indicate that the angle differences are usually less than 0.3 degree when the rotation angle small enough ($< 5^\circ$). Even when the rotation angles become big, the angel differences would be less than 0.5 degree.

(2) The differences of translation bias happen. The experiments of large rotation bias even up to 90 degree are implemented. The results indicated that the difference will increase along with the increment of rotation bias. The reason for this phenomena is that the image center, to which the rotation angle $\Delta\alpha$ is relative, is not actually the rotation center, just as what has been predicted in Section 3.

(3) The evaluation parameters are not actually equal to zero. The error mainly derives from the discrete distribution of image. But the error is small enough relative to the largest intensity 255 of MR image. Therefore, the result of registration algorithm when the translation and rotation take place synchronously is satisfying.

5 Conclusions and Recommendations

Taking advantage of the invariant characteristics of translation, rotation and a little distortion in PCNN, this paper proposes a new registration algorithm to apply to the MR image. Lots of experiment results have shown that this method for MR medical image registration is effective and has high precision. Furthermore, the computational load of the automatic algorithm is small and it needn't face those problems in the algorithm based on projective method, such as how to choose the initial value and when the translation or rotation bias becomes large, its search extent and computational load will increase greatly. The existence of Gauss noise is familiar in the MR image. In this new method, many neurons will fire synchronously each iteration, which makes the location of barycenters more fixed, so it is more effective and steadier, and insensitive to the noise, all of which will be benefit for the medical MR image registration. However, besides medical MR image, the algorithm can only be applied to the image of nonsymmetrical structure. When the image of nonsymmetrical structure is applied as input to PCNN, the barycenters of all groups are dispersed and the registration result is satisfying and has high precision. For symmetrical structure, the barycenters of all groups for the image may overlap. As a result, the registration algorithm won't work well. However, for the medical MR image, the probability of symmetrical structure is so small that this case can be considered negligible. In order to extend the utilization of this algorithm, how to improve the mechanism of firing to make the barycenters apart from each other is the point that we are focusing on in future.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant No. 60574039) and the “863” Project of National Ministry of Science and Technology (Grant No. 2006AA03A175).

References

1. Chen, H., Yao, D.: A fMRI Data Processing Method Using Projection Technique. Robotics, Intelligent Systems and Signal Processing. Lecture Notes in IEEE, 3 Changsha (2003) 1235-1239
2. Mouravliansky, N., Matsopoulos, G. K., Delibasis, K., Nikita, K.S.: Automatic Retinal Registration Using Global Optimization Techniques. IEEE Trans. Engineering in Medicine and Biology Society **20** (1998) 567-570
3. Johnson, L., Padgett, L.: PCNN Models and Applications. IEEE Trans. Neural Networks **10** (1999) 480-498
4. Johnson, J.L.: Overview of Pulse Coupled Neural Network. IEEE Trans. Neural Networks **10** (1999) 461-463
5. Johnson, J.L.: Pulse-coupled Neural Nets: Translation, rotation, Scale, Distortion and Intensity Signal Invariance for Images. Appl. Opt. **33** (1994) 6239-6253
6. Johnson, J.L., Ranganath, H., Kuntimad, G.: Pulse Coupled Neural Networks. Neural Networks and Pattern Recognition, San Diego (1998) 1-56
7. Eckhorn, R., Reitboeck, H.J., Arndt, M., et al.: Feature Linking via Synchronization Among distributed Assemblies: Simulation of Results from Cat Cortex. Neural Computation **2** (1990) 293-307
8. Xue, Y., Yang, S.: Image Segmentation Using Watershed Transform and Feed-back Pulse Coupled Neural Network. Lecture Notes in Computer Science, **3696** (2005) 531-536
9. Ranganath H.S., Kuntimad G., Johnson J.L.: A Neural Network for Image Understanding. In: Riesler E and Beale R: Handbook of Neural Computation, Oxford UK (1997) G1.6.1-G1.6.6
10. Gu, X., Yu, D., Zhang, L.: Image Thinning Using Pulse Coupled Neural Network. Pattern Recognition Letters **25** (2004) 1075-1084
11. David B.F., James C.B., Bosacchi B.: Pulse-Coupled Neural Networks (PCNN) and New Approaches to Biosensor Applications. Applications and Science of Computational **3390** (1998) 79-88
12. Rui, L., Jeffrey, L.K., Martin, J.M.: an Information-theoretic Criterion for Intrasubject Alignment of fMRI Time Series: Motion Corrected Independent Component Analysis. IEEE Trans. Medical Image **25** (2005) 29-44

Image Magnification Based on the Properties of Human Visual Processing

Sung-Kwan Je¹, Kwang-Baek Kim², Jin-Young Lee³, and Jae-Hyun Cho⁴

¹ Dept. of Computer Science, Pusan National University
jimmy374@pusan.ac.kr

² Dept. of Computer Engineering, Silla University
gbkim@silla.ac.kr

³ Dept. of Digital Design, Busan Digital University
cyber@bdu.ac.kr

⁴ Dept. of Computer Engineering, Catholic University of Pusan
jhcho@cup.ac.kr

Abstract. Image magnification is among the basic image processing operations. The most commonly used techniques for image magnification are based on interpolation method. However, the magnified images produced by the techniques, such as nearest neighbor, bilinear and cubic method, often appear a variety of undesirable image artifacts such as 'blocking' and 'blurring' into the several processing for image magnification. In this paper, we propose image magnification method by properties of human visual system which reduce information during transforming from receptors to ganglion cells in retina and magnify information at visual cortex. Our method uses the whole image to exactly detect the edge information of the image and then emphasizes edge information. Experiment results show that the proposed method solves the drawbacks of the image magnification, such as blocking and blurring, and has a higher PSNR and Correlation than the traditional methods.

Keywords: Image magnification, edge detection, image reduction, retina, visual cortex.

1 Introduction

Image magnification is among the basic image processing operations and has many applications in a various area. In recent, image equipments, CCD camera, digital camera and cell phone are now widespread and as a result, computer users can buy them and acquire many digital images as desired. This is why the need to display and print them also increases [1,2].

However, such various images with optical industry lenses are used to get high-resolution. The lenses are not only expensive but also too big for us to carry. So, they are using the digital zooming method with the lenses to solve the problems. This is why the common use is cheaper and faster than the lenses. The digital zooming method generally uses the nearest neighbor interpolation method, which is simpler and faster than other methods. But it has a drawback such as blocking phenomenon

when an image is enlarged. Also, to improve the drawbacks, there are the bilinear interpolation method and the cubic convolution interpolation commercially used in the software market. The bilinear method uses the average of 4 neighborhood pixels. It can solve the blocking phenomenon but brings about the loss of the image like the blurring phenomenon when the image is enlarged. Cubic convolution interpolation is improved in image loss like the nearest neighbor interpolation and bilinear interpolation. But it has another problem that its processing time is long as it uses the offset of 16 neighborhood pixels [3-5].

A number of methods for magnifying images have been proposed to solve the problems until now. However, proposed methods on magnifying images have the disadvantage that either the sharpness of the edges cannot be preserved or that some highly visible artifacts are produced in the magnified image. Although previous methods show a high performance in special environment, there are still the basic problems left.

Recently, researches on Human vision processing have been in the rapid progress. In addition, a large number of models for modeling human vision system have been proposed to solve the drawbacks of machine vision such as object recognition and object detection [6].

This paper presents a method for magnifying images that produces high quality images based on human visual properties which have image reduction on retina cells and information magnification of input image on visual cortex.

The rest of this paper is organized as follows. Section 2 presents the properties on human visual system and related works that have proposed for magnifying image. Section 3 presents our method that extracts the edge information using wavelet transform and uses the edge information base on the properties of human visual processing. Section 4 presents the results of the experiment and some concluding remarks are made in Section 5.

2 Related Works and Human Visual Processing

2.1 Related Works

The simplest way to magnify images is the nearest neighbor method by using the pixel replication and basically making the pixels bigger. It is defined by equation 1. However, the resulting magnified images have a blocking phenomenon [7].

$$Z(i, j) = I(k, l), \quad 0 \leq i, j, \text{ integer}$$

$$k \equiv \text{int} \left[\frac{i}{2} \right], l = \text{int} \left[\frac{j}{2} \right], \text{ where } Z(i, j) \text{ is a magnified image} \quad (1)$$

Other method is bilinear method, which determines the value of a new pixel based on a weighted average of the 4 pixels in the nearest 2×2 neighborhood of the pixel in the original image [7]. Therefore this method produces relatively smooth edges with hardly any blocking and is the better than the nearest neighbor but appears blurring phenomenon. It is defined by equation 2.

$$\begin{aligned} Z(i, 2j) &= I(k, l), \quad Z(i, 2j+1) = \frac{1}{2}[I(k, l), I(k, l+1)] \\ Z(2i, j) &= I_i(k, l), \quad Z(2i+1, j) = \frac{1}{2}[I_i(k, l), I_i(k+1, l)] \end{aligned} \quad (2)$$

More elaborate approach uses cubic convolution interpolation which is more sophisticated and produces smoother edges than bilinear interpolation. Bicubic interpolation is a bicubic function using 16 pixels in the nearest 4×4 neighborhood of the pixel in the original image and is defined by equation 3. This method is most commonly used by image editing software, printer drivers and many digital cameras for re-sampling images. Also, Adobe Photoshop offers two variants of the cubic convolution interpolation method: bicubic smoother and bicubic sharper. But this method arises in another problem that the processing time takes too long by the computation for the offsets of 16 neighborhood pixels [8].

$$f(x) = \begin{cases} (a+2)|x|^3 - (a+3)|x|^2 + 1, & 0 \leq |x| < 1 \\ a|x|^3 - 5a|x|^2 + 8a|x| - 4a, & 1 \leq |x| < 2 \\ 0, & \text{elsewhere} \end{cases} \quad \text{where } a=0, \text{ or } -1 \quad (3)$$

Recently, research on interpolation images taking into account the edges has gained much attention. Allebach and Wong and Salisbury et al. proposed methods that search for edges in the input images and use them to assure that the interpolation does not cross them. The problem is one of how to define and find the important edged in the input image [9].

Other edge-adaptive methods have been proposed by Jensen and Anastassiou[10]. The commercial software Genuine Fractals also used an edge adaptive method to magnify images, but the details of the algorithm are not provided. Currently, the methods presented in [10,11] are the most widely known edge-adaptive methods. They can well enough avoid jagged edges, but a limitation is that they sometimes introduce highly visible artifacts into the magnified images, especially in areas with small size repetitive patterns [12].

In section 3, we will propose an efficient method by image reduction and edge enhancement based on the properties on human visual processing.

2.2 Human Visual Processing

In the field of computer vision, many researches have been conducted in relation with edge information to solve the problem of magnification. Image information received from retina in Human visual system is not directly transmitted to the cerebrum when we recognize it. This is why there are the properties of many cells in Human visual system [13].

First, the visual process begins when visible light enters the eye and forms images on the retina, a thin layer of neurons lining the back of the eye. The magnified view of the retina consists of a number of different types of neurons, including the rod and cone receptors, which transform light energy into electrical energy, and fibers that transmit electrical energy out of the retina in the optic nerve. Second, The signals

generated in the receptors trigger electrical signals in the next layer of the retina, the bipolar cells, and these signals are transmitted through the various neurons in the retina, until eventually they are transmitted out of the eye by ganglion cell fibers. These ganglion cell fibers flow out of the back of the eye and become fibers in the optic nerve [13,14].

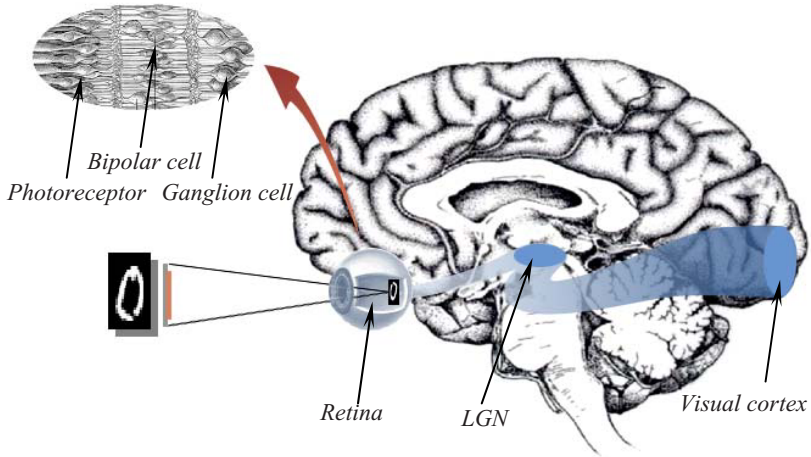


Fig. 1. The processing steps of human vision system

Third, most of these optic nerve fibers reach the lateral geniculate nucleus (LGN), the first major way station on the way to the brain. The LGN is a bilateral nucleus, which means that there is an LGN on the left side of the brain, and also one on the right side. Finally, Fibers from the LGN stream to the primary visual receiving area, the striate cortex or V1 in the occipital lobe. In conclusion, the main properties in human visual processing are as follows:

First, in retinal cells, the large difference between the number of receptors and the number of ganglion cells means that signals from many receptors converge onto each ganglion cell. Second, in visual cortex, this cell responds to the directions such as vertical, horizontal and orthogonal. Finally, the signal from ganglion cells coming from retina in fovea needs more space on the cortex than the signals from retina in periphery. The result is the cortical magnification factor [13].

We propose the magnification method considering the properties of human visual processing in section 3.

3 Image Magnification by the Properties of Human Vision System

Based on the properties of human visual processing discussed in section 2, we now describe a magnification method for improving the performance of conventional image magnification methods. Human vision system does not transfer image information from retina to visual cortex in the brain directly. By the properties of

retinal cells, there is the reduction of information when vision information is transferred from receptors to ganglion cells. In addition, the reduced information from retina is transfer to the visual cortex with the magnified information. We proposed the magnified method with the properties. The schematic diagram of the method is shown in fig.2.

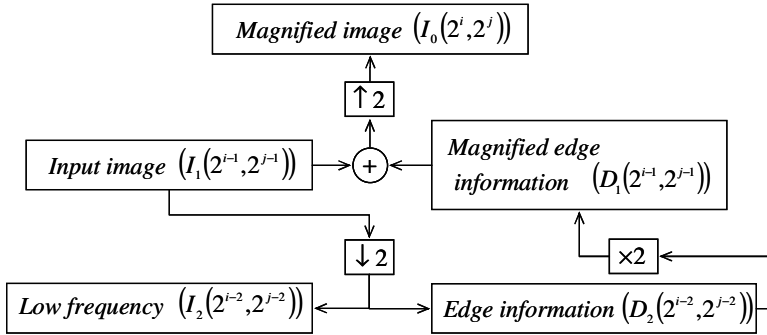


Fig. 2. The basic diagram of proposed method

The input image to be magnified is denoted as I_1 . First the input image is reduced in size by a factor of 2 in both dimensions such as I_2 and D_2 , where I_2 is also half of I_1 and D_2 is edge information such as vertical, horizontal and orthogonal information. For convenience, we will only set the case of magnification by a zooming factor of 2^k , where k is an integer. Wavelet transform is used to reduce input image. In the contrary, the input image is able to magnify if the edge information is calculated.

We were able to obtain the wavelet coefficients with quarter of image from the input image. The wavelet transform is used for detecting the edge of input image. The information by wavelet transform has the edge information in several directions such as vertical, horizontal and diagonal information. In this paper, we enhanced magnification algorithm to remove the blurring phenomenon using high-frequency information of sub-bands.

Fig. 3 shows the detailed algorithm. The main steps are as follows: First, decomposes the input image with wavelet analysis in order to obtain the edge information from the input image as convergence of receptors onto ganglion cells in the retina. Second, In order to obtain a edge information, it make used of the high frequency, which have the vertical, horizontal and orthogonal information, from wavelet transformation. This process is similar to magnify the image information to visual cortex which has the properties that magnify information from retina.

And then, generate the approximated image by compose the magnified edge and the input image. Finally, normalize the generated magnification image by Gaussian filter. Gaussian filter is similar to the acuity in human response degree.

Harr function is used to obtain the edge information. This is why Harr has a linear computation and has the properties of the orthogonal and the normalization.

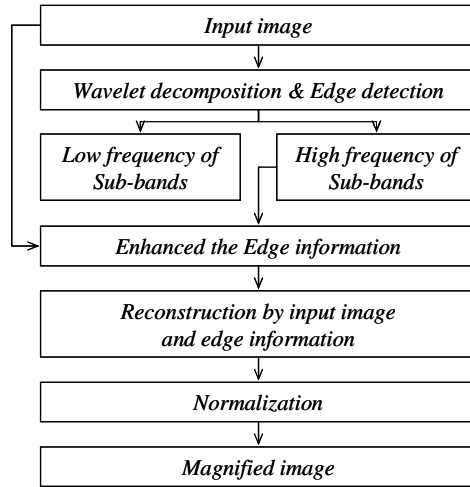


Fig. 3. The detail algorithm of proposed method

In addition, it is a simple and basic function among the mother wavelets. In next step, we use bilinear method to emphasize the edge information. And equation 4 shows that the magnified image is reconstructed by the input image I_1 and edge information D_1 .

$$\begin{aligned}
 Zoom_{temp}[j,i] &= (I_1[k,i] + LH_1[k + half, i])/2 \\
 Zoom_{temp}[j+1,i] &= (HL_1[k,i] - HH_1[k + half, i])/2 \\
 I_0[i,j] &= (Zoom_{temp}[i,k] + Zoom_{temp}[i,k + half])/2 \\
 I_0[i,j+1] &= (Zoom_{temp}[i,k] - Zoom_{temp}[i,k + half])/2
 \end{aligned} \tag{4}$$

Here, LH_1 is the horizontal direction of the edge information, HL_1 is the vertical direction of the edge information, and HH_1 is the diagonal direction of it, and we calculate by moving to the point of adding half of the size because the size of the input image is half size of the magnified image. We calculate the magnified image I_0 through combination and decomposition by using the edge information D_1 magnified by bilinear interpolation.

As a last step, the magnified image is convolved with a Gaussian low pass filter mask in equation 5.

$$MI[i,j] = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(i^2 + j^2)}{2\sigma^2}} * I_0[i,j] \tag{5}$$

The filter is used for removing stray noise and used for having the effect of slightly blurring similar to the properties of visual cortex. The Gaussian filter is mainly used in analyzing brain waves in visual cortex. And once a suitable mask has been calculated, then the Gaussian smoothing filter can be performed through standard convolution methods. By setting the standard deviation of input image as σ and the Gaussian operator as $\frac{1}{\sqrt{2\pi\sigma^2}}e^{-\frac{(i^2+j^2)}{2\sigma^2}}$, the magnification image $MI[i, j]$ is obtained.

4 Experimental Results

We used the Matlab 6.0 in a Pentium 2.4GHz with 512MB memory in a Windows XP environment and a lot of images were simulated by several methods. We used the SIPI Image Database and HIPR Image Library package used in other research papers on image processing [14,15]. In order to evaluate the performance of the proposed algorithm, we calculated with the PSNR and the correlation, which is numerically expressing the difference between the original image and the magnified image, in comparison with the previous algorithms in order to evaluate objectively.

First, we calculated the processing time taken for the 256×256 sized of the Lena image to become enlarged to a 512×512 sized Lena image in evaluated performance. Fig. 4 shows the processing time of the algorithms. The nearest neighbor interpolation is very fast in the processing time (0.016s), but it loses a part of the image due to the blocking phenomenon.

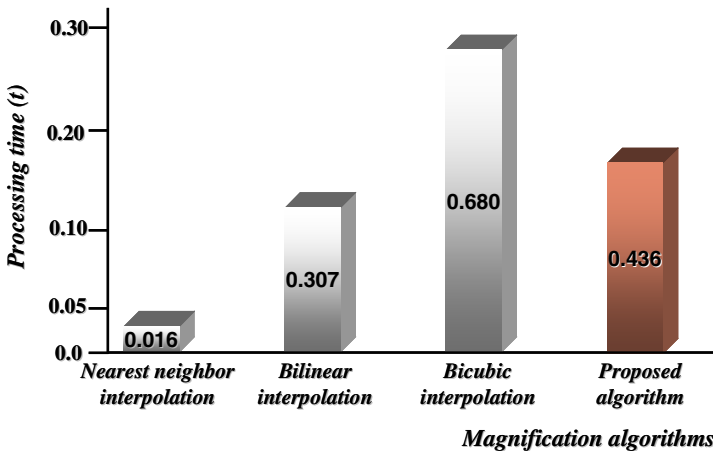


Fig. 4. The processing time in each algorithm

The nearest neighbor method is faster than our method in the processing time (0.110s), but it also loses a part of the image because of the blurring phenomenon. The cubic convolution interpolation does not have any image loss such as the blocking and blurring phenomenon, but is too slow in the processing time (0.307s)

because it uses 16 neighborhood pixels. The proposed algorithm solves the problem of image loss and is faster than the cubic convolution and bilinear interpolation in the processing time (0.047s). And then, figure 5 shows a reduced image of the 512×512 sized Lena image to a 256×256 sized Lena image by averaging 3×3 windows. This is followed by an enlargement of the 512×512 image through the use of each algorithm.

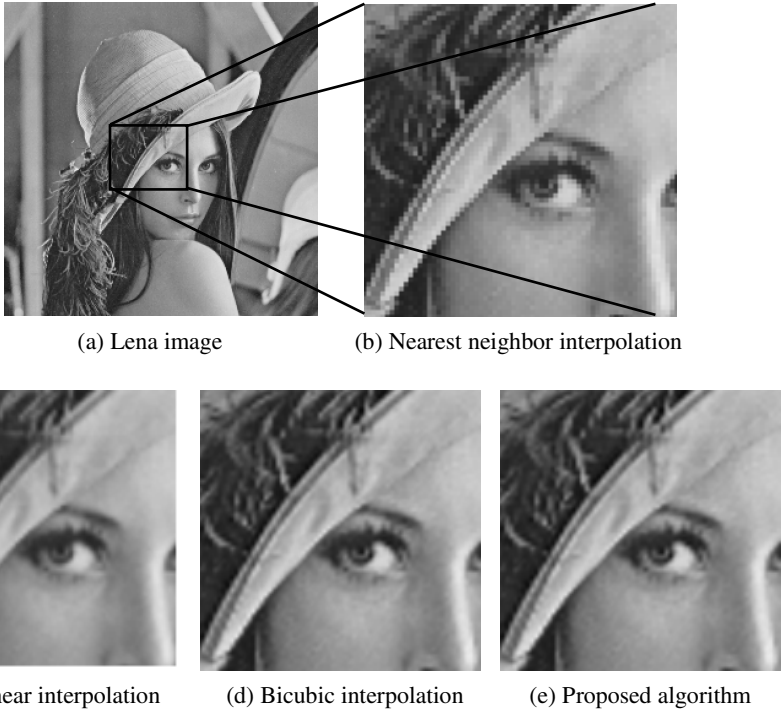


Fig. 5. Close-up comparison of Lena image

Fig. 5 compares the results when closing up the central part of Lena image 8 times to present a vision performance. For comparison, we can find the blocking phenomenon within vision in the nearest neighborhood interpolation (b). And we can also find the blurring phenomenon within vision in the bilinear interpolation(c). The proposed algorithm has better resolution than the cubic convolution interpolation and emphasizes the edge information in figure 4(d, e). Second, we calculated the PSNR with an objective decision.

Table 1 presents PSNR and Correlation in each algorithm. The MSE is a mean square error between the original image and the magnified image. Generally, the PSNR value is 20-40db. Our method is better than any other algorithm the PSNR value is 29.94.

Table 1. Comparison of the PSNR & Correlation of each algorithm

Methods	PSNR(db)	Cross-correlation
Nearest neighborhood interpolation	19.54	0.978983
Bilinear interpolation	29.36	0.983878
Cubic convolution interpolation	29.93	0.985839
Proposed algorithm	29.94	0.985559

Also, table 1 shows that the cross-correlation is used to compare objectively in other images. The cubic convolution interpolation is better than another method in Correlation. But it also has similar results in cross-correlation. So we tested other images (Baboon, Pepper, Aerial, Airplane, and Barbara) by the cross-correlation and PSNR in Table 2 and 3. Table 2 and 3 show that the proposed algorithm is better than any other methods in PNSR and Correlation on other images.

Table 2. Comparison of the PSNR of our method and general methods in several images

Methods	Baboon	Pepper	Aerial	Airplane	Barbara
Nearest neighborhood interpolation	21.91	28.18	23.54	32.55	24.04
Bilinear interpolation	22.13	29.06	24.45	33.44	24.12
Cubic convolution interpolation	22.22	29.17	24.67	33.72	24.25
Proposed algorithm	22.44	30.08	24.69	34.18	24.33

Table 3. Comparison of the correlation value of our method and general methods in several images

Methods	Baboon	Pepper	Aerial	Airplane	Barbara
Nearest neighborhood interpolation	0.87687	0.98326	0.90590	0.96654	0.95657
Bilinear interpolation	0.88288	0.98648	0.92402	0.97378	0.95743
Cubic convolution interpolation	0.88522	0.98677	0.92750	0.97556	0.95860
Proposed algorithm	0.89096	0.98934	0.92729	0.97475	0.95928

5 Conclusions

In image processing, the interpolated magnification method brings about the problem of image loss such as the blocking and blurring phenomenon when the image is enlarged. In this paper, we proposed the magnification method considering the properties of human visual processing to solve such problems. As a result, our method is faster than any other algorithm and is capable of removing the blocking and

blurring phenomenon when the image is enlarged. The cubic convolution interpolation in image processing can obtain a high resolution image when the image is enlarged. But the processing is too slow as it makes use of the average of 16 neighbor pixels. The proposed algorithm is better than the cubic convolution interpolation in the processing time and performance. In the future, to reduce the error ratio, we will enhance the normalization filter which has reduced the blurring phenomenon because the Gaussian filter is a low pass one.

References

1. Battiato, S., Mancuso, M.: An Introduction to the Digital Still Camera Technology. *ST Journal of System Research. Special Issue on Image Processing for Digital Still Camera* (2001)
2. Battiato, S., Gallo, G., Stanco, F.: A Locally Adaptive Zooming Algorithm for Digital Images. *Image and Vision Computing. Elsevier Science B.V.* **20** (2002) 805-812
3. Aoyama, K. and Ishii, R.: Image Magnification by Using Spectrum Extrapolation. *IEEE Proceedings of the IECON* **3** (1993) 2266 -2271
4. Candocia, F.M. and Principe, J.C.: Superresolution of Images Based on Local Correlations. *IEEE Transactions on Neural Networks* **10** (1999) 372-380
5. Biancardi, A., Cinque, L., Lombardi, L.: Improvements to Image Magnification. *Pattern Recognition. Elsevier Science B.V.* **35** (2002) 677-687
6. Suyung, L.: A Study on Artificial Vision and Hearing Based on Brain Information Processing. *BSRC Research Report* (2001)
7. Gonzalez, R.C., Richard E. W.: *Digital Image Processing. Second edition*, Prentice Hall (2001)
8. Keys, R.G.: Cubic Convolution Interpolation for Digital Image Processing. *IEEE Transaction on Acoustics, Speech, and Signal Processing* **29** (1981) 1153-1160
9. Salisbury, M., Anderson, C., Lischinski, D., and Salesin, D.H.: Scale-dependent Reproduction of Pen-and Ink Illustration. In *Proceedings of SIFFRAPH* **96** (1996) 461-468
10. Li, X., Orchard, M.T.: New Edge-directed Interpolation. *IEEE Transactions on Image Processing* (2001) 1521-1527
11. Muresan, D.D., and Parks, T.W.: Adaptively quadratic image interpolation. *IEEE Transaction on Image Processing* (2004) 690-698
12. Johan, H., and Nishita, T.: A Progressive Refinement Approach for Image Magnification. In *Proceedings of the 12th Pacific Conference on Computer Graphics and Applications*, (2004) 351-360
13. Bruce, G.E.: *Sensation and Perception*, Sixth edition (2002)
14. Duncan, J.: Selective Attention and the Organization of Visual Information. *Journal of Experimental Psychology: General.* **113** (1984) 501-517
15. The HIPR Image Library. <http://homepages.inf.ed.ac.uk/rbf/HIPR2/>.

Incorporating Image Quality in Multimodal Biometric Verification

Hong Huang¹, Jianwei Li¹, Zezhong Ma², and Hailiang Feng¹

¹ Key Lab. on Opto-electronic Technique and systems, Ministry of Education, Chongqing University, 400030 Chongqing, China

² Chongqing Institute of Surveying and Planning for Land, Jiangbei, 400020 Chongqing, China

Hhuang.cqu@gmail.com, Jiangweilee@cqu.edu.cn, Mazhcq@163.com, Fenghailiang@cqu.edu.cn

Abstract. The effect of image quality on the performance of multimodal biometric verification is studied. A biometric system based solely on single modality is often not able to meet the system performance requirements for poor image quality. Prior studies of multimodal biometric authentication have shown that it can improve performance over use of a single unimodal biometric. The well-known multimodal methods do not consider the quality information of the data used when combining the results from different matchers. In the paper, a novel SVM-based multimodal biometric authentication system is presented. It is based on SVM classifiers and quality measures of the input biometric signals. Experimental results on a prototype application based on fingerprint and face are reported. The proposed scheme is shown to outperform significantly multimodal systems without considering quality signals and unimodal systems over a wide range of image quality.

1 Introduction

Automatic access of persons to services is becoming increasingly important in the information era. Although person authentication by machine has been a subject of study for more than thirty years, it has not been until recently that the matter of combining a number of different traits for person verification has been considered [1], because single unimodal biometric is still facing numerous problems, some of them inherent to the technology itself (e.g., bad quality fingerprints due to some people whose fingers are too dry, wet and temporal or permanent damages or Changes in hairstyle, makeup, facial hair, etc in face verification). As a result, much research work has been done in multimodal biometric system.

From a practical point of view, multimodal verification has also been studied as a two-class classification problem by using a number of machine learning paradigms. Some design guidelines for a multi-biometric system are known and well accepted. Previous related literatures can be found in various publications. Hong et al [2] introduced the method of integrating faces and fingerprints, however, without the

evaluation of fingerprint image quality, the system may be unable to extract features from fingerprints associated with specific individuals, due to the poor quality of the fingerprints. Souheil Ben-Yacoub et al [3] used SVM-based method to fuse the voice and faces, but the accuracy of the multi-model system may be unsatisfied for the reason of complex background and the change of voice. Marcos et al [4] presented a multiple impression system, the system captures and processes two or more fingerprint images to improve robust, as in [5], the fingerprints of specific individuals are too dry or moist, which will decline the performance.

Current trends in multimodal biometrics research include the exploitation of quality signals. In this paper, we propose a novel quality-based adaptive trained multimodal fusion scheme based on support vector machines. With adaptive fusion scheme, we mean that the fusion scheme readapts to each identity claim as a function of the estimated quality of the input biometric signal.

The paper is organized as shows. The evaluation of biometric signals quality is briefly introduced in Section 2. In Section 3, the proposed quality-based multimodal biometric system is presented. The results based on the proposed algorithm are discussed in Section 4, Conclusions will be finally given in section 5.

2 Automatic Image Quality Assessment

The benefits of having an automatic quality estimate include the following: in a multimodal biometric system involving several traits, e.g. face, fingerprint and speech, the quality of the presented images influences the weight given to the respective expert at fusion stage, where a final decision is made. However, a universal quality method or model appears to be impossible: One application may use information of an image not useful to another application. In biometrics, for example, a face image contains information not useful to a fingerprint machine expert, so we use different model when trying to estimate the quality of biometric images.

2.1 Fingerprint Quality Assessment

By human experts, the quality of a fingerprint image is usually expressed in terms of the clarity of ridge and valley structures, as well as the extractability of certain points (minutiae, singular points). For efficiency reasons, we use the method proposed by Hong huang [5]. The automatic evaluation of fingerprint image quality includes the following:

- 1) Direction and foreground estimation. This step determines if a given block depicts a portion of a fingerprint and extracts a nominal direction from a foreground block.
- 2) Dominant direction. After the foreground blocks are marked, it is determined if the resulting direction for each block is prominent. The idea is that a block with a prominent direction should exhibit a clear ridge/valley direction that is consistent with most of the pixel directions in the block.
- 3) Image quality computation. Since regions (or accordingly minutiae) near the centroid are likely to provide more information for biometrics authentication, the overall quality of the fingerprint image is computed

from the directional blocks by assigning relative weight w_i for foreground block i at location x_i given by

$$w_i = \exp\{-\|x_i - x_c\|^2 / (2q^2)\}, \tag{1}$$

where x_c is the centroid of foreground, and q is a normalization constant. The overall quality Q of a fingerprint image is obtained by computing the ratio of total weights of directional blocks to the total weights for each of the blocks in the foreground by

$$Q = \sum D W_i / \sum F W_i. \tag{2}$$

Here D is the set of directional blocks and F the set of foreground blocks. The quality Q is used as a measure of how much reliable directional information is available in a fingerprint image. If the computed Q is less than the quality threshold, T , the image is considered to be of poor quality.

2.2 Face Quality Assessment

When it comes to estimate the quality of face images, the quality of a face image is usually expressed in terms of sample properties. The sample property includes two aspects, the first is Character: features of the sample source (e.g. pose expression), the second is Fidelity: accuracy with that the sample represents its source (e.g. sharpness, resolution). We can model the behavior of the sample property with following features: sharpness, openness of eyes, deviation from frontal pose (here, frontal means: within 5 degrees yaw and pitch angle) and wearing of glasses. Sample properties found to have considerable influence on the performance of face verification subsystem [6].

Consider a data set, F , containing the samples of face images, then we apply 3×3 mean filter to F , we can get the result F' . The sharpness problem is solved using the representation

$$D = \text{abs}(F - F'). \tag{3}$$

Sharpness is defined by the average pixel value over image D . Well-focused images get high sharpness values, blurred images low ones. Other three sample properties (openness of eyes, deviation from frontal pose, wearing of glasses) are determined as follows:

- 1) Apply wavelet transform to relevant portion of face image.
- 2) To result, apply support vector machine trained on labeled image samples.

3 Our Proposed Multimodal Biometric System

Our proposed system consists of four phases as follows: image-acquisition phase, image quality assessment phase, fingerprint and face verification subsystems, fusion and decision phase (see Fig. 1).

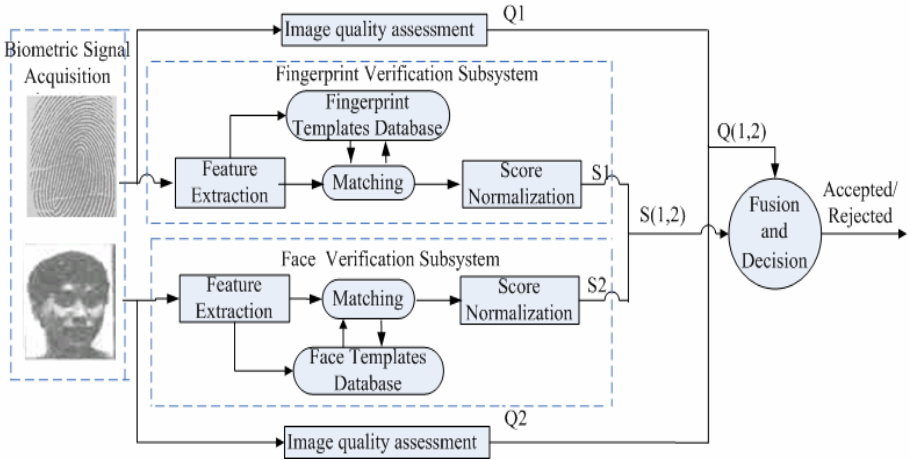


Fig. 1. Block-diagram of the proposed multimodal biometric system

3.1 Fingerprint Verification Subsystem

Most fingerprint identification systems represent the uniqueness of a fingerprint by means of its minutiae pattern; we will now adopt the system proposed by Jain [7], because it is a complete and well documented system which is very suitable for explanation and further comparison with other systems proposed in the literature. A method was developed consisting of several steps that we will now describe briefly (see Fig. 2):

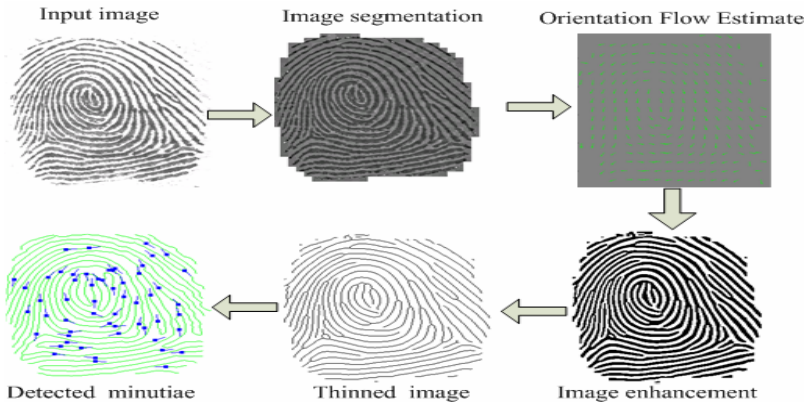


Fig. 2. Block-diagram of the proposed multimodal biometric system

(1) First, an orientation field is estimated. This field represents the orientation of the ridges and valleys at each region of the image, and it is estimated taking into account the vertical and horizontal gradients along all pixels in the image.

(2) After the region of interest has been delimited, the ridges are extracted and thinned.

(3) At this point, minutiae points can be easily found. For each minutiae point, its position is stored, as well as the orientation field in this point and a segment of the associated ridge.

Once the minutiae points have been found, a matching strategy has to be developed. As the minutiae representation scheme does not take into account the possible variability between several fingerprint images from the same finger, this problem has to be dealt with in matching stage. Factors that may cause two representations to be different, even though they come from the same individual, include possible rotations, nonlinear deformations caused by the pressure of the finger upon the sensor and the inherent imprecise nature of the extraction minutiae procedure. Therefore, the matching procedure must be based on a somehow elastic comparison between both point (minutiae) patterns. The system uses a matching strategy that is divided into two stages (see Fig. 3):

(1) As the alignment of point patterns is generally a hard task, specially in the presence of noise and deformations, the ridges associated to each minutiae point are used, so that using these corresponding curve segments makes the problem easier and the results more robust.

(2) The minutiae matching score is computed then by using a variant of the edit distance on polar coordinates and based on a size-adaptive tolerance box.

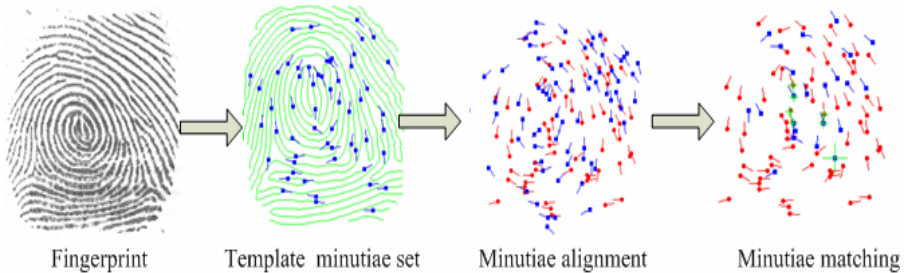


Fig. 3. Flowchart of the minutia matching algorithm

3.2 Face Verification Subsystem

Face verification involves extracting a feature set from a two-dimensional image of the user's face and matching it with the template stored in the database. The feature extraction process is often preceded by a face detection process during which the location and spatial extent of the face is determined within the given image. This is a difficult process given the high degree of variability associated with human faces (color, texture, expression, pose, etc.). The problem is further compounded by the presence of complex backgrounds and variable lighting conditions. A variety of techniques have been described in the literature to locate the spatial coordinates of a face within an image (Burel and Carel, 1994; Rowley et al., 1998; Yang and Huang, 1994). Once the boundary of the face is established, we use the eigenface approach to extract features from the face (Kirby and Sirovich, 1990; Turk and Pentland, 1991). In this approach a set of orthonormal vectors (or images) that span a lower dimensional

subspace is first computed using the principal component analysis (PCA) technique. The feature vector of a face image is the projection of the (original face) image on the (reduced) eigenspace. Matching involves computing the Euclidean distance between the eigenface coefficients of the template and the detected face.

3.3 Support Vector Machine

Support vector machine (SVM) is based on the principle of structural risk minimization [8]. It aims not only to classify correctly, all the training vectors, but also to maximize the margin from both classes. The optimal hyperplane classifier of a SVM is unique, so the generalization performance of SVM is better than other methods that possible lead to local minimum.

Given a set $(x_i, y_i), i=1, \dots, n, x \in R^d, y \in \{+1, -1\}$, where the x_i belongs to either of two classes, w_1, w_2 which are assumed to be linearly separable, y_i is the indicator(+1 for $w_1, -1$ for w_2). In this paper, we take into account the case of non-separable patterns, the training errors are allowed and the problem formulation in that case is called as soft margin SVM. This can be done by introducing positive slack variables ξ_i , in constraints with (4):

$$y_i(\langle w, \Phi(x_i) \rangle_H + w_0) \geq 1 - \xi_i, \forall i, \tag{4}$$

$$\xi_i \geq 0, i = 1, \dots, N. \tag{5}$$

Hence a logical way to assign an extra cost for errors is to change the objective function to be minimized into:

$$\min \left\{ \frac{1}{2} \|w\|^2 + C \cdot \left(\sum_i \xi_i \right) \right\}, i = 1, \dots, N, \tag{6}$$

where C is a chosen parameter and ξ_i are slack variables. The optimization problem in (4), (5) and (6) is solved using the dual representation [9]:

$$\max_{a_1, a_2, \dots, a_N} \left(\sum_{i=1}^N a_i - \frac{1}{2} a_i a_j y_i y_j K(x_i, x_j) \right) \tag{7}$$

subject to

$$0 \leq a_i \leq C, i = 1, \dots, N, \tag{8}$$

$$\sum_{i=1}^N a_i y_i = 0. \tag{9}$$

3.4 SVM-Based Multimodal Fusion Using Quality Signals

Given a multimodal biometric verification system consisting of R different unimodal systems $r = 1, \dots, R$, each one computes a similarity score $x_r \in R$ between an input biometric pattern and the enrolled pattern of the claimant. Let the similarity scores, provided by the different unimodal systems, be combined into a multimodal score $x = [x_1, \dots, x_R]^T$, where $'$ denotes transpose. The design of a trained fusion scheme consists

in the estimation of a function $f:R^R \rightarrow R$ based on empirical data so as to maximize the separability of client $\{f(x)|\text{client attempt}\}$ and impostor $\{f(x)|\text{impostor attempt}\}$ fused score distributions[12].

The fused score s_T of a multimodal test pattern x_T is defined as follows:

$$s_T = f(x_T) = \langle w^*, \Phi(x_T) \rangle_H + w_0^*, \tag{10}$$

which, applying the Karush-Kuhn-Tucker (KKT) conditions to the problem in (4), (5) and (6) can be shown to be equivalent to the following sparse expression

$$s_T = f(x_T) = \sum_{i \in SV} a_i^* y_i K(x_i, x_T) + w_0^*, \tag{11}$$

where (w^*, w_0^*) is the optimal hyperplane, (a_1^*, \dots, a_N^*) is the solution to the problem in (7), (8), (9), and $SV = \{i | a_i^* > 0\}$ indexes the set of support vectors. w_0^* is obtained from the solution to the problem in (7), (8), (9) by using the KKT conditions.

As a result, the training procedure in (7), (8), (9) and the testing strategy in (11) are obtained for the problem of multimodal fusion.

Let $q = [q_1, \dots, q_R]$ denote the quality value vector of the multimodal and R is the number of modalities, q_r is supposed to be in the range $[0, 1]$, and Q_{max} corresponds to the highest quality. As a result, $q = [q_1, \dots, q_R]$ is computed from quality measures on the fingerprint- or face-based input biometric signals. The proposed quality-guided fusion scheme (from now on also referred to as SVM_Q) is based on using the quality vector q as follows (the bimodal case $R = 2$ is described):

- 1) Training phase: an initial fusion scheme (SVM) is trained as described above by using

$$C_i = C \left(\frac{q_{i,1} q_{i,2}}{Q_{max}^2} \right)^{\alpha_1}, \tag{12}$$

where $q_{i,1}$ and $q_{i,2}$ are the components of the quality vector q_i associated with training sample (x_i, y_i) and C is a positive constant. As a result, the higher the overall quality of a multimodal training scores the higher its contribution to the fusion scheme. Additionally, two SVMs of dimension one (SVM_1 and SVM_2) are trained by using training data from respectively first and second traits. Similarly to Eq.(12), $C_j = C(q_{i,j}/Q_{max})^{\alpha_j}$ for SVM_j with $j=1, 2$.

- 2) Authentication Phase: at this step, the three above-mentioned classifiers SVM , SVM_1 and SVM_2 are trained (i.e., the combining functions $f_{SVM}(\bullet)$, $f_{SVM1}(\bullet)$ and $f_{SVM2}(\bullet)$ introduced in (10) are available). An input multimodal biometric sample with quality vector $q_T = [q_{T,1}, q_{T,2}]$ (suppose $q_{T,1} > q_{T,2}$, otherwise interchange indexes) claims an identity and thus generates a multimodal matching score $x_T = [x_{T,1}, x_{T,2}]$. The combined quality-based matching score is computed as follows:

$$f_{SVM_Q}(x_T) = \beta f_{SVM_1}(x_{T,1}) + (1 - \beta) f_{SVM}(x_T), \tag{13}$$

where

$$\beta = \left(\frac{q_{T,1} - q_{T,2}}{Q_{max}} \right)^{\alpha_2}. \tag{14}$$

4 Experiments Results

4.1 Experimental Setup

To evaluate the performance of the system a database containing fingerprint and face samples was required. The XM2VTS [11] frontal-face-images database was used as the face database and the CQU-Veridicom fingerprint database of chongqing university was used as the fingerprint database. The CQU-Veridicom fingerprint database contains a total of 2,500 fingerprint images (8-bit, 500 dpi, 256×300 array, 256 grey levels.) from 250 individuals with 10 images per individual, which were captured from a set of volunteers in the 20-73 age range (55% male) with a solid-state sensor manufactured by Veridicom. As the fingerprint and the face databases contain samples belonging to different people, a “chimerical” multimodal database was created using pairs of artificially matched fingerprint and face samples that were made for testing purposes.

The database was divided into two sets: the training set and the testing set. The training set consisted of 500 image pairs of 125 people (4 image pairs per person) and was used as a training database for individual modalities, to get the distributions of the unimodal matching scores used in the decision fusion module and to get the weightings associated with different modalities.

The testing dataset consisted of 1000 image pairs of 125 people (8 image pairs per person) and was used exclusively for the evaluation of the system performance. Out of 8 image pairs for each person, 5 were used in the enrolment stage and 3 were used for testing. The tests involved trying to verify every test pair for every one of the 131 people enrolled in the database. This setup makes for 375 (125 x 3) valid client experiments and 46,500 (125 x 3 x 124) impostor experiments.

4.2 Results

In the following, the proposed quality-based multimodal approach ($\alpha_1 = 0.5$, $\alpha_2 = 1$ and $C = 100$) is compared to multimodal fusion without quality assessment ($q=1$ for all signals). Comparative performance results are given in Fig. 4.

Figure 4 shows the performance of the fusion based on the different fusion rules. Fusion of fingerprint and face using the score fusion rule gives a large improvement in the GAR compared to the best single modality (here, fingerprint). The quality-based fusion rule further improves the GAR. For example, at a FAR of 0.1%, the GAR of the fingerprint modality is 84%, while the GAR of the score and quality-based product fusion rules are 94.5% and 98.0%, respectively. From figure 2, we also observe that the performance of the quality-based fusion rule is comparable to the score normalization fusion method proposed.

Figure 5 shows the biometric samples of a user whose face images are of good quality ($q_1 = 0.8$), but fingerprint images are of poor quality ($q_2 = 0.3$). The direct fusion score is low, resulting in a false reject. However, the quality-based fusion rule implicitly assigns a higher weight to the modality with better quality (face). Hence, the quality-based is high and the user is accepted by the multimodal biometric system.

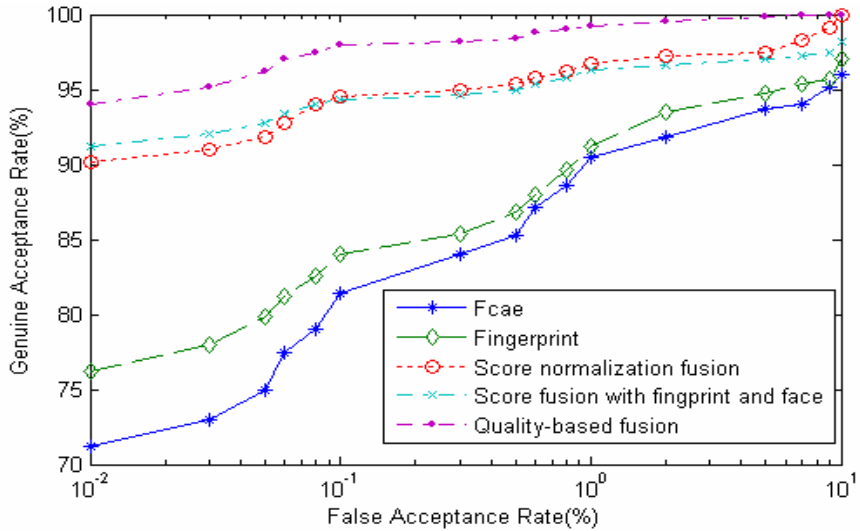


Fig. 4. ROC Curves of the two single modal and multi-modal biometric verification systems

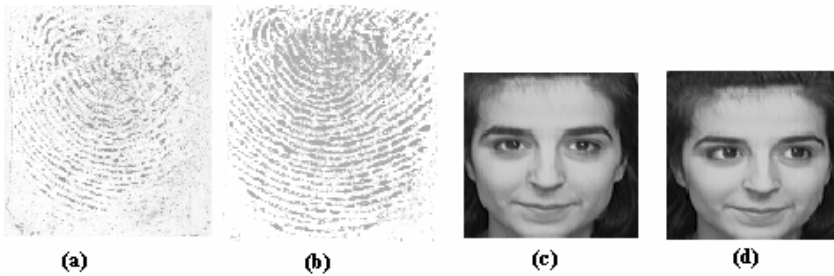


Fig. 5. Illustration of quality-based fusion. This user's fingerprints ((a) template and (b) query) are of poor quality and the face images ((c) template and (d) query) are of good quality. Quality-based fusion rule implicitly assigns a higher weight to the face modality, resulting in a correct acceptance of a genuine user who was falsely rejected by the simple direct fusion rule.

5 Conclusions

“Multimodal Technology makes Biometrics work” – this was the advertising slogan that we have started with [10]. We have proposed a SVM-based scheme to achieve quality-dependent match score fusion. The proposed method does not use any ad-hoc weighting scheme to combine the match scores. Instead, by estimating the quality of fingerprint and face data. The proposed quality-based fusion scheme provides significant improvement in the performance of a multimodal biometric system.

Future work includes the investigation of automatic quality measures for the different biometric signals, the generalization of the proposed scheme to the case of

combining more than two modalities and the comparison of the reported scheme with other quality-based strategies.

Furthermore, considerable advantage of SVM in addition to other methods is the fact that very few parameters need to be fixed or estimated. In other methods the choice of parameters is very difficult and time-consuming and demands often a lot of a priori knowledge about the training data. The choice of the few parameters of an SVM (kernel function/parameter of kernel function/penalty of misclassification) is more general and does not necessarily influence the final results in a considerable way.

Acknowledgments. My heart-felt gratitude first goes to Professor Jianwei Li, my supervisor, for his insightful advice in my paper writing. Furthermore, cordial thanks are extended to cordial thanks are extended to Science& Technology department of Chongqing for offer fund under projects CSTC.2004AA2001- 8277-02.

References

1. Fierrez-Aguilara, J., Ortega-Garcia, J., et al.: Kernel-based Multimodal Biometric Verification using Quality Signals. *Biometric Technologies for Human Identification, Proceedings of SPIE* **5404** (2004) 544-554
2. Lin, H., Anil, J.: Integrating Faces and Fingerprints for Personal Identification. *IEEE transaction on pattern analysis and machine intelligence* **20** (12) (1998) 1295-1307
3. Souheil, B.Y., Yousri, A.: Fusion of Face and Speech Data for Person Identity Verification. *IEEE transaction on Neural Networks* **10** (5) (1999) 1065-1074
4. Faundez-Zanuy, M.: Data Fusion in Biometrics. *IEEE A&E SYSTEMS MAGAZINE* **1** (2005) 34-38
5. Huang, H., Li, J., He, W.: A Fingerprint Capture System and the Corresponding Image Quality Evaluation Algorithm Based on FPS200. *PRICAI 2006, LNAI 4099* (2006) 1058-1062
6. Fronthaler, H., Kollreider, K., Bigun, J.: Automatic Image Quality Assessment with Application in Biometrics. *Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)* (2006) 2646-2652
7. Prabhakar, S., Jain, A.K.: Decision-level Fusion in Fingerprint Verification. *Pattern Recognition* **35** (2002) 861-874
8. Vapnik, V. N.: *Statistical Learning Theory*. Wiley Interscience (1997)
9. Stitson, M.O., Weston, J.A.E., Gammerman, A., Vovk, V., Vapnik, V.: *Theory of Support Vector Machines*. Royal Holloway Technical Report CSD-TR-96-17 (1996)
10. Aggarwal, G., Nalini, K. R., Verification, B.: Looking Beyond Raw Similarity Scores. *Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop(CVPRW'06)* (2006)
11. Messer, K., Matas, J., Kittler J., Luettin, J., Maitre, G.: XM2VTSDB: The Extended M2VTS Database. *Second International Conference on Audio and Video-based Biometric Person Authentication (AVBPA'99)*, Washington D.C. (1999) 72-77

Efficient Motion Estimation Scheme for H.264 Based on BP Neural Network

Wei Zhou¹, Haoshan Shi¹, Zheming Duan¹, and Xin Zhou²

¹ Department of Electronic Engineering, Northwestern Polytechnical University, XI'AN

² Department of Computer Science & Engineering, Northwestern Polytechnical University, XI'AN

Abstract. In the new H.264 video coding standard, motion estimation takes up a significant encoding time especially when using the straightforward full search algorithm (FS). We present an efficient scheme based on BP neural network algorithm which we believe can overcome to a significant degree this shortcoming. The mean squared error (MSE) between the current block and the same position in the reference frame is an often used matching criteria in block matching process. The scheme presented is very well suited to neural network training where the performance index is the mean squared error. The experimental results in Table 1 and Table 2 in the full paper compare our method with the full search algorithm. These comparisons show preliminarily but clearly that our method dose overcome to a significant degree the shortcoming of FS mentioned at the beginning of this abstract with neglectable coding efficiency loss.

Keywords: Motion Estimation; mean squared error; BP neural network.

1 Introduction

Motion Estimation (ME) is an important part of any video compression system, since it can achieve significant compression by exploiting the temporal redundancy existing in a video sequence. Unfortunately it is also the most computationally intensive function of the entire encoding process. In motion estimation the current image is divided into Macro-Blocks (MB) and for each MB, a similar one is chosen in a reference frame, minimizing a distortion measure. The best match found represents the predicted MB, while the displacement from the original MB to the best match gives the so-called Motion Vector (MV). Only the MV and the residual need to be encoded and transmitted into the final stream.

Full Search Block-Matching motion estimation is the technique suggested in the reference software models of all the previous video coding standards, such as MPEG-1/2/4 and H.261/3. The FS algorithm exhaustively checks all the macro-blocks in the SW, thus finding always the optimum match, but it is the most computationally intensive ME algorithm possible and the most CPU-consuming part of the entire encoding process.

In the new, emerging H.264 standard^[1], each 16x16 pixels MB can be sub-partitioned into smaller blocks, down to sizes of 4x4 pixels. This feature gives the ME process the ability to adapt to the local characteristics of the image, but it makes ME even more computationally intensive than in the case of other previous standards. It has been proven that with generalized predictors selection, zonal search and early termination criteria it is possible to achieve almost the same visual quality of the FS, while doing a dramatically reduced number of matches between blocks.

In this paper we present an efficient scheme based on BP neural network algorithm for H.264 encoding schemes. The high computational demands of the H.264 video encoder are reduced by the scheme presented.

2 Block Matching

In the popular video coding standards (H.261, H.263, MPEG-1, MPEG-2 and MPEG-4), motion estimation and compensation are carried out on 8 x 8 or 16 x 16 blocks in the current frame. Motion estimation of complete blocks is known as block matching.

For each block of luminance samples (say 16 x 16) in the current frame, the motion estimation algorithm searches a neighbouring area of the reference frame for a ‘matching’ 16 x 16 area. The best match is the one that minimizes the energy of the difference between the current 16 x 16 block and the matching 16 x 16 area. The area in which the search is carried out may be centred around the position of the current 16 x 16 block, because (a) there is likely to be a good match in the immediate area of the current block due to the high similarity (correlation) between subsequent frames and (b) it would be computationally intensive to search the whole of the reference frame.

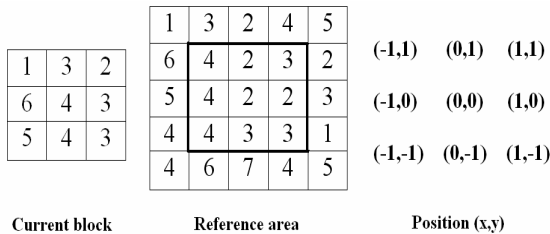


Fig. 1. Current 3 x 3 block and 5 x 5 reference area

Figure 1 illustrates the block matching process. The current ‘block’ (in this case, 3 x 3 pixels) is shown on the left and this block is compared with the same position in the reference frame (shown by the thick line in the centre) and the immediate neighbouring positions (± 1 pixel in each direction). The mean squared error (MSE) between the current block and the same position in the reference frame (position (0, 0)) is given by

$$\left[\begin{aligned} & (1-4)^2 + (3-2)^2 + (2-3)^2 + (6-4)^2 + (4-2)^2 \\ & + (3-2)^2 + (5-4)^2 + (4-3)^2 + (3-3)^2 \end{aligned} \right] / 9 = 2.44 \tag{1}$$

3 Block-matching Scheme Based on BP Neural Network Algorithm

The algorithm used is a variation of Newton's method and Levenberg-Marquardt algorithm that was designed for minimizing function that are sums of squares of nonlinear functions^[2]. This is very well suited to neural network training where the performance index is the mean squared error.

Let's begin by considering the form of newton's method where the performance index is a sum of squares. Newton's method for optimizing a performance index $F(x)$ is

$$x_{k+1} = x_k - A_k^{-1} g_k, \quad (2)$$

where $A_k \equiv \nabla^2 F(x)|_{x=x_k}$ and $g_k \equiv \nabla F(x)|_{x=x_k}$.

If we assume that $F(x)$ is a sum of squares function:

$$F(x) = \sum_{i=1}^N v_i^2(x) = v^T(x)v(x). \quad (3)$$

Then the j th element of the gradient would be

$$[\nabla F(x)]_j = \frac{\partial F(x)}{\partial x_j} = 2 \sum_{i=1}^N v_i(x) \frac{\partial v_i(x)}{\partial x_j}. \quad (4)$$

The gradient can therefore be written in matrix form:

$$\nabla F(x) = 2J^T(x)v(x), \quad (5)$$

where $J(x)$ is the Jacobian matrix. Next we want to find the Hessian matrix. The k, j element of the Hessian matrix would be

$$[\nabla^2 F(x)]_{k,j} = \frac{\partial^2 F(x)}{\partial x_k \partial x_j} = 2 \sum_{i=1}^N \left\{ \frac{\partial v_i(x)}{\partial x_k} \frac{\partial v_i(x)}{\partial x_j} + v_i(x) \frac{\partial^2 v_i(x)}{\partial x_k \partial x_j} \right\}. \quad (6)$$

The Hessian matrix can then be expressed in matrix form:

$$\nabla^2 F(x) = 2J^T(x)J(x) + 2S(x), \quad (7)$$

where

$$S(x) = \sum_{i=1}^N v_i(x) \nabla^2 v_i(x). \quad (8)$$

If we assume that $S(x)$ is small, we can approximate the Hessian matrix as

$$\nabla^2 F(x) = 2J^T(x)J(x). \tag{9}$$

This leads to the Levenberg-Marquardt algorithm according to Gauss-Newton^[3]:

$$x_{k+1} = x_k - [J^T(x_k)J(x_k) + \mu_k I]^{-1} J^T(x_k)v(x_k). \tag{10}$$

This algorithm has the very useful feature that as μ_k is increased it approaches the steepest descent algorithm with small learning rate:

$$x_{k+1} \cong x_k - \frac{1}{\mu_k} J^T(x_k)v(x_k) = x_k - \frac{1}{2\mu_k} \nabla F(x). \tag{11}$$

Now let’s see how we can apply the Levenberg-Marquardt algorithm to the multilayer network training problem. The performance index for multilayer network training is the mean squared error. If each target occurs with equal probability, the mean squared error is proportional to the sum of squared errors over the Q targets in the training set:

$$F(x) = \sum_{q=1}^Q (t_q - a_q)^T (t_q - a_q) = \sum_{q=1}^Q e_q^T e_q = \sum_{q=1}^Q \sum_{j=1}^{S^M} (e_{j,q})^2 = \sum_{i=1}^N (v_i)^2, \tag{12}$$

where $e_{j,q}$ is the j th element of the error for the q th input/target pair.

The backpropagation process computed the sensitivities through a recurrence relationship from the last layer backward to the first layer. We can use the same concept to compute the terms needed for the Jacobian matrix if we define a new Marquardt sensitivity:

$$\tilde{s}_{i,h}^m \equiv \frac{\partial v_h}{\partial n_{i,q}^m} = \frac{\partial e_{h,q}}{\partial n_{i,q}^m}, \tag{13}$$

where, $h = (q - 1)S^M + k$.

Now we can compute elements of the Jacobian by

$$[J]_{h,l} = \frac{\partial v_h}{\partial x_l} = \frac{\partial e_{k,q}}{\partial w_{i,j}^m} = \frac{\partial e_{k,q}}{\partial n_{i,q}^m} \times \frac{\partial n_{i,q}^m}{\partial w_{i,j}^m} = \tilde{s}_{i,h}^m \times \frac{\partial n_{i,q}^m}{\partial w_{i,j}^m} = \tilde{s}_{i,h}^m \times \tilde{a}_{j,q}^{m-1}. \tag{14}$$

The Marquardt sensitivities can be computed through the same recurrence relations as the standard sensitivities with one modification at the final layer, which for standard back propagation is computed with Eq.(13). For the Marquardt sensitivities at the final layer we have

$$\tilde{s}_{i,h}^M = \frac{\partial v_h}{\partial n_{i,q}^M} = \frac{\partial e_{k,q}}{\partial n_{i,q}^M} = \frac{\partial (t_{k,q} - a_{k,q}^M)}{\partial n_{i,q}^M} = \frac{\partial a_{k,q}^M}{\partial n_{i,q}^M} = \begin{cases} -f^M(n_{i,q}^M), \text{ for, } i = k \\ 0, \text{ for, } i \neq k \end{cases}. \tag{15}$$

Therefore when the input P_q has been applied to the network and the corresponding network output a_q^M has been computed, the Levenberg-Marquardt back propagation is initialized with

$$\tilde{S}_q^M = -\dot{F}^M(n_q^M), \quad (16)$$

where $F^M(n^M)$ is defined in Eq.(12). Each column of the matrix \tilde{S}_q^M must be back propagated through the network produce one row of the Jacobian matrix. The column can also be back propagated together using

$$\tilde{S}_q^M = -\dot{F}^m(n_q^m)(W^{m+1})^T \tilde{S}_q^{m+1}. \quad (17)$$

The total Marquardt sensitivity matrices for each layer are the created by augmenting the matrices computed for each input:

$$\tilde{S}^m = [\tilde{S}_1^m | \tilde{S}_2^m | \dots | \tilde{S}_Q^m]. \quad (18)$$

So, the iterations of the back propagation algorithm can be summarized as follows:

- (1) Present all inputs to the network and compute the corresponding network outputs (using Eq.(13)) and the errors $e_q = t_q - a_q^M$. Compute the sum of squared errors over all inputs, $F(x)$, using Eq.(12).
- (2) Compute the Jacobian matrix. Calculate the sensitivities with the recurrence relations Eq.(17), after initializing with Eq.(16). Augment the individual matrices into the Marquardt sensitivities using Eq.(18). Compute the elements of the Jacobian matrix with Eq.(14).
- (3) Solve Eq.(10) to obtain Δx_k .
- (4) Recompute the sum of squared errors using $x_k + \Delta x_k$. If this new sum of squares is smaller than that computed in step1, then divide μ by ϑ , let $x_{k+1} = x_k + \Delta x_k$ and go back to step1. If the sum of squares is not reduced, then multiply μ by ϑ and go back to step3.

The algorithm is assumed to have converged when the norm of the gradient, Eq.(5), is less than some predetermined value, or when the sum of squares has been reduced to some error goal.

4 Results

We present the experimental results with the algorithm implemented into the reference software model JM 11.0. In Table 1, a comparison between Block-matching scheme based on BP neural network algorithm (BPNN) and FS is reported for QCIF and CIF sequences compressed at QP=31, Intra period=12 and two different SR

values (16 and 32). As the tests were made at constant QP, the most important result is the length of the encoded bit stream (in Bytes/Picture), which indicates the compression rate achieved for the given visual quality QP parameter. The 6 column reports the per cent increase of bytes-per-picture for BPNN against FS. Table 1 also shows the average Peak Signal-to-Noise Ratio (PSNR) and the number of matches performed per block (NM column). For FS the latter is a constant value equal to $(2*SR+1)^2$, whereas for BPNN it is an average value.

Table 1. Comparison Between FS and BPNN

Seq	SR	Algorithm	B/Pict	PSNR	NM
Foreman QCIF	16	FS	466.70	33.86	1083
	16	BPNN	471.96	33.74	10.7
	32	FS	466.58	33.91	4217
	32	BPNN	471.91	33.82	10.5
Football CIF	16	FS	3052.89	32.81	1084
	16	BPNN	3160.41	32.69	10.6
	32	FS	3043.36	32.87	4217
	32	BPNN	3144.81	32.75	10.9

Table 2. Comparison Between FS and BPNN at 64kbit/s

Seq	SR	Algorithm	PSNR	NM
Foreman QCIF	16	FS	30.87	1083
	16	BPNN	30.54	10.6
	32	FS	31.02	4217
	32	BPNN	30.84	10.1
Teeny QCIF	16	FS	25.33	1084
	16	BPNN	25.12	9.7
	32	FS	25.16	4217
	32	BPNN	25.04	10.2

Tables 2 presents a comparison between the two algorithms made at constant bit rate. Table 2 refers to QCIF sequences encoded at 64 kbit/s. As it can be seen in the two tables, BPNN ensures a PSNR very close to the one of FS at low bit rates, and a speedup factor around 100 with Search Range of 16 pixels, and around 400 with Search Range of 32 pixels.

5 Conclusion

In this paper, an efficient motion estimation technique suitable for H.264/AVC compression schemes is implemented. The high computational demands of the H.264 video encoder are reduced by the block-matching algorithm based on BP neural network algorithm. Our results demonstrate that the efficient motion estimation algorithm presented can produce similar and in some cases better results compared to even FS algorithm while having considerably lower complexity.

References

1. Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC) - Joint Committee Draft. Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG [C], 7th Meeting: Pattaya, Thailand (2003) 7-14
2. Martin, T.Hagan, Howard, B.Demuth, Mark, Beale: Neural Network Design [M]. Bei Jing: China Machine Press (2003) 1219-1249
3. Scales, L.E.: Introduction to Non-Linear Optimization. New York: Springer-Verlag (1985)
4. Xue, Jin-zhu, Shen, Lan-sun: An Efficient Block-matching Motion Estimation Algorithm for H.264/AVC [J]. Acta Electronic Sinica **32** (4) (2004) 583-586
5. He, Gui-ming, Wu, Bao-yuan, et al: The Technology of Video Coding and Communications Control Based on Object [M]. Wu Han: Publishing House of Wu Han University (2005) 177-188
6. Hosur, P. I., Ma, K. K.: Motion Vector Field Adaptive Fast Motion Estimation [C]. Proc. 2nd Int. Conf. Information, Communications and Signal Processing (ICICS'99), Singapore (1999) 7-10

Neural Network Based Visual Tracking with Multi-cue Adaptive Fusion

Yongwei Li, Shiqiang Hu, and Peng Guo

College of Electrical and Information Engineering, Hebei University of Science and Technology, Shijiazhuang 050018, China
05991i@163.com, sqhu@sjtu.edu.cn, guopeng993@163.com

Abstract. Visual tracking has been an active area of research in computer vision. However, robust tracking is still a challenging task due to cluttered backgrounds, occlusions and pose variations in the real world. To improve the tracking robustness, this paper proposes a tracking method based on multi-cue adaptive fusion. In this method, multiple cues, such as color and shape, are fused to represent the target observation. When fusing multiple cues, fuzzy logic is adopted to dynamically adjust each cue weight in the observation according to its associated reliability in the past frame. In searching and tracking object, neural network algorithm is applied, which improves the searching efficiency. Experimental results show that the proposed method is robust to illumination changes, pose variations, partial occlusions, cluttered backgrounds and camera motion.

1 Introduction

Visual tracking has become a popular topic in the field of computer vision. Its potential applications include smart surveillance, virtual reality, perceptual interface, video conferencing, etc. Although visual tracking has been intensively studied in the literature, developing an algorithm that is robust to a wide variety of conditions is still an open problem.

Visual tracking can be considered to match coherent relations of image features between frames. In the last decades, various algorithms have been proposed, such as Kalman filter [1], Condensation [2], Mean Shift [3], Cam Shift [4], etc. However, most of them are based on a single image cue. It is clear that no single image cue can be robust enough to successfully deal with various conditions occurring in the real-world environments. Shape-based trackers or color-based trackers, for example, are distracted by other targets with similar shape or color characteristics. To overcome the weak robustness of single-cue tracking, many algorithms have been proposed based on multi-cue fusion. The fusion of multiple cues not only can provide more reliable observation when estimating a state, but different cues may be complementary in that one may succeed when another fails. The key challenge for this kind of algorithm is how to optimally fuse multiple cues. In most algorithms [5-8], the fusion scheme is non-adaptive, in which the reliability of each cue is assumed to be unchanged during

the tracking. However, such assumption is often invalid due to the dynamically changing environments. In this paper we propose a tracking method based on multi-cue adaptive fusion. In this method, color and shape cues are fused to represent the target observation. During the tracking, fuzzy logic is applied to dynamically adjust the each cue weight in the observation. By employing fuzzy logic, multiple cues are adaptively fused, which greatly increases the reliability of the observation. In searching and tracking object, genetic algorithm is applied, which improves the searching efficiency. Experimental results show that the proposed method is robust to illumination changes, pose variations, partial occlusions, cluttered backgrounds and camera motion.

2 Algorithm Description

In this paper, target tracking is implemented by using genetic searching. We adopt genetic algorithm (GA) to find the candidate target in each frame that best matches the target model. The target observation is represented by the color and shape cues. The GA mainly depends on the observation similarity between the target model and the candidate target to find the target in each frame. During the tracking process, the color and shape cue may change due to the target movement and the dynamically changing backgrounds, so in this paper fuzzy logic is adopted to adaptively fuse color and shape cue, which greatly increases the reliability of the observation. The whole tracking process is shown in Figure 1.

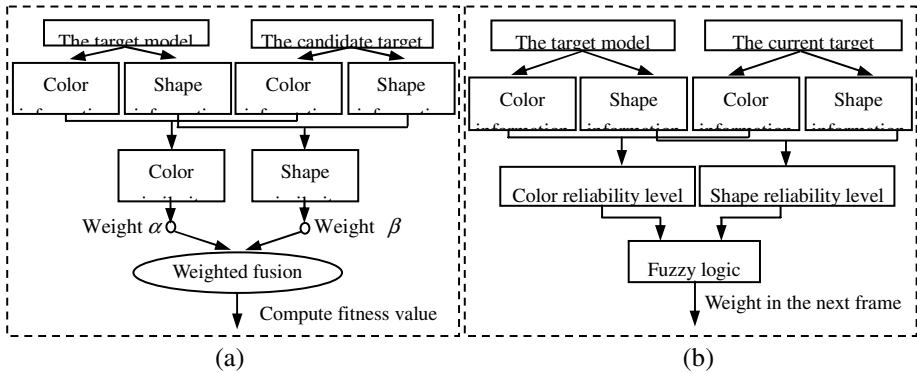


Fig. 1. Flowchart of the tracking process

3 Observation Representation

3.1 Color Cue

Color distribution of the target is represented by color histogram. In our experiments, color histogram is calculated with m ($m=8 \times 8 \times 8$) bins in RGB space. We only consider the color distribution inside the target region. In this paper, the contour of the target is

approximated by an ellipse, so the target region is an elliptic region. Assume the target region (elliptic region) is centered at $x = (x, y)$ with a size of $\mathbf{h} = (h_x, h_y)$, where h_x and h_y are the lengths of two half axes of the elliptic region, and let $\mathbf{x}_i = (x_i, y_i), i = 1, \dots, n_h$, be the locations of the pixels in the target region, then the color distribution can be calculated as

$$\hat{p}^{(u)}(\mathbf{x}, \mathbf{h}) = \frac{\sum_{i=1}^{n_h} k\left(\left\|\frac{\mathbf{x} - \mathbf{x}_i}{\mathbf{h}}\right\|\right) \delta[b(\mathbf{x}_i) - u]}{\sum_{i=1}^{n_h} k\left(\left\|\frac{\mathbf{x} - \mathbf{x}_i}{\mathbf{h}}\right\|\right)}, \quad u = 1, \dots, m, \tag{1}$$

where δ is the Kronecker delta function, $b(\mathbf{x}_i)$ is a function which associates the pixel at location \mathbf{x}_i with the bin index $b(\mathbf{x}_i)$ of the histogram, $k(\cdot)$ is a weighting function that has the following form

$$k(\|\mathbf{r}\|) = \begin{cases} 1 - \|\mathbf{r}\|^2, & \text{if } \|\mathbf{r}\| < 1, \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

If we denote $\hat{q} = \{\hat{q}^{(u)}\}_{u=1, \dots, m}$ as the color distribution of the target model and $\hat{p}(\mathbf{x}, \mathbf{h}) = \{\hat{p}^{(u)}(\mathbf{x}, \mathbf{h})\}_{u=1, \dots, m}$ as a candidate target, the similarity between \hat{q} and $\hat{p}(\mathbf{x}, \mathbf{h})$ can be measured by the Bhattacharyya distance [3]

$$d_c[\hat{q}, \hat{p}(\mathbf{x}, \mathbf{h})] = \sqrt{1 - \rho[\hat{q}, \hat{p}(\mathbf{x}, \mathbf{h})]}, \tag{3}$$

where $\rho[\hat{q}, \hat{p}(\mathbf{x}, \mathbf{h})]$ is the Bhattacharyya coefficient that has the following form

$$\rho[q, p(\mathbf{x}, \mathbf{h})] = \sum_{u=1}^m \sqrt{\hat{q}^{(u)} \hat{p}^{(u)}(\mathbf{x}, \mathbf{h})}, \tag{4}$$

After obtaining a distance d_c on the RGB color histogram, we define a color likelihood function as follows

$$p(\mathbf{z}_c | \mathbf{x}, \mathbf{h}) = \frac{1}{\sqrt{2\pi}\sigma_c} \exp\left(-\frac{d_c^2[\hat{q}, \hat{p}(\mathbf{x}, \mathbf{h})]}{2\sigma_c^2}\right), \tag{5}$$

where σ_c is the Gaussian variance. In our experiments, σ_c is selected as 0.2. Equation (5) shows that the larger $p(\mathbf{z}_c | \mathbf{x}, \mathbf{h})$ is, the more similarity between two histograms.

3.2 Shape Cue

In this paper, the tracked target is a human head. In general, the human head can be approximated by an ellipse. So we use an ellipse to model the contour of the head in our experiments. It means that the shape template is an ellipse.

We use Chamfer distance to measure the similarity of the shape between the template and the candidate target. Given a binary image T of the shape template, a binary image I (Figure 2(b)), and the distance image DI of I (Figure 2(c)), about the distance image, the reader can refer to [9],[10] for more details, and assume the candidate target is at location $\mathbf{x} = (x, y)$ in the image I , then the Chamfer distance between the shape template and the candidate target can be calculated as

$$d_s[T(\mathbf{x}, \mathbf{h}), I] = \sqrt{\frac{1}{|T|} \sum_{t \in T} DI(t)}, \quad (6)$$

where $T(\mathbf{x}, \mathbf{h})$ denotes the shape template T with a size of $\mathbf{h} = (h_x, h_y)$ centered at $\mathbf{x} = (x, y)$ in the image I , $|T|$ denotes the number of features in the image T , and $DI(t)$ is the value of the pixel in the distance image DI which lies under the t th feature (pixel) of the image T . After obtaining the Chamfer distance between two shapes, we define a shape likelihood function as follows

$$p(\mathbf{z}_s | \mathbf{x}, \mathbf{h}) = \frac{1}{\sqrt{2\pi}\sigma_s} \exp\left(-\frac{d_s^2[T(\mathbf{x}, \mathbf{h}), I]}{2\sigma_s^2}\right), \quad (7)$$

where σ_s is the Gaussian variance. We set σ_s as 0.5 in our experiments. Equation (7) shows that the larger $p(\mathbf{z}_s | \mathbf{x}, \mathbf{h})$ is, the more similarity between two shapes.

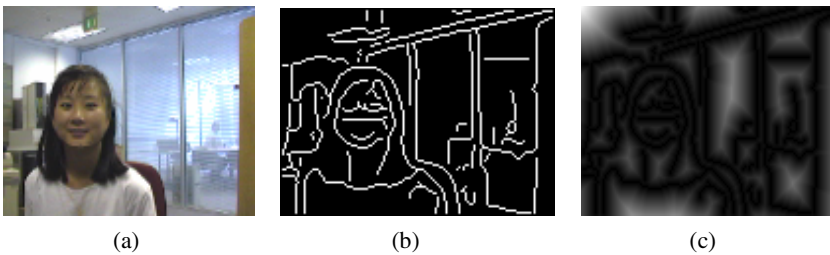


Fig. 2. (a) original image (b) binary image (c) distance image

3.3 Multi-cue Adaptive Fusion

The observation information is represented in this paper by two kinds of image cues: color cue and shape cue. Assume the n th candidate target region (elliptic region) has a size of $\mathbf{h}_n = (h_x^{(n)}, h_y^{(n)})$ and is centered at $\mathbf{x}_n = (x^{(n)}, y^{(n)})$ in the image I , then the entire observation likelihood function of the n th candidate target can be calculated as

$$p(\mathbf{z} | \mathbf{x}_n, \mathbf{h}_n) = \alpha p(\mathbf{z}_c | \mathbf{x}_n, \mathbf{h}_n) + \beta p(\mathbf{z}_s | \mathbf{x}_n, \mathbf{h}_n), \alpha + \beta = 1, \tag{8}$$

where $p(\mathbf{z}_c | \mathbf{x}_n, \mathbf{h}_n)$ and $p(\mathbf{z}_s | \mathbf{x}_n, \mathbf{h}_n)$ are the likelihoods of the color and shape cues, respectively, which are defined in equation (5) and equation (7), $0 \leq \alpha \leq 1$ is the weight of the color cue, $0 \leq \beta \leq 1$ is the weight of the shape cue. The cue weights in most algorithms are assumed to be unchanged during the tracking. However, such assumption is often invalid in practice. Instead of assuming the fixed weights, we use fuzzy logic to adjust the weights dynamically according to the former reliability of the cue.

It is worth pointing out here that the proposed fusion scheme is an open framework into which other cues can be easily included too.

3.3.1 Adjusting Cue Weights Using Fuzzy Logic

The main parts of the fuzzy logic are fuzzification, fuzzy rule base, fuzzy inference, and defuzzification. In this paper, the fuzzy logic is designed based on the singleton fuzzification, product inference, and centroid defuzzification, and the fuzzy rule is defined as

$$R^j : \text{If } e_1 \text{ is } A_1^j \text{ and } e_2 \text{ is } A_2^j \dots \text{ and } e_l \text{ is } A_l^j \text{ then } u \text{ is } B^j, j=1, \dots, L, \tag{9}$$

where R^j is the j th fuzzy rule, L is the total number of the fuzzy rule, $\mathbf{e} = (e_1, \dots, e_l)^T$ and u are the input and output of the fuzzy logic, respectively, A_i^j and B^j are fuzzy linguistic terms characterized by membership function.

In our experiments, the inputs of the fuzzy logic are the reliability levels of the color and shape cues in the current frame, while the output is the color weight α in the next frame. The shape weight β is calculated by $\beta = 1 - \alpha$. We denote the reliability levels of the color and shape cues by e_c and e_s , respectively, which can be calculated by equation (5) and equation (7). The fuzzy sets of e_c and e_s are both {SR, S, M, B, BR} and α is {ST, VS, SR, S, M, B, BR, VB, BT}, where ST stands for smallest, VS for very smaller, SR for smaller, S for small, M for middle, B for big, BR for bigger, VB for very bigger, BT for biggest. The membership functions of e_c , e_s and α are all Gaussian functions, which are shown in Figure 3. The fuzzy rule base is shown

in Table 1. By employing fuzzy logic, the unreliable cues are suppressed quickly while cues that have proved to be reliable in the recent past are given larger weights.

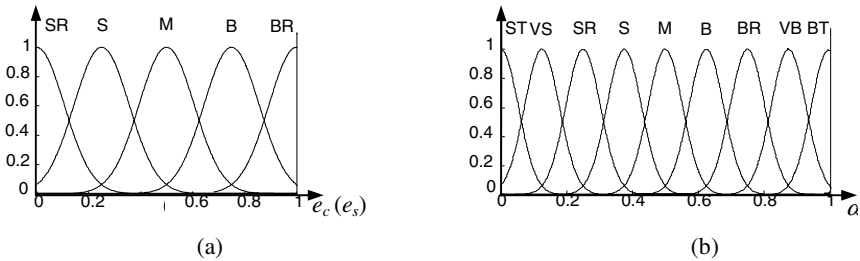


Fig. 3. (a) Membership functions for e_c and e_s , (b) Membership function for α

Table 1. Fuzzy rule base

α		e_c				
		SR	S	M	B	BR
e_s	SR	M	B	BR	VB	BT
	S	S	M	B	BR	VB
	M	SR	S	M	B	BR
	B	VS	SR	S	M	B
	BR	ST	VS	SR	S	M

4 Target Searching Based on Neural Network Algorithm

Neural network algorithm is a powerful searching algorithm. It has proved to be a robust and efficient way of solving optimization problem. In this paper, we adopt neural network algorithm to find the target in the current frame. Assume the target state is denoted by $S = (x, y, h_x, h_y)$, where (x, y) is the position center of the target and (h_x, h_y) is the size, then the parameters in S are the searched parameters.

Given an input-output pair $(f_{ij}, net_{0j}), i = 1,2,3; j = 1, \dots, p$, our task is to train the neural network off-line by the back-propagation learning algorithm with a momentum term such that

$$E = \frac{1}{2} \sum_i \sum_j (net_{0j} - f_{ij})^2, \tag{10}$$

is minimized. Letting w_{ij} be the adjustable weights of the neural network, the training rule is

$$\Delta w_{ij}(k) \propto -\frac{\partial E}{\partial w_{ij}}, \tag{11}$$

$$w_{ij}(k+1) = w_{ij}(k) + \alpha \left(-\frac{\partial E}{\partial w_{ij}} \right) + \eta \Delta w_{ij}(k), \tag{12}$$

where $\alpha \in (0,1)$ is the learning rate, and $\eta \in (0,1)$ is a momentum factor.

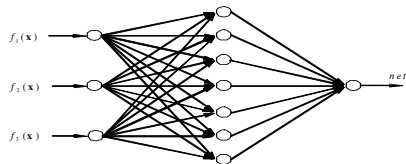
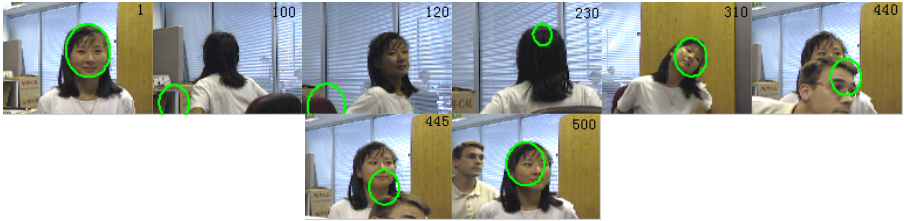


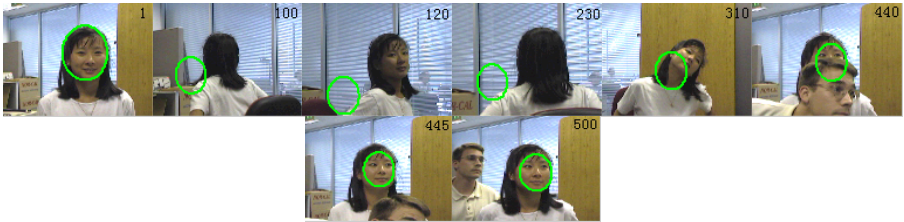
Fig. 4. The structure of three-layer feed forward neural network

5 Experimental Results and Analysis

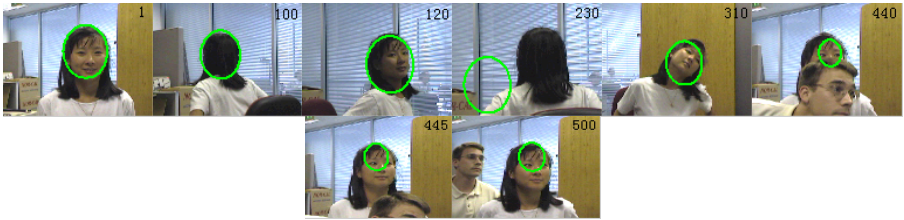
In this section we evaluate our method on a sequence image with 500 frames¹. This sequence simulates various tracking conditions, including illumination changes, pose variations, partial occlusions, cluttered backgrounds and camera motion. In the experiments, the target model is initialized by hand in the first frame. We present the comparison between the results obtained by four kinds of methods. The tracking results



(a) Tracking results based on color cue



(b) Tracking results based on shape cue



(c) Tracking results based on multi-cue non-adaptive fusion



(d) Tracking results based on multi-cue adaptive fusion

Fig. 5. Tracking results using four kinds of methods

¹ <http://vision.stanford.edu/~birch>

are shown in Figure 5. In image (a), the tracking is only based on color cue. This method can succeed when target color has no great change. But if target color changes dramatically, the tracking will be lost, such as the 100th frame. Image (b) shows the tracking result based on shape cue. This method can work well in simple background. However, when the background is cluttered, such as the 100th and 120th frame, the tracking will fail. Image (c) shows the tracking results using multiple cues. We can see that the tracking results are better than image (a) and (b). But the tracking still fails in the 230th frame. It's mainly because that in this method, the reliability of each cue is assumed to be unchanged during the tracking. Such assumption is invalid in our experiments due to the head rotation and the camera moving. Image (d) gives the results of our method. The tracking results show that our method tracks the head very robustly and accurately throughout the whole sequence image. This fact can be explained by multi-cue adaptive fusion. Figure 5 shows the evolution of the color and shape cue. We can see that the proposed fusion scheme can successfully suppress the cue that is unreliable for tracking. For example, at about the 65th frame, the color cue becomes unreliable due to the head rotation, while the shape cue can work well, so the color weight automatically decreases and the shape weight correspondingly increases to keep tracking robustly. At about the 270th frame, the color weight begins to increase, it's because that the color cue becomes reliable due to the face turning to camera again. From about the 420th to the 480th frame, the weights change dramatically, the reason behind this is that during this period the target is partially occluded by another face.

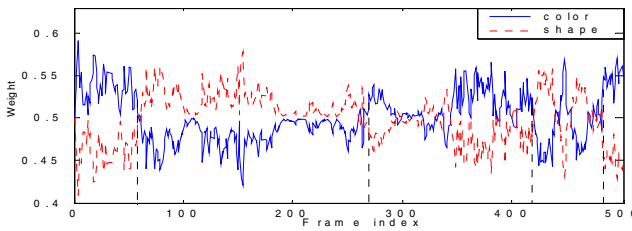


Fig. 6. Weight curves of the color and shape cues

6 Conclusions

In this paper we proposed a tracking method based on multi-cue adaptive fusion. In this method, the color and shape cue are fused to represent the target observation. When fusing multiple image cues, an adaptive fusion scheme is adopted. In this fusion scheme, fuzzy logic is applied to dynamically adjust the weight of each cue, which greatly improves the reliability of the observation. The experimental results show that with adaptive fusion of multiple image cues, the tracker becomes more robust to illumination changes, pose variations, partial occlusions, cluttered backgrounds and camera motion.

Acknowledgements

This paper was supported by the National Natural Science Foundation of China (Grant No. 60674107), the Natural Science Foundation of Hebei Province (GrantNo. F2006000343).

References

1. Blake, A., Isard, M., Reynard, D.: Learning to track the visual motion of contours. *Artificial Intelligence* **78** (1995) 179-212
2. Isard, M., Blake, A.: Contour tracking by stochastic propagation of conditional density. In *Proceeding of the 4th European Conference on Computer Vision*, Cambridge, UK, Apr, (1996) 343-356.
3. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **5** (5) (2003)564-577
4. Bradski, G.R.: Computer vision face tracking as a component of a perceptual user interface. In *IEEE Workshop on Applications of Computer Vision*, Princeton, NJ, Oct, (1998) 214-219
5. Birchfield, S.: Elliptical head tracking using intensity gradients and color histograms. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, Santa Barbara, California, USA, Jun (1998) 232-237
6. Tao, X., Christian, D.: Monte Carlo visual tracking using color histograms and a spatially weighted oriented Hausdorff measure. In *Proceedings of the Conference on Analysis of Images and Patterns*, Groningen, The Netherlands, Aug (2003) 190-197
7. Kwolek, B.: Stereovision-based head tracking using color and ellipse fitting in a particle filter. In *Proceedings of the 8th European Conference on Computer Vision*, Prague, Czech Republic, May (2004) 192-204
8. Spengler, M., Schiele, B.: Towards robust multi-cue integration for visual tracking. *Machine Vision and Applications* **14** (2003) 50-58.
9. Huttenlocker, D.P., Klanderman, G. A., Rucklidge, W. J.: Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **15** (9) (1993) 850-863
10. Gavrilu, D.M.: Multi-feature hierarchical template matching using distance transforms. In *the Proceedings of the 14th International Conference on Pattern Recognition*, Brisbane, Australia, Aug (1998) 439-444

One-Class SVM Based Segmentation for SAR Image

Jianjun Yan and Jianrong Zheng

Center for Mechatronics Engineering, East China University of Science and Technology,
Shanghai 200237, P.R. China

Abstract. Image segmentation is of great importance in the field of image processing. A wide variety of approaches have been proposed for image segmentation. However, SAR image segmentation poses a difficult challenge owing to the high levels of speckle noise. In this paper, we proposed a SAR image segmentation method based on one-class support vector machines (SVM) to solve this problem. One-class SVM and two-class SVM for segmentation is discussed. One-class way is a kind of unsupervised learning, and one-class SVM based segmentation method reduces greatly human interactions, while yielding good segmentation results compared to two-class SVM based segmentation method. The segmentation results based on SVM are also compared to threshold method and adaptive threshold method. Experimental results demonstrate that the proposed method works well for image segmentation while reducing the speckle noise.

1 Introduction

Synthetic aperture radar (SAR) is an important use for military reconnaissance and civil activity, so it has practical meaning and application prospect to study feature extraction and object recognition for SAR images. SAR Image segmentation is the most difficult and important preprocess technique in the process of SAR. A wide variety of approaches have been proposed for image segmentation, such as threshold segmentation using maximum entropy, cluster segment and so on. But a major issue in SAR images is that they are generally affected by multiplicative speckle noise which damages radiometric resolution and affects the tasks of human interpretation and scene analysis. So, it is very important to segment SAR image accurately and efficiently. Traditional segmentation methods demand removing noise before segmentation, thus lose inevitably some targets and structural information, affecting the final segmentation result. So traditional segmentation techniques of images cannot obtain desired segmentation results, and affect the recognition rate of target. Recently, a kernel based classification technique: Support Vector Machine (SVM) [1, 2] is widely used in feature recognition. Studies on training algorithms for SVM are important issues in the field of machine learning. It is a challenging task to improve the efficiency of the algorithm without reducing the generalization performance of SVM. This paper is structured as follow; firstly, we briefly introduce two-class SVM, which is common classifier. Then we discuss one-class SVM, and make comparison between one-class classification and two-class classification. Thirdly the experiments

of SAR images segmentation are executed through respectively using traditional methods, one-class SVM based method and two-class SVM based method. Segmentation results are given and analyzed through comparison with other segmentation methods. The final results demonstrate SVM based SAR image segmentation is better than traditional method and one-class SVM is suitable for SAR image segmentation, outperforming the two-class SVM.

2 Support Vector Machine

For the binary classification problems, n-dimensional training vectors x_i labeled by $y \in \{-1, 1\}$. The goal of learning is to find a suitable value of α parameter in the decision function $f(x, \alpha)$, which makes that $f(x, \alpha) = y, \forall x$. In other words, we can get a correct classification result. Suppose we have some hyperplane which separates the positive from the negative examples. The vectors x lie on the hyperplane satisfy $w \cdot x + b = 0$, where w is normal to the hyperplane, $|b|/\|w\|$ is the perpendicular distance from the hyperplane to the origin. Let d_+ (d_-) be the shortest distance from the separating hyperplane to the closest positive (negative) example. Define the “margin” of a hyperplane to be $(d_+ + d_-)$. It is the hyperplane with largest margin that support vector algorithm wants to look for. Suppose that all the training examples satisfy the following constraints:

$$w \cdot x_i + b \geq 1, \quad \text{if } y_i = 1, \tag{1}$$

$$w \cdot x_i + b \leq -1, \quad \text{if } y_i = -1. \tag{2}$$

Now consider the cases that the equality constraints above hold on. The points for which the equality in Eq. (1) holds lie on the hyperplane $H_1: w \cdot x_i + b = 1$ with normal w and perpendicular distance from the origin $|1 - b|/\|w\|$. Similarly, the points for which the equality in Eq. (2) holds lie on the hyperplane $H_2: w \cdot x_i + b = -1$ with normal again w and perpendicular distance from the origin $|-1 - b|/\|w\|$. Hence $d_+ = d_- = 1/\|w\|$ and the margin of the separating hyperplane is simply $d_+ + d_- = 2/\|w\|$. Therefore our task is to make the margin largest, that is to say we try to minimize $\|w\|^2$ subject to constraints (1) and (2).

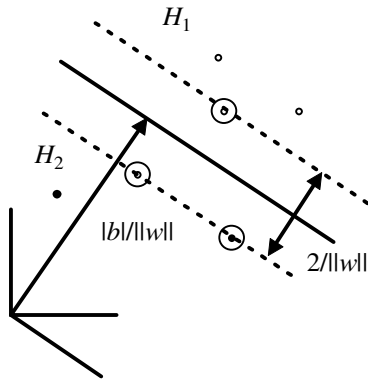


Fig. 1. Linear separating hyperplane for the separable case

There is shown the solution for a typical two dimension case in Fig. 1. Those training examples for which the equalities in constraints (1) and (2) are called support vectors; they are indicated in Fig. 1 by the extra circles.

Consider the points for which the inequalities in constraints (1) and (2) do not hold, which are nonseparable datas. Vapnik and Cortes (1995) introduced positive slack variables (Here, slack variables are the errors of classification):

$$\xi_i \geq 0, i=1, \dots, l. \tag{3}$$

They gave a quadratic convex programming [3]:

$$\begin{aligned} &\text{minimize } \frac{1}{2} \|w\|^2 + C \cdot \sum_{i=1}^l \xi_i, \\ &\text{subject to } y_i((w \cdot x_i) + b) \geq 1 - \xi_i, \dots \\ &\quad \xi_i \geq 0, i = 1, \dots, l, \end{aligned} \tag{4}$$

where $C > 0$ is a parameter to be chosen by user, a larger C corresponding to assigning a higher penalty to errors. Minimizing the first term of objective function gives the maximum margin, while minimizing the second term assures the minimum total error of all training examples.

By using Lagrange multiplier techniques and kernel functions, one can have the following optimization problem:

$$\begin{aligned} &\text{maximize } W(\alpha) = \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l \alpha_i y_i \alpha_j y_j K(x_i \cdot x_j), \\ &\text{subject to } \sum_{i=1}^l \alpha_i y_i = 0, \alpha_i \in [0, C], i = 1, \dots, l. \end{aligned} \tag{5}$$

Therefore we have the decision function of nonlinear support vector machine:

$$f(x) = \text{sgn} \left(\sum_{i=1}^l \alpha_i y_i K(x_i, x) + b \right). \tag{6}$$

If $\alpha_i \neq 0$, the corresponding training example x_i is called support vector. Note that w can be completely described as a linear combination of support vectors. In a sense, the complexity of function's representation by SVs is independent of the dimension of the input space, and depends only on the number of SVs. This is useful for deal with high input dimensional problems.

3 Support Vector Machine Based Segmentation of SAR Image

In two-class classification, the data from two classes are available. Most two-class classifiers assume more or less equally-balanced data classes and thus do not work well when one class is severely undersampled or even completely absent. To achieve a satisfied classified result by using two-class classification, we need to know the number of classes and the distributions of target data and non-target data of SAR images in order to achieve representative and balanced data samples. The requirement is impractical and often impossible to meet. These disadvantages make the conventional segmentation systems not very useful. So the one-class way of

unsupervised learning is also used for segmentation of SAR images. One of the advantages of this one-class SVM based segmentation is that human interactions have been greatly reduced, while yielding better segmentation results compared to other supervised two-class or multi-class based segmentation methods.

One-class SVM was proposed by Schölkopf et al. in 2001 for estimating the support of a high-dimensional distribution. One-class classification is a kind of unsupervised learning mechanism, which trains on unlabelled data, trying to assess whether a test point is likely to belong to the distribution underlying the training data. Given training vectors $x_i \in \mathbb{R}^n, i = 1, \dots, l$ without any class information, the primal form [4] is:

$$\begin{aligned} &\text{minimize } \frac{1}{2} \|w\|^2 + \frac{1}{\nu l} \sum_{i=1}^l \xi_i - \rho, \\ &\text{subject to } \langle w, \Phi(x_i) \rangle \geq \rho - \xi_i, . \\ &\quad \xi_i \geq 0, i = 1, \dots, l. \end{aligned} \tag{7}$$

The dual is:

$$\begin{aligned} &\text{minimize } \frac{1}{2} \alpha^T Q \alpha, \\ &\text{subject to } 0 \leq \alpha_i \leq 1/(\nu l), i = 1, \dots, l, . \\ &\quad e^T \alpha = 1, \end{aligned} \tag{8}$$

where $Q_{i,j} = K(x_i, x_j) \equiv \varphi(x_i)^T \varphi(x_j)$.

The decision function is

$$\text{sgn} \left(\sum_{i=1}^l \alpha_i K(x_i, x) - \rho \right). \tag{9}$$

The parameter ν is of significance, which characterizes the fractions of SVs and outliers. So the selection of the parameter ν is very important. In this paper, polynomial kernel is used to train data for one-class based SVM segmentation for SAR images. Experiments are done by using the LIBSVM [5].

4 Experiments and Results

In this paper, the performances of SAR image segmentation based on one-class SVM and two-class SVM are tested, and are compared with the traditional segmentation methods. The traditional methods used in this paper include threshold segmentation and adaptive threshold segmentation. The segment results of airplane1 and aireplane2 with above methods are shown in Fig.2 and Fig.3, respectively. From the Fig.2 and Fig.3, we can see that the traditional methods do not work well in the presence of spectacle noise, while the SVM based segmentation method can obtain desired result in both extracting object and reducing spectacle noise, outperforming the other methods. The segmentation results of one-class SVM are better than that of two-class SVM. And for one-class SVM based segmentation, the selection of parameter ν is important, and segmentation results with the smaller ν will lose some details.

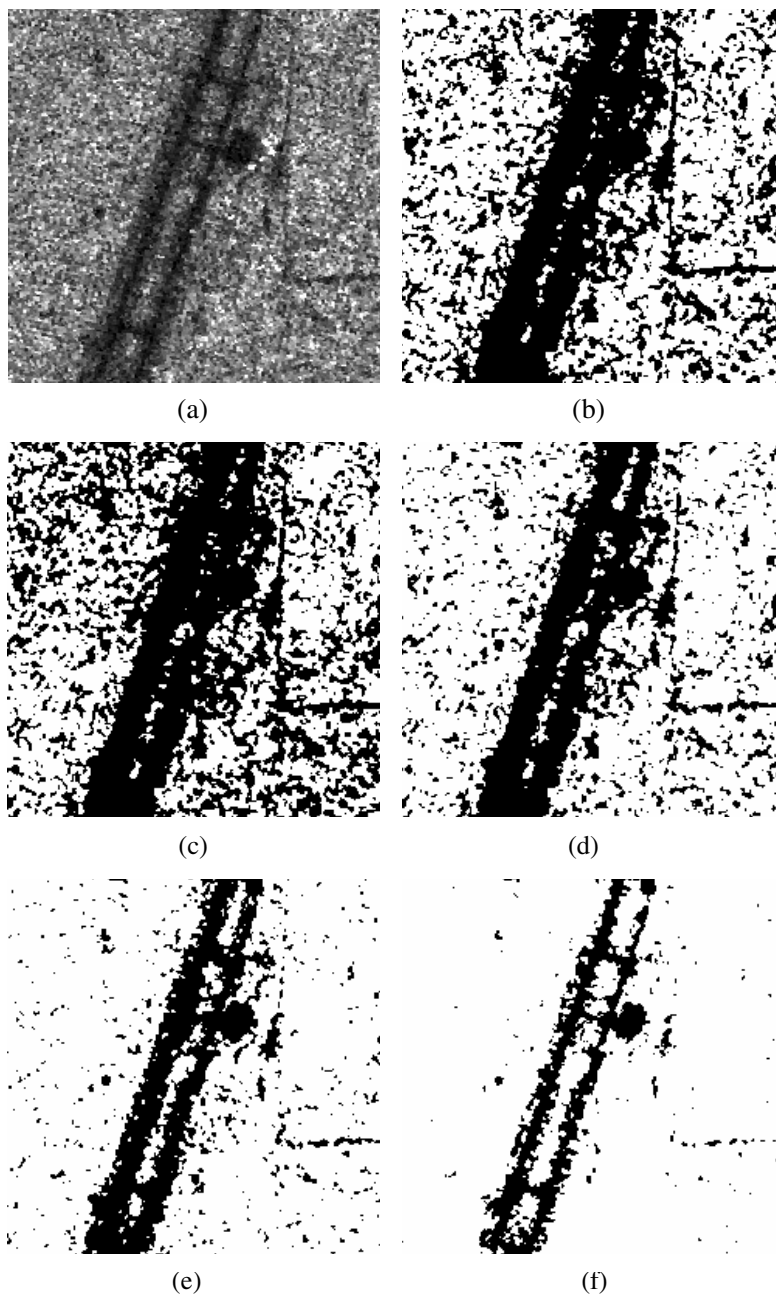


Fig. 2. Segmentation for Airplane1: (a) Original Image, (b) Threshold Segmentation, (c) Adaptive Threshold Segmentation, (d) Two-class SVM Based Segmentation, (e) One-class SVM Based Segmentation with $\nu=0.5$, (f) One-class SVM Based Segmentation with $\nu=0.2$

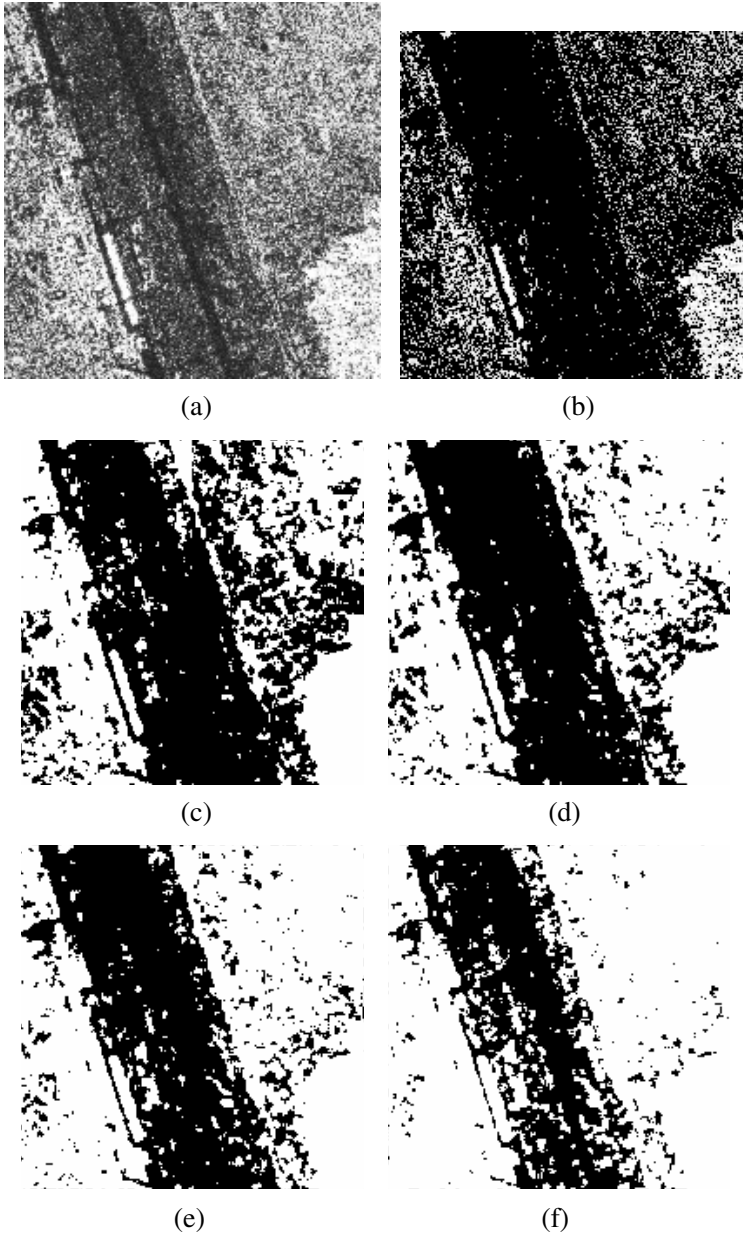


Fig. 3. Segmentation for Airplane2: (a) Original Image, (b) Threshold Segmentation, (c) Adaptive Threshold Segmentation, (d) Two-class SVM Based Segmentation, (e) One-class SVM Based Segmentation with $\nu=0.5$, (f) One-class SVM Based Segmentation with $\nu=0.3$

For segmentation of SAR image, there is not a general evaluation standard. In this paper, the grey-level contrast and region non-uniformity are applied to evaluate the

segmentation results. In evaluation of grey-level contrast (GC) [6], for two adjacent regions which grey-level means are respectively f_1 and f_2 , the grey-level contrast are calculated as:

$$GC = \frac{|f_1 - f_2|}{f_1 + f_2}. \tag{10}$$

Region non-uniformity [6, 7] is defined as:

$$NU = \frac{|F_r| \sigma_f^2}{|F_r + B_r| \sigma^2}, \tag{11}$$

where σ^2 represents the variance of the whole image, σ_f^2 represents the foreground variance. F_r and B_r denote the foreground and background area pixels in the test image. It is expected that a well-segmented image will have a non-uniformity measure close to 0, while the worst case of $NU=1$ corresponds to an image for which background and foreground are indistinguishable up to second order moments.

From Table 1, Fig. 2, Table 2 and Fig. 3, we can deduce that the SVM based segmentation for SAR image can effectively segment noisy SAR image. And the results of one-class SVM based segmentation are better than that of two-class SVM based segmentation.

Table 1. Segmentation results of airplane1 for different segmentation methods

Segmentation method	Grey-level contrast	Non-uniformity
Threshold Segmentation	0.30719	0.65085
Adaptive Threshold Segmentation	0.28892	0.68932
Two-class SVM Based Segmentation	0.34048	0.53698
One-class SVM Based Segmentation $\nu=0.5$	0.36949	0.56019
One-class SVM Based Segmentation $\nu=0.2$	0.52386	0.28304

Table 2. Segmentation results of airplane2 for different segmentation methods

Segmentation method	Grey-level contrast	Non-Uniformity
Threshold Segmentation	0.40665	0.92038
Adaptive Threshold Segmentation	0.33735	0.72519
Two-class SVM Based Segmentation	0.37244	0.70132
One-class SVM Based Segmentation $\nu=0.5$	0.35797	0.71422
One-class SVM Based Segmentation $\nu=0.3$	0.39741	0.53126

5 Conclusion

SAR images are characterized by the intrinsic multiplicative noise, which affects negatively image analysis techniques. In this paper, we presented a one-class SVM based segmentation method for SAR images, which can effectively segment noisy SAR images. One-class SVM based segmentation is not necessary to provide a

representative sample because of unsupervised learning. Experimental results show that proposed method has better segmentation performance than two-class SVM based segmentation and the traditional segmentation methods. The one-class SVM is suitable for segmentation of SAR image. The selection of the parameter ν in one-class SVM based segmentation impacts the performance of segmentation.

References

1. Vladimir N., Vapnik.: The Nature of Statistical Learning Theory. New York: Springer-Verlag (1995)
2. Cortes, C., Vapnik, V.: Support Vector Networks. *Machine Learning* **20** (3) (1995) 273-297
3. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery* **2** (2) (1998) 1-47
4. Schölkopf, B., Platt, J.C., et al.: Estimating the support of a high-dimensional distribution. Microsoft Research Corporation Technical Report MSR-TR-99-87 (1999)
5. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. (2001) Software available at <http://www.csie.ntu.edu.tw/~cjlin/lib>.
6. Levine, M.D., Nazif, A.M.: Dynamic measurement of computer generated image segmentations. *IEEE Trans. on Pattern Recognition Analysis and Machine Intelligence* **7** (1985) 155-164
7. Zhang, Y.J.: A survey on evaluation methods for image segmentation. *Pattern Recognition*. **29** (1996) 1335-1346

A New Segmentation Method Based on SVM for HIFU Image-Guided System

Zhao Zhang¹, Su Zhang¹, Wei Yang¹, Ya zhu Chen¹,
and Hong tao Lu²

¹ Biomedical Instrument Institute, Shanghai Jiao Tong University,
Shanghai 200030, China

² Dept. of Computer Science, Shanghai Jiao Tong University,
Shanghai 200030, China

{z_ball, suzhang}@sjtu.edu.cn

Abstract. High Intensity Focused Ultrasound (HIFU) is one of promising non-invasive thermal ablation techniques of tumor. In this paper, we present a segmentation method based on Support Vector Machine (SVM) for HIFU image-guided system where SVM is used to construct the prior model about the intensity and the shape of the structure from the training set of images and the boundaries. When segmenting a novel image, we improved level set method by incorporating this prior model. Segmentation results are demonstrated on ultrasonic images. It shows that the prior model makes segmentation process more robust and faster.

1 Introduction

HIFU is receiving increasing attention as a non-invasive method of destroying deep-seated tumors. Image guidance is required to target and monitor therapy. Currently, ultrasound is leading imaging modalities for guidance of HIFU therapy due to advantages in cost, ease of integration, and real-time implementation. Segmentation is the key part of image-guided system. In HIFU image-guided system, a series of consecutive two dimensional ultrasonic slices are gathered in order to produce full three dimensional volumes of the anatomy. When we perform the segmentation slice by slice, it is extremely tedious and time consuming. In the face of this challenge prior knowledge needs to be added to make the segmentation methods more robust and faster. SVM approach is considered as a good candidate for utilizing the prior knowledge because of its high generalization performance and sparse solution. In the sequential images, there are many similar factors in neighboring slices, such as intensity and shape of the structure. So we select some slices segmented manually to be training samples. Then SVM for density estimation are used to construct a prior model of the structure based on these training data. To segment object from a neighboring slice, the level set method is improved by incorporating the prior model.

2 Density Estimation Based on SVM

The density, $P(\mathbf{x})$, is the solution of the following linear operator equation [3]:

$$F(\mathbf{x}) = \int_{-\infty}^{x_1} \cdots \int_{-\infty}^{x_N} P(\mathbf{t}) dt_1 \cdots dt_N, \tag{1}$$

where $\mathbf{x}=(x_1, \dots, x_N) \in \mathbb{R}^n$. Since $F(\mathbf{x})$ is unknown and a random independent sample is given as $\mathbf{x}_1, \dots, \mathbf{x}_l$, the empirical distribution function can be evaluated as

$$F_l(\mathbf{x}) = \frac{1}{l} \sum_{i=1}^l \theta(x_1 - x_{i,1}) \cdots \theta(x_N - x_{i,N}),$$

where $\theta(x)$ is 1 when $x > 0$, otherwise, it is 0.

Construct $(\mathbf{x}_1, F_l(\mathbf{x}_1)), \dots, (\mathbf{x}_l, F_l(\mathbf{x}_l))$, The problem of finding the solution of linear operator equations is equivalent to the regression problem in image space used SVM method.

Kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{r=0}^{\infty} \psi_r(\mathbf{x}_i) \psi_r(\mathbf{x}_j)$ is used in image space. Cross kernel is $\kappa(\mathbf{x}_i, \mathbf{x}) = \sum_{r=0}^{\infty} \psi_r(\mathbf{x}_i) \phi_r(\mathbf{x})$, then the solution of Eq.1 is:

$$P(\mathbf{x} | \alpha) = \sum_{i=1}^l \alpha_i (\Psi(\mathbf{x}_i) \cdot \Phi(\mathbf{x})). \tag{2}$$

In order to satisfy Eq.2 being density, the weights α_i should be as following:

$$\sum_{i=1}^l \alpha_i = 1, \alpha_i \geq 0, i = 1, \dots, l. \tag{3}$$

Then when we use the technique of Linear SVM and ϵ -insensitive loss function, the problem of density estimation is equivalent to the Linear Programming problem as:

$$\min \sum_{i=1}^l \delta_i \alpha_i + C \sum_{i=1}^l \xi_i + C \sum_{i=1}^l \xi_i^*, \tag{4}$$

subject to the constraints are Eq.3 and

$$F_l(\mathbf{x}_i) - \epsilon_i - \xi_i \leq \sum_{j=1}^l \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) \leq F_l(\mathbf{x}_i) + \epsilon_i + \xi_i^*, i = 1, \dots, l, \tag{5}$$

$$\xi_i \geq 0, \xi_i^* \geq 0, i = 1, \dots, l, \tag{6}$$

where $\epsilon_i = \lambda \sqrt{\frac{1}{l} F_l(\mathbf{x}_i)(1 - F_l(\mathbf{x}_i))}$, $\delta_i = \frac{1}{l} \sum_{j=1}^l \|\mathbf{x}_i - \mathbf{x}_j\|^2$. We can get the solution of density estimation by substitute α_i into Eq.2.

3 Segmentation Algorithm Based on SVM

3.1 Shape Model Based on SVM

Signed distance function is used as the shape representation. We construct a statistical relationship between final curve, involving curve and the gradient of image as

$P(\phi^* | \phi, \nabla I)$, where ϕ is the evolving surface in some step, ∇I is the gradient of image, ϕ^* is the final surface. At each evolving step, the MAP of ϕ^* with the conditions of ϕ and ∇I is estimated:

$$\phi_{MAP}^* = \arg \max_{\phi^*} P(\phi^* | \phi, \nabla I) . \tag{7}$$

According to [1], $P(\phi^* | \phi, \nabla I)$ can be expressed as:

$$P(\phi^* | \phi, \nabla I) \propto P(\phi | \phi^*)P(\nabla I | \phi^*)P(\phi^*) . \tag{8}$$

The last term $P(\phi^*)$ on right hand side of Eq.8 can be modeled by training the samples segmented manually. The training set of shape can be defined as $T = \{\phi_1^*, \phi_2^*, \dots, \phi_n^*\}$. Here ϕ_i^* is aligned as column vector in sequence. It is infeasible to train T directly because each signed distance function include a large amount of data. So we use the method similar to the Eigenface method [2] to transform the training set T , lower its dimension. Define $\mu = \frac{1}{n} \sum \phi_i^*$, $\hat{\phi}_i = \phi_i^* - \mu$ and $M = [\hat{\phi}_1^* \quad \hat{\phi}_2^* \quad \dots \quad \hat{\phi}_n^*]$. The

covariance matrix $\frac{1}{n} MM^T$ is decomposed as $U \Sigma U^T = \frac{1}{n} MM^T$, where U is a matrix whose column vectors represent the set of orthogonal modes of shape variation and Σ is a diagonal matrix of corresponding singular values. An estimate of a novel shape, ϕ^* , of the same class of object can be represented by k principal components in a k -dimensional vector of coefficients, $\alpha = U_k^T (\phi^* - \mu)$, Where U_k is a matrix consisting of the first k columns of U that is used to project a surface into the eigen-space. Given the coefficient α , an estimate of the shape $\tilde{\phi}^*$, is reconstructed from U_k and μ , namely $\tilde{\phi}^* = U_k \alpha + \mu$. So the training set is transformed as $\tilde{T} = \{\alpha_1, \alpha_2, \dots, \alpha_n\}$. The shape distribution model $P(\phi^*)$ can be transferred to $P(\alpha)$. It can be derived using the method of SVM described in the last section.

The second term $P(\nabla I | \phi^*)$ on right hand side of Eq.8 describes the relationship between the signed distance function and the gradient of the image. We can find the relationship through regression estimation method based on SVM. Training set can be construct as $T = \{ \langle \phi(x_1), |\nabla I(x_1)| \rangle, \dots, \langle \phi(x_n), |\nabla I(x_n)| \rangle \}$. Find $f(\phi^*)$, the regression estimation of $|\nabla I|$, then define $P(\nabla I | \phi^*)$ as:

$$P(\nabla I | \phi^*) = \exp(-|f(\phi^*) - |\nabla I||^2) . \tag{9}$$

The first term $P(\phi | \phi^*)$ on right hand side of Eq.8 describes the relationship between final surface and evolving surface. According to [1], supposing initial curve lies inside the object we want to segment, $P(\phi | \phi^*)$ can be defined as:

$$P(\phi | \phi^*) = \exp(-V_{outside}) . \tag{10}$$

Here $V_{outside}$ is the volume of the evolving curve that lies outside the final curve.

After construct the shape model, at each step during the surface evolution process, we use simple gradient ascent on the log probability function in Eq.8 to estimate ϕ_{MAP}^* . In order to evolve the surface towards the MAP estimate, the surface increment induced from the shape model is:

$$\Delta\phi_S(\mathbf{x}, t) = \phi^*(\mathbf{x}, t) - \phi(\mathbf{x}, t) . \tag{11}$$

3.2 Intensity Model Based on SVM

We construct a statistical relationship between the intensity of image and the surface as $P(\phi(\mathbf{x}) | I(\mathbf{x}), \phi(N(\mathbf{x})))$, where $N(\mathbf{x})$ is the neighborhood of the point \mathbf{x} . It expresses the probability of the value of the surface at the point \mathbf{x} , given the intensity value of the image at the same point and the neighboring values of the surface. This conditioned distribution can be approximated as [4]:

$$P(\phi(\mathbf{x}) | I(\mathbf{x}), \phi(N(\mathbf{x}))) \propto P(I(\mathbf{x}), \phi(\mathbf{x}))P(\hat{\phi}(t+), \hat{\phi}(t-) | \phi(\mathbf{x}))P(\hat{\phi}(n+), \hat{\phi}(n-) | \phi(\mathbf{x})) . \tag{12}$$

here, four neighbors in the direction of the local normal ($\hat{n}+$, $\hat{n}-$) and the tangent ($\hat{t}+$, $\hat{t}-$) are used to the embedded curve. During the segmentation process, we maximize the probability to estimate each surface point $\phi(\mathbf{x})$ independently while assuming the rest of the surface is constant:

$$\phi(\mathbf{x}) = \max_{\phi(\mathbf{x})} \log P(\phi(\mathbf{x}) | I(\mathbf{x}), \phi(N(\mathbf{x}))) . \tag{13}$$

In each evolving step, the surface increment induced from intensity model is:

$$\begin{aligned} \Delta\phi_I(\mathbf{x}, t) = & \frac{d}{d\phi(\mathbf{x}, t)} \log P(I(\mathbf{x}), \phi(\mathbf{x}, t)) + \frac{d}{d\phi(\mathbf{x}, t)} \log P(\hat{\phi}(t+), \hat{\phi}(t-) | \phi(\mathbf{x}, t)) \\ & + \frac{d}{d\phi(\mathbf{x}, t)} \log P(\hat{\phi}(n+), \hat{\phi}(n-) | \phi(\mathbf{x}, t)) . \end{aligned} \tag{14}$$

The first term on right hand side of Eq.12 relates the intensity and the surface at \mathbf{x} . The training set of this term is $T = \{ \langle I_1, \phi_1 \rangle, \dots, \langle I_n, \phi_n \rangle \}$. This term can be derived from the training set using the method of SVM described in the last section. Then the first

term in Eq.14 is computed by taking the gradient of the sampled probability in the direction of $\phi(\mathbf{x})$.

The middle term on right hand side of Eq.12 reflects the curvature profile of the training data. According to [4], curvature k reflects the distribution of $\phi(\hat{t}+)$, $\phi(\hat{t}-)$ and $\phi(\mathbf{x})$. So we use $\{k_1, \dots, k_n\}$ as the training set. This term is derived from the training set using the method of SVM described in the last section. Then the middle term in Eq.14 can be computed by taking the gradient of the sampled probability in the direction of $\phi(\mathbf{x})$.

The last term on right hand side of Eq.12 prevent the surface from evolving arbitrarily. According to [4], The last term in Eq.14 is defined as:

$$\frac{d}{d\phi(\mathbf{x}, t)} \log P(\phi(\hat{n}+), \phi(\hat{n}-) | \phi(\mathbf{x}, t)) = \alpha(\phi(\hat{n}+) + \phi(\hat{n}-) - 2\phi(\mathbf{x}, t)). \tag{15}$$

3.3 Evolving the Surface

Here we improved the level set method by incorporating the prior knowledge into the surface evolving process. The update rule for the surface is:

$$\phi(\mathbf{x}, t + 1) = \phi(\mathbf{x}, t) + \beta_1 \Delta \phi_S(\mathbf{x}, t) + \beta_2 \Delta \phi_I(\mathbf{x}, t) . \tag{16}$$

The two parameters β_1 and β_2 are used to balance the influence of the shape model and intensity model. Evolving the surface accords to Eq.16 until there is little change. Then extract the zero level set, it is the boundary of the desired structure.

4 Experiment Results and Discussion

Let us present an experiment to illustrate our segmentation method. There is a set of egg in ultrasonic images. We select 20 slices as training samples. They are segmented manually. Select a neighboring slice to be segmented. We segmented the image using level set method based on C-V model (Fig.1) and the method researched in this paper (Fig.2) independently. The figures illustrate the initial, middle, and final steps in the evolution process of segmentation.

Comparing Fig.1 with Fig.2, we can find that the method in this paper have better result than the method based on C-V model. Furthermore, to achieve the final curve from the initial curve, Fig.1 iterated 70 times, while Fig.2 needs 45 iterates. Since incorporating the prior information, our method converges on the boundary faster and more robustly. In our segmentation method, SVM was used to construct the prior model. The experiments showed that the number of support vectors only amounts to 20% ~ 40% of the number of training samples. Because of the sparse solution, SVM method to build the prior model reduced the computational labor of updating the surface, compared to Parzen method.

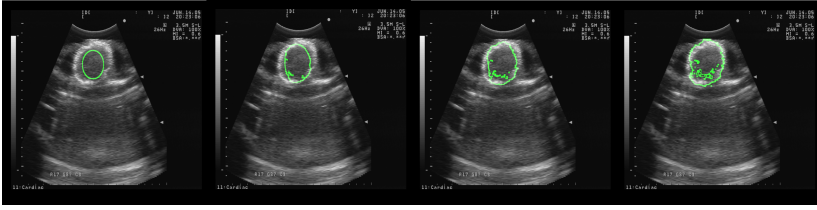


Fig. 1. Segmentation of egg in ultrasonic image using C-V method

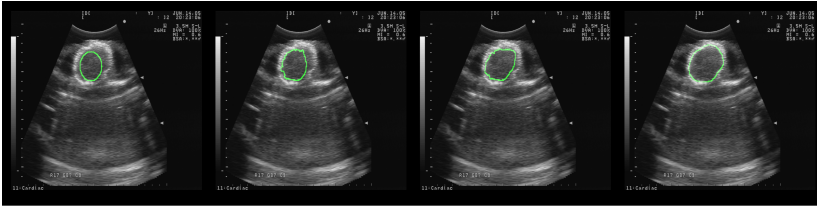


Fig. 2. Segmentation of egg in ultrasonic image using the method researched in this paper.

Acknowledgments. This study is supported by the National Basic Research Program of China (973 Program) (No. 2003CB716103) and NFSC under project No.60573033.

References

1. Leventon, M.E., Grimson, W.E., Faugeras, O.: Statistical Shape Influence in Geodesic Active Contours. *Comp. Vision Pat. Recog* (2000)
2. Turk, M.A., Pentland, A.P.: Eigenfaces for Recognition. *Cognitive Neurosci* **3** (1) (1991) 7186
3. Vapnik, V., Mukherjee, S.: Support Vector Method for Multivariate Density Estimation. *Advances in Neural Information Processing Systems*, MIT Press, Massachusetts (2000) 659-665
4. Zhang, Z., Zhang, S., Zhang, C.X., Chen, Y.Z.: SVM for Density Estimation and Application to Medical Image Segmentation. *Journal of Zhejiang University*, **7** (5) (2006) 365-372

Greenhouse Air Temperature and Humidity Prediction Based on Improved BP Neural Network and Genetic Algorithm

Fen He¹, Chengwei Ma¹, Junxiong Zhang², and Ying Chen²

¹ The Key Laboratory of Agricultural Bio-environment Engineering, Ministry of China, China Agricultural University, Beijing 100083, China
3600seconds@163.com, macwbs@cau.edu.cn

² College of Engineering, China Agricultural University, Beijing 100083, China
{zhang_junxiong, cau2004}@163.com

Abstract. The adequacy of improved back propagation (IBP) neural network to model the inside air temperature and humidity of a production greenhouse as a function of outside parameters including temperature, relative humidity, wind speed, and solar radiation was addressed. To avoid standard BP algorithm's shortcoming of trapping to a local optimum and to take advantage of the genetic algorithm (GA)'s globe optimal searching, a new kind of hybrid algorithm was formed based on the IBP neural network and GA. BP neural network was improved by adding the inertia impulse and self-adaptation learning rate to lessen convergence vibration and increase the learning speed. Then the initialized weights and thresholds of IBP neural network were optimized with GA. Through carrying out the experiments, the specimen data were collected on half-hourly basis in a greenhouse. After the network structure and parameters were determined reasonably, the network was trained. A comparison was made between measured and predicted values of temperature and relative humidity, and the results showed that the IBP neural network model combined with GA given a good prediction for inside temperature and humidity. By using the root mean square error (RMSE) algorithm, the RMSE between temperature predicted and measured was 0.8°C, and the relative humidity RMSE was 1.1%, which can satisfy with the demand of greenhouse climate environment control.

1 Introduction

The greenhouse microclimate is a typical complicated nonlinear system, which provides the plants with good environmental conditions for growing. Temperature and humidity are considered key factors in greenhouse climate and they are the results of complex and interactive heat and mass exchanges between the inside air and the several other elements of the greenhouse (e.g., construction, vegetation, controllers, etc.) and the outside boundaries (e.g., outside air, sky, solar radiation, etc.). It is hard to build the greenhouse mechanism model with simple mathematical formulas or transform functions. However, the method of building model with artificial neural network has strong ability of nonlinear function mapping, which is applied to many production process systems.

Over the last decades, a large effort was devoted to study the greenhouse with the method of artificial neural networks [1]-[5]. Hugo and Linker trained a neural network using experimental data to model the internal temperature [6] [7]. And Ferreira etc. also created a RBF neural network model, and it was incorporated in an environment control strategy, which made the model very complicated [8]. By comparison, the present study extended the use of neural network models by not only fitting models to the experimental data, but also by using them to predict the greenhouse climate inside. For few utilized the BP neural network and GA to investigate the greenhouse climate, this paper combined IBP neural network with GA to predict the inside air temperature and humidity by using outside parameters entirely in a glass greenhouse.

2 Improved BP Neural Network Model

Standard BP neural network consists of an input layer, an output layer and hidden layers. The hidden layer can has one or more layers with each having several neurons. It possesses fault-tolerant and generalization ability for prediction. However, the standard BP neural network is prone to trapping to a local optimum and the convergence speed and learning rate is slow, which will influence the precision of prediction.

In order to overcome the disadvantages aforementioned, the standard BP neural network needs to be modified to predicting greenhouse temperature and humidity. Two aspects were improved:

- 1) Adding the inertia impulse α into the formula of network weights. This method can consider not only the error effect on grads, but also the influence of change trend on error curve when modifying the weights. It can filter the high frequency vibration, smoothen the network convergence process, and increase the training speed. The weights modifying formulas with inertia impulse are:

$$\begin{cases} \Delta w_{ij}(t+1) = (1-\alpha)\eta\delta_i p_j + \alpha\Delta w_{ij}(t), \\ \Delta b_i(t+1) = (1-\alpha)\eta\delta_i + \alpha\Delta b_i(t). \end{cases} \quad (1)$$

In (1), t is the training times, η is the learning rate.

- 2) In standard BP algorithm, the learning rate η is a fixed value. For high prediction precision, η should be small enough, which results in the slow learning process. To resolve this, in training process, the method of adjusting η automatically was adopted. The adjusting formula is as follows and $SSE(t)$ is the network square sum of error:

$$\eta(t+1) = \begin{cases} 1.05\eta(t), & SSE(t+1) < SSE(t), \\ 0.7\eta(t), & SSE(t+1) > 1.04SSE(t), \\ \eta(t), & otherwise. \end{cases} \quad (2)$$

3 Methods and Procedures

3.1 Hybrid Algorithm Based on Improved BP Neural Network and GA

The IBP algorithm adopted the error derivative to instruct the learning process, it was still the local optimum searching. For ensuring the precision of network model, GA was chosen to change this status. GA is an algorithm of globe optimum inspired from biology and anthropology, and it is good at globe optimum searching.

In network training, GA was used to determine the network initial weights and thresholds. The steps of GA optimizing the network weights and thresholds are as follows:

- 1) Initialize a population, including its scale, selection probability, crossover probability, mutation probability and the individual generated randomly. Each individual of population represents a group of initial weights of the whole network.
- 2) Decode each individual to network weight, input the training specimen, and calculate the neural network error function corresponding to every group of weights, then determine the fitness of each individual. The fitness function is:

$$f(X^q) = \frac{1}{\sqrt{\sum_{p=1}^N \sum_{k=1}^m (t_{pk} - O_{pk}^q)^2}} \quad (3)$$

In (3), N is the training specimen number, m is the output node number, the expected output of network is t_{pk} , $p = 1, 2, \dots, N$, the practical output is O_{pk}^q , $q = 1, 2, \dots, M$, M is the individual number of population.

- 3) If the population can't satisfy the precision expected (after optimization with GA, network square sum of error $SSE' > \varepsilon_2$), generate the new population.
- 4) Return to 2) with the new population, and circulate and repeat the process of fitness evaluation, population judgment, selection, crossover, and mutation. Make the individual fitness and average fitness in population increase till the optimal individual fitness reach the demand expected, then the algorithm finishes. Decode the individual searched by the optimum fitness, and the optimum network weights and thresholds are obtained.

After GA optimized the network weights and threshold, if the error of network still can't reach the precision expected, then IBP algorithm was used to continue searching in space of network weights and threshold searched by GA for optimizing the weight space.

3.2 Data Source and Pretreatment

Outside temperature T_{out} , relative humidity RH_{out} , wind speed W_{out} , and solar radiation SR_{out} are the main factors affecting greenhouse inside air temperature T_{in} and humidity RH_{in} in the summer [9]. The relationship between network input and data output was built, which aimed at model creating and prediction by neural network.

The specimen data were all measured in a glass greenhouse of China Agricultural University, and the experiment data were acquired on half-hourly basis from 4-19 August 2005. Inside temperature and relative humidity were measured by the hygrothermograph automatically. The outside weather factors were collected by greenhouse computer controlling system, and the parameters measured including outside temperature, humidity, wind speed, and solar radiation.

For input specimen belonged to different dimension, make the input specimen unitary and fall within a specified range. The method of proportion compress was chosen, and the following formula was used to scale the input specimen into the range of 0 to 1:

$$T = T_{\min} + \frac{T_{\max} - T_{\min}}{X_{\max} - X_{\min}}(X - X_{\min}). \tag{4}$$

In (4), X is the original data, X_{\max} and X_{\min} are the maximum and minimum of the original data. T is the target data, T_{\max} and T_{\min} are the maximum and minimum of the target data, and $T_{\max} = 0.9$, $T_{\min} = 0.1$. Then make the data obtained finally revert, the formula is:

$$X = X_{\min} + \frac{X_{\max} - X_{\min}}{T_{\max} - T_{\min}}(T - T_{\min}). \tag{5}$$

After all specimen data were measured, organized, and analyzed, 300 specimen groups were selected to predict temperature and humidity. Pick out 60 specimen groups for prediction, and 70 groups chosen randomly from 240 left for validation.

3.3 Determination of Neural Network Structure

The hidden layer number and nodes are key factors of neural network structure. Basing on the Kosmogorov approaching theory, the BP neural network with a hidden layer can approach any continuous function with any precision, the neural network with only one hidden layer was chosen. According to the main factors affecting the greenhouse inside climate, the network input layer node was set as 4. The output layer node was 2. For the determination of the node number in the hidden layer, we chosen different nodes of hidden layer to try after inputting the same training and validating specimen. Table 1 was the error comparison of different nodes in hidden layer. When the node in hidden layer was 9, after inputting the training specimen and validation specimen for network training, the network output node error sum of squares was the smallest, which showed the network structure was better.

According to analysis, the neural network model was 3 layers BP neural network with the structure of 4-9-2. The transfer function was:

$$f(x) = 1 / (1 + e^{-x}). \tag{6}$$

By using the hybrid algorithm to train the network and some correlative parameters were determined by self-adjusting in model running, the parameters corresponding with BP neural network: the learning rate $\eta = 0.3$, the inertia impulse $\alpha = 0.95$, the

maximal circulation time $t_{\max} = 10000$, the network precision $\varepsilon_1 = 0.0001$. The parameters corresponding with GA: the population scale $N=40$, the network precision $\varepsilon_2 = 0.001$, crossover probability $P_c = 0.70$, mutation probability $P_m = 0.05$. After the network weights and thresholds were obtained by network training, specimen groups selected were input for validation and prediction.

Table 1. Error comparison of different nodes in hidden layer

Nodes in hidden layer	Network error sum of squares ($\times 10^{-4}$)	
	Training specimen	Validating specimen
7	4.7762	4.5561
8	5.6643	3.4416
9	0.9305	0.9851
10	2.0292	3.3905
11	4.2264	4.8115
12	5.6678	5.9456

4 Results and Discussion

Fig. 1 and Fig. 2 were the comparison curves between measured and predicted values of the inside temperature and relative humidity. Fig. 3 and Fig.4 were the fit curves between measured and predicted inside temperature and relative humidity. The temperature relative error was within $\pm 6\%$, which was lower than the humidity fit precision, but it also can be applied in practice. The relative humidity relative error was within $\pm 3\%$, which was difficult to obtain by physical model. Owing to the process of building the physical model, the ununiformity of greenhouse humidity in

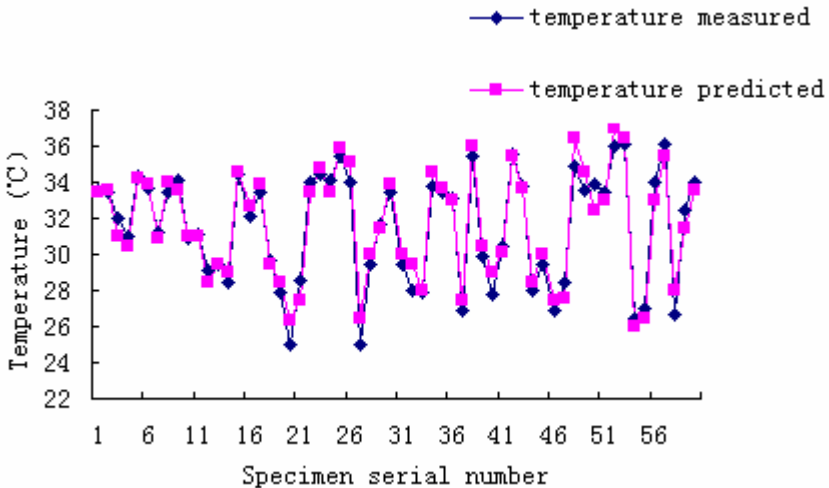


Fig. 1. Comparison between measured and predicted values of inside temperature

space and measuring error will influence the result. But when creating model with the method of neural network, the weather in periods of experiments is the same approximately, and the measuring sensors positions are not changed, in the process of network training and validating, it can remember these characteristics of system, which makes the prediction precision higher than physical model.

The results obtained showed that there was excellent agreement between measured and predicted values. By utilizing the RMSE algorithm to analyze the values, the RMSE between the temperature measured and predicted was $0.8\text{ }^{\circ}\text{C}$, which illuminated the neural network model had the ability of prediction. And the RMSE between the relative humidity measured and predicted was 1.1%. The results demonstrated the neural network model can predict the change of inside temperature and relative humidity of greenhouse accurately.

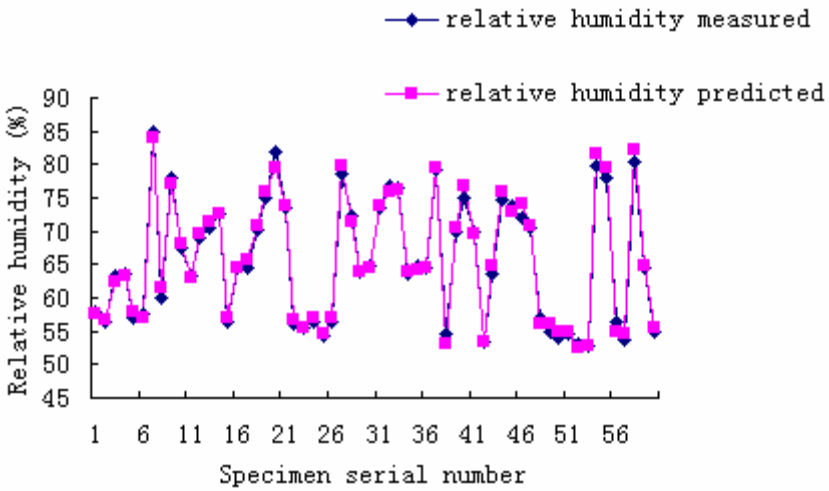


Fig. 2. Comparison between measured and predicted values of inside relative humidity

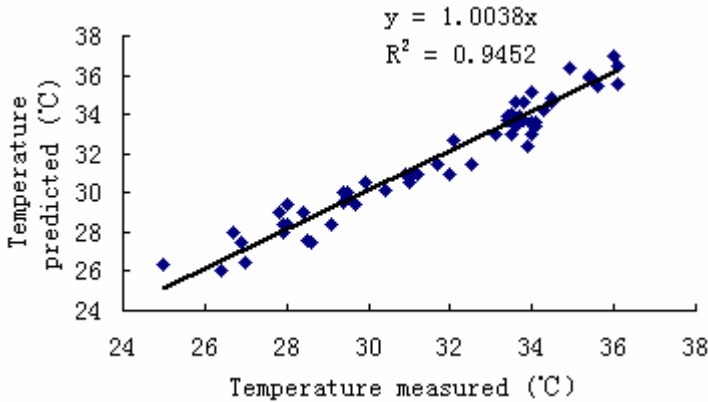


Fig. 3. The fit curve between measured and predicted temperature

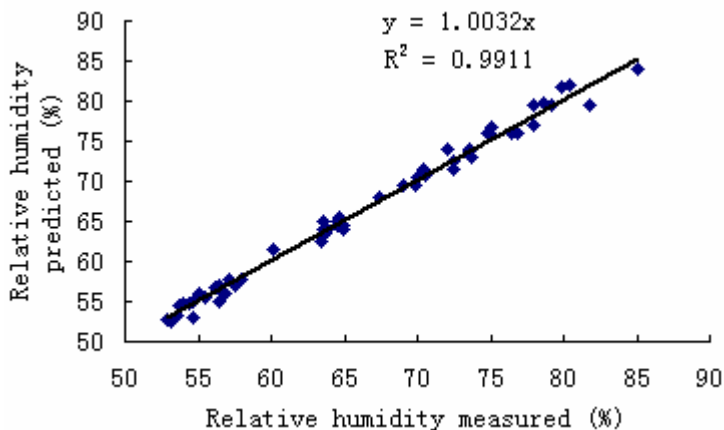


Fig. 4. The fit curve between measured and predicted relative humidity

5 Conclusions

An IBP neural network combining with GA was proposed for modeling the internal greenhouse temperature and humidity and the network structure was determined reasonably. Compared with the traditional method of building greenhouse model, the neural network can reflect the greenhouse nonlinear characteristics. And through network training and validation, the predicting results obtained showed a good performance and fitness between measured and predicted data, which indicated that the neural network model can be used in practice to predict greenhouse internal temperature and relative humidity by making use of external weather data, and it also can meet the need of greenhouse environment prediction.

References

1. Seginer, I., Boulard, T., Bailey, B.J.: Neural Network Models of the Greenhouse Climate. *J. Agric. Eng. Res.* **59** (1994) 203–216
2. Kok, R., Lacroix, R., Clark, Taillefer, G., E.: Imitation of a Procedural Greenhouse Model with an Artificial Neural Network. *Can. Agric. Eng.* **36** (1994) 117–126
3. Seginer, I.: Some Artificial Neural Network Applications to Greenhouse Environmental Control. *Computers and Electronics in Agriculture* **18** (1997) 167–186
4. Linker, R., Seginer, I., Gutman, P.O.: Optimal CO₂ Control on a Greenhouse Modeled with Neural Networks. *Computers and Electronics in Agriculture* **19** (1998) 289–310
5. Ferreira, P.M., Ruano, A.E.: Predicting the Greenhouse inside Air Temperature with RBF Neural Networks. 2nd IFAC/CIGR Workshop on Intelligent Control for Agricultural Applications, Bali Indonesia (2001)
6. Hugo Uchida Frausto, Peters, J.G.: Modelling Greenhouse Temperature Using System Identification by Means of Neural Networks. *Neurocomputing* **56** (2004) 423–428

7. Linker, R., Seginer, I.: Greenhouse Temperature Modeling: A Comparison between Sigmoid Neural Networks and Hybrid Models. *Mathematics and Computers in Simulation* **65** (2004) 19-29
8. Ferreira, P.M., Faria, E.A., Ruano, A.E.: Neural Network Models in Greenhouse Air Temperature Prediction. *Neurocomputing* **43** (2002) 51–75
9. Jolliet, O.: HORTITIRANS, A Model for Predicting and Optimizing Humidity and Transpiration in Greenhouse. *J. Agric. Eng. Res.* **57** (1994) 23–37

A Modified RBF Neural Network and Its Application in Radar

Yang Jun, Ma Xiaoyan, Lu Qianhong, Liu Bin, and Deng Bin

Air Force Radar Academy, Wuhan, 400019, China
yangjem@126.com

Abstract. Aiming at the problem of parameter estimation in radar detection, a modified RBF neural network is proposed to estimate parameter accurately because of its good approximation ability to random nonlinear function and quick convergence speed. Two classical detection methods, which widely used in radar field, are listed in this paper, and their corresponding parameters are estimated with modified RBF neural network. Theoretical analysis and numerical results both show that the proposed method has good parameter estimation accuracy and quick convergence speed.

1 Introduction

In radar detection field, parameter estimation problem is often encountered to detect target. In the view of math, such case can be considered as a mapping between variables and functions, i.e. there is certain mapping relationship between variable t and expression $f(t)$ (both t and $f(t)$ belong to real value), however, either $f(t)$ or t may not be expressed by each other with closed-form expression. For expression $f(t)$ with a given value, we need estimate its corresponding parameter to variable t , two conventional ways are used to solve such problem, one is numerical searching method and the other is Monte-carlo simulation method, but in both of the methods, the former needs a long time to search and the latter needs large samples or long simulation time. Considering the advantages of radial basis function (RBF) neural network, such as good nonlinear approximation ability and quick convergence speed^[1,2], and the character of parameter estimation in radar detection, a modified neural network (MNNT) via RBF is proposed to estimate parameter accurately. And reason of the MNNT's good approximation is analyzed in theory. Finally, as classical examples, two detection methods, which are widely used in radar, are listed in this paper. Theoretical analysis and numerical results show that the parameters estimation method has a good parameter estimation accuracy and short convergence speed time.

2 Modified RBF Neural Network

The RBF NNT can approximate any nonlinear function in theory, i.e. the mapping between variable t and function $f(t)$ can be approximated with it (both t and $f(t)$

belong to real). The block diagram of conventional RBF NNT (CNNT) is shown in Fig.1, and the mapping can be represented as $f^{-1}[f(t)]:R \rightarrow R$, where $f^{-1}(\cdot)=t \in R$ and R denote real value. Their relationships can be written as

$$t = f^{-1}[f(t)] = \sum_{j=1}^M w_j \varphi(\|f(t) - c_j\|) \tag{1}$$

where $\varphi(\cdot)$ is a RBF, and Gaussian function is used in this paper, i.e.

$$\varphi(\|f(t) - c_j\|) = \exp\left(\frac{-\|f(t) - c_j\|^2}{\sigma_j^2}\right) \tag{2}$$

where parameters c_j and σ_j ($j=1,2,\dots,M$) are center and width of Gaussian function for the j th hidden layer, respectively, parameter w_j ($j=1,2,\dots,M$) is linking weight for the j th hidden layer.

Block diagram of the modified RBF neural network (MNNT) is shown in Fig.2 in this paper, and the relationship between variable t and function $f(t)$ is

$$t = \sum_{j=1}^M w_j \varphi(\|G[f(t)] - c_j\|) \tag{3}$$

where the parameters c_j, σ_j ($j=1,2,\dots,M$), w_j and $\varphi(\cdot)$ have the same meanings as that in eqn.2, the relationship between function $G[f(t)]$ and function $f(t)$ is

$$G[f(t)] = k \lg[f(t)] \tag{4}$$

where parameter k is a constant, but not zero, and its value is related to the given problem.

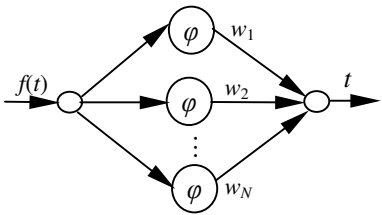


Fig. 1. Block diagram of the CNNT

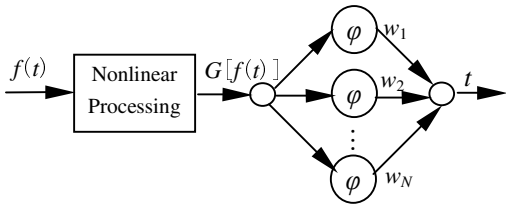


Fig. 2. Block diagram of the MNNT

The main reason for using the MNNT in radar is that the values of given $f(t)$ are usually very small (for example 10^{-6}), and each value of $f(t)$ maps only one value of variable t , inversely, different value of t maps different value of $f(t)$. Such a mapping relationship can be approximated by a NNT, whose input samples and output samples are a set of values of $f(t)$ and t respectively, thus such mapping problem in math can be equivalent to pattern classification problem, therefore, different value of t maps different value of $f(t)$ can be equivalent to that different value of t belongs to different pattern value of $f(t)$. It is well known that the accuracy

of pattern classification is related to the distance among samples, i.e. the less the distance between samples/patterns is, the larger the classification error is. For a trained CNNT, in which the variables c_j, σ_j, w_j and M are fixed, less distance among samples will lead to larger estimation error, so we can't directly use the CNNT to estimate radar detection parameter t . For the reason above, in order to improve classification/approximation accuracy, one effective way is to enlarge the distance among samples, which can be achieved by nonlinear transformation to the input samples, thus we can estimate parameter t accurately for a given value of $f(t)$.

In eqn.2 and eqn.3, the MNNT parameters M, c_j, σ_j and $w_j (j=1,2,\dots,M)$ can be obtained through training the neural network. Here, we use K-means algorithm^[2,3] to obtain the network parameters c_j and σ_j ; and the linking weights $w_j (j=1,2,\dots,M)$ between hidden layer and output layer are achieved by using orthogonal least squares learning algorithm^[4,5], the orthogonal least squares learning algorithm can not only obtain linking weights and the number of hidden nodes, but also ensure that the network has the least clustering center, which can decrease the network dimension and quicken the network operation speed. The paper [6] listed the MNNT training steps in detail. Before the training of the MNNT, the maximum number of initial nodes in hidden layer is equal to the number of input samples.

3 Parameters Estimation Problem in Radar Detection

In this section, we list two classical parameters estimation problem in radar detection. In fact, there are many similar cases in radar detection.

3.1 Parameter Estimation for Binary Window Integration Detection

As a conventional radar detection algorithm, Binary Sliding Window Integration Detection (BSWID) mainly including two parts, one part is binary hypothesis test (0 or 1 declared), the other part is m/n sliding window integration detection. The binary hypothesis test is usually implemented with a given detection rule such as CFAR detection, likely ratio test rule or Neyman-Pearson rule etc, here, we respectively denote P_{fa} and P_{d1} as the binary hypothesis test's false probability and detection probability. For a detector, there are two steps to evaluate its detection performance, the first step is to estimate detection threshold (DT) by a given P_{fa} , the second step is to obtain relationship between DT, signal to noise/clutter ratio (SNR/SCR) and P_{d1} . For the m/n sliding window integration detection part, a target is declared once there exists m 's 1 in n 's echoes ($n>m$). If the BSWID's total false alarm probability and total detection probability are denoted by P_{fa} and P_d , respectively, then the relationship between P_{fa} and P_{fa1} can be represented as^[7]

$$P_{fa} = \sum_{i=m}^n \binom{n}{i} P_{fa1}^i (1 - P_{fa1})^{n-i} \tag{5}$$

and relationship between P_d and P_{d1} can be written as^[7]

$$P_d = \sum_{i=m}^n \binom{n}{i} P_{d1}^i (1 - P_{d1})^{n-i} \tag{6}$$

where $\binom{n}{i}$ is binomial coefficients $n! / [(n-i)!i!]$. In practice, we need estimate the value of P_{fa1} in the presence of a given value of P_{fa} , obviously, it is difficult to obtain the closed-form of P_{fa1} with respect to P_{fa} from eqn.5, however, with given P_{fa1} , we can obtain P_{fa} from eqn.5, then we can use values of P_{fa} as input samples and values of P_{fa1} as output samples to train the MNNT, once the network is trained, we can easily estimate the corresponding value of P_{fa1} in the given value of P_{fa} (for example 10^{-6}).

3.2 Parameter Estimation in CFAR Detection

Constant false alarm rate (CFAR) detection plays an important role in radar detection, and it is widely used in practice. It is not only used to judge whether the target is present or not, but also can hold CFAR during detection by adjusting detection threshold adaptively, which is more important. The detection threshold, denoted by D , is obtained by the product of background noise/clutter power level, denoted by Z , and scaling factor, denoted by S , their relationship is

$$D = SZ \tag{7}$$

In eqn.7, the scaling factor parameter S needs to be estimated, and the greatest-of (GO) CFAR detection algorithm is usually used in practice, with the assumption of that Gaussian noise/clutter background and square law detector, the target fluctuating model is Swerling II, false alarm probability of the GO CFAR detector, denoted by $P_{fa,GO}$, can be written as^[7]

$$P_{fa,GO} = 2(1+S)^{-\frac{L}{2}} - 2 \sum_{i=0}^{\frac{L}{2}-1} \binom{\frac{L}{2}+i-1}{i} (2+S)^{-\left(\frac{L}{2}+i\right)} \tag{8}$$

Detection probability of the GO CFAR detector, denoted by $P_{d,GO}$, can be written as^[7]

$$P_{d,GO} = 2 \left(1 + \frac{S}{1+\lambda}\right)^{-\frac{L}{2}} - 2 \sum_{i=0}^{\frac{L}{2}-1} \binom{\frac{L}{2}+i-1}{i} \left(2 + \frac{S}{1+\lambda}\right)^{-\left(\frac{L}{2}+i\right)} \tag{9}$$

where variable L is the number of the reference cell of GO CFAR detector, variable λ is signal to noise/clutter ratio (SNR/SCR). Similar to the section 3.1, the value of variable S needed estimated in the case of a given value of $P_{fa,GO}$. It is difficult to obtain the closed-form of S with respect to $P_{fa,GO}$ from eqn.8, however, with given P_{fa1} , we can obtain P_{fa} from eqn.8, then we can use values of $P_{fa,GO}$ as input

samples and values of S as output samples to train the MNNT, once the network is trained, we can easily estimate the corresponding value of S in presence of a given value of $P_{fa,GO}$.

4 Numerical Analysis

Numerical analysis includes following parts: compare parameters estimated accuracy using the MNNT algorithm and the CNNT algorithm, the corresponding values of variables are set as following in numerical analysis, for the BSWID detector, $n=15$, $m=6$, $P_{fa}=10^{-6}$, the input samples range of P_{fal} is $10^{-2} \sim 10^{-1}$, $k=-20$. And for the GO CFAR detector, corresponding parameters are set as following, $n=32$, $P_{fa,GO}=10^{-6}$, the input samples range of $P_{fa,GO}$ is $10^{-3} \sim 10^{-9}$ and $k=-1$. Numerical results are shown in Fig.3 to Fig.8 and Table 1. Relationship between variable P_{fa} and variable P_{fal} is shown in the Fig.3, Relationship between variable $-20\lg P_{fa}$ and variable P_{fal} is shown in the Fig.4, the Fig.5 shows trained results of the MNNT and the CRBF. The Fig.6 is relationship between variable P_{fa} and variable S, Relationship between variable

Table 1. Parameters estimation results of two detectors in $P_{fa} = 10^{-6}$

Parameters	Estimated	Estimated value of P_{fa} via estimated value (P_{fal}/S)	P_{fa} Estimation error
P_{fal}	0.02497488	$1.000002407 \times 10^{-6}$	2.407×10^{-12}
S	0.982770951	$9.999274278 \times 10^{-7}$	7.25722×10^{-11}

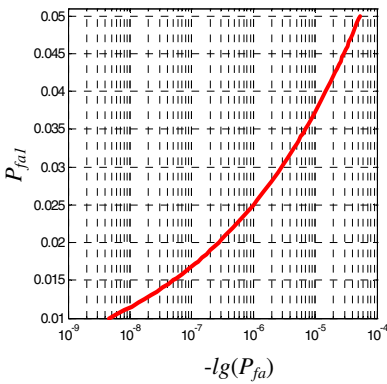


Fig. 3. Relationship between Pfa and $Pfal$ for BSWID

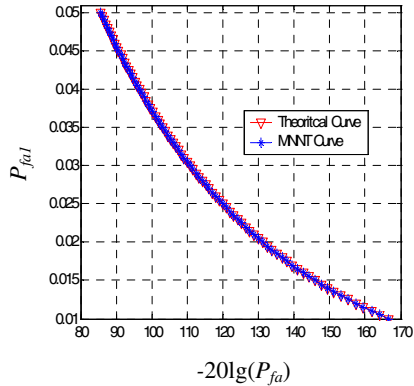


Fig. 4. Relationship between $G[Pfa]$ and $Pfal$ for BSWID

$-\lg P_{fa}$ and variable S is shown in the Fig.7, the Fig.8 shows trained results of the MNNT and the CRBF. Parameters estimation error, using MNNT, is listed in Table 1 for the two detectors. It takes very short time to estimate the corresponding parameters with the MNNT algorithm, it is not listed here for the limit of paper.

Some conclusions can be drawn from the Fig.3 to Fig. 8 and Table 1. Firstly, if the input samples are not transformed with nonlinear method, the distance among input samples so small that it can not effectively approximate using CNNT algorithm, however, once using MNNT algorithm, the mapping relationship can be accurately represented between input samples and output samples. Secondly, parameters can be estimated effectively using the MNNT algorithm, but it cannot be done well with CNNT. Lastly, since The MNNT algorithm has few restricting factors, it can be widely used in radar field.

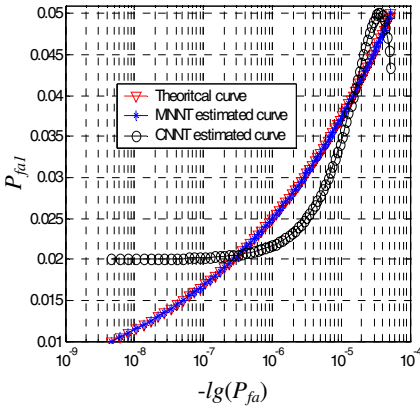


Fig. 5. The MNNT and CNNT trained results for BSWID

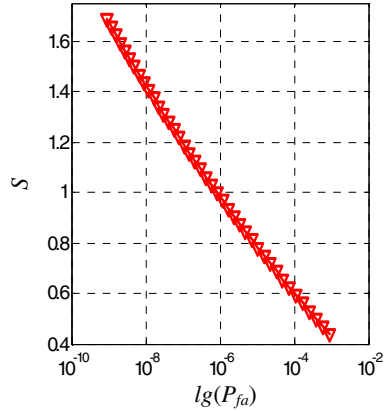


Fig. 6. Relationship between P_{fa} and T for GO CFAR

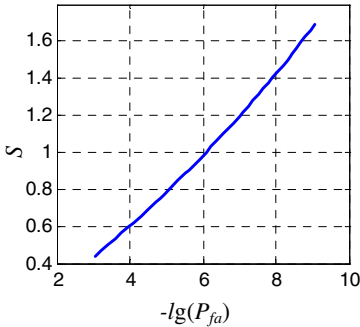


Fig. 7. Relationship between $G[P_{fa}]$ and S for GO CFAR

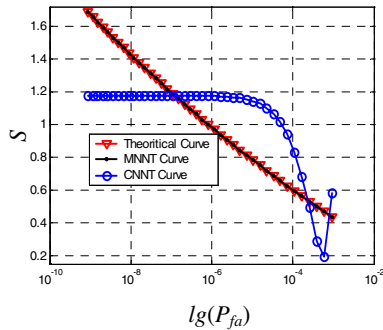


Fig. 8. The MNNT and CNNT trained results for GO CFAR

5 Conclusion

In order to estimate parameter accurately in radar detection, the RBF NNT is applied because of its advantages, such as good nonlinear approximation ability and quick convergence speed; meanwhile, considering the characters of parameter estimation in radar detection, a MNNT via RBF is proposed to estimate parameters more accurately, and as examples, two classical radar detection problems are listed in this paper. Theoretical analysis and numerical results both show that the MNNT algorithm has good parameters estimation accuracy, and can be widely used in radar field.

References

1. Schilling, R.J., Carroll, J.J., Al-Ajlouni, A.F.: Approximation of Nonlinear Systems with Radial Basis Function Neural Networks [J], *IEEE Trans. on Neural Networks* **12** (1) (2001) 1-15
2. Leshno, M., Lin, V.Y., Pinkus, A., Schocken, S.: Multilayer Feedforward Networks with a Nonpolynomial Activation Function can Approximate any Function [J], *IEEE Trans. on Neural Networks* **6** (1) (1993) 861-867
3. Moody, J., Darken, C.: Fast Learning in Networks of Locally-Tuned Processing Units [J], *Neural Computation* **2** (1) (1989) 281-294
4. Orr, M.J. L.: Regularization in the Radial Basis Function Centers[J], *Neural Computation* **7** (3) (1995) 606-620
5. Chen, S., Cowan, C.F.N., Grant, P.M.: Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks [J], *IEEE Trans. on Neural Networks* **2** (2) (1991) 302-309
6. Chen, S., Grant, P.M., Cowan, C.F.N.: Orthogonal Least Squares Learning Algorithm for Training Multioutput Radial Basis Function Networks [J], *IEE Proc.-F***139** (6) (1992) 378-384
7. Skolnik, Merrill I.: *Radar Handbook* [M], The McGraw Companies, Inc. 2nd Edition, 1990

Driving Load Forecasting Using Cascade Neural Networks^{*}

Zhancheng Wang¹, Guoqing Xu², Weimin Li³, and Yangsheng Xu¹

¹ The Chinese University of Hong Kong, Shatin NT, Hong Kong
{zawang, ysxu}@mae.cuhk.edu.hk

² Tongji University, Shanghai, China

³ Shanghai Jiao Tong University, Shanghai, China

Abstract. This paper presents an approach for solving the driving load forecasting problem based on Cascade Neural Networks with node-decoupled extended Kalman Filtering (CNN-NDEKF). Because of the inherent advantages, hybrid electric vehicles (HEV) are being given more and more attention. The power control strategy of HEVs is the key technology which determines the HEV's efficiency and pollutive emission level. Since the extent of improvement involved with HEV power control strategies greatly depends on the future driving load forecasting, in this paper, we attempt to achieve driving load forecasting using CNN-NDEKF. Instead of forecasting the entire load sequence, we define 5 load levels by a fuzzy logic method and then we forecast the load level. Simulation study is given to illustrate the feasibility of the driving load forecasting approach.

1 Introduction

Aimed at solving the more and more serious problems of energy and pollution, HEV is one of the best practical applications for transportation with high fuel economy and low emission. The power control strategy of HEVs is the key technology which makes the HEVs more efficient and less pollutive. Since the extent of improvement involved with HEV power control strategies greatly depends on the driving load forecasting, CNN is used to forecast the driving load in this paper.

The main control objective of power control is to satisfy power requirement with the cooperation of electric power source and fuel power source. At the same time, the corresponding fuel consumption and emissions should be as low as possible. For a HEV, the fundamental problem is the optimal split of the power of combustion engine and electric motor at an optimum engine speed. It has to be solved “online” during the operation. In general, power control strategies can be roughly classified into three kinds.

The first approach is rule-based algorithm, such as energy follow and thermostatic. In energy follow strategy, the engine is always on, and it changes the

^{*} The work described in this paper is supported by the Innovation Technology Fund of the Hong Kong Special Administrative Region (GHP/011/05).

energy output based on the energy requirement. The thermostatic control strategy obtains electric power through generator and engine, and it turns the engine on and off based on SOC [1]. This kind of control strategy works fast and reliable, but often produces results far away from an optimal control.

The second approach is based on static optimization method. The optimization solutions figure out the proper split between the electric power and fuel power using steady-state efficiency maps, and realize HEV optimization based on the operation with the optimal parameters [2], [3]. However, how to determine the efficiency maps and optimal parameters are the challenges of this kind of approach.

The third approach is real-time optimization. Several algorithms have been proposed, including fuzzy logic controller [4], energy-flow analysis [5], Dynamic Programming [6], etc. The control strategy with real-time optimization calculates the optimal torque based on the feature parameters of vehicle, and decides the actual torque output by modifying the optimal torque based on real-time road situation and SOC. Dynamic optimization parameters can be changed based on real-time operation state and energy requirement, then present a real-time optimal solution for power control. This kind of approach are more accurate under transient conditions than the first two methods, but the inherent disadvantages of heavy computational cost limits its application.

In general, existing approaches have some limitations that they used only the current vehicle state for decision-making in power distribution; little consideration is generally given to the future load. As a result, the existing approaches do not consider the effects of variations on the vehicle emissions and fuel consumption over the future forecasting load to which the vehicle may be subjected. After studying the methods for HEV power control strategy, in this paper, we use CNN to forecast the future load of HEVs.

The rest of the paper is organized as follow: In section 2, the definition of load level will be introduced. The load forecasting approach based on CNN-NDEKF are described in Section 3. Section 4 is devoted to the presentation of simulation results. Conclusions are presented in Section 5.

2 Definition of Load Level

The driving load in a future time slot is proportional to the vehicle net wheel torque as calculated by equation (1):

$$T_{T_s} = \frac{1}{T_s} \int_0^{T_s} T_{wh(t)} dt \quad (1)$$

$$T_{wh}(t) = a(t)M_r r_d + B_{wh}V(t) + \frac{V(t)r_d^2}{|V(t)|}(F_r + F_a(V(t)))$$

where T_{T_s} is the average torque of the future time slot T_s , $T_{wh(t)}$ is the net wheel torque at time t , $a(t)$ is the acceleration of the vehicle at time t , $M_r = M_v + J_r/r_d^2$ is the effective mass of the vehicle, M_v is the mass of the vehicle, J_r is the equivalent moment of inertia of the rotating components in the vehicle. r_d is

the dynamic tire radius, B_{wh} is the viscous damping, F_r and F_a are the rolling resistance force and the aerodynamic drag force. Obviously, the relationship between the driving load and the acceleration/speed of the vehicle is complex and nonlinear.

There are infinite kinds of future load sequences. Theoretically speaking, it may be best if we could forecast an entire load sequence, however, forecasting the entire load sequence will cost enormous computation and is not necessary. After considering the trade-offs between computational cost and HEV power control strategy requirement, in this paper, we divide the driving load into 5 levels which are very low (VL), low (L), medium (M), high (H) and very high (VH) and a fuzzy logic approach is used to determine the driving load level.

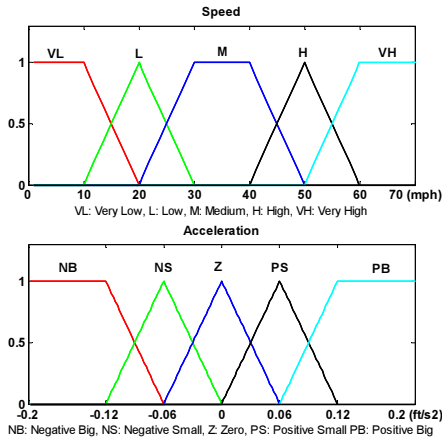


Fig. 1. Membership function for input variables

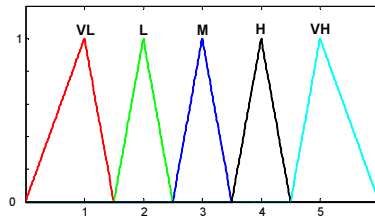


Fig. 2. Membership function for output variables

The average acceleration and speed are chosen as the input parameters of the fuzzy logic approach and are divided into 5 levels. Membership functions of the fuzzy sets are shown in Figure (1) and Figure (2). The fuzzy rules are represented in a linguistic form in Table 1. The rules are formulated based on the experience gained. To reduce the computational burden of defuzzification, the centroid method is selected to be the defuzzifier. After gaining a single output value, we round it to one of the 5 driving load levels.

Table 1. Fuzzy rules

		<i>Acceleration</i>				
		<i>NB</i>	<i>NS</i>	<i>Z</i>	<i>PS</i>	<i>PB</i>
<i>Speed</i>	<i>VL</i>	<i>VL</i>	<i>VL</i>	<i>VL</i>	<i>L</i>	<i>L</i>
	<i>L</i>	<i>VL</i>	<i>L</i>	<i>L</i>	<i>L</i>	<i>M</i>
	<i>M</i>	<i>L</i>	<i>M</i>	<i>M</i>	<i>M</i>	<i>H</i>
	<i>H</i>	<i>M</i>	<i>M</i>	<i>H</i>	<i>H</i>	<i>VH</i>
	<i>VH</i>	<i>H</i>	<i>H</i>	<i>VH</i>	<i>VH</i>	<i>VH</i>

VL: Very Low, L: Low, M: Medium, H: High, VH: Very High

NB: Negative Big, NS: Negative Small, Z: Zero,

PS: Positive Small, PB: Positive Big

For example, if the membership of i th speed is μ_{S_i} , and that of acceleration is μ_{A_i} , the firing strength, μ_i , of the premise is calculated based on min operator. The firing strength of each rule is calculated as follows:

$$\mu_i = \min(\mu_{S_i}, \mu_{A_i}) \quad (2)$$

After the firing strength μ_i is calculated for all 25 rules, the corresponding driving load level is obtained by a weighted average given by the equation (3)

$$L = \text{round}\left(\frac{\sum_{i=1}^{25} \mu_i \alpha_i}{\sum_{i=1}^{25} \mu_i}\right) \quad (3)$$

where α_i is a value of consequent parts on i th rule.

3 Load Forecasting Approach Based on CNN-NDEKF

Our goal is to decide the driving load level in a future time slot T_s based on the past driving sequence so that the value of the load level can be used in the HEV power control strategy to reduce the fuel consumption and emission. We address this problem as a pattern recognition problem using CNN-NDEKF.

3.1 A Brief Introduction of CNN-NDEKF

In recent years, neural networks have shown great promise in identifying complex non-linear mappings from observed data, and have found many applications in non-linear system. Despite significant progress in the application of neural networks to many real-world problems, however, the vast majority of neural network research still relies on fixed-architecture networks trained through backpropagation or some other slightly enhanced gradient descent algorithm. There are two main problems with this prevailing approach. First, the ‘‘appropriate’’ network architecture varies from application to application; yet, it is difficult to guess this architecture, the number of hidden units and number of layers - a priori for a specific application without some trial and error. Even within the same application, functional complexity requirements can vary widely, as is the case, for example,

in modeling human tracking strategies from different individuals [7]. Second, the backpropagation and other gradient descent techniques tend to converge rather slowly, often exhibit oscillatory behavior, and frequently convergence to poor local minima.

Therefore, Nechyba and Xu [8] developed a new neural network learning architecture to counter these problems mentioned above. This neural network is well known flexible Cascade Neural Network with Node-Decoupled Extended Kalman Filtering (CNN-NDEKF). Below, we briefly summarize the CNN-NDEKF training algorithm and why we selected this learning algorithm to forecast the future driving load level.

Firstly, no a priori model structure is assumed; the neural network automatically adds hidden units to an initially minimal network as the training requires. Fig 3 illustrates how a two point, single-output network grows as two hidden units are added. Thus, a cascade network with inputs, hidden units and outputs, has connection where,

$$n_w = n_{in}n_0 + n_h(n_{in} + n_0) + (n_h - 1)\frac{n_h}{2} \tag{4}$$

Secondly, hidden unit activation function are not constrained to be a particular type. Rather, for each new hidden unit, the incrementally learning algorithm can select that functional form, which maximally reduces the residual error over the training data. Typical alternatives to the standard sigmoidal function are sine, cosine, and the Gaussian function.

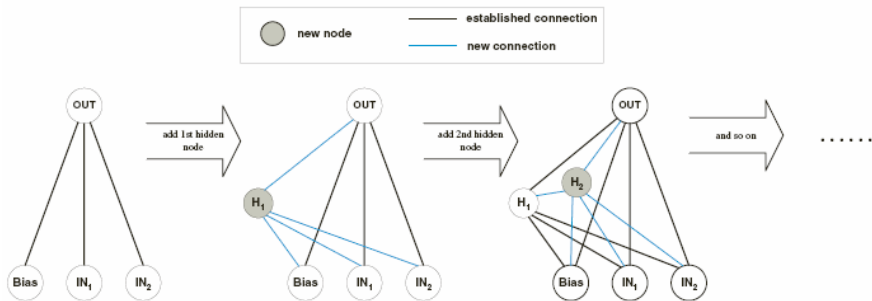


Fig. 3. The cascade learning architecture

Finally, it has been shown that NDEKF, a quadratically convergent alternative to slower gradient descent training algorithms, such as backpropagation or quickprop, fits well within the cascade learning framework and converges to good local minima with less computation. NDEKF is a natural formulation for cascade learning for we only train the input-side weights of one hidden neuron and the output units at any one time; we can partition the m weights by unit into groups—one group for the current hidden unit, groups for the output units. In fact,

by iteratively training one hidden unit at a time and then freezing that unit's weights, we minimize the potentially detrimental effect of the node-decoupling.

Denote ω_k^i as the input-side weight vector of lengths m_i at iteration k , for unit $i \in 0, 1, \dots, n_0$, where $i = 0$ corresponds to the current hidden unit being trained, and $i \in 0, 1, \dots, n_0$ corresponds to the i th output unit.

The NDEKF weight-update recursion is given by

$$\omega_{k+1}^i = \omega_k^i + (\psi_k^i)^T (A_k \xi_k) \phi_k^i \tag{5}$$

where ξ_k is the n_0 -dimensional error vector for the current training pattern, ψ_k^i is the n_0 -dimensional error vector for the partial derivatives of the network's output unit signals with respect to the i th unit's net input, and

$$\phi_k^i = P_k^i \zeta_k^i \tag{6}$$

$$A_k = (I + \sum_{i=0}^{n_0} [(\psi_k^i)^T \phi_k^i] [\phi_k^i (\phi_k^i)^T])^{-1} \tag{7}$$

$$P_{k+1}^i = P_k^i - \phi_k^i (\phi_k^i)^T (A_k \phi_k^i) \phi_k^i (\phi_k^i)^T + \eta I \tag{8}$$

where $(\psi_k^i)^T$ is the m_i -dimensional input vector for the i th unit, P_{k+1}^i is the $m_i \times m_i$ approximate conditional error covariance matrix for the i th unit, and η is a small real number which alleviates singularity problem for P_{k+1}^i .

The flexible functional form which cascade learning allows, is ideal for forecasting the future driving load levels. By making as few prior assumptions as possible in driving load forecasting, we improve the likelihood that the learning algorithm will converge to a good predictive model of the driving data.

The goal that we are achieving here is to forecast the future driving load level. We consider the driving load sequence as a measurable stochastic process and the knowledge behind it as a underlying stochastic process. A CNN-NDEKF is employed to generate classifier for future load level identification, and the model parameters are updated through a learning process which ensures the model best represents the training data. The procedures for CNN-NDEKF-based learning can be summarized as follows:

1. Initially, there are no hidden units in the network, only direct input-output connections. These weights are trained first, thereby capturing any linear relationship between the inputs and the outputs.
2. With no further significant decrease in the root mean square (RMS) error between the network outputs and the training data (eRMS), a first hidden unit is added to the network from a pool of candidate units. These candidate units are trained independently and in parallel with different random initial weights by using the quick-prop algorithm.
3. The best candidate unit will be selected and installed into the network if no more appreciable error reduction occurs, therefore, the first hidden node is produced.
4. Once the hidden unit is installed, the hidden-unit input weights are frozen, while the weights to the output unit is going to train again. This allows for such

faster convergence of the weights during training than a standard multi-layer feed-forward network.

5. This process (from step2-step4) is repeated until the eRMS reduces sufficiently for the training set or the number of hidden units reach a predefined maximum number.

In modeling future driving load level, as with other poorly understood phenomena, we must rely on modeling by observation, or learning, rather than theoretical or physical derivation. A future driving load level is characterized by unique, complex, and unknown properties; so we require a learning paradigm that can cope with many difficult challenges, first of all, little if anything is known a priori about the a) structure, b) order, c) granularity, or d) delay. Second, future driving load level is dynamic, stochastic, and nonlinear in nature. It is prone to gradual changes over time. In order to address these challenges, the CNN-NDEKF mentioned above can satisfy the requirement by learning driving load data.

3.2 Training Samples Build-Up

In order to develop a load forecasting classifier, a CNN-NDEKF is selected due to its effectiveness in the classification of complex and nonlinearly separable target classes. A CNN-NDEKF classifies its input vector into one of 5 target load levels through a two stage process.

To train the CNN network for load forecasting classification problem, the statistics of three typical drive cycles, i.e. US06 (average speed: 77.2mh/h), NEDC (average speed: 33.3mh/h) and Manhattan (average speed: 10.98mh/h) will be calculated to generate the training database. These cycles represent a wide variety of conditions because no two cycles seem to provide similar features they are significantly different in acceleration rate, average speed.

The corresponding flow diagram to prepare training data is shown in Figure 4 and the procedure for one driving cycle is described as below:

1. Initialize the segment lengths of T_p and T_f . T_p represents a past time slot which are used to forecast future load level, while T_f represents an objective future time slot.
2. Randomly choose a start point T_s , if $T_s + T_p + T_f$ is larger than the driving cycle length or T_s is smaller than T_p , go to step 2.
3. Apply fast fourier transform (FFT) on the past driving cycle to select characteristic features and generate CNN-NDEKF input vector P .
4. Calculate the average speed and acceleration of the future driving cycle. Apply the fuzzy logic approach introduced in Section 2 to calculate the load level as CNN-NDEKF output y . Generate a training sample $\{P, y\}$
5. Go to step 2 until there are 100 training samples.

Since three driving cycles are considered in this paper and one hundred training samples are generated from each driving cycle, we can obtain 3×100 training samples totally.

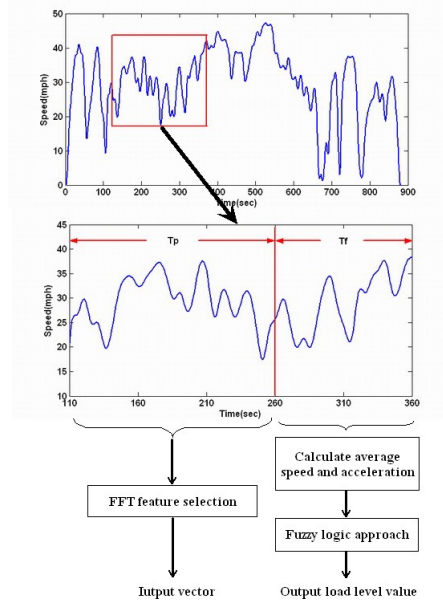


Fig. 4. Flow of training data preparation

4 Simulation Study

4.1 Segment Selection

The selection of T_p and T_f will influence the performance of training significantly. Generally speaking, a larger T_p means more past information will be considered so that the corresponding result will be better while the computational cost will be much more. Similarly, a smaller T_f implies that the classifier will work more frequently but cost more computation.

In order to balance the performance and cost, in this paper, we let $T_p = 150$ and $T_f = 100$. The reason is that the typical cycles in urban traffic situations is approximately 180 seconds. So the value of T_p , i.e. 150 seconds, which is slightly less than 180 seconds, will be enough to reflect the past driving cycle. The value of T_f , i.e. 100 seconds, will make the classifier work in every 100 seconds so that it will not miss any traffic situation changes.

4.2 Feature Selection

The past driving cycle can not be used to train directly because the dimension of the past driving cycle, i.e. 150, is so large that it requires numerous instances to determine the result. In our application, fast fourier transform (FFT) is used on past driving cycle to select characteristic features and reduce the dimension. The first n coefficients of FFT transform are chosen to form a new input vector.

Table 2. Training performance comparison of different number of coefficients

Features Number	4	6	8	10	13	15	20
Error Rate	36.67%	22.67%	10.67%	7.5%	7.333%	6.67%	6%

In Table 2, we compare the training performance by choosing different numbers of FFT coefficients. It shows we can achieve better accuracy by choosing more coefficients, while the time consumption increases. It can be seen that when the number of coefficients is bigger than 10, the accuracy increases slowly. Therefore, in the experiments of this paper, we choose 10 features to generate new input vectors. Moreover, a 92.5% accuracy proves the feasibility of CNN-NDEKF to future driving load classification problem.

4.3 Error and Hidden Unit

In CNN-NDEKF, *MaxHidden* is an very important parameter to the classification result. In Table 3, we compare the number of Maximum Hidden Unit and error classification rate of the CNN-NDEKF classifier with different *MaxHidden* value.

Table 3. Number of max hidden unit versus error rate

MaxHidden	2	4	5	6	8	10	15	20
Error Rate	0.2667	0.1333	0.075	0.1733	0.2267	0.2333	0.2333	0.24

As shown in Table 3, when *MaxHidden* is 5, we get a minimum classification error rate 7.5%. This result shows that our approach can get a high accuracy in future driving load level classification. When *MaxHidden* increases from 2 to 5, the error rate reduces from 26.67% to 7.5%. Contrarily, it increase from 7.5% to 24% when *MaxHidden* increases from 5 to 20. A larger *MaxHidden* might correspond to higher testing accuracy, as well as more iterations, and however over fitting will occur. To avoid the over fitting, it can not be too large. In this paper, we choose the max hidden unit number as 5.

4.4 Case Study

After obtaining the CNN-NDEKF classifier in the previous sections, we use 5 different standard driving cycles which are Highway Fuel Economy Test (HWFET), Urban Dynamometer Driving Schedule (UDDS), City Driving for a Heavy Vehicle (WVUCITY), Interstate Driving for a Heavy Vehicle (WVUINTER) and New York City Cycle (NYCC) to evaluate it. Obviously, these cycles represent a wide variety of conditions. It is effective to validate the classification performance by these driving cycles.

Table 4. Training performance pomparison of different number of coefficients

Driving Cycle	HWFET	UDDS	WVUCITY	WVUINTER	NYCC
Error Rate	11.33	8.67	11.33	16.67	17.33

Simulation results of this case are shown in Table 4. It can be seen that an average error rate of 13.066% is gained, which proves the feasibility of the our driving load forecasting approach.

5 Conclusions

In this paper, we present a future driving load forecasting approach based on CNN-NDEKF classification method. We use a fuzzy logic method to determine the driving load and forecast the load level instead of forecasting the entire load sequence. Simulation study is given to illustrate the feasibility of the driving load forecasting approach. Further works focus on utilizing the future driving load level in the HEV power control approach to improve the control results.

References

1. Jalil, N., Kheir, N.A., Salman, M.: A rule-based energy management strategy for a series hybrid vehicle. Proceedings of the American Control Conference, Albuquerque, New Mexico, June 1997
2. Kim, C., NamGoong, E., Lee, S.: Fuel economy optimization for parallel hybrid vehicles with CVT. SAE, Paper no. 1999-01-1148
3. Paganelli, G., Ercole, G., Brahma, A., Guezennec, Y., Rizzoni, G.: A general formulation for the instantaneous control of the power split in charge-sustaining hybrid electric vehicles. Proceedings of 5th International Symposium of Advanced Vehicle Control, Ann Arbor, MI, 2000
4. Lee, H.D., Sul, S.K.: Fuzzy-logic-based torque control strategy for parallel-type hybrid electric vehicle. IEEE Transactions, Industrial Electronics **45** (1998) 625-632
5. Piccolo, A., Ippolito, L., Galdi, V.Z., Vaccaro, A.: Optimization of energy flow management in hybrid electric vehicles via genetic algorithm. Proceedings of IEEE/ASME International Conference on Advanced Intelligent Mechatronics, Como, Italy, July 2001
6. Lin, C.C., Peng, H., Grizzle, J.W., Kang, J.M.: Power management strategy for a parallel hybrid electric truck. IEEE Transactions on Control Systems Technology **11** (2003) 830-849
7. Nechyba, M., Xu, Y.S.: Cascade neural networks with node decoupled extended Kalman filtering. Proc. IEEE Int. Comput. Intell. Robot. Automat. Symp., Monterey, CA, July 1997
8. Nechyba, M., Xu, Y.S.: Human control strategy: abstraction, verification and replication. IEEE Control System Magazien, October, 1997

Minimal Resource Allocation on CAN Bus Using Radial Basis Function Networks*

Yan Hao Wei¹, Mu-Song Chen, Chuan Ku Lin, and Chi-Pan Hwang²

¹ Department of Electrical Engineering, Da-Yeh University
chenms@mail.dyu.edu.tw

² Department of Electrical Engineering, National Chung-Hwa University
cphwang@cc.ncue.edu.tw

Abstract. Optimal message scheduling is one of the key issues in the field of controller area network (CAN) bus system. There are numerous approaches related to this issue. Most of them are essentially based on priority-based strategies. In 1, we utilized Radial Basic Function (RBF) network 2 as a message scheduling controller to dynamically schedule messages. Furthermore, an online Backward-Through-Time (BTT) algorithm is presented for parameter optimization under *a priori* fixed network structure. Intuitively, an inappropriate RBF network structure leads to performance degradation. In the worst case, the CAN system diverges. In this paper, we extend our previous works by including Minimal Resource Allocation (MRA) algorithm for structure determination. In this way, both problems of parameter optimization and structure determination can be resolved at the same time. Simulation results demonstrated that the proposed BTT with MRA methods outperform our previous results in terms of convergence time, stability, and the number of required hidden neurons (or radial basis functions).

1 Introduction

CAN 34 is a real-time distributed bus protocol with many desirable properties for embedded and online systems. It is a priority-based mechanism where collisions are avoided by using priorities for bus arbitration. The priority-based arbitration mechanism requires that different CAN nodes never simultaneously send messages with equal identifiers. According to this mechanism, as soon as the bus is idle, each node competing for the bus begins to send the arbitration field of its message. At the end of the arbitration field, only the node which is sending the message with the lowest arbitration field value, will be transmitting.

CAN must carry both periodic and sporadic real-time messages, as well as non real-time messages. All these messages must be properly scheduled on the network so that real-time messages meet their deadlines while co-existing with non real-time messages. Previous work regarding scheduling such messages on CAN bus focused on fixed-priority mechanisms 56. Although static systems can be scheduled easily by fixed-priority scheduling, it does not suitable on scheduling of dynamic systems, where an

* This research was supported by the National Science Council under contract number NSC95-2221-E-212-062.

offline feasibility test has incomplete knowledge about the future behavior of the system. In general, fixed-priority schemes give lower utilization than other schemes such as non-preemptive earliest deadline first (EDF) 7. Based on the EDF access mechanism, soft real-time communication will be scheduled optimally with EDF approach. To guarantee deadlines of hard real-time communication, calendar-based resource reservation is applied. An application of this scheduling approach in a distributed object-oriented real-time control system has been introduced by 8. The drawback of EDF is its high overhead which makes EDF impractical for CAN system. Recently, a novel approach 1 based on the concept of controller-plant model is proposed to realize real-time scheduling on CAN system (or plant) for different messages scheduling. These messages are classified into three broad categories: (1) hard real-time (HRT) messages, (2) soft real-time (SRT) messages, and (3) non real-time (NRT) messages, respectively. The controller is implemented by RBF network. Due to its self-learning capability, the RBF network is treated as an adaptive message scheduling controller (MSC), which dynamically accommodates itself to the current network flow rate and waiting time for existing messages. It is apparent that the purposes of the MSC are to avoid delay transmissions. Simulation results show that the MSC can reduce failure rate of messages sending efficiently, even in high flow rate. However, the network structure of the MSC, i.e., the number of RBFs¹ is determined by an ad-hoc method. Although the structure of the MSC can be changed to achieve an optimal configuration, several trials have to be applied such that the minimum failure rate can be attained. From our knowledge, the number of RBFs influences the failure rate significantly. During stage of adding/deleting neurons, undesired transition time makes the system unstable and leads to performance degradation. To overcome this problem, a more advanced and efficient structure adaptation algorithm called Minimal Resource Allocation (MRA) method 9, is applied to reduce this instability.

The rest of the paper is organized as follows. Section 2 introduces the basic structure of MSC. The advantages and drawbacks of the MSC are discussed. To defeat the problems of the given MSC, the MRA algorithm is described in section 3. Experimental simulations compared with 1 are conducted in section 4. Finally, section 5 gives our conclusions and future works.

2 Message Scheduling Controller (MSC)

In this section, the complete framework of the proposed controller-plant model is depicted in Fig 1 and the functions of MSC are introduced briefly. In Fig 1, MSC is constructed by a RBF network which is a special structure of neural networks with single hidden layer. The structure of RBF network is described in Fig 2. The values of the input variables formulate an input vector \mathbf{x} , which is forwarded from the input layer to the hidden layer. The hidden layer is comprised of a number of nonlinear processing nodes or radial basis function $\psi_k(\mathbf{x})$. In this paper, we employ Gaussian function as the radial basis function

$$\psi_k(\mathbf{x}) = \exp\left(-\frac{1}{2} \frac{\|\mathbf{x} - \mathbf{c}_k\|^2}{\sigma_k^2}\right), \quad \mathbf{x} \in R^n \quad (1)$$

¹ In this paper, the number of RBFs is the same as the number of hidden neurons. In the rest of the paper, "RBFs" and "hidden neurons" will be used interchangeably.

where \mathbf{c}_k and σ_k are center vector and width, respectively. The output of each hidden node is then multiplied by a particular synaptic weight w_{kj} , while the final output of the network is a simple summation of all the weighted hidden node activations. In formula,

$$y_i = \sum_{k=1}^p w_{ki} \psi_k(\mathbf{x}), \quad \forall i \tag{2}$$

In 0, w_{ki} is the connection weight from k th RBF to output y_i . The output vector $\mathbf{y}(t) = [y(\text{HRT},t) \ y(\text{SRT},t) \ y(\text{NRT},t)]^T$ should denote corresponding probabilities for different types of messages Ω_i and $\Omega_i \in \{\text{HRT}, \text{SRT}, \text{NRT}\}$. The one with maximum value in \mathbf{y} ($k = \arg \max_i \{y_1(t), y_2(t), y_3(t)\}$, $\Omega_k \in \{\text{HRT}, \text{SRT}, \text{NRT}\}$) indicates the most appropriate message type for transmitting in the next time instant. In 1, input vector is defined as $\mathbf{x}(t) = [\mathbf{D}(t) \ \mathbf{F}(t)]^T$, where $\mathbf{D}(t) = [D(\text{HRT},t) \ D(\text{SRT},t) \ D(\text{NRT},t)]^T$ and $\mathbf{F}(t) = [F(\text{HRT},t) \ F(\text{SRT},t) \ F(\text{NRT},t)]^T$ are waiting time and flow rate of messages, respectively. In our design, HRT messages should deserve more bandwidth than the others, while NRT messages have the lowest priority.

It is obvious that the fitting parameter vector in Fig 2 can be expressed as $\Theta = [w_{ij}, \mathbf{c}_i, \sigma_i]^T, \forall i, j$ and w_{ij}, \mathbf{c}_i , and σ_i can be tuned by backpropagation learning on the basis of steepest descent iterations

$$\begin{aligned} w_{ij} &\leftarrow w_{ij} + \eta e_j \psi_i \\ \mathbf{c}_i &\leftarrow \mathbf{c}_i + \eta e_j w_{ij} \psi_i \frac{\|\mathbf{x} - \mathbf{c}_i\|}{\sigma_i^2} \\ \sigma_i &\leftarrow \sigma_i + \eta e_j w_{ij} \psi_i \frac{\|\mathbf{x} - \mathbf{c}_i\|^2}{\sigma_i^3} \end{aligned} \tag{3}$$

as long as the desired outputs t_k can be obtained. Unfortunately, the desired target values are always unavailable and the supervised learning can not proceed. Therefore, a Backward-Through-Time algorithm (BTT) is presented for online supervised learning. Detail descriptions of BTT algorithm can refer to 1.

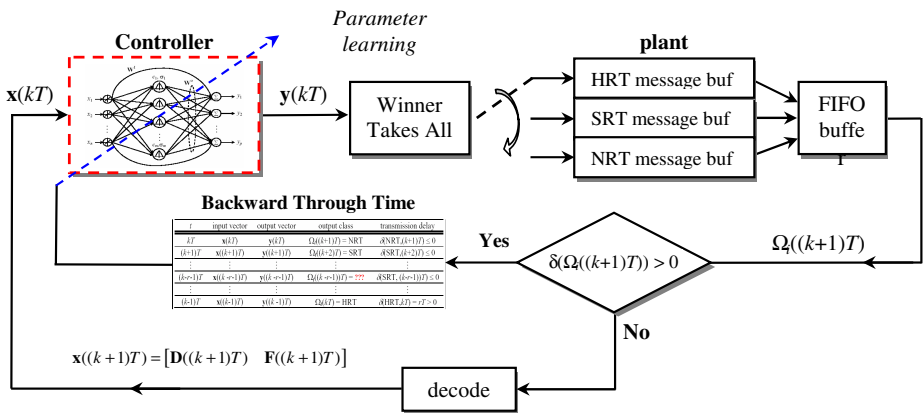


Fig. 1. The proposed framework of controller-plant model for message scheduling on CAN bus

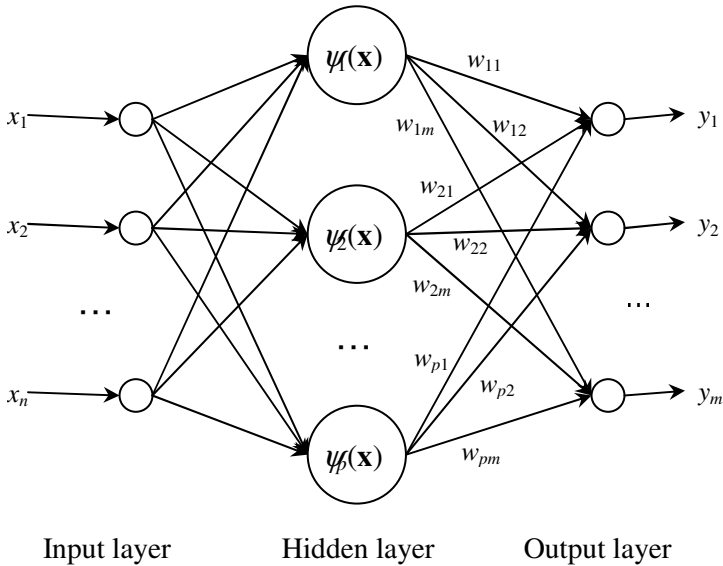


Fig. 2. RBF network with n inputs, p RBFs, and m outputs

Although the supervised learning can minimize output errors, it is apparent that most of the literature related to RBF network optimization considers a fixed topology and proposes methods for optimizing the set of parameters under that unchangeable structure, i.e., a fixed number of RBFs. The specifications of the topology can be supplied either by human experts or by exhaustive trial-and-error studies. However, consulting an expert may be difficult and/or expensive; furthermore, there may be even no human experts available when the RBF network must be constructed. Therefore, structure identification is crucial and important. The fact that much more effort has been dedicated to dealing with the problem of parameter adjustment than that of structure identification is understandable since the latter is a very complicate task. An additional difficulty is that unlike methods for parameter adjustment, it is not possible to test structure identification algorithms without using parameter adjustment algorithms. Thus, both problems must be tackled simultaneously. Recently, an online structural adaptive learning algorithm is suggested by Lu Yingwei et al. 9. The MRA algorithm combines the growth criteria of resource allocation with a pruning strategy to realize a minimal network structure. In MRA, radial basis functions are added based on the novelty of the new data and the weights connecting the hidden neurons to the outputs are estimated using the least squares method or its alternatives. On the other hand, neurons that consistently made little contributions to the network output are considered to be removed. In the following section, we will discuss the MRA algorithm more in detail.

3 Minimal Resource Allocation (MRA)

In the MRA algorithm, the RBF network begins with no hidden units. As long as each new input-output data, which is decided by BTT algorithm, is received, the network is

built up based on certain growing and pruning criteria. MRAN came with a dynamic network topology structure by adding the hidden neurons sequentially based on the novelty of the input data or by deleting neurons on the basis of the normalized contribution of each hidden neuron to the overall outputs. A new input pattern \mathbf{x} is considered as novel (1) if that sample is far away from the existing nearest center \mathbf{c}_{ir} and (2) if the error between the output $\mathbf{y}(kT)$ and the target $\mathbf{d}(kT)$ is large, and (3) If the root mean square value of the output error over a sliding window M is above a predefined value. In formula

$$\|\mathbf{x} - \mathbf{c}_{ir}\| > \theta_1 \tag{4}$$

$$\|\mathbf{y}(kT) - \mathbf{d}(kT)\| > \theta_2 \tag{5}$$

$$\sqrt{\frac{1}{M} \sum_{k=q-(M-1)}^q \|\mathbf{y}(kT) - \mathbf{d}(kT)\|^2} > \theta_3 \tag{6}$$

The purpose of introducing a sliding window reduces the effect of noise and ensures smooth change of the number of hidden neurons. If the input pattern satisfies the above-mentioned criteria in Esq. 0, 0, and 0, a new hidden neuron is added and the corresponding parameters \mathbf{c}_{new} , σ_{new} , and \mathbf{w}_{new} should be initialized accordingly to ensure stable transition

$$\begin{aligned} \mathbf{c}_{new} &= \mathbf{x}, \quad \sigma_{new} = \kappa \|\mathbf{x} - \mathbf{c}_{ir}\| \\ \mathbf{w}_{new} &= \|\mathbf{y}(kT) - \mathbf{d}(kT)\| \end{aligned} \tag{7}$$

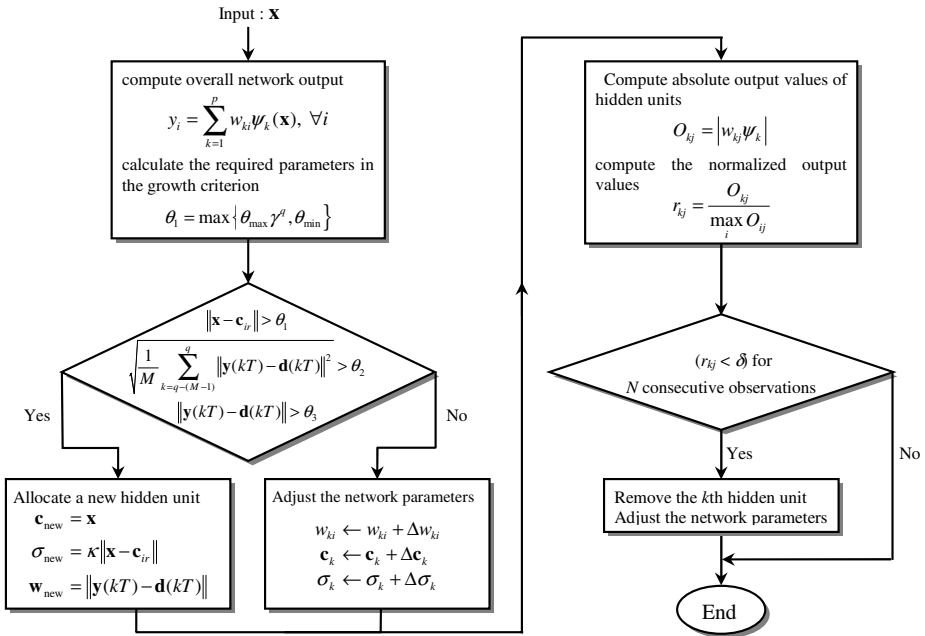


Fig. 3. The flowchart of MRA algorithm

If the input pattern does not pass the criteria for novelty, then no hidden neuron will be added and the network parameters will be adjustment either by the Least Mean Square (LMS) or Extended Kalman Filter (EKF) algorithms.

To avoid excessive number of hidden neurons, which make the network susceptible to noise and cause the problem of overfitting, a pruning process proceeds to get rid of the inactive neurons. In pruning mechanism, a redundant neuron which has smallest contribution to overall RBF network is always removed. The contributions r_{kj} of each hidden neuron is measured by the normalized output of each hidden neuron to all the outputs

$$r_{kj} = \frac{O_{kj}}{\max_{\forall i} \{O_{ij}\}}, \quad \forall k, j \tag{8}$$

where $O_{kj} = |w_{kj}\psi_k|$. The normalized output of each neuron r_{kj} is then observed for N consecutive inputs. Hidden neuron k is pruned, if its output r_{kj} falls below a threshold value for the N consecutive inputs. Fig 3 shows the flowchart of MRA algorithm.

4 Simulation Results

In this section, the applicability of the proposed BTT with MRA method is demonstrated through simulation examples and the results are compared with our previous works. The messages are generated randomly in a predefined range and the flow rate is defined by the traffic load $F(\Omega_i)$, where $\Omega_i \in \{HRT, SRT, NRT\}$. Table 1 summarizes these essential information, including $D_{\max}(\Omega_i)$ and $F(\Omega_i)$. It should be noted that the dimensions of F are defined as the number of messages have to be served in per time unit T .

Table 1. Simulation conditions

Maximum transmission time	$D_{\max}(HRT) = 120, D_{\max}(SRT) = 200, D_{\max}(NRT) = 400$
Traffic load	$F(HRT) = 30\sim 55, F(SRT) = 30\sim 55, F(NRT) = 30\sim 55$

Fig 4. shows the simulation results by using BTT method for parameter identification and MRA algorithm for structure determination. The MSC starts with no hidden neuron. The bar chart in Fig 4(b)(c)(d) indicates the number of delay-transmitted messages in each 500 T interval. In the first 1,000 T, all messages suffer from a high delay rates. This is due to the fact that the MSC gains less experience from the input data. Therefore, it fails to capture the dynamics of the underlying system. As simulation goes on, the MRA algorithm starts to add more hidden neurons to accommodate with the incoming data. After 1,000 T, the system is complex enough to manage messages scheduling properly and no more delay transmission happens. Finally, the number of necessary hidden neurons is six in this simulation as shown in Fig 4(a).

To make the simulation results more comparable with the first one, we conduct the second experiment using an ad-hoc structure determination method proposed by the first author 10. In this trial, all simulation conditions are the same as listed in Table 1.

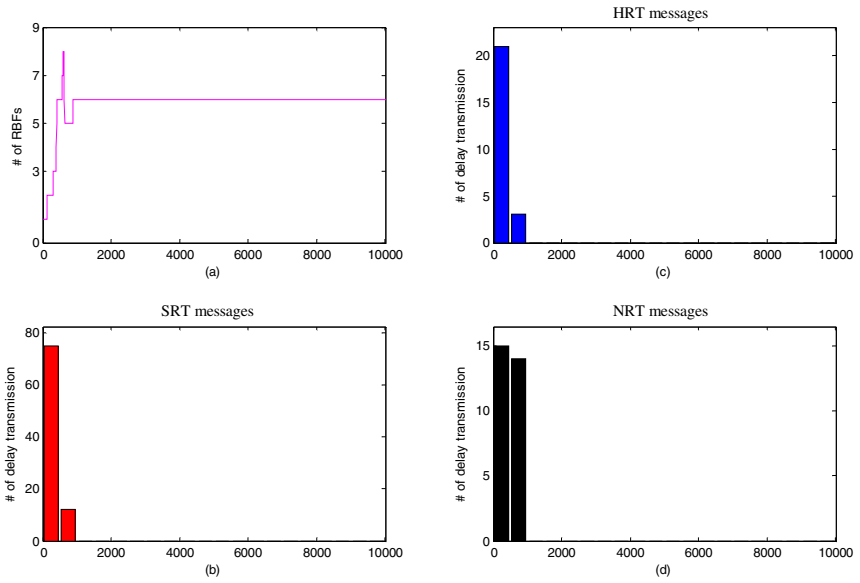


Fig. 4. Message scheduling on CAN bus using BTT and MRA algorithms. (a). The required number of hidden neurons and (b)(c)(d) the number of delayed-transmitted HRT, SRT, and NRT messages, respectively.

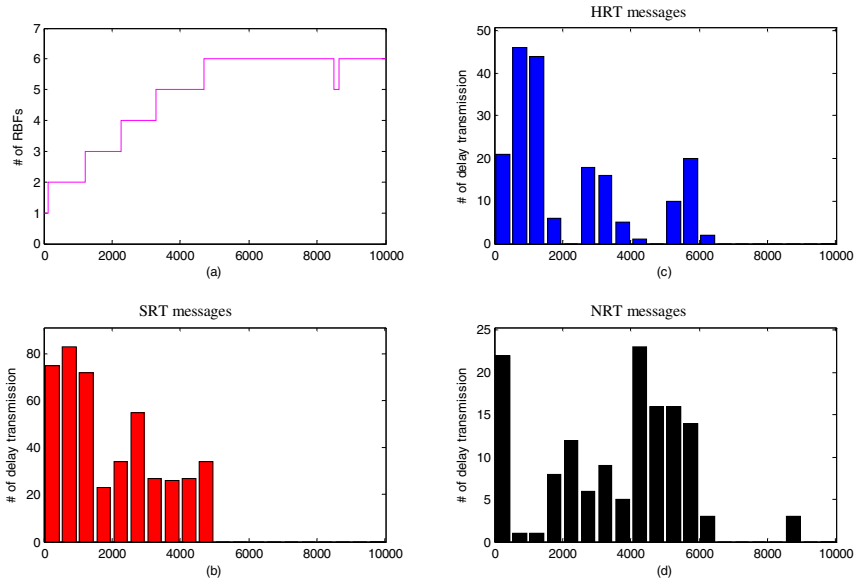


Fig. 5. Message scheduling on CAN bus using BTT and 10 for structure determination

In Fig 4, beginning of the simulation results bears the similarity as shown in the first experiment. The number of hidden neurons increases steadily. However, the CAN system becomes stable until 7,000 T for each type of messages. There is no surprise with these consequences since the ad-hoc method requires more parameter optimization epochs to stabilize the system. Therefore, the convergence rate is slow when compared with the first example.

5 Conclusions

In this paper, we present a complete framework for online adaptation of the RBF network that can be applied for dynamical modeling of time-varying nonlinear CAN systems. The proposed BTT methodology and MRA algorithm use combined structure and parameter adaptation so that changes in the dynamics of the given system can be tracked more accurately. The quality of the performance of the proposed methods is evident in the examples provided, which show how the algorithm, when it modifies the structure of the RBF network, takes the correct decisions of message scheduling. There is, nevertheless another important issue that must be taken into account for our future work: how to estimate the bandwidth in the given CAN system. This is a problem that, perhaps, may be tackled by estimating the available bandwidth using fuzzy inference system 11 in advance.

References

1. Lin, C., Yen, H., Chen, M., Hwang, C., Thanh, N.M.: A Neural Network Approach for Controller Area Network Message Scheduling Control. The International MultiConference of Engineers and Computer Scientists, Hong Kong, June 20-22, (2006) 36-41
2. Haykin, S.: Neural Networks: A Comprehensive Foundation. Macmillan College Publishing Company, New York (1994)
3. Farsi, M., Ratcliff, K., Barbosa, M.: An Overview of Controller Area Network. Computing & Control Engineering Journal **10**(3) (1999) 113-120
4. Bosch, R.: CAN Specification Version 2.0. Bosch, Sep. (1991)
5. Zuberi, K.M., Shin, K.G.: Non-preemptive Scheduling of Messages on Controller Area Network for Real-time Control Applications. Proc. of the first IEEE Real-time Technology and Applications Symposium (1995) 240-249
6. Tindell, K., Burns, A., Wellings, A.J.: Calculating Controller Area Network (CAN) Message Response Times. Contr. Eng. Practice **3**(8) (1995) 1163-1169
7. Livani, M., Kaiser, J.: EDF Consensus on CAN Bus Access for Dynamic Real-Time Application. In Proceedings of the 6th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS'98), Orlando, Florida, USA Mar. (1998)
8. Kaiser, J., Livani, M.A.: Invocation of Real-Time Objects in A CAN Bus-System. First Int'l Symposium on Object-Oriented Distributed Real-Time Computing Systems, Kyoto (1998)
9. Lu, Y., Sundararajan, N.: Performance Evaluation of A Sequential Minimal Radial Basis Function (RBF) Neural Network Learning Algorithm. IEEE Trans. Neural Networks **9**(2) (1998)
10. Yen, H.: Real Time Message Scheduling and Bandwidth Allocation on CAN Bus. Master thesis, Dept. of Electrical Engineering, Da-Yeh University June (2005)
11. Jang, J.S., Sun, C.T., Mizutani, E., Neuro-Fuzzy and Soft Computing, Prentice Hall, (1997)

Neural Network Based High Accuracy Frequency Harmonic Analysis in Power System

Xiaohua Wang^{1,2}, Yigang He², and Ying Long³

¹ Department of Electrical and Information Engineering,
Changsha University of Science & Technology, Changsha 410077, P.R. China

² Department of Electrical and Information Engineering,
Hunan University, Changsha 410082, P.R. China

³ Department of Electronic & Communication Engineering,
Changsha University, Changsha 410003, P.R. China
cslgwxh@163.com

Abstract. A back-propagation neural network method is proposed for accurate frequencies, amplitudes and phases estimation from periodic signals in power systems, and the convergence theorem shows that the proposed algorithm can be convergent asymptotically to its global minimum. The method is aimed at the system in which the sampling frequency cannot be locked on the actual fundamental frequency. Some simulating examples are given and the results show that the accuracy of the estimates provided by the proposed approach in the asynchronous case is relatively better than that of the estimates obtained with the conventional harmonic analysis methods.

1 Introduction

In recent years the harmonic pollution is becoming more and more serious with the widely use of nonlinear components in electric and electronic devices [1]. It has become a curse to power system, which deteriorates the quality of electric energy and greatly affects the safe and economical operation of electric power system. Therefore, it is necessary to measure the real-time content of harmonic and control the status of harmonic in electric network.

Presently, the spectral analysis of continuous periodic signals, such as voltages and currents in power networks, is based on the Fast Fourier transform (FFT). However, because of the grid effect and energy leakage of FFT, the calculated signal parameters, including frequencies, amplitudes and especially phases, are not precise and cannot meet the need of harmonic measurement. To improve the precision of FFT algorithm, the windowed interpolating algorithms [2]–[6] are proposed. The interpolating algorithms can eliminate the errors caused by grid effect, but the errors produced by leakage effect must be reduced by windowing the signals. Recent techniques employed are based on wavelet transform theory [7]–[9], which exploits time–frequency characterization of input signal to identify particular harmonics within sub-bands of interest. However, this technique requires a complex procedure.

The algorithm proposed in this paper is developed for high accuracy estimation of frequencies, amplitudes and phases of power system, under asynchronous sampling case. It is aimed at the system in which the sampling frequency cannot be locked on the actual fundamental frequency.

2 The Proposed Algorithm

For a sampling frequency of f_s , the following discrete-time signal model is considered

$$y(m) = \sum_{n=1}^N A_n \sin(\omega_n m T_s + \theta_n), \quad m = 1, 2, \dots, M \tag{1}$$

where, A_n is the amplitude, θ_n the phase, ω_n the angular frequency of the n th-harmonic, and T_s sampling period. In power system, equation (1) can be expressed as

$$\begin{aligned} y(m) &= \sum_{n=1}^N \{A_n \sin \theta_n \cos(n\omega_0 m T_s) + A_n \cos \theta_n \sin(n\omega_0 m T_s)\} \\ &= \sum_{n=1}^N \{a_n \cos(n\omega_0 m T_s) + b_n \sin(n\omega_0 m T_s)\} \end{aligned} \tag{2}$$

where $a_n = A_n \sin \theta_n$, $b_n = A_n \cos \theta_n$, and ω_0 is the fundamental angle frequency. The matrix expression of (2) can be written as

$$\mathbf{y}(\mathbf{x}) = \mathbf{a}^T \mathbf{C} + \mathbf{b}^T \mathbf{S} \tag{3}$$

where

$$\mathbf{a} = [a_1, a_2, \dots, a_N]^T \tag{4}$$

$$\mathbf{b} = [b_1, b_2, \dots, b_N]^T \tag{5}$$

$$\mathbf{x} = \begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} \tag{6}$$

$$\mathbf{C} = \begin{bmatrix} \cos[\omega_0 T_s] & \cos[2\omega_0 T_s] & \dots & \cos[N\omega_0 T_s] \\ \cos[2\omega_0 T_s] & \cos[4\omega_0 T_s] & \dots & \cos[2N\omega_0 T_s] \\ \vdots & \vdots & \ddots & \vdots \\ \cos[N\omega_0 T_s] & \cos[2N\omega_0 T_s] & \dots & \cos[N^2\omega_0 T_s] \end{bmatrix} \tag{7}$$

$$\mathbf{S} = \begin{bmatrix} \sin[\omega_0 T_s] & \sin[2\omega_0 T_s] & \dots & \sin[N\omega_0 T_s] \\ \sin[2\omega_0 T_s] & \sin[4\omega_0 T_s] & \dots & \sin[2N\omega_0 T_s] \\ \vdots & \vdots & \ddots & \vdots \\ \sin[N\omega_0 T_s] & \sin[2N\omega_0 T_s] & \dots & \sin[N^2\omega_0 T_s] \end{bmatrix}. \tag{8}$$

Now we define error function as

$$\mathbf{e} = \mathbf{z} - \mathbf{y} \tag{9}$$

where \mathbf{z} is the actual distorted vector, and \mathbf{y} is the output vector of the neural network. The performance criterion uses the mean-squared error function defined as

$$J(\mathbf{e}) = \frac{1}{M} \sum_{m=1}^M e^2(m). \tag{10}$$

To solve the design problem of minimizing J , we take a back-propagation neural network of size $2N$, whose neuron states are $x(n)$ ($n=1,2,\dots,2N$), then the steepest descent algorithm for the approximate square error is

$$\Delta \mathbf{x}_k = -\eta \frac{\partial J}{\partial \mathbf{x}_k} = \eta \frac{2}{M} \begin{bmatrix} \mathbf{C} \\ \mathbf{S} \end{bmatrix} \mathbf{e}_k^T \tag{11}$$

where $\eta > 0$ is learning rate.

In order to ensure convergence of the neural network, it is important to select proper learning rate η . Assume that the fundamental frequency of power system is known, we have the following theorem.

Theorem 1. If learning rate satisfies $\eta < M / \|\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}\|_2$, the neural networks algorithm is stable, and convergent asymptotically to its global minimum.

Proof. Define (10) as a Lyapunov function. Since

$$\Delta \mathbf{e}_k = (\Delta \mathbf{x}_k)^T \frac{\partial \mathbf{e}_k}{\partial \mathbf{x}_k} = -\eta \frac{2}{M} \mathbf{e}_k (\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}) \tag{12}$$

then

$$\begin{aligned} \Delta J(\mathbf{e}_k) &= \frac{1}{M} \left\| \mathbf{e}_k - \eta \frac{2}{M} \mathbf{e}_k (\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}) \right\|_2^2 - \frac{1}{M} \|\mathbf{e}_k\|_2^2 \\ &\leq \frac{1}{M} \left\| \mathbf{I} - \eta \frac{2}{M} (\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}) \right\|_2^2 \|\mathbf{e}_k\|_2^2 - \frac{1}{M} \|\mathbf{e}_k\|_2^2 \\ &= \frac{1}{M} \|\mathbf{e}_k\|_2^2 \left(1 - \eta \frac{4}{M} \|\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}\|_2 + \eta^2 \frac{4}{M^2} \|\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}\|_2^2 \right) - \frac{1}{M} \|\mathbf{e}_k\|_2^2 \\ &= \frac{4}{M^2} \eta \|\mathbf{e}_k\|_2^2 \|\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}\|_2 \left(\eta \frac{1}{M} \|\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}\|_2 - 1 \right) \end{aligned} \tag{13}$$

It is easy to see from (13) that if

$$\eta < M / \|\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}\|_2 \tag{14}$$

then $\Delta J(\mathbf{e}_k) \leq 0$. When $\Delta J(\mathbf{e}_k) = 0$, we have

$$\frac{4}{M^2} \eta \|\mathbf{e}_k\|_2^2 \|\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}\|_2 = 0. \tag{15}$$

Since

$$\|\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}\|_2 > 0 \tag{16}$$

thus

$$\mathbf{e}_k = 0 \tag{17}$$

$$\Delta \mathbf{x}_k = 0 \tag{18}$$

$$J(\mathbf{e}_k) = 0. \tag{19}$$

Therefore, if $\eta < 4 / \|\mathbf{C}^T \mathbf{C} + \mathbf{S}^T \mathbf{S}\|_2$, the neural network is stable, and the neural is convergent asymptotically to its global minimum. The theorem is proved completely.

In the above neural network algorithm, only weighted coefficient vector \mathbf{x} is updated, this means that if the sampling frequency can be locked on the actual fundamental frequency, then accurate harmonic amplitude and phase estimation can be obtained when the neural network is convergent. However, in some applications, it is not possible to make the sampling rate synchronizing to the actual fundamental frequency. So it is necessary to make the parameter ω_0 approximate the actual fundamental angle frequency. Now we define

$$\mathbf{D} = \begin{bmatrix} 1 & 2 & \dots & M \\ 2 & 4 & \dots & 2M \\ \vdots & \vdots & \vdots & \vdots \\ N & 2N & \dots & NM \end{bmatrix} \tag{20}$$

then

$$\begin{aligned} \omega_0(k+1) &= \omega_0(k) - \beta \frac{\partial J}{\partial \omega_0(k)} \\ &= \omega_0(k) + \beta T_s \frac{2}{M} \mathbf{e}_k \{ \mathbf{D}^T .* \mathbf{C}^T * \mathbf{b}_k - \mathbf{D}^T .* \mathbf{S}^T * \mathbf{a}_k \} \end{aligned} \tag{21}$$

where β , $\beta > 0$, is the learning rate to ω_0 , and “.*” is matrix element group multiplication operation in MATLAB, i.e., $\mathbf{D}.*\mathbf{C} = [D_{ij}C_{ij}]_{N \times M}$.

Obviously, it can be seen from above neural network algorithm that accurate harmonic frequency, amplitude and phase estimation can be obtained by training the neural network.

3 Algorithm Implementation

Due to the illustration purpose we designated for the present paper, some design guidelines listed below are used to simplify the numerical simulation procedure.

- 1) Initial value: Produce a random initial weight vector \mathbf{x} , let initial fundamental frequency $f_0 = 50$ Hz, and specify sampling frequency f_s , sample data length M , error bound ε , and suitable learning rate η and β .
- 2) Produce new predicted output \mathbf{y} of the neural network using (3).
- 3) Calculate \mathbf{e} and $J(\mathbf{e})$ via (9) and (10).
- 4) Update the weighted coefficient vector \mathbf{x} according to (11), and the parameter ω_0 to (21).
- 5) If $J(\mathbf{e}) > \varepsilon$, go to step 2), otherwise, close the training of the neural network.

4 Simulation Results

In this section, we present some simulation results in order to evaluate the performance of the neural network algorithm. Considering the signal in [6], signal with four harmonics are generated following the model in (1). The magnitudes are defined as $A_1=1$, $A_2=1/8$, $A_3=1/4$, $A_4=1/16$; and phases as $\theta_1=\pi/8$, $\theta_2=\pi/4$, $\theta_3=3\pi/8$, $\theta_4=\pi/2$. The fundamental frequency can take as values in the continuous interval defined by the $\pm 1\%$ deviations tolerated in the power system. The signal is sampled as frequency $f_s=1000$ Hz and the length of data is 25 sampling points. Specify $\varepsilon = 10^{-26}$, $\eta = 0.9$.

Now we study the sensitivity of fundamental frequency estimation with different values 49.5, 50, and 50.5 Hz of actual fundamental frequency depending on the different learning rate β . Fig. 1 illustrates the sensitivity of estimation. The case with $\beta = 2800$ is used to test the proposed algorithm in this section. In this case, the estimation errors are the least.

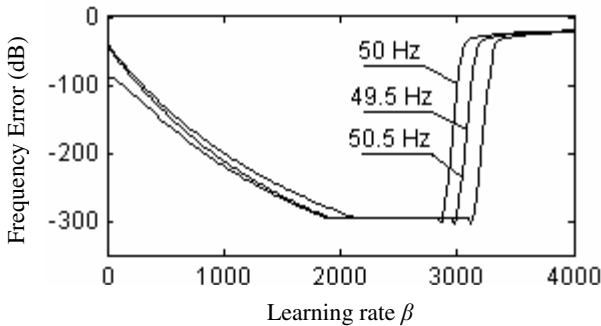


Fig. 1. Frequency estimation sensitivity depending on the value of learning rate β

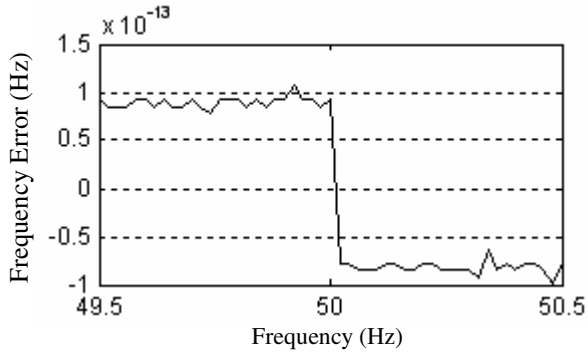


Fig. 2. Frequency estimation error

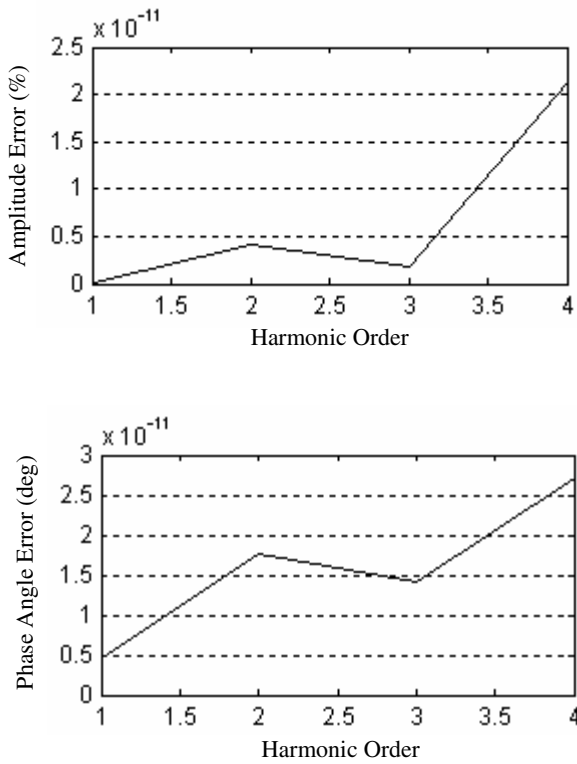


Fig. 3. Amplitude and phase errors by 376 times training and 0.078 seconds estimation time, at $f = f_0$

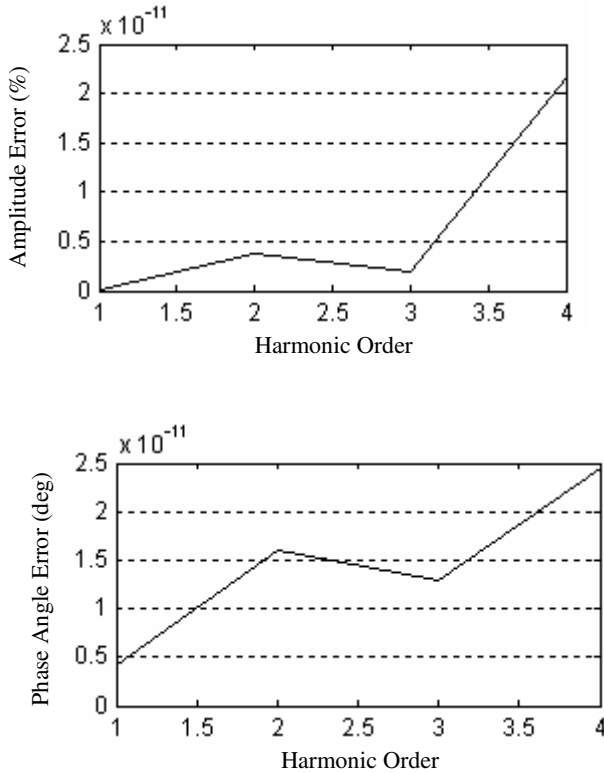


Fig. 4. Amplitude and phase errors by 377 times training and 0.078 seconds estimation time, at $f = 0.99f_0$

Fig. 2 illustrates the fundamental frequency estimation error, and Figs. 3–5 show the relative-error percentage on amplitude and the absolute error on phase (in degrees) of the harmonic estimations obtained from the test signal using the proposed approach. Fig. 3 shows the synchronous sampling case, Fig. 4 and 5 show the extreme asynchronous sampling cases. In all cases, the proposed method provides similar excellent results, with magnitude and phase errors being less than 10^{-10} , and fundamental frequency errors less than 10^{-12} . Comparing with the results in [6], the standard raised-cosine (SRC), generalized raise-cosine (GRC), and Hanning windows are used to estimate the harmonics of this test signal. In all cases, the SRC pulse provides better results: in the synchronous sampling case, it obtains excellent results with amplitude and phase errors being less than 10^{-13} , while in the extreme asynchronous sampling cases, the approach provides errors ranging between one thousandth and one hundredth of a percentage point. This means that the proposed approach is better than the method in [6].

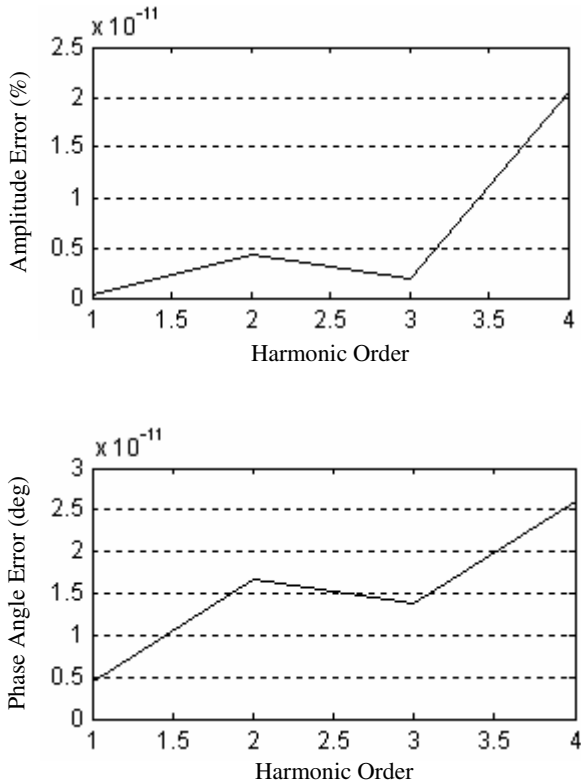


Fig. 5. Amplitude and phase errors by 399 times training and 0.079 seconds estimation time, at $f = 1.01f_0$

5 Conclusion

In the extreme asynchronous sampling cases, the accuracy of the harmonic estimations obtained with the proposed neural network algorithm is relatively better than that of the estimations provided by the shaping pulses without amplitude compensation and the Hanning window with amplitude compensation (in [6]). The results of simulation analysis with the help of MATLAB software packages confirm the validity of the proposed method for harmonic estimations at high accuracy of 99.99999999%.

Acknowledgements

This work was supported by the National Natural Foundation of China under grant No. 50677014, the Doctoral Special Fund of Education Ministry of China under grant No. 20060532006, the Program for New Century Excellent Talents in University of

China (NCET-04-0767), and the Foundation of Hunan Provincial Science and Technology of China under grant No. 06JJ2024, 03GKY3115, 04FJ2003, and 05GK2005.

References

1. Davis, E.J., Emanuel, A.E.: Harmonic Pollution Metering: Theoretical Considerations. *IEEE Trans. Power Del* **15** (Jan. 2000) 19-23
2. offelli, C., Petri, D.: The Influence of Windowing on the Accuracy of Multifrequency Signal Parameter Estimation. *IEEE Trans. Instrum. Meas* **41** (Apr. 1992) 256-261
3. Testa, A., Gallo, D., Langella, R.: On the Processing of Harmonics and Interharmonics: Using Hanning Window in Standard Framework. *IEEE Trans. Power Del* **19** (Jan. 2004) 28-34
4. Barros, J., Diego, R.I.: On the Use of the Hanning Window for Harmonic Analysis in the Standard Framework. *IEEE Trans. Power Del* **21** (Jan. 2006) 538-539
5. Zhang, F., Geng, Z. Yuan, W.: The Algorithm of Interpolating Windowed FFT for Harmonic Analysis of Electric Power System. *IEEE Trans. Power Del* **16** (Apr. 2001) 160-164
6. Serna, J.A.: On the Use of Amplitude Shaping Pulses as Windows for Harmonic Analysis. *IEEE Trans. Instrum. Meas* **50** (Dec. 2001) 1556-1562
7. Kuang, W.T., Morris, A.S.: Using Short-Time Fourier Transform and Wavelet Packet Filter Banks for Improve Frequency Measurement in A Doppler Robot Tracking System. *IEEE Trans. Instrum. Meas* **51** (June 2002) 440-445
8. Pham, V.L., Wong, K.P.: Antidistortion Method for Wavelet Transform Filter Banks and Nonstationary Power System Waveform Harmonic Analysis. *IEE Proc. Gener. Transm. Distrib* **148** (Mar. 2001) 117-122
9. Pham, V.L., Wong, K.P.: Wavelet-Transform-Based Algorithm for Harmonic Analysis of Power System Waveforms. *IEE Proc. Gener. Transm. Distrib* **146** (May 1999) 249-254

Hardware/Software Partitioning of Core-Based Systems Using Pulse Coupled Neural Networks

Zhengwei Chang and Guangze Xiong

School of Computer Science and Engineering,
University of Electronic Science and Technology of China,
Chengdu 610054, P.R. China
changzw@ustc.edu, gzxiong@uestc.edu.cn

Abstract. Hardware/software partitioning of System-on-chip (SoC partitioning) has a significant effect on the cost and performance of the SoC. Given an embedded system specification and an available core library, the goal of low power SoC partitioning is to select appropriate intellectual-property (IP) cores or software components for the SoC, such that the power consumption of the SoC is minimized under price and timing constraints. SoC partitioning is first formulated to the constrained single-pair shortest-path problem in a directed, weighted graph, and then a novel discrete pulse coupled neural network (PCNN) approach is proposed to get the optimal solution. Autowaves in PCNN are designed specially to meet the constraints and find the optimal path in the constructed graph. Experimental results are given to demonstrate the feasibility and effectiveness of the proposed method.

1 Introduction

Technology development has made it possible to implement an entire system on a single die, a so-called System-on-chip (SoC). SoC development is based on the design reuse philosophy, using pre-designed, pre-verified cores known as intellectual-property (IP) cores in hardware and components in software. Different from traditional method that designs hardware and software independently, hardware/software co-design has evolved as a new style of SoC design [1]. Hardware/software partitioning of SoC (SoC partitioning) is a crucial step in it. Starting from system specifications, SoC partitioning determines the corresponding hardware and software portions of a SoC, thus it has a significant effect on the cost and performance of the SoC. In [2-4], the hardware/software partitioning problem was addressed by using heuristic techniques, such as genetic algorithm, simulated annealing, taboo search, and Hopfield neural networks. But most of them did not target core-based SoCs.

Pulse coupled neural networks (PCNN) is different from traditional artificial neuron networks [5, 6]. PCNN models have biological background. Lately, PCNN is applied in many fields [7], such as image processing, pattern recognition, and optimization. Since the original work of Caulfield and Kinser [8] in 1999, there are several researches [9-11] using autowaves in PCNNs to solve the combinatorial optimization problems, especially the shortest-path problems.

In this paper, we first formulate SoC partitioning to the constrained single-pair shortest-path problem of a directed, weighted graph, and then propose a novel PCNN approach to get the optimal solution. Autowaves in PCNN are designed specially to meet the constraints and find the optimal path in the constructed graph. Different from traditional heuristic techniques, all the solutions found by this algorithm are globally optimal solutions.

This paper is organized as follows. In Section 2, the description and formulation of SoC partitioning is presented. The model of a novel PCNN and its application to find the constrained shortest-path are described in Section 3. Experimental results from a practical SoC are given in Section 4 followed by conclusions in Section 5.

2 Formulation of SoC Partitioning

Most embedded systems are designed today as core-based SoCs to produce high performance and shorten time-to-market. Resorting to hardware/software partitioning, the design process must determine a SoC that achieves low power consumption, low cost, and high performance, from the design space defined by the set of cores and the set of tasks that need to execute on them.

2.1 Definition of SoC Model

Before introducing the system representation, we list the assumptions made for this work.

1. The SoC is specified in terms of a task graph. Nodes of the task graph represent tasks, and edges represent the data or control dependencies between the tasks. Each task may be performed on an IP core in hardware or a component in software.
2. A library including IP cores and software components is built up. For each task of the SoC, a wide range of cores is available in the library for implementing it.
3. Each core within the library has its specific performance measures, such as price, runtime and power consumption.
4. The total price of a SoC is equal to the sum of the prices of all the cores used by the SoC. For runtime and power consumption, it is the same case.

Given the task graph and the core library of a SoC, the main objective of low power SoC partitioning is to optimally allocate the task nodes to the IP cores and software components under price and timing constraints, such that the power consumption of the SoC is minimized.

Formally, we define $T = \{t_1, t_2, \dots, t_n\}$ where t_i represents a task of the SoC, and $CL = \{Core_1, Core_2, \dots, Core_n\}$, in which $Core_i = \{Core_{i1}, Core_{i2}, \dots, Core_{im}\}$ is a set of candidates for task t_i , $m = |Core_i|$ is the size of $Core_i$. For $Core_i$ and $Core_j$ where $i \neq j$, m may have different value. $Core_{ij}$ indicates a core model, and is defined as $Core_{ij} = (P_{ij}, C_{ij}, T_{ij})$, in which P_{ij} is the power consumption of $Core_{ij}$ to perform t_i , C_{ij} is the price of $Core_{ij}$, and T_{ij} is the runtime of $Core_{ij}$ to perform t_i . We define C_{max} and T_{max} as the maximum price and runtime the SoC must not exceed respectively.

SoC partitioning is defined as the following constrained optimization problem of finding a solution $S = (Core_{1i}, Core_{2j}, \dots, Core_{nk})$:

$$\min f_P(S) = (P_{1i} + P_{2j} + \dots + P_{nk}), \tag{1}$$

s.t.

$$g_C(S) = (C_{1i} + C_{2j} + \dots + C_{nk}) \leq C_{max}, \tag{2}$$

$$g_T(S) = (T_{1i} + T_{2j} + \dots + T_{nk}) \leq T_{max}, \tag{3}$$

where $Core_{1i} \in Core_1, Core_{2j} \in Core_2, \dots, Core_{nk} \in Core_n$.

2.2 Formulating SoC Partitioning to the Constrained Single-Pair Shortest-Path Problem

Based on the definitions in Section 2.1, we can convert SoC partitioning to the constrained single-pair shortest-path (CSPSP) problem in a directed, weighted graph, as shown in Fig. 1.

1. A vertex $Core_{ij}$ (i representing the task index and j representing the alternative core index) is connected to all vertices in the next column (i.e., $Core_{i+1,k}$) with directed, weighted edges. The weight of the edge connecting vertices $Core_{ij}$ and $Core_{i+1,k}$ is $P_{i+1,k}$.
2. In addition, two special nodes, the source $Start$ and the destination End are added. Node $Start$ connects the vertices $Core_{1i}$ ($1 \leq i \leq a, a = |Core_1|$) with a weight P_{1i} . The vertices $Core_{ni}$ ($1 \leq i \leq c, c = |Core_n|$) connect to node End with a weight $P_{End} > 0$.

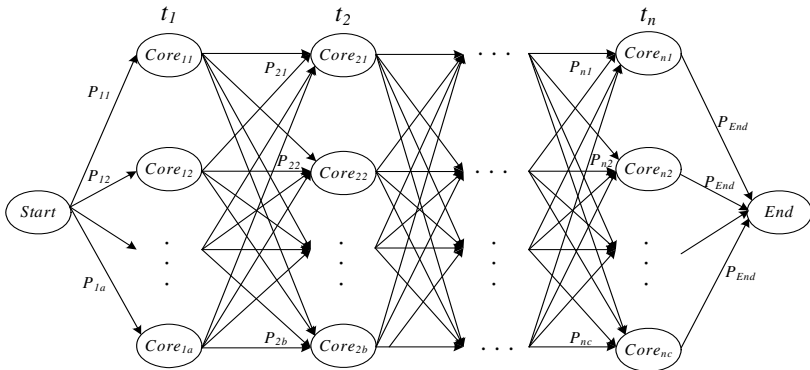


Fig. 1. A directed, weighted graph constructed from SoC partitioning

Accordingly, SoC partitioning is converted to the problem of finding a shortest-path $p_c = (Start, Core_{1i}, Core_{2j}, \dots, Core_{nk}, End)$ from $Start$ to End , subject to two additive constraints as Eqs. (2) - (3). The power consumption of the SoC can be calculated by

$$f_P(S) = W(p_c) - P_{End}, \quad W(p_c) \text{ is the weight of path } p_c. \tag{4}$$

3 A PCNN Approach

In [9-11], the authors utilize autowaves in PCNNs to find the shortest-paths in a non-deterministic and parallel way, but there is no constraints on the paths. For the CSPSP problem, we use the discrete-time PCNN (DPCNN) network based on Output-threshold coupled neural network [10].

3.1 The DPCNN Model

Given a weighted graph G with N nodes, W_{IJ} denoting the weight of the edge that connects node I and J , we will introduce the designation of a DPCNN model to find an unconstrained shortest-path p between the source nodes S and the destination node D in graph G .

Each neuron in the network corresponds to a node in the graph. In addition, each directed connection between neurons in the network corresponds to the directed edge between the nodes in the graph, i.e. the network itself is isomorphic to the graph. Each neuron I has an output Y_I , and Y_I is designed as the following function:

$$Y_I(k) = \text{step}(U_I(k) - \theta_I(k)) = \begin{cases} 1, & \text{if } U_I(k) > \theta_I(k), \\ 0, & \text{otherwise,} \end{cases} \quad I = 1, 2, \dots, N, \tag{5}$$

where $U_I(k)$ and $\theta_I(k)$ are the internal activity and threshold at time k , respectively. $U_I(k)$ is always set to 0 at any time, i.e. $U_I(k) = 0$ for $k \geq 0, I = 1, 2, \dots, N$. All of the neurons are set reset in initial time instant, i.e. $\theta_I(0) = 0$, for $I = 1, 2, \dots, N$.

The threshold of neuron I is designed as Eqs. (6) - (8). It is clear that the threshold of neuron I is decrease linearly respected to a constant value Δt .

$$\theta_{JI}(k) = \begin{cases} V_\theta, & \text{if } Y_J(k-1) = 0 \text{ and } Y_J(k) = 1, \\ \theta_{JI}(k-1) - \frac{V_\theta}{W_{JI}} \Delta t, & \text{if } Y_J(k-1) = 1 \text{ and } Y_J(k) = 1, \\ 0, & \text{if } Y_J(k) = 0, \end{cases} \quad \text{for } W_{JI} \neq 0, \tag{6}$$

$$\theta_I(k) = \min\{\theta_{JI}(k) : W_{JI} \neq 0\}, \tag{7}$$

$$Y_I(k) = \begin{cases} 1, & \text{if } Y_I(k-1) = 1 \\ \text{step}(-\theta_I(k)), & \text{otherwise} \end{cases} \tag{8}$$

where V_θ is a predetermined positive constant and its value is problem independent, W_{JI} is the linking strength from neuron J to neuron I . If there have no linking connection between neuron J and neuron I , then $W_{JI} = 0$. θ_{JI} is the threshold contribution of neuron J to neuron I .

Neuron I is said to fire if its output changes from 0 to 1. It is known from Eqs. (6)-(8) that when the autowave travels along a connection between neuron J and neuron I , the time periods from J fires to I fires is proportional to the link strength W_{JI} . When m neurons J_1, J_2, \dots , and J_m are linked to neuron I , and at least one of these neurons fires before neuron I , the autowave that arrives at neuron I at the earliest time is allowed to continue traveling.

The algorithm of DPCNN for finding the unconstrained shortest-path p in graph G is expressed in the following steps.

- (1) Initialize the network, set iterate number $k = 0$, $Y_I(k) = 0$ and $\theta_I(k) = 0$, ($I = 1, 2, \dots, N$).
- (2) Set $k = k + 1$, and $\theta_S(k) = -1$, then the neuron S fires first, as shown in

$$\begin{cases} Y_S(k) = 1, \\ Y_I(k) = 0, \end{cases} \quad (I = 1, 2, \dots, N \text{ and } I \neq S),$$
 autowaves are generated and neuron S is the source of the autowaves.
- (3) Calculate $\theta_I(k)$ and $Y_I(k)$ according to Eqs. (6) - (8), ($I = 1, 2, \dots, N$). For $W_{JI} \neq 0$, if $Y_I(k) == 1$ and $Y_I(k-1) == 0$ and $\theta_I(k) == \theta_{JI}(k)$, then update the route.
- (4) Set $k = k + 1$, repeat step 3 until neuron D in the network is in fired state. Go reverse the route table, path p and the weight of the shortest-path from S to D : $W(p)$ is determined.

3.2 The Algorithm for Solving the CSPSP Problem Using DPCNN

While DPCNN is applied to the CSPSP problem shown in Fig. 1, all of the autowaves in the network travel toward the same direction, i.e. they have the same sequence (*Start, Core_{1i}, Core_{2j}, ..., Core_{nk}, End*) of fired neurons. But the constraints require to be considered while the autowave is traveling ahead to the destination node *End*. As a result, we must check whether each autowave meets the constraints at every iteration, so the satisfied autowaves may continue to travel. On the other hand, the ones that cannot satisfy the constraints will be deleted, guaranteeing that other links with a larger weight or loose constraints have the opportunity to be chosen.

We illustrate how the autowaves are designed to meet the constraints. Consider the case when an autowave has neurons *Start, Core_{1i}, Core_{2j}, ..., and Core_{uk}* fired, and I represents the neuron fired latest among them. Before this autowave explores all the neurons which neuron I connects, it is examined whether the constraints can be satisfied if them fires next. Among these neurons, the neuron J not satisfying the constraints would not allow to be fired by this autowave, thus we set $\theta_{JI}(k) = +\infty$. When no one of these neurons is allowed to fire next, neuron I would return to unfired state, thus its output Y_I is reset to 0. If I is neuron *Start*, we can draw a conclusion that there is no valid solution for this problem. Otherwise, assuming that neuron H fired previous to neuron I , we set $\theta_{HI}(k) = +\infty$. As a result, neuron I may fire from other neurons.

The algorithm of DPCNN for the CSPSP problem is described below.

// k : the current iterate;

// W_{IJ} : the linking strength from neuron I to neuron J ;

// $Core_{0l}$: neuron $Start$, set $C_{0l} = 0$, $T_{0l} = 0$, and $Core_0 = \{Core_{0l}\}$;

// $Core_{n+1,l}$: neuron End , set $C_{n+1,l} = 0$, $T_{n+1,l} = 0$, and $Core_{n+1} = \{Core_{n+1,l}\}$;

//Price[I]: the sum prices of the neurons along the autowave traveling from neuron $Start$ to neuron I ;

//Time[I]: the sum runtime of the neurons along the autowave traveling from neuron $Start$ to neuron I ;

//PreNode[I]: an array to record the routes, PreNode[I] = H , if neuron H connects neuron I , and the autowaves travel from H to I ;

- (1) Initialize the network, set $k = 0$, and for each neuron $I = Core_{ij}$, ($0 \leq i \leq n+1, 1 \leq j \leq |Core_i|$), in the network, $Y_I(k) = 0$, $\theta_I(k) = 0$, Price[I] = 0, Time[I] = 0, PreNode[I] = $Start$.
- (2) Set $k = k+1$, and $\theta_{Start}(k) = -1$.
- (3) For each neuron $I = Core_{ij}$ in the network, ($0 \leq i \leq n+1, 1 \leq j \leq |Core_i|$)
 - (3.1) Calculate $\theta_I(k)$ and $Y_I(k)$ according to Eqs. (6) - (8).
 - (3.2) If $Y_I(k) == 1$ and $Y_I(k-1) == 0$ and $I \neq End$
 - (3.2.1) For each neuron H in the previous column, (i.e. $W_{HI} \neq 0$)
If $\theta_I(k) == \theta_H(k)$, then set PreNode[I] = H .
 - (3.2.2) Set Price[I] = Price[PreNode[I]] + C_{ij} , and Time[I] = Time[PreNode[I]] + T_{ij} .
 - (3.2.3) For each neuron $J = Core_{i+1,r}$ in the next column (i.e. $W_{IJ} \neq 0$)
If Price[I] + $C_{i+1,r} > C_{max}$ or Time[I] + $T_{i+1,r} > T_{max}$, then set $\theta_{IJ}(k) = +\infty$.
 - (3.2.4) If $\min\{\theta_{IJ}(k): J = Core_{i+1,r}, \forall J \in Core_{i+1}\} == +\infty$,
 - (3.2.4.1) Return neuron I to unfired state, set $Y_I(k) = 0$.
 - (3.2.4.2) If $I == Start$, then this algorithm ends, and there is no valid solution.
 - (3.2.4.3) Set $H = PreNode[I]$, and $\theta_H(k) = +\infty$.
- (4) Set $k = k+1$, repeat step 3, until $Y_{End}(k) == 1$ and $Y_{End}(k-1) == 0$.
- (5) Go reverse the route according to PreNode[I], then the constrained optimal path p_c and its weight $W(p_c)$ is determined. $g_C(S)$ and $g_T(S)$ is record in Price[End], and Time[End], respectively.

4 Experimental Results

To verify the feasibility and effectiveness of our method for real applications, we apply this algorithm to a SoC that is designed as video/audio sender subsystem of a PDA platform. The task graph of this SoC is shown in Fig. 2, and Table 1 shows the core library.

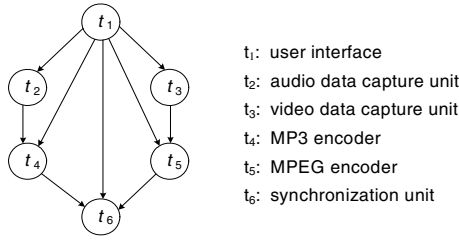


Fig. 2. Task graph of the example SoC of a PDA platform

A directed, weighted graph is constructed as described in Section 2.2, shown in Fig. 3. The weight of all the edges connecting node *End* is: $P_{End} = 0.005$. In our experiments, we set $V_0 = 5$, and $\Delta t = 0.005$. DPCNN is applied to find the constrained optimal path from *Start* to *End* in this graph, and then we get the optimal solutions for the SoC under different constraints. The experimental results are shown in Table 2, and the entry with *N/A* indicates that there is no valid solution satisfying such constraints.

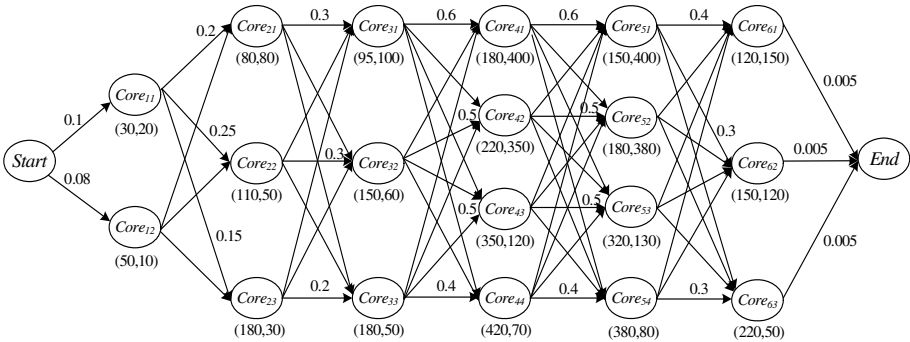


Fig. 3. The target graph constructed from the example SoC

When $C_{max} = 1500 \text{ } \text{Y}$, and $T_{max} = 500 \text{ ms}$, the solution found by DPCNN is $S = (Core_{12}, Core_{23}, Core_{33}, Core_{44}, Core_{54}, Core_{63})$, and its optimized power consumption is $f_P(S) = 1.53 \text{ W}$. As a result, task t_1 is implemented as components in software, and the other 5 tasks are implemented as IP in hardware.

When $C_{max} = 1000 \text{ } \text{Y}$, and $T_{max} = 1000 \text{ ms}$, a different solution is found. Tasks t_2 and t_3 are implemented in hardware, and the other 4 tasks are implemented in software.

The experimental results demonstrated that the solutions found by DPCNN are globally optimal solutions, since they are same as solutions found by exhaustive search.

Table 1. The core library for the example SoC of a PDA platform

	Core ID	Type	Power/W	Price/ ¥	Time/ms
<i>Core₁</i>	<i>Core₁₁</i>	Component	0.1	30	20
	<i>Core₁₂</i>	Component	0.08	50	10
<i>Core₂</i>	<i>Core₂₁</i>	Component	0.2	80	80
	<i>Core₂₂</i>	IP	0.25	110	50
	<i>Core₂₃</i>	IP	0.15	180	30
<i>Core₃</i>	<i>Core₃₁</i>	Component	0.3	95	100
	<i>Core₃₂</i>	IP	0.3	150	60
	<i>Core₃₃</i>	IP	0.2	180	50
<i>Core₄</i>	<i>Core₄₁</i>	Component	0.6	180	400
	<i>Core₄₂</i>	Component	0.5	220	350
	<i>Core₄₃</i>	IP	0.5	350	120
	<i>Core₄₄</i>	IP	0.4	420	70
<i>Core₅</i>	<i>Core₅₁</i>	Component	0.6	150	400
	<i>Core₅₂</i>	Component	0.5	180	380
	<i>Core₅₃</i>	IP	0.5	320	130
	<i>Core₅₄</i>	IP	0.4	380	80
<i>Core₆</i>	<i>Core₆₁</i>	Component	0.4	120	150
	<i>Core₆₂</i>	Component	0.3	150	120
	<i>Core₆₃</i>	IP	0.3	220	50

Table 2. The solutions for the example SoC found by DPCNNs

Constraints		Results			
C_{max}	T_{max}	Solution S	$f_P(S)$	$g_C(S)$	$g_T(S)$
1500	500	$(Core_{12}, Core_{23}, Core_{33}, Core_{44}, Core_{54}, Core_{63})$	1.53	1430	290
1000	1000	$(Core_{12}, Core_{23}, Core_{33}, Core_{42}, Core_{52}, Core_{62})$	1.73	960	940
500	1500	N/A	N/A	N/A	N/A

5 Conclusions

In this paper, we have addressed the problem of low power hardware/software partitioning of core-based SoCs subject to price and timing constraints. Based on a system specification and an available core library, SoC partitioning is formulated to the constrained single-pair shortest-path problem in a directed, weighted graph. Autowaves in PCNN are designed specially to satisfied the price and timing constraints, and find the optimal path in the constructed graph. This is a nondeterministic approach that would guarantee the globally optimal solutions. It has been demonstrated that PCNN is feasible and effective to solve the SoC partitioning problems. Due to the highly parallel computation of the network, it would be much faster to find the solution if the network is realized with VLSI.

References

1. Wolf, W.: A Decade of Hardware/software Codesign. *IEEE Computer* **36** (4) (2003) 38-43
2. Saha, D., Mitra, R. S., Basu, A.: Hardware Software Partitioning using Genetic Algorithm. *Proc. of 10th International Conference on VLSI Design* (1997) 155-160
3. 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
4. Guo, B., Wang, D., Shen, Y., Liu, Z.: Hardware-Software Partitioning of Real-Time Operating Systems using Hopfield Neural Networks. *Neurocomputing* **69** (16-18) (2006) 2379-2384
5. Johnson, J.L., Ritter, D.: Observation of Periodic Waves in a Pulse-Coupled Neural Network. *Opt. Lett.* **18** (15) (1993) 1253-1255
6. Johnson, J.L.: Pulse-Coupled Neural Nets: Translation, Rotation, Scale, Distortion, and Intensity Signal Variance for Images. *Appl. Opt.* **33** (26) (1994) 6239-6253
7. Johnson, J.L., Padgett, M.L.: PCNN models and Applications. *IEEE Transactions on Neural Networks* **10** (3) (1999) 480-498
8. Caulfield, H.J., Kinser, J.M.: Finding the Shortest Path in the Shortest Time using PCNN's. *IEEE Transactions on Neural Networks* **10** (3) (1999) 604-606
9. Gu, X., Zhang, L., Yu, D.: Delay PCNN and Its Application for Optimization. In: Yin, F., Wang, J., Guo, C. (eds.): *ISNN 2004. Lecture Notes in Computer Science*, **3173** Springer-Verlag, Berlin Heidelberg (2004) 413-418
10. Zhang, J., Wang, D., Shi, M.: Output-Threshold Coupled Neural Network for Solving the Shortest Path Problems. *Science in China, Ser. F* **47**(1) (2004) 20-33
11. Qu, H., Zhang, Y.: A New Algorithm for Finding the Shortest Paths using PCNNs. *Chaos, Solitons & Fractals*, Accepted

Technical and Fundamental Analysis for the Forecast of Financial Scrip Quotation: An Approach Employing Artificial Neural Networks and Wavelet Transform

Anderson da Silva Soares¹, Maria Stela Veludo de Paiva¹,
and Clarimar José Coelho²

¹ São Carlos Engineering School, Department of Electric Engineer - University of São Paulo - Av. Trabalhador São Carlense 4000, São Carlos, Brazil
engsoares@gmail.com, mstela@sel.eesc.usp.br

² Department of Computer Science - Catholic University of Goiás - Av. Universitária 1440, Goiânia - Brazil
clarimarc@gmail.com

<http://www.springer.com/lncs>

Abstract. This paper presents a method for predicting nonlinear time series. It is based on the multiscale filtering, fundamental and technical model and artificial neural networks. In the technical model we used wavelet transform for disjoin the time series trends then to smooth the economic time series by multiscale filtering. We used too the fundamental analysis, that is, financial and macroeconomics variables to improve the network forecasting. The results were compared with the technical analysis showing that the multiscale filtering and addition of the fundamental variables increase the network forecasting ability.

1 Introduction

About the technical analysis, the Dow theory assert that there are three kinds of trends in temporal economics series: short-time trends, intermediate-time trends and long-time trends. The long-time trends and intermediate-time trends are related with the just price of the scrip while that short-time trends are related with the random behavior of the prices. The short-time trends are hardly models because is hardship to find variables that influence your behavior pattern. The short-time trends are more correlated with the rumors and “humorlessly” of the stock exchange.

In this paper, we explored the wavelet transform multiscale capability it can take advantage of the fact that the multiscale decomposition separates the trend from the signal for smooth the short-time trends. In the context of the signal processing long-time trends and intermediate-time trends are related with the low frequency and short-time frequency are related with the high frequency.

The first suggestion is that, in many instances, the trend affects the low frequency components, while the high frequencies may still be purely stochastic

and can be smooth. The second suggestion is that, the prices quotation forecast can be improved using a Artificial Neural Network (ANN) to find economics variables (fundamental analysis) correlated with stock exchange behavior after to smooth random behavior.

2 Our Approach

Various methods, such as linear regression methods, Fourier transform based methods have been used to analyze time series and make predictions [4,8]. However, these models are based on the assumption of stationarity and require transformation of non-stationary data for analysis. Both linear and non-linear models have limitations when it comes to analyzing non-stationary data. The wavelet transform (WT) has been proposed for time series analysis in many papers over the last few years [11]. The big advantage of a WT is the time-frequency resolution that enable the localization of high-frequency in fine time intervals, in finance, they often mean observations taken daily or at a finer time scale. The idea behind these localization time-frequency is to cut the signal of interest into several parts and then analyze the parts separately with the aim to smooth random behavior.

In your discrete version the WT is give to:

$$f(t) = \sum_k c_k^j \phi_{j,k}(t) + \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} d_k^j \Psi_k^j(t) \tag{1}$$

where the function

$$\Psi_k^j(t) = \Psi(2^j t - k) \tag{2}$$

is called mother wavelet and the function

$$\phi_{j,k}(t) = 2^{\frac{j}{2}} \phi(2^j t - k) \tag{3}$$

is called father wavelet or scaling function and the coefficients j, k are the wavelet transform coefficients.

Wavelet transforms specify location (via translation) and frequency (via dilation) using the father and mother wavelet, that is, using base functions ϕ and Ψ respectively. The wavelet function is in effect a band-pass filter and scaling it for each level halves its bandwidth. The signal is decomposed simultaneously using a high-pass filter. The outputs giving the detail coefficients (from the high-pass filter) and approximation coefficients (from the low-pass). This decomposition is repeated to further increase the frequency resolution and the approximation coefficients decomposed with high and low pass filters and then down-sampled. For time series we used 2 level wavelet decomposition. In wavelet decomposition, universal threshold and soft thresholding have been applied as techniques [2,3]. Equation 4 presents the universal threshold expression.

$$\sigma \sqrt{2 \log n} \tag{4}$$

Then, the equation 4 is applied in detail coefficients for each level wavelet decomposition, afterwards the smoothed signal is recovered by inverse wavelet transform.

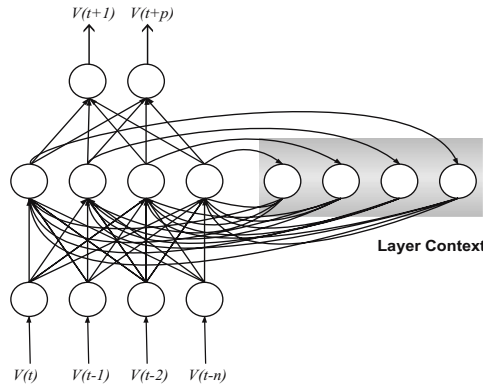


Fig. 1. Elman's network

The ANN showed in figure 1 implement two memory mechanisms. The first one is the input layer by delayed time data. The data window ($V(t - 1)$, $V(t - 2)$, $V(t - 3)$, \dots , $V(t - n)$) applied to the network input layer implement temporal memory of the p past data, where the n numbers of past time are called delay window. The network forecast the $t + p$ future prices in agreement with the $t - n$ input delay. The second memory mechanisms is implemented in the hidden layer, where each hidden unit is connected to itself and also connected to all the other hidden units. The network hidden layer makes a copy of the past inputs characteristics in the context layer. The recurrence indirectly memorize all the previous input values presented to the network. For the network training is used the backpropagation algorithm that employ the descending gradient to adjust the network weights for the training data. The weights adjust is given by equation 5 where η is a positive constant called learning tax.

$$\Delta w_{ij} = \eta \frac{\partial E(t', t)}{\partial w_{ij}} \tag{5}$$

For more detail about the backpropagation algorithm, is recommended the reading of Rumelhart's original paper [12].

2.1 Joining Fundamental and Technical Analysis

In Jang [5] is proposed the use of two ANN for technical analysis: an ANN to forecast the price in long-time and other to forecast the price in short-time. It was used the following data: peak values, low values, close values, and volume negotiated for each day. In Kim [6] was used the feature transformation based on

domain knowledge for the forecast of futures quotations. The data filters classify and extract rules to reduce the feature space dimensionality.

Although the Jang [5] and Kim [6] works present relevant results, they do not consider economic aspects related with the data. Qi [9] and Racine [10] demonstrated that there are economic variables correlated with the financial scrips quotations. We used the Spearman rank correlation coefficient (SRCC) for select important variables. SRCC is commonly used to find relationship between two variables that can be non-linear relationship. The Spearman’s rank correlation coefficient r is give to:

$$|r| = \frac{\sum_{i=1}^n R(x_i)R(y_i) - n \left(\frac{n+1}{2}\right)^2}{\left(\sum_{i=1}^n R(x_i)^2 - n \left(\frac{n+1}{2}\right)^2\right)^{0.5} \left(\sum_{i=1}^n R(y_i)^2 - n \left(\frac{n+1}{2}\right)^2\right)^{0.5}} \quad (6)$$

where $R(x)$ and $R(y)$ are the ranks of a pair of variables (x and y) and r varies from 1 (perfect correlation) through 0 (no correlation or independence). The decision criterion for the addition of variable is $|r| > 0.1$.

We extracted too financial statement data from the economic report of the companies and macroeconomic variables from the Getúlio Vargas foundation, main organization of financial studies in the Brazil. Based on recommendations from previous studies [1,7,9,10] we tested 5 financial statement variables and 6 macroeconomic variables as the predictor attributes because we judge that the variable number is sufficient for the study. The definitions of the variables used are given in table 1.

Table 1. Fundamental variables

Financial variables	
Variable	Description
v_1	Current assets/Current liabilities
v_2	Net sales/Total assets
v_3	Market capitalization = Stock price x Common shares outstanding
v_4	Earnings per share
v_5	Capital expenditure
Macroeconomic variables	
v_6	Government interest rate
v_7	Consumer price index
v_8	Effective exchange rate
v_9	Purchase price of crude oil
v_{10}	BOVESPA index [†]
v_{11}	Country risk classification [‡]

[†] The Bovespa Index is the main indicator of the Brazilian stock market’s average performance.

[‡] The country risk classification method measures the likelihood that a country will service its external debt.

The technical model variables, quotations of the opening price, maximum price, minimum price, closing price and trading volume are joining the fundamental model variables what satisfy the SRCC test for the ANN input.

To evaluate the network predict capacity is necessary to measure the error. The error quantifiers for this task are the PME (Percentile Mean Error) defined in the equation 7 and PQME (Root of Percentile Quadratic Mean Error) defined in the equation 8.

$$PME = \frac{1}{N} \sum_{t=1}^N \frac{(V(t) - V(pt))}{V(t)} \quad (7)$$

$$PQAE = \sqrt{\frac{1}{N} \sum_{t=1}^N \frac{(V(t) - V(pt))^2}{V(t)}} \quad (8)$$

In both equations 7 and 8 $V(t)$ is a true value in time t , $V(pt)$ is a quotation for the time t and N is the number of observations at validation set. For the equation (7) the result is positive in case of underestimated quotation forecast and negative in case of overestimated. For the equation (8) the result indicates the forecasting mean error, where the minimum value is zero.

3 Results and Discussion

The preferential scrips prices of the type Pn from distinct economic sectors were obtained from São Paulo stock exchange (BOVESPA): Pão de açúcar group (supermarket), ITAUSA group (financial group), EMBRAER (aircraft industry), Vale do rio doce company (iron ore extraction), Petrobrás (exploration and production of petroleum) and America drink company (drinks company).

The referring quotations from the period of 2004 till 2005 year will be used in the training set of the network. After the data normalization the data set were smooth by wavelet transform describe in the section 2. The most suitable resolution level is identified based on the smoothness of the approximation signal at that level, that is, having all the high frequency components smoothed. The desired approximation signal should depict a general pattern of its original. For the proposed model, different resolution level are tested and found that approximation signal at resolution level two is sufficiently smooth to represent a general pattern of the original signal.

The figure 2 shows the smoothing results of short-time trend related previous. Observe in the figure 2 that fine time intervals with low frequency not are sliced, this fact is not possible using Fourier transform. The high frequency located in particular slices (for example in the space of time 0 to 100 daily evolution) of the series are smoothed preserve fine peaks with low frequency. This is possible with the localization time-frequency of the wavelet transform. Eventually the window Fourier transform can be used, but the dilemma is determine the ideal window length for each time series.

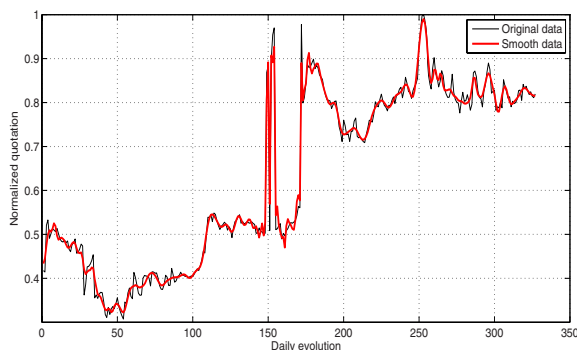


Fig. 2. Quotation Forecast Results for the Pão de Açúcar Group

Table 2. Spearman rank correlation coefficient results for fundamental variables

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	Choice variables
C_1	0.19	0.23	0.05	0.20	0.01	0.08	0.12	0.03	0.01	0.13	0.27	$v_1, v_2, v_4, v_7, v_{10}$ and v_{11}
C_2	0.10	0.04	0.08	0.22	0.05	0.13	0.08	0.07	0.01	0.11	0.23	v_1, v_4, v_6, v_{10} and v_{11}
C_3	0.12	0.22	0.05	0.21	0.03	0.01	0.01	0.04	0.01	0.10	0.20	v_1, v_2, v_4, v_{10} and v_{11}
C_4	0.11	0.12	0.01	0.21	0.04	0.01	0.01	0.11	0.02	0.12	0.27	$v_1, v_2, v_4, v_8, v_{10}$ and v_{11}
C_5	0.11	0.01	0.01	0.18	0.00	0.10	0.07	0.09	0.19	0.21	0.19	v_1, v_4, v_9, v_{10} and v_{11}
C_6	0.12	0.14	0.02	0.23	0.03	0.06	0.10	0.03	0.01	0.15	0.23	$v_1, v_2, v_4, v_4, v_{10}$ and v_{11}

where C_1 : Pão de Açúcar Group, C_2 : ITAUSA Group, C_3 : Embraer, C_4 : Vale do rio doce company, C_5 : Petrobrás and C_6 : AMBEV.

With the time series smoothed we applied the SRCC test for addition fundamental variables. The results of Spearman rank correlation coefficient and the

Table 3. Final results obtained

Company	Method A ¹		Method B ²		Method C ³	
	EMP	REMQP	EMP	REMQP	EMP	REMQP
C_1^\diamond	$-1.94x10^{-3}$	$1.37x10^{-2}$	$-1.88x10^{-5}$	$9.32x10^{-3}$	$2.04x10^{-5}$	$1.89x10^{-4}$
C_2^\diamond	$5.04x10^{-2}$	$1.20x10^{-1}$	$3.36x10^{-4}$	$4.820x10^{-2}$	$1.92x10^{-5}$	$1.45x10^{-3}$
C_3^\diamond	$-8.04x10^{-1}$	2.06	$3.94x10^{-1}$	$7.07x10^{-2}$	$-4.31x10^{-3}$	$7.20x10^{-2}$
C_4^*	$7.61x10^{-5}$	$5.88x10^{-2}$	$6.02x10^{-5}$	$4.90x10^{-2}$	$8.32x10^{-4}$	$9.57x10^{-4}$
C_5^*	$4.71x10^{-2}$	$3.53x10^{-1}$	$4.00x10^{-2}$	$3.53x10^{-1}$	$4.34x10^{-3}$	$2.13x10^{-2}$
C_6^*	$7.01x10^{-2}$	$8.99x10^{-2}$	$3.74x10^{-3}$	$5.30x10^{-3}$	$4.22x10^{-3}$	$5.08x10^{-3}$

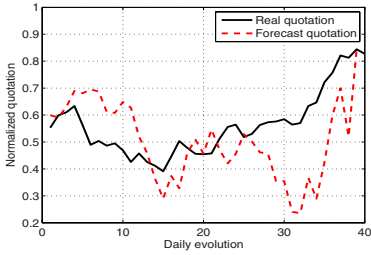
[◇] Forecasting using 40 neurons in the output layer.

^{*} Forecasting using 150 neurons in the output layer.

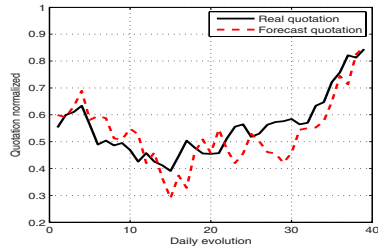
¹ Method A: Only past data in ANN input (Technical Model).

² Method B: Past Data smoothed by wavelet transform in ANN input.

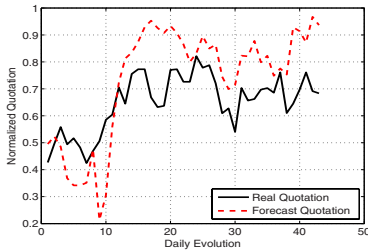
³ Method C: Past Data smoothed by wavelet transform and fundamental variables (Fundamental model) in ANN input.



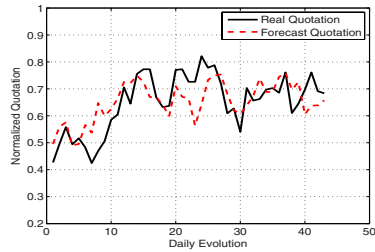
(a) Pão de açúcar group scrips quotation forecasting using method A.



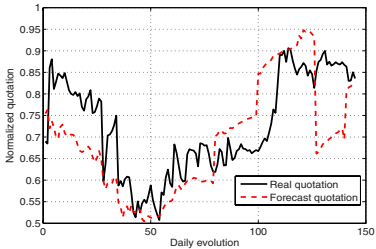
(b) Pão de açúcar group scrips quotation forecasting using method C.



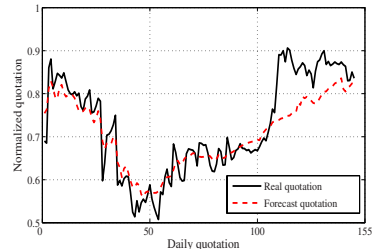
(c) ITAUSA group scrips quotation forecasting using method A.



(d) ITAUSA group scrips quotation forecasting using method C.



(e) Embraer scrips quotation forecasting using method A.



(f) Embraer scrips quotation forecasting using method C.

Fig. 3. Scrips quotation forecasting

resume of choice variables after of SRCC test are shows in table 2. The results of table 2 shows that for each economic time series distinct variables can be correlated. The addition of no-correlated variables with the serie can not help in the forecasting. For Pão de Açúcar group the variables v_2, v_4, v_7, v_{10} and v_{11} can be correlated with the time series behavior and variables v_1, v_3, v_5, v_6, v_8 and v_9 not. However for the Petrobrás the variables v_1, v_4, v_6, v_{10} and v_{11} can be correlated with te time series behavior and v_2, v_3, v_5, v_6 and v_8 not. For each time series the fundamental variables choice are addition in the ANN input for future quotations forecast.

The quotations forecasting refer to mean values for the future days from the analyzed values. After the data normalization the network was trained with the

learning algorithm backpropagation using sixty delay input, sixty neurons in the hidden layer and the number of neurons in the output layer is determinate by the forecasting length. The final results are showed in table 3 and figures 3(a, b, c, d, e and f).

The table 3 show that the better results were obtained using the wavelet transform and fundamental variables. In spite of wavelet transform help the ANN generalization the ANN get to find behavior patterns with addition of fundamental variables. The use only wavelet transform or other method that not use economic aspects can not be appropriate. The forecasting can be make, but which are the economic conditions for the forecasting? Is possible foresee the economic recession? We think that for better that it is the method, it he must consider the economic conditions for forecast the economic time series. Long-time forecast can be made but the success of forecast quotation will depend of ideal economic scenery. Another approach is to generate futures economic sceneries with the possible fundamental variables condition.

It is observed in the figure 3 that the network had an improvement in the prices quotation forecast. The empiric results indicate that the short-time trend are little related with market behavior pattern and long-time and intermediate trends are correlated with fundamental variables. The result demonstrates that not only the historical data affect the quotation behavior. The investors do not follow a logic established only by past quotation. The past quotations are important but not the unique factor.

4 Conclusion

In this paper, we presented a method for predicting nonlinear time series. It is based on the multiscale filtering, fundamental and analysis model and neural networks modelling. The series obtained after wavelet decomposition filter contains information about the trends with the random behavior smoothed. The results presented show that wavelet transform is an efficient tool for the time series analysis. Using the wavelet transform were possible to smooth the random behavior of the time series. Using the Elman's network was possible make quotation forecasting with addition of economic variables that influence the financial scrips quotations. The combination of data considered presented when comparative relatively satisfactory performance to the traditional models that are only based on the technical analysis.

The methodology used for future quotation forecast provides an improvement in the forecast capacity of ANNs offering results more parsimonious.

4.1 Future Works

For the future works is propose the M -channels filter bank in wavelet transform. The time series is decomposed by M -channels, so-called subband signals, in M frequencies on the contrary of only high frequencies and low frequencies.

Each one of the subband signals carries information on the time series in a particular frequency band that can be correlated with distinct trends and economics variables.

References

1. Manjeet S. Dhatt, Yong H. Kim, and Sandip Mukherji, *Relations between stock returns and fundamental variables: Evidence from a segmented market*, *Asia-Pacific Financial Markets* **6** (1999), 221–233.
2. David L. Donoho, *De-noising by soft-thresholding*, *IEEE Transactions on Information Theory* **41** (1995), no. 3, 613–627.
3. T.R. Downie and B.W. Silverman, *The discret multiple wavelet transform and thresholding methods*, *IEEE Transactions on Information Theory* **46** (1998), no. 9, 2558–2561.
4. Ramazan Gencay, Brandon Whitcher, Ramazan Gengay, and Faruk Selguk, *An introduction to wavelets and other filtering methods in finance and economics*, Academic Press, 2001.
5. Gia Shuh Jang, *An intelligent stock portfolio management system based on short-term trend prediction using dual-module neural networks*, *International Conference on Artificial Neural Networks* **1** (1991), 447–452.
6. Kyoung-Jae Kim, *Artificial neural networks with feature transformation based on domain knowledge for the prediction of stock index futures*, *Intelligent Systems in Accounting, Finance and Management* **3** (12), no. 167-176, 2004.
7. Monica Lam, *Neural network techniques for financial performance prediction: integrating fundamental and technical analysis*, *Decision Support Systems* **37** (2004), no. 4, 567–581.
8. Donal B. Percival and Andrew T. Walden Percial, *Wavelet methods for time series analysis*, Cambridge University Press, 2000.
9. Min Qi, *Nonlinear predictability of stock returns using financial and economic variables*, *Journal of Business and Economic Statistics* **17** (1999), no. 4, 419–429.
10. J. Racine, *On the nonlinear predictability of stock returns using financial and economic variables*, *Journal of Business and Economic Statistics*, **19** (2001), no. 3, 380–382.
11. O. Renaud, J.L Starck, and F. Murtagh, *Wavelet-based combined signal filtering and prediction*, *IEEE Transaction on Systems, Man and Cybernetics* **35** (2005), no. 6, 1241–1251.
12. D. E. Rumelhart and J. L. McClelland, *Parallel distributed processing*, vol. 1, The MIT Press, 1986.

Multi-view Moving Human Detection and Correspondence Based on Object Occupancy Random Field

Jian Xu, Xiao-qing Ding, Sheng-jin Wang, and You-shou Wu

State Key Laboratory of Intelligent Technology and Systems, Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
{xujian,dxq,wsj,wuys}@ocrserv.ee.tsinghua.edu.cn

Abstract. In this paper, we address the problem of detection and correspondence of moving persons in a multi-camera set. A novel algorithm is proposed based on object occupancy random field (abbreviated by OORF), which has a robust performance even under severe occlusions and a fast implementation speed. The core of our algorithm is OORF and object window structure. The latter is essential to compute OORF and provides a scheme to detect and correspond objects simultaneously.

Keywords: Detection, correspondence, occlusion, multi-view, object occupancy random field, ground plane, occupancy probability.

1 Introduction

In this paper, we address the problem of detection and correspondence of multiple moving persons in a multi-view set. Human detection is a common problem in surveillance applications. And object correspondence is a special one in multi-camera situation where persons in the scene may emerge in more than one views at the same time, and it is required labelling the consistent ones across different views. Both of the two problems are crucial in multi-view systems and are the base for following operations such as tracking, behavior analysis, information fusion and so on. We are interested in crowded situations where partial or total occlusions are common and it can not guarantee that any of the people will be visually isolated. In this set, an efficient algorithm should face two difficulties: occlusion and heavy computational burden. However, few algorithms can deal well with these two problems. For most existing systems, human detection is first performed in the 2D image of each view and then corresponded across views. In densely crowded scenes, correct detection is difficult due to occlusion and the following correspondence based on the former detection results cannot be guaranteed to be right and robust. Another strategy, which first fuses information from all views and then deals with the problem in fusion space, has a better performance against occlusion but suffers from heavy computational burden.

The paper propose a novel approach which unifies human detection and correspondence in multi-view set under a uniform framework. The idea is straightforward. The algorithm's core is object occupancy random field (OORF) and

object window structure. OORF depicts the probabilistic property that an object occupies a position on the ground plane. The object window structure helps compute the occupancy probability and is also the tool to detect and correspond people across views. Experiment results show that the algorithm can deal with occlusion correctly and robustly and be performed in real time.

2 Related Work

Multi-camera surveillance is a relatively new domain in computer vision, which is gaining increasing interest recently. Human detection and correspondence are two substantial problems in multi-view environment. Related work on this topic can be organized into two loose categories by what types of space information are dealt with in, i.e. image space and fusion space.

The image-space based scheme implements people detection in 2D image space first, and then associates detection results of different views to label them. Most multi-view approaches adopt this scheme [1]-[4]. Utsumi et al. [1], assuming that cameras are calibrated, detect objects relying on human models and label them by features such as position, velocity, height and visual angle constraints. Javed et al. [4] model connected blobs obtained from background subtraction to extract human and use color histogram to label them. To be robust against occlusion, this scheme usually needs a sufficient number of occlusion-free views for each object and cannot work well under severe occlusions.

The fusion-space based scheme solves the problems not in image space but in fusion space where all information from different views are combined together. This scheme is a newly-emerging one and there is just a little related literature [8][9]. Fleuret et al. [8] introduce fixed point probability field in top view which is computed from binary foreground image of all views and estimate the human position in probability field. Saad et al. [9] select one view as a reference image and use a planar homograph constraint to resolve occlusions and determine people's locations. The planar homograph constraint combines foreground likelihood information probability of a pixel in the image belonging to the foreground from different views. These approaches have a better performance to tackle occlusion but suffer from the heavy burden of computation.

3 Object Occupancy Random Field

The section details object occupancy random field. We assume that cameras are calibrated, and that people are moving on a calibrated ground plane.

3.1 The Ground Plane Π

Multiple cameras monitor a scene, as can be seen in Fig. 1. We call the intersected area of all views the valid area and restrict it a ground plane Π . The ground plane constraint will simplify the problem and is reasonable in most applications such as crossroads, corridors, and airdromes are approximately planar.

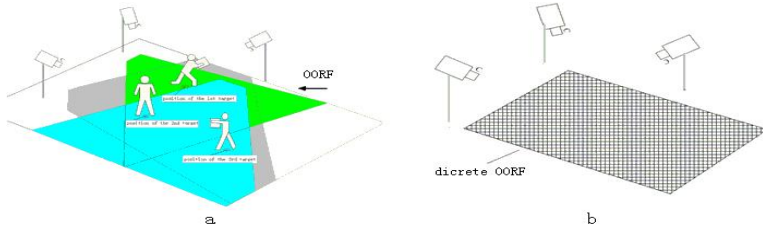


Fig. 1. Object occupancy field. (a) Valid area for multiple cameras. (b) Discrete object occupancy field.

3.2 Definition of OORF

The valid monitored area with the ground plane constraint is called object occupancy field due to observable objects moving on this site. We discretize object occupancy field with equidistance grids, as Fig. 1-b shows. Now we give the definition of object occupancy random field $p(I)$, describing the probabilistic property that an object occupies one position in the field. Let $E_i \in \{0, 1\}$ denote a binary random variable which means that an object exists at position i if $E_i = 1$ while not existing if $E_i = 0$. Let n denote the number of cameras monitoring the scene and I_k be the observed image obtained from the k_{th} camera. $I = \{I_1, I_2, \dots, I_n\}$ means observed image set from all the cameras in the scene. Let θ_k be intrinsic and external parameters for the k_{th} camera and $\Theta = \{\theta_1, \theta_2, \dots, \theta_n\}$ represents parameter set of all cameras. Now let $V = \{I, \Theta\}$ combine all observed images and parameters from the n cameras into a set. Given observed images and parameters of all cameras, the occupancy probability is written as

$$P(E_i = 1|V) \tag{1}$$

we call object occupancy field with the definition of occupancy probability object occupancy random field.

4 Computation of the Occupancy Probability

The core of OORF is how to compute occupancy probability. The paper computes occupancy probability using local images through a structure called object window structure. The object window structure is also the key element that unifies the process of human detection and correspondence.

4.1 Object Window Structure S

We define a structure called object window structure according to the imaging principle of a camera, denoted by S . Let M be a primitive human model which is a cylinder with radius r and height h as Fig. 2(a). Let W denote a set which

consists of n sub-windows (w_1, w_2, \dots, w_n) . The k_{th} sub-window w_k is the projection of human model M at a position in OORF to the k_{th} camera. To compute conveniently, the sub-window is simplified to a rectangle. S is composed of M and W and is written as $S = (M, W)$.

The definition of object window structure is too abstract to be comprehended. Now we interpret the physical meaning of the object window structure according to the imaging process. As can be seen in Fig. 2, there is a configuration of 3 cameras with known parameters $\Theta = \{\theta_1, \theta_2, \theta_3\}$ on OORF Π . The observed binary foreground images from all views are $I = \{I_1, I_2, I_3\}$. We place one primitive human model on each of the 3 positions in the OORF in Fig. 2 and consider the case that the 2_{nd} position is occupied by a person and others not occupied. Project a human model to individual cameras according to their camera parameters and then we get 3 projected areas which are approximately rectangular (in order to computer conveniently we simplify them as rectangles). The projected areas in images for the 1_{th} , 2_{nd} and 3_{rd} human models are individually represented by yellow, purple and red rectangles. It can be seen from the figure that the projected points of a human model at a position will only fall within the area of the associate sub-windows in images. It indicates that the occupancy probability in a position merely has relation with its sub-windows. Consequently we will consider only the area in the sub-windows when computing the occupancy probability. It can be seen from Fig. 2 that foreground pixels will only fall within most area of the associate sub-windows if a human model occupied by a person. Let $fore(w_k^i)$ denote the number of foreground pixels that fall within the k_{th} sub-window of the position i and $area(w_k^i)$ the number of pixels that the k_{th} sub-window of the position i contains. We use $filldensity = \frac{fore(w_k^i)}{area(w_k^i)}$ to denote the ratio of the foreground pixel number to the pixel number that the sub-window contains.

We can see from Fig. 2 that:

1. If a person contained in a human model stands at a position, then his/her projected pixels will fall only in area of the associate sub-windows in images.
2. A person will not occupy only one single position since he/she is not a point object. We call the center of those positions that a person occupies the person's position. And the *filldensity* of the position occupied by a person will decrease as the distance away from the person's center position increases.
3. The sub-windows for a not-occupied position may be partially filled by foreground pixels due to occlusion or image noise. However, the *filldensity* of these sub-windows will be not all high.

The 1_{st} property indicates that the occupancy probability of a position has only relation with its object window structure and others tell that the *filldensity* will influence the occupancy probability.

4.2 Computing Occupancy Probability

Now we will show how to compute occupancy probability. We first provide the deduction of the computational formula and then give the implementation of the

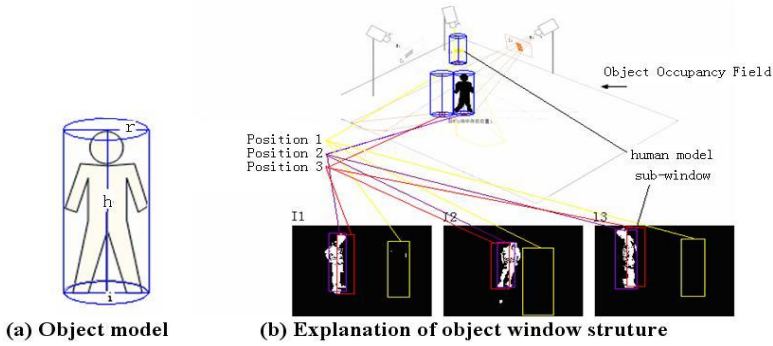


Fig. 2. Object window structure

computational formula in practice. Before the deduction we state the following two assumptions.

- Assumption 1: Occupancy probabilities of different positions are independent.
- Assumption 2: Individual observed images simultaneously captured from different cameras are conditional independent given all cameras' parameters.

Then we can deduce the occupancy probability formula as follows.

$$\begin{aligned}
 p(E_i = 1|V) &= p(E_i = 1|I, \Theta) \\
 &= \frac{p(I|E_i = 1, \Theta)p(E_i = 1|\Theta)}{p(I|\Theta)} \\
 &= \prod_{k=1}^n p(I_k|E_i = 1, \Theta) \frac{p(E_i = 1|\Theta)}{\prod_{k=1}^n p(I_k|\Theta)} \\
 &= \frac{p(E_i = 1|\Theta)}{\prod_{k=1}^n p(I_k|\Theta)} \prod_{k=1}^n \frac{p(E_i = 1|I_k, \Theta)p(I_k|\Theta)}{p(E_i = 1|\Theta)} \\
 &= \frac{1}{(p(E_i = 1|\Theta))^{n-1}} \prod_{k=1}^n p(E_i = 1|I_k, \Theta) \tag{2}
 \end{aligned}$$

As the term $\frac{1}{(p(E_i=1|\Theta))^{n-1}}$ has nothing with observed images, it can be got rid of. We can write occupancy probability form as

$$p(E_i = 1|V) \propto \prod_{k=1}^n p(E_i = 1|I_k, \Theta) \tag{3}$$

Formula 3 tells us, $p(E_i = 1|V)$ is determined by the individual term $p(E_i = 1|I_k, \Theta)$. As the previous discussion indicates that the occupancy probability

of a position has only relation with its object window structure and that the *filldensity* will influence the occupancy probability, we compute $p(E_i = 1|I_k, \Theta)$ as follows

$$p(E_i = 1|I_k, \Theta) = \text{filldensity} = \frac{\text{fore}(w_k^i)}{\text{area}(w_k^i)} \quad (4)$$

Plugging [4](#) into [3](#) we get,

$$p(E_i = 1|V) \propto \prod_{k=1}^n \frac{\text{fore}(w_k^i)}{\text{area}(w_k^i)} \quad (5)$$

The formula [5](#) is the practical computational form of occupancy probability.

5 OORF Based Human Detection and Correspondence

Fig. [3](#) shows the flow chart for implementing the algorithm and an implementing example. The flow char consists of 4 parts: 1_{st} is foreground extraction, 2_{nd} OORF computation, 3_{rd} object position extraction in OORF and 4_{th} object extraction and correspondence. The following content will detail the flow chart.

5.1 Foreground Extraction

From the previous section we know that the occupancy probability is calculated based on binary foreground images from individual views. This part provides the foregrounds of individual images by the mixed Gaussian model [6](#). Fig. [3](#).b shows foreground images from two cameras.

5.2 OORF Computation

The binary foregrounds are sent to this part to compute occupancy probability of OORF according to the formula [5](#) and Fig. [3](#).c shows the computing result.

5.3 Object Position Extraction in OORF

This part analyzes the computing result of OORF. It cannot be guaranteed due to occlusion, not-point objects, or noise that the value of occupancy probability is 1 at the occupied position and 0 at the not-occupied position. In practice, the probability is high at the center position of a person and is getting lower and lower as it is away from the center. Therefore, it is essential to analyze the OORF to obtain true object positions. Here we do as follows. First select an experiential threshold for occupancy probability to remove those positions with low value. Then we implement morphologic operator to dispel those isolated pixels because real occupied positions will form a crest not just a vertex. At last, we will label the connected area and select the center of each connected area as the object's positions. Fig. [3](#).d shows the post-processing result of OORF.

5.4 Object Extraction and Correspondence

According to the estimated positions obtained from OORF, we use the sub-windows of object window structure to extract objects, which, at the same time, indicates the correspondence relationship of objects between different views, as we can see from Fig. 3.e.

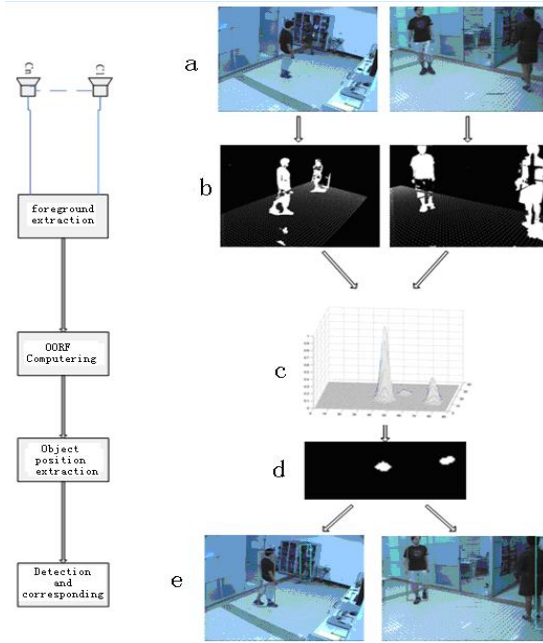


Fig. 3. Implementing process of our algorithm. The left part is the flow chart while the right part gives an example with the configuration of 2 cameras. (a) input images, (b) binary foreground images, (c) computation result of OORF, (d) result after processing OORF, the white areas are object areas, (e) the last result for human detection and correspondence. The grids in images are discrete positions.

6 Results and Discussions

In this section, we show experimental results on two sequences shot in a room about $4 \times 5m^2$. Cameras are located at three corners of the room. The two sequences are obtained by 2 and 3 cameras respectively. The experiments are implemented on a 3.0G Hz Intel Pentium-4 CPU with 1.0G RAM.

6.1 Results

Fig. 4 demonstrates a few frames of each sequence. In each case, we display the original images with the results of detection and correspondence based on object

window structure, and the OORF result map with the locations of detected people. The top three rows are the result of a two-person sequence by 2 cameras and the bottom three a three-person sequence by 3 cameras. Table 1 shows some evaluation parameters we obtain from two sequences of 300 frames, including processing time (time span for implementing our algorithm once), false negative rate and false positive rate.

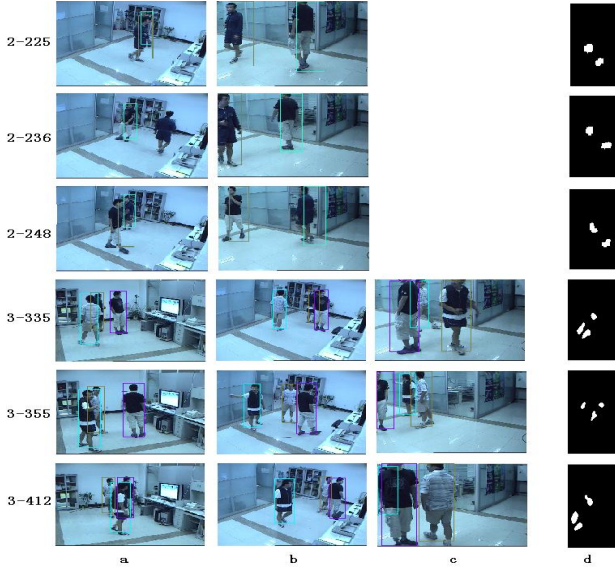


Fig. 4. Results. The top three rows are results under a 2-view set and the bottom three under a 3-view set. The left numbers are frame sequence indices, a, b and c are results of detection and correspondence results and d is the post-processing map result of OORF.

Table 1. Test performance

parameter	performance
processing time 2-view	18ms
processing time 3-view	34ms
False negative 2-view	12.3%
False negative 3-view	9.1%
False positive 2-view	9.8%
False positive 3-view	6.5%

6.2 Occlusion

Occlusion is common in our experiments. As can be seen, occlusion is more severe in Fig. 4. For example, total occlusion occurs in the left image of frame

2-225 and partially occlusion in Frame 2-248. However, the performance of our algorithm is robust. The reason that our algorithm has a powerful ability against occlusion is that we fuse foreground information of all views into 3-dimensional space to locate humans rather than detect humans directly in 2-dimensional image space. Under occlusion, the occluded parts cannot provide information for human detection directly in image space but they still fall in the foreground area, which will support locating them in 3-dimensional space. Then we detect and correspond humans across views based on their location in OORF.

6.3 Computational Complexity

Most implementing time is spent on the part of the computation of the probability of OORF, so we just analyze the computational complexity of OORF. To compute conveniently, we simplify the projection area of human models on each view as rectangular area. Let N be the number of discrete positions of OORF. Assume that the height of the rectangular projection area in the k_{th} camera of a human model at position i is $h_{k,i}$ (represented by pixel unit) and the width is fixed as $w_{k,i} = h_{k,i}/4$. The computation of $\frac{fore(w_k^i)}{area(w_k^i)}$ needs $w_{k,i} \times h_{k,i} - 1$ additions, 1 products and 1 division. Thus, the total computational cost is $\sum_{i=1}^N \sum_{k=1}^n w_{k,i} \times h_{k,i} - 1$ additions, $N \times n$ products and $N \times n$ divisions. For our experiments $n = 3$ and the room $4 \times 5m^2$. The discrete square grid of OORF is $75mm$ so $4 \times 5m^2$ so $N \approx 4000$. One total computation of OORF needs nearly 1.2×10^7 additions, 1.2×10^4 products and 1.2×10^4 divisions. It can be tackled quickly for a 3.0GHz Intel Pentium-4 CPU with 1.0G RAM. Table I shows that we can perform the algorithm about 55 times per second under 2 cameras and 30 times per second under 3 cameras. (The original images are of size 768×576 and we scale them to 384×288 . One implementation process includes the computation and analysis of OORF, human detection and correspondence, but don't include foreground extraction.)

7 Conclusion

The paper proposes a novel unified algorithm of multi-view human detection and correspondence based on object occupancy random field. The main contributions of our algorithm are as follows: the first is the presentation of the unified scheme for solving detection and correspondence simultaneously; the second is a fast computational approach for OORF. Both of these contributions are based on a structure called object window. The experiment results demonstrate that the algorithm is robust and efficient even under occlusion while the implementation speed is fast.

References

1. Utsumi, A., Mori, H., Ohya, J. and Yachida, M.: Multiple-View-Based Tracking of Multiple Humans. IEEE Proceedings of Int. Conf. on Pattern Recognition (1998) 597-601.
2. Orwell, J., Massey, S., Remagnino, P., Greenhill, D. and Jones, G.A.: A Multi-agent Framework for Visual Surveillance. Int. Conf. on Image Processing (1999) 1104-1107.

3. Collins, R.T., Lipton, A.J., Fujiyoshi, H. and Kanade, T.: Algorithms for Cooperative Multisensor Surveillance. *Proc. IEEE* **89** (2001) 1456-1477.
4. Javed, O., Rasheed, Z., Shafique, K. and Shah, M.: Tracking Across Multiple Cameras with Disjoint Views. *IEEE Proceedings of Int. Conf. on Computer Vision* (2003) 952-957.
5. Otsuka, K. and Mukawa, N.: Multi-view Occlusion Analysis for Tracking Densely Populated Objects Based on 2-d Visual Angles. *IEEE Proceedings of Computer Vision and Pattern Recognition* **1** (2004) 1-90.
6. Stauffer, C. and Grimson, W.: Adaptive Background Mixture Models for Real-time Tracking. *IEEE Proceedings of Computer Vision and Pattern Recognition* (1999) 245-252.
7. Krumm, J., Harris, S., Meyers, B., Brumitt, B., Hale, M., and Shafer, S.: Multi-camera Multi-person Tracking for EasyLiving. *IEEE Proceedings of Int. Workshop on Visual Surveillance* (2000) 3-10.
8. Fleuret, F., Lengagne, R., and Fua, P.: Fixed Point Probability Field for Complex Occlusion Handling. *IEEE Proceedings of Int. Conf. on Computer Vision* **1** (2005) 694-700.
9. Saad, A., Khan, M. and Shah, M.: A Multiview Approach to Tracking People in Crowded Scenes Using a Planar Homography Constraint. *European Conf. on Computer Vision* (2006) 133-146.

Determination of Storage Time of Rice Seed Using ANN Based on NIRS

Zhu Dengsheng¹ and Li Xiaoli²

¹ Jinghua College of Profession & Technology, 321007, Jinghua

² College of Biosystems Engineering and Food Science, Zhejiang University, 310029, Hangzhou, China

Abstract. A simple, fast and nondestructive approach was put forward to classify rice seed of different storage time. This discrimination was conducted by integrated with wavelet transform (WT), principal component analysis (PCA) and artificial neural networks (ANN) based on near infrared reflectance spectroscopy (NIRS). Four classes' samples from four different storage times were used for Vis/NIR spectroscopy on 325-1075 nm using a field spectroradiometer. WT and PCA were used to reduce spectral data dimension and extract diagnostic information from spectra data. The first eight PCs, which accounted for 99.94% of the raw spectral variables, were used as input of the ANN model. The ANN model yielded high discrimination accuracy. The discrimination accuracy was 97.5% for rice seed samples of four different storage years.

1 Introduction

Rice sustains two-thirds of the world's population. So, a quick, accurate and nondestructive way is needed to classify the rice seed of different storage time. Near infrared spectroscopy (NIRS) has been used to measure the quality of rice and classified many materials from different class successfully. However, few researches focused on discrimination the rice seeds of different storage time based on Vis/NIR spectroscopy technique.

NIR spectroscopy technology has been effectively combined with chemometrics such as partial least squares (PLS), multiple partial least squares (MPLS), principal component analysis [1][2] and discriminant analysis for classification, discrimination and authentication purposes [3]. These methods failed with many variables, common solutions are to reduce the dimension of the predictor matrix by using data compressed arithmetic and then apply LDA. Wavelet transform [4] [5], PCA [6] [7] has been proved as effective tool for feature extraction. In qualitative and quantitative analysis, artificial neural networks are more and more widely applied during the past several years. Compared with SIMCA, PLS, DPLS, QDA and LDA et al method, the better advantage of ANN is its anti-jamming, anti-noise and robust nonlinear transfer ability [8]. But the shortcoming of ANN is nonconvergent mostly when the input data is mass. So, the spectral data must be compressed as low-dimension data before ANN.

Inspired by this, we present a novel chemometrics method for differentiating the rice seed of different storage time, by integrating artificial neural network with wavelet extraction and principal component analysis. The wavelet transform was used to extract features from the spectrum, and the features were visualized by principal component analysis in PCs space, then the PCs which is closely correlative with the categories of these samples were used as the input of an ANN model for discrimination the classes of samples with different storage time.

2 Materials and Method

2.1 Materials

210 Samples of rice seed of same varieties produced from 2002 to 2005 were taken in experiment. The rice seed samples were obtained from Grain Supply Center of Hangzhou, Zhejiang, China. The corresponding samples were storage for one year (OYS), two year (TWYS), three year (THYS) and four year (FYS). All samples were stored without any chemical or biological preservative treatment. The samples states can be seen in table 1.

Table 1. Status of the samples in the research

Storage time	Varieties	Producing area	Storage area	No.
One year	early-indica type rice	Jiangxi,China	Hangzhou,China	54
Two year	early-indica type rice	Jiangxi,China	Hangzhou,China	51
Three year	early-indica type rice	Jiangxi,China	Hangzhou,China	52
Four year	early-indica type rice	Jiangxi,China	Hangzhou,China	53

2.2 NIR Spectra Collection

A Vis/NIR spectroradiometer (Handheld FieldSpec) was used to collect spectrum from 325 to 1075 nm at 3.5 nm bandwidth. The uniform glass vessel (diameter: $d=60\text{mm}$, height: $h=14\text{mm}$) was used to load the rice seed, and the vessel was filled with paddy. The spectroradiometer was fixed at 120 mm above the surface of the sample with the visual angle of 25° . The light source of a Lowell pro-lam 14.5 V Bulb tungsten halogen that could be used both in the visible and near infrared region was placed about 100 mm away from the sample surface.

A 100 mm² thick Teflon® disk was used as the optical reference standard for the system. A reflectance (R) was calculated by comparing near infrared energy reflected from the sample with the standard reference. Due to imperfection in the own system, a big scattering can be observed in the beginning and end of the spectral data, affecting the measurement accuracy, so the first 75 and the last 75 wavelengths data were taken out of all analysis, starting from here all the considerations were based on this range of wavelengths. The absorbance spectra of all the four classes can be seen in fig.1.

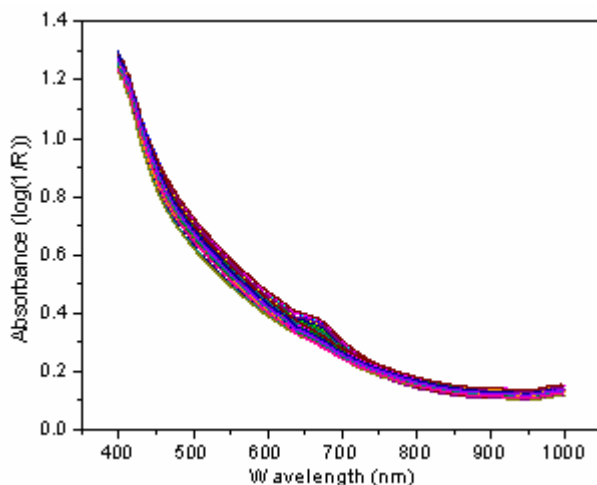


Fig. 1. Absorbance spectra of four classes samples

2.3 Chemometrics

The wavelet transform (WT) enables the signal (spectrum) to be analyzed as a sum of functions (wavelets) with different spatial and frequency properties [2]. The generated waveforms are analyzed with wavelet multiresolution analysis to extract sub-band information from the simulated transients. Principal component analysis (PCA) is a very effective data reduction technique for spectral data [6]. It summarizes data by forming new variables, which are linear composites of the original variables. Artificial neural networks (ANN) are known as useful tools for pattern recognition, identification, and classification. And such a model can provide data approximation and signal-filtering functions beyond optimal linear techniques. PCA was performed using the Unscrambler 9.5 software (CAMO). The matlab Wavelet Toolbox was used to perform the standard wavelet packet transform using the predefined wavelet filters. The matlab Neural Networks Toolbox was used to build the back-propagation network model.

3 Results and Discussion

Fig.1 shows the average absorbance spectra of paddy samples from four different storage time. Seemingly, there isn't a remarkable difference among these four classes in these spectra range. After comparing in detail, some differences can be detected from 600 nm to 700 nm, which makes it possible to discriminate the samples with difference storage. The differences may be caused by the different internal attribute of these samples, such as the starch and protein. The baseline drift in the spectra (shown in fig.1.) is mostly due to system noise, which can be eliminated by smoothing pre-treatment.

3.1 Feature Extraction

In this research, the wavelet transform was used to select features from the raw spectra. The WT was implemented using a dyadic filter tree. After trying, daubechies2 wavelet was selected as the suitable function to decompose the spectral signal. Then the spectra data, which have 210 rows and 601 columns, were decomposed at third level by db2 wavelet. After decomposition, low-frequency coefficients (cA3) and high-frequency coefficients cD3, cD2 and cD1 were obtained. It can be found that the high-frequency signals cD3, cD2 and cD1 contain mass high-frequency noise, and it can barely give any help to classify the varieties. So, the low-frequency coefficients (cA3) were used to replace the spectral signal with low-frequency data.

3.2 Data Visualization by Principal Components Analysis

Principal component analysis aimed to mapping the wavelet coefficients in a low-dimension space with the largest variability. PCA was performed on the 77-wavelet coefficients of each sample, the dimensionality of the wavelet coefficients was reduced from 77 to 8 by PCA, and hence 8 principal components could be achieved. Fig.2 is the scatter image on the PC1 (variability, 84.4%) vs. PC2 (variability, 9.7%) scores space. The PCs 1 and 2 accounted for 94.1% of the variation.

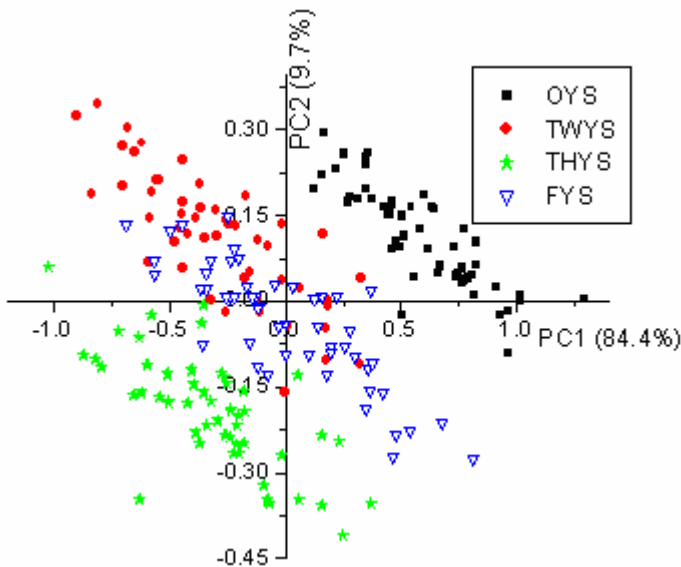


Fig. 2. Scatter plot of PC1 vs PC2 scores of all samples

The image defined by the PCs 1 and 2 shows some difference among the OYS, TWYS, THYS and FYS samples. The four classes samples are distributed in parallel strap shape. Analyzing fig.2, it is obvious that the PC1 and PC2 promote the pretty good separation of the samples of the OYS class, TWYS class and THYS class. The samples of OYS class, THYS class and FYS class can be differentiate by the scores

of PC1 and PC2. But the TYS and FYS samples are overlapped in the image. A more accuracy and clear separation need to be made. So, an artificial neural network algorithm was applied to classify these classes with digital discriminative result. It can be found that the cumulative reliabilities of the first 8 principal components are 99.9%. It means these components can explain 99.9% of the variables the rest components don't give more useful information for detecting the classes.

3.3 Discrimination Sample by ANN

The whole samples were separated randomly into two parts, the randomly selected 170 samples were used as calibration samples, and the remains 40 were used as prediction samples. The first eight principal components from PCA were used as input vector. As there were four different classes samples, the output vectors of these samples were assumed to be as a matrix with one row and four column, only one element was assigned 1, the other 3 were assigned 0. Such as, the output binary vector (1 0 0 0) was denoted as the paddy seed of one-year storage time. A BP neural network model with three-layers was built. The transfer function of hidden layer was tansig function. The transfer function of output layer was logsig function. The train function was trainlm. The best neural network architecture was obtained with 8-8-4. The threshold value was 0.1. The neural network yielded a very high discrimination accuracy, all of the OYS, TWYS and THYS samples were correctly classified for the calibration and prediction sample sets, respectively. The FYS rice seed was more difficult to classify in prediction. However, 90% of FYS samples were correctly classified in the prediction set. The total accuracy rate was 97.5% for all the four classes. The classification and prediction rate of this model can be seen in table 2.

Table 2. Classification and prediction rate of this model

Classes	Classification			Prediction		
	No.	FNo.	Ar	No.	FNo.	Ar
OYS	44	0	100%	10	0	100%
TWYS	41	0	100%	10	0	100%
THYS	42	0	100%	10	0	100%
FYS	43	0	100%	10	1	90%
Total	170	0	100%	40	1	97.5%

Note: No.-number, FNo.-fault number, Ar-accuracy rate.

4 Conclusion

The proposed method of integration of the wavelet transform, principal component analysis and artificial neuron network has shown a pretty good separation of rice seed of different storage time based on Vis/NIRS technology. In this application, wavelet transforms was used as a tool for dimension reduction and noise removal. The principal component analysis mapped the wavelet coefficients in a low-dimension PCs space with the largest variability. The structures of dataset correlation with the storage

time were discovered in the PCs space. The diagnostic spectral information was used as input to build the ANN model, this model shown a very good result for classifying the four classes. It means that NIR spectroscopy can be used to classify rice seed of different storage time nondestructively. And this chemometrics way combined with WT, PCA and ANN is effective to extract diagnostic information and build quantitative discrimination model.

References

1. Donald, D., Everingham, Y., Coomans, D.: Integrated Wavelet Principal Component Mapping for Unsupervised Clustering on Near Infrared Spectra. *Chemometrics and Intelligent Laboratory Systems* **77** (2005) 32-42
2. He, Y., Li, X. L., Deng, X. F.: Discrimination of Varieties of Tea Using Near Infrared Spectroscopy by Principal Component Analysis and BP Model. *Journal of Food Engineering* **79** (2007) 1238-1242
3. Boscaini, E., Mikoviny, T., Wisthaler, A., Hartungen, E.V., Mark, T.D.: Characterization of Wine with PTR-MS. *International Journal of Mass Spectrometry* **239** (2004) 215-219
4. Vannucci, M., Sha, N.J., Brown, P.J.: NIR and Mass Spectra Classification: Bayesian Methods for Wavelet-based Feature Selection. *Chemometrics and Intelligent Laboratory System* **77** (2005) 139-148
5. Cocchi, M., Corbellini, M., Foca, G., Lucisano, M., Pagani, M.A., Tassi, L., Alessandro U.: Classification of Bread Wheat Flours in Different Quality Categories by a Wavelet-based Feature Selection/Classification Algorithm on NIR Spectra. *Analytica Chimica Acta* **544** (2005) 100-107
6. Karoui, R., Mouazen, A.M., Ramon, H., Schoonheydt, R., Baerdemaeker, J.D.: Feasibility Study of Discriminating the Manufacturing Process and Sampling Zone in Ripened Soft Cheeses Using Attenuated Total Reflectance MIR and Fiber Optic Diffuse Reflectance VIS-NIR Spectroscopy. *Food Research International* **39** (2006) 588-597
7. Pontes, M.J.C., Santos, S.R.B., Araujo, M.C.U., Almeida, L.F., Lima, R.A.C., Gaiao, E.N., Souto, U.T.C.P.: Classification of Distilled Alcoholic Beverages and Verification of Adulteration by Near infrared Spectrometry. *Food Research International* **39** (2006) 182-189
8. Dardenne, P., Pierna, J.A.F.: A NIR Data Set Is the Object of a Chemometric Contest at 'Chimiometrie 2004'. *Chemometrics and Intelligent Laboratory System* **80** (2006) 236-242

Selected Problems of Knowledge Discovery Using Artificial Neural Networks

Keith Douglas Stuart¹ and Maciej Majewski²

¹ Polytechnic University of Valencia, Department of Applied Linguistics
Camino de Vera, s/n, 46022 Valencia, Spain
kstuart@idm.upv.es

² Koszalin University of Technology, Faculty of Mechanical Engineering
Raclawicka 15-17, 75-620 Koszalin, Poland
maciej.majewski@tu.koszalin.pl

Abstract. The paper describes the application of an artificial neural network in natural language text reasoning. The task of knowledge discovery in text from a database, represented with a database file consisting of sentences with similar meanings but different lexico-grammatical patterns, was solved with the application of neural networks which recognize the meaning of the text using designed training files. We propose a new method for natural language text reasoning that utilizes three-layer neural networks. The paper deals with recognition algorithms of text meaning from a selected source using an artificial neural network. In this paper we present that new method for natural language text reasoning and also describe our research and tests performed on the neural network.

1 Introduction

For linguistic research, there is a need for consciously created and organized collections of data and information that can be used to carry out knowledge discovery in texts and to evaluate the performance and effectiveness of the tools for these tasks. Knowledge discovery in text is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in unstructured textual data [14,15]. These patterns are unknown, hidden or implicit in semi-structured and unstructured collections of text. Below are some of the kinds of knowledge discovery tasks that many subject disciplines are interested in:

- Identification and retrieval of relevant documents from one or more large collections of documents.
- Identification of relevant sections in large documents (passage retrieval).
- Co-reference resolution, i.e., the identification of expressions in texts that refer to the same entity, process or activity.
- Extraction of entities or relationships from text collections.
- Automated characterization of entities and processes in texts.

- Automated construction of ontologies for different domains (e.g., characterization of medical terms).
- Construction of controlled vocabularies from fixed sets of documents for particular domains.

The need to construct controlled vocabularies for subject domains has meant that terminological extraction from corpora has become an important process in tasks related to knowledge discovery in text [14].

The proposed system for knowledge discovery in text uses neural networks for natural language understanding in Fig. 1.

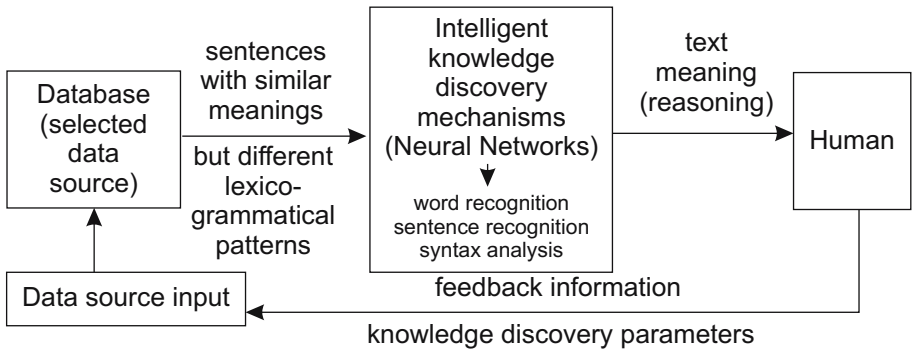


Fig. 1. Steps involved in proposed knowledge discovery in text

The system consists of a selected data source, 3-layer artificial neural networks, network training sets, letter chain recognition algorithms, syntax analysis algorithms, as well as coding algorithms for words and sentences.

2 The State of the Art

Knowledge discovery is a growing field: There are many knowledge discovery methodologies in use and under development. Some of these techniques are generic, while others are domain-specific.

Learning algorithms are an integral part of knowledge discovery. Learning techniques may be supervised or unsupervised. In general, supervised learning techniques enjoy a better success rate as defined in terms of usefulness of discovered knowledge. According to [1,2], learning algorithms are complex and generally considered the hardest part of any knowledge discovery technique. Machine discovery is one of the earliest fields that has contributed to knowledge discovery [4]. While machine discovery relies solely on an autonomous approach to information discovery, knowledge discovery typically combines automated approaches with human interaction to assure accurate, useful, and understandable results.

There are many different approaches that are classified as knowledge discovery techniques [16]. There are quantitative approaches, such as the probabilistic and statistical approaches. There are approaches that utilize visualization

techniques. There are classification approaches such as Bayesian classification, inductive logic, data cleaning/pattern discovery, and decision tree analysis [2,4]. Other approaches include deviation and trend analysis, genetic algorithms, neural networks, and hybrid approaches that combine two or more techniques.

The probabilistic approach family of knowledge discovery techniques utilizes graphical representation models to compare different knowledge representations [7]. These models are based on probabilities and data independencies. The statistical approach uses rule discovery and is based on data relationships. An inductive learning algorithm can automatically select useful join paths and attributes to construct rules from a database with many relations [3]. This type of induction is used to generalize patterns in the data and to construct rules from the noted patterns.

Classification is probably the oldest and most widely-used of all the knowledge discovery approaches [3,7,16]. This approach groups data according to similarities or classes. There are many types of classification techniques e.g. the Bayesian approach, pattern discovery and data cleaning, and the decision tree approach. Pattern detection by filtering important trends is the basis for the deviation and trend analysis approach. Deviation and trend analysis techniques are normally applied to temporal databases [4,6].

Neural networks may be used as a method of knowledge discovery. Neural networks are particularly useful for pattern recognition, and are sometimes grouped with the classification approaches. A hybrid approach to knowledge discovery combines more than one approach and is also called a multi-paradigmatic approach. Although implementation may be more difficult, hybrid tools are able to combine the strengths of various approaches. Some of the commonly used methods combine visualization techniques, induction, neural networks, and rule-based systems to achieve the desired knowledge discovery. Deductive databases and genetic algorithms have also been used in hybrid approaches.

3 Method Description

In the proposed knowledge discovery system shown in Fig. 2, sentences are extracted from the database. The separated words of the text are the input signals of the neural network for recognizing words [5]. The network has a training file containing word patterns. The network recognizes words as the sentence components, which are represented by its neurons in Fig. 3. The recognized words are sent to the algorithm for coding words [12]. Then, the coded words are transferred to the sentence syntax analysis module. It is equipped with the algorithm for analysing and indexing words. The module indexes words properly and then they are sent to the algorithm for coding sentences [13]. The commands are coded as vectors and they are input signals of the sentence recognition module using neural networks. The module uses the 3-layer Hamming neural network in Fig. 4, either to recognize the sentence in order to find out its meaning or just does not recognize the sentence. The neural network is equipped with a training file containing patterns of possible sentences whose meanings are understood.

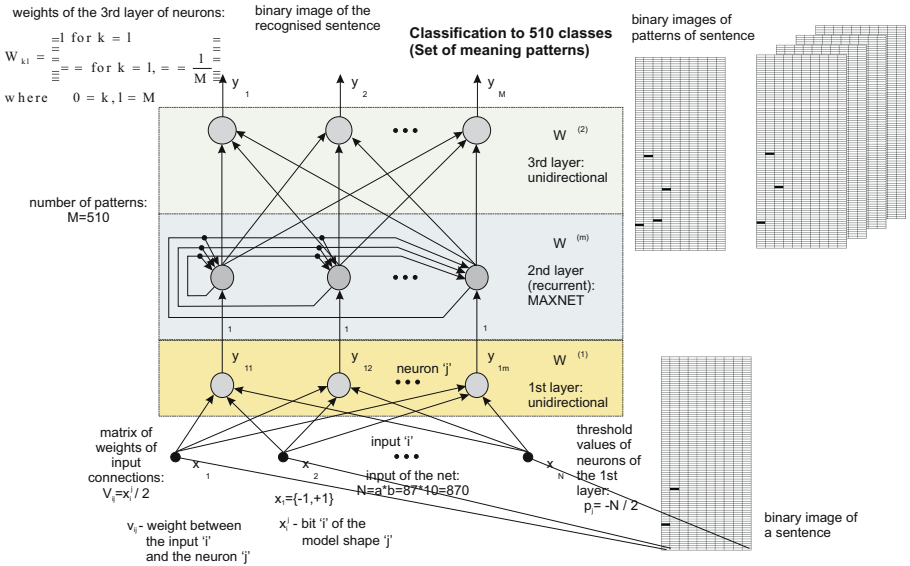


Fig. 4. Scheme of the 3-layer neural network for sentence recognition

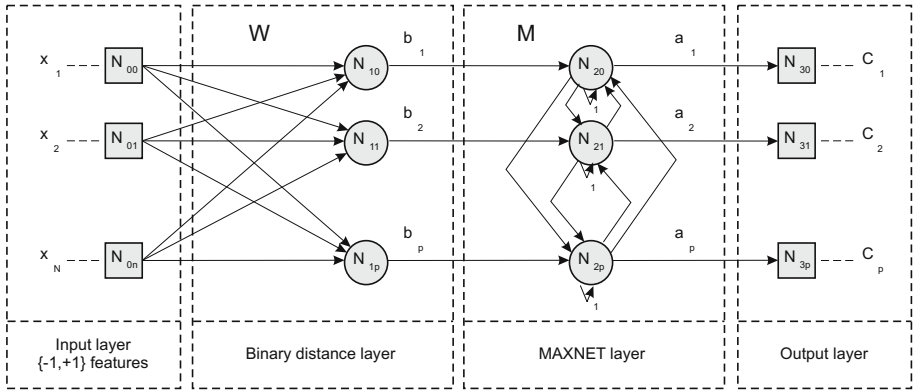


Fig. 5. Structure of the Hamming neural network as a classifier-expert module

Because of the binary input signals, the Hamming neural network is chosen in Fig. 5 which directly realizes the one-nearest-neighbour classification rule [9,10,11]. Each training data vector is assigned a single class and during the recognition phase only a single nearest vector to the input pattern x is found and its class C_i is returned. There are two main phases of the operation of the expert-network: training (initialization) and classification. Training of the binary neural network consists of copying reference patterns into the weights of the matrix W_{pn} , as follows (II):

$$w_i = x_i, \quad 1 \leq i \leq p \tag{1}$$

where p is the number of input patterns-vectors x , each of the same length n , w_i is the i -th row of the matrix W of dimensions p rows and n columns. For given n the computation time is linear with the number of input patterns p .

The goal of the recursive layer N_2 is selection of the winning neuron. The characteristic feature of this group of neurons is a self connection of a neuron to itself with a weight $m_{ii}=1$ for all $1 \leq i \leq p$, whereas all other weights are kept negative. Initialization of the N_2 layer consists in assigning negative values to the square matrix M_{pp} except the main diagonal. Originally Lippmann proposed initialization [8] (2):

$$\begin{aligned} m_{kl} &= -(p-1)^{-1} + \xi_{kl} \quad \text{for } k \neq l, \quad 1 \text{ for } k = l \\ \text{where } &1 \leq k, l \leq p, p > 1 \end{aligned} \tag{2}$$

where ξ is a random value for which $|\xi| \ll (p-1)^{-1}$. However, it appears that the most efficient and still convergent solution is to set equal weights for all neurons N_2 which are then modified at each step during the classification phase, as follows (3):

$$\begin{aligned} m_{kl} &= \varepsilon_k(t) = -(p-t)^{-1} \quad \text{for } k \neq l, \quad 1 \text{ for } k = l \\ \text{where } &1 \leq k, l \leq p, p > 1 \end{aligned} \tag{3}$$

where t is a classification time step. In this case the convergence is achieved in $p-1-r$ steps, where $r > 1$ stands for the number of nearest stored vectors in W .

In the classification phase, the group N_1 is responsible for computation of the binary distance between the input pattern z and the training patterns already stored in the weights W . Usually this is the Hamming distance (4):

$$b_i(z, W) = 1 - n^{-1} D_H(z, w_i), \quad 1 \leq i \leq p \tag{4}$$

where $b_i \in [0, 1]$ is a value of an i -th neuron in the N_1 layer, $D_H(z, w_i) \in \{0, 1, \dots, n\}$ is a Hamming distance of the input pattern z and the i -th stored pattern w_i (i -th row of W).

In the classification stage, the N_2 layer operates recursively to select one winning neuron. This process is governed by the following equation (5):

$$a_i[t+1] = \varphi \left(\sum_{j=1}^n m_{ij} a_j[t] \right) = \varphi \left(a_i[t] + \sum_{j=1, i \neq j}^n m_{ij} a_j[t] \right) \tag{5}$$

where $a_i[t]$ is an output of the i -th neuron of the N_2 layer at the iteration step t , φ is a threshold function given as follows (6):

$$\varphi(x) = x \quad \text{for } x > 0, \quad 0 \text{ otherwise} \tag{6}$$

Depending on the chosen scheme (2)-(3) of the m_{ij} weights in (5), we obtain different dynamics of the classification stage. The iterative process (5) proceeds up to a point where only one neuron has value different than 0 - this neuron is a winner.

4 Research Results

The dataset for the tests carried out contained a database of 1500 sentences, files consisting of 522 letter chains, 87 word training patterns and 510 sentence meaning training patterns. The first test measured the performance of the

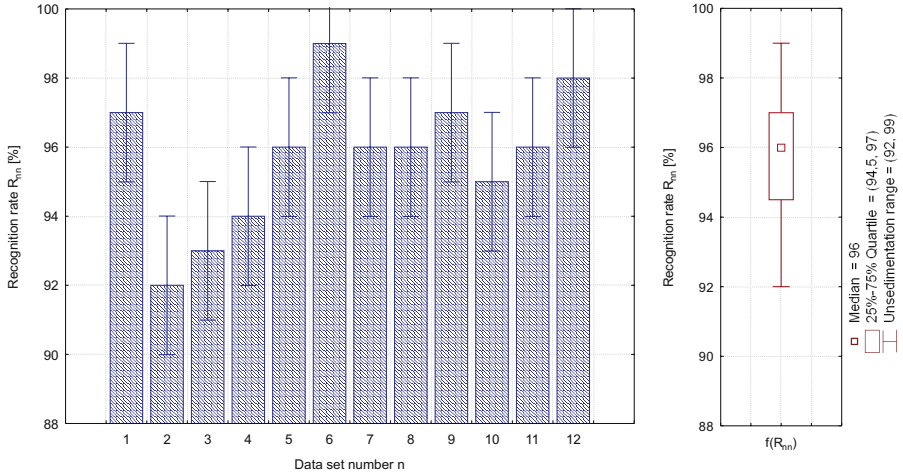


Fig. 6. Sentence meaning recognition rate as a set of words recognised earlier

sentence meaning recognition with the sentence recognition module using artificial neural networks as a set of words recognised earlier in Fig. 6.

As shown in Fig. 7a, the ability of the implemented neural network for word recognition to recognise the word depends on the number of letters. The neural network requires a minimum number of letters of the word being recognized as its input signals. As shown in Fig. 7b, the ability of the neural network for sentence meaning recognition to recognise the sentence depends on the number of sentence component words. Depending on the number of component words of the sentence, the neural network requires a minimum number of words of the given sentence as its input signals.

5 Conclusions and Perspectives

Knowledge discovery is a rapidly expanding field with promise for great applicability. Knowledge discovery purports to be the new database technology for the coming years. The need for automated discovery tools had caused an explosion in research.

The motivation behind using the binary neural networks in knowledge discovery comes from the possible simple binarization of words and sentences, as well as very fast training and run-time response of this type of neural networks.

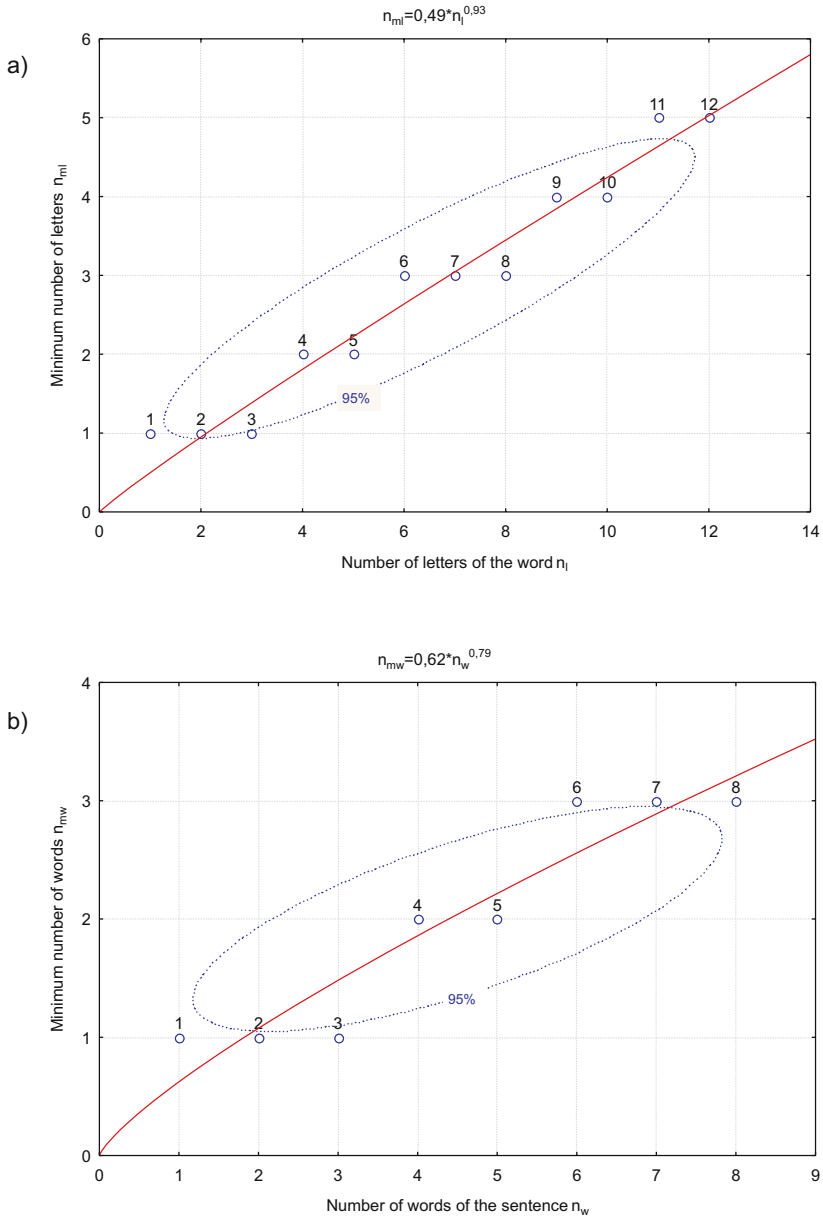


Fig. 7. a) Sensitivity of word recognition: minimum number of letters of the word being recognized to number of word component letters; b) Sensitivity of sentence meaning recognition: minimum number of words of the sentence being recognized to number of sentence component words

Application of binary neural networks allows for recognition of sentences in natural language with similar meanings but different lexico-grammatical patterns, which can be encountered in documents, texts, vocabularies and databases. The presented methods can be easily extended.

It is anticipated that commercial database systems of the future will include knowledge discovery capabilities in the form of intelligent database interfaces. Some types of information retrieval may benefit from the use of knowledge discovery techniques. Due to the potential applicability of knowledge discovery in so many diverse areas there are growing research opportunities in this field.

References

1. Berry, M., Linoff, G.: *Mastering Data Mining*. John Wiley & Sons (2000)
2. Dunham, M.H.: *Data Mining Introductory and Advanced Topics*. Prentice Hall (2003)
3. Han, J., Kamber, M.: *Data Mining: Concepts and Techniques*. 2nd edition. Morgan Kaufmann (2006)
4. Hand, D.J., Mannila, H., Smyth, P.: *Principles of Data Mining*. MIT Press (2000)
5. Kacalak, W., Douglas Stuart, K., Majewski, M.: *Intelligent Natural Language Processing*. International Conference on Natural Computation ICNC2006. Xi'an, China. *Lectures Notes in Artificial Intelligence* **4221** (2006) 584-587
6. Kantardzic, M.: *Data Mining: Concepts, Models, Methods, and Algorithms*. Wiley-IEEE Press (2002)
7. Larose, D. T.: *Discovering Knowledge in Data: An Introduction to Data Mining*. John Wiley & Sons, Inc. (2004)
8. Lippman, R.: *An Introduction to Computing with Neural Nets*. *IEEE Trans. Acoustic, Speech, and Signal Processing* **3** (1987) 4-22
9. Majewski, M., Kacalak, W.: *Intelligent Layer of Two-Way Voice Communication of the Technological Device with the Operator*. *Lectures Notes in Artificial Intelligence* **3683** (2005) 930-936
10. Majewski, M., Kacalak, W.: *Intelligent Human-Machine Voice Communication System*. *Engineering Mechanics International Journal for theoretical and applied mechanics* **12** (2005) 193-200
11. Majewski, M., Kacalak, W.: *Automatic Recognition and Evaluation of Natural Language Commands*. *Lecture Notes in Computer Science* **3973** (2006) 1155-1160
12. Majewski, M., Kacalak, W.: *Natural Language Human-Machine Interface using Artificial Neural Networks*. *Lecture Notes in Computer Science* **3973** (2006) 1161-1166
13. Majewski, M., Kacalak, W., Douglas Stuart, K.: *Selected Problems of Intelligent Natural Language Processing*. *International Journal of Research on Computing Science* **20** (2006) (ISSN 1665-9899)
14. Stuart, K.: *Corpus Linguistics: Theory, Method and Applications*. XXIV International AESLA Conference, Language Learning, Language Use and Cognitive Modelling: Applied Perspectives across Disciplines. Madrid, Spain (2006)
15. Stuart, K.: *A Software Application for Text Analysis*. International Conference on Engineering Education, Valencia, Spain (2003)
16. Tan, P.N., Steinbach, M., Kumar, V.: *Introduction to Data Mining*. Pearson Addison Wesley (2005)

An Intelligent Differential Evolution Algorithm for Designing Trading-Ratio System of Water Market

Ying Liu¹, Bo Liu², Jikun Huang¹, Yunhua Wu¹, Ling Wang², and Yihui Jin²

¹ Center for Chinese Agricultural Policy, Institute of Geographical Sciences and Natural Resource Research, Chinese Academy of Sciences, Beijing 100101, China
Liuying.04b@igsnr.ac.cn

² Department of Automation, Tsinghua University, Beijing 100084, China
Liub01@mails.tsinghua.edu.cn

Abstract. As a novel optimization technique, neural network based optimization has gained much attention and some applications during the past decade. To enhance the performance of Differential Evolution Algorithm (DEA), which is an evolutionary computation technique through individual improvement plus population cooperation and competition, an intelligent Differential Evolution Algorithm (IDEA) is proposed by incorporating neural network based search behaviors into classic DEA. Firstly, DEA operators are used for exploration by updating individuals so as to maintain the diversity of population and speedup the search process. Secondly, a multi-layer feed-forward neural network is employed for local exploitation to avoid being trapped in local optima and improve the convergence of the IDEA. Simulation results and comparisons based on well-known benchmarks and optimal designing of trading-ratio system for water market demonstrate that the IDEA can effectively enhance the searching efficiency and greatly improve the searching quality.

1 Introduction

Neural network (NN) has shown to be an effective optimization technique for complex function optimization problems. Since the pioneer work of Tank and Hopfield's in 1986 [1], NN-based searching algorithms have aroused intense interests. However, simple NN-based search often needs a large number of iterations to reach the global optimum and are sensitive to the initial conditions.

Recently, a new evolutionary technique, differential evolution algorithm (DEA), has been proposed [2] as an alternative to genetic algorithm (GA) [3] and particle swarm optimization (PSO) [4] for unconstrained continuous optimization problems. Although the original objective in the development of DEA was for solving the Chebyshev polynomial problem, it has been found to be an efficient and effective solution technique for complex functional optimization problems. In a DEA system, a population of solutions is initialized randomly, which is evolved to find optimal solutions through the mutation, crossover, and selecting operation procedures. Compared with GA and PSO, DEA has some attractive characteristics. It uses simple differential operator to create new candidate solutions and one-to-one competition scheme to

greedily select new candidate, which work with real numbers in natural manner and avoid complicated generic searching operators in GA. It has memory, so knowledge of good solutions is retained in current population, whereas in GA, previous knowledge of the problem is destroyed once the population changes and in PSO, a secondary archive is needed. It also has constructive cooperation between individuals, individuals in the population share information between them. Due to the simple concept, easy implementation and quick convergence, nowadays DEA has attracted much attention and wide applications in different fields mainly for various continuous optimization problems [5].

To the best of our knowledge, there is no any published work for dealing with numerical optimization by hybridizing DEA and NN-based local search. And there is also no research on DEA for optimal designing of trading-ratio system for water market. In this paper, an intelligent DEA (IDEA) is proposed by incorporating differential evolution algorithm (DEA) and neural network based search behaviors into classic DEA. Firstly, DEA operators are used for exploration by updating individuals so as to maintain the diversity of population and speedup the search process. Secondly, a multi-layer feed-forward neural network is employed for local exploitation to avoid being trapped in local optima and improve the convergence of the IDEA. Simulation results and comparisons based on well-known benchmarks and optimal designing of trading-ratio system for water market demonstrate that the IDEA can effectively enhance the searching efficiency and greatly improve the searching quality.

2 Formulation of Water Right Trading System

In this section we design a water right trading system [6], which could avoid third-party effects as reducing the instream flow of the intermediate stream and downstream. The initial water right is distributed to meet the minimum instream flow standard which the authority sets exogenously. Then water users may trade water right according to a set of given trading ratios. The trading ratios are set exogenously based on return flow parameters to ensure that any trading would not violate minimum instream flow standards. The trading rules can take care of the location effects of users and may achieve social optimum [7].

The social planner’s problem is to maximize the social benefits of all traders:

$$\text{Max } \sum B_i(q_i) = \sum a_i q_i^{b_i}, \tag{1}$$

$$\text{s.t.: } V_0 - \sum_{j=1}^i (1 - R_j) iniq_j = F_i, \tag{2}$$

$$q_i \leq iniq_i + \sum_{j=1}^{i-1} \frac{1 - R_j}{1 - R_i} tq(j, i) - \sum_{j=i+1}^n tq(i, j), \quad i = 1, \dots, n, \tag{3}$$

$$\frac{V_0 - F_i}{1 - R_i} \geq q_i \geq 0, \tag{4}$$

$$tq(\cdot) \geq 0, \tag{5}$$

where q_j is quantitative water use of user j ; $B_i(q_i)$ is the utility function of user i with water quantity q_j ; a_i and b_i are the given parameters; V_0 is the headstream water supply; R_i is the return flow; $iniq_i$ denotes the initial endowment, which is derived from equation (2); F_i is the minimum instream flow requirement of user i ; $(1 - R_i)/(1 - R_j)$ is the trading ratio between down stream user j and up stream user i ; $tq(i, j)$ represent the water quantity transformed from i to j . Equation (2) denotes that the initial endowment should satisfy minimum instream flow requirement. Equation (3) requires that each user’s trading balance should under the trading rules.

3 Introduction on DEA and NN-Based Local Search

3.1 DEA

DEA is a population-based evolutionary computation technique, which uses simple differential operator to create new candidate solutions and one-to-one competition scheme to greedily select new candidate. The theoretical framework of DEA is very simple and DEA is easy to be coded and implemented with computer. Besides, it is computationally inexpensive in terms of memory requirements and CPU times. Thus, nowadays DEA has attracted much attention and wide applications in various fields [5].

In DEA, it starts with the random initialization of a population of individuals in the search space and works on the cooperative behaviors of the individuals in the population. Therefore, it finds the global best solution by utilizing the distance and direction information according to the differentiations among population. However, the searching behavior of each individual in the search space is adjusted by dynamically altering the differentiation’s direction and step length in which this differentiation performs.

The i -th individual in the d -dimensional search space at generation t can be represented as $X_i(t) = [x_{i,1}, x_{i,2}, \dots, x_{i,d}]$, ($i = 1, 2, \dots, NP$, where NP denotes the size of the population). At each generation t , the *mutation* and *crossover* operators are applied on the individuals, and a new population arises. Then, *selection* takes place, and the corresponding individuals from both populations compete to comprise the next generation.

For each target individual $X_i(t)$, according to the *mutation* operator, a *mutant vector* $V_i(t+1) = [v_{i,1}(t+1), \dots, v_{i,d}(t+1)]$ is generated by adding the weighted difference

between a defined number of individuals randomly selected from the previous population to another individual, which is described by the following equation:

$$V_i(t+1) = X_{best}(t) + F(X_{r_1}(t) - X_{r_2}(t)), \tag{6}$$

where $r_1, r_2 \in \{1, 2, \dots, N\}$ are randomly chosen and mutually different and also different from the current index i . $F \in (0, 2]$ is constant called scaling factor which controls amplification of the differential variation $X_{r_1}(t) - X_{r_2}(t)$, and NP is at least 4 so that the mutation can be applied. $X_{best}(t)$, the base vector to be perturbed, is the best member of the current population so that the best information could be shared among the population.

After the *mutation* phase, the *crossover* operator is applied to increase the diversity of the population. Thus, for each target individual $X_i(t)$, a *trial vector* $U_i(t+1) = [u_{i,1}(t+1), \dots, u_{i,d}(t+1)]$ is generated by the following equation:

$$u_{i,j}(t+1) = \begin{cases} v_{i,j}(t+1), & \text{if } (rand(j) \leq CR) \text{ or } j = randn(i), \\ x_{i,j}(t), & \text{otherwise.} \end{cases} \quad j = 1, 2, \dots, d \tag{7}$$

where $rand(j)$ is the j -th independent random number uniformly distributed in the range of $[0, 1]$. $randn(i)$ is a randomly chosen index from the set $\{1, 2, \dots, d\}$. $CR \in [0, 1]$ is constant called crossover parameter that controls the diversity of the population.

Following the *crossover* operation, the *selection* arises to decide whether the *trial vector* $U_i(t+1)$ would be a member of the population of the next generation $t+1$. For a minimum optimization problem, $U_i(t+1)$ is compared to the initial target individual $X_i(t)$ by the following one to one based greedy selection criterion:

$$X_i(t+1) = \begin{cases} U_i(t+1), & \text{if } F(U_i(t+1)) < F(X_i(t)), \\ X_i(t), & \text{otherwise.} \end{cases} \tag{8}$$

where F is the objective function under consideration, $X_i(t+1)$ is the individual of the new population. The procedure described above is considered as the standard version of DEA, and it is denoted as DE/best/1/bin. Several variants of DEA have been proposed, depending on the selection of the base vector to be perturbed, the number and selection of the differentiation vectors and the type of crossover operators [5].

The key parameters in DEA are NP (size of population), F (scaling factor) and CR (crossover parameter). Proper configuration of the above parameters would achieve good tradeoff between the global exploration and the local exploitation so as to increase the convergence velocity and robustness of the search process. Some basic

principles have been given for selecting appropriate parameters for DEA [5]. In general, the population size NP is choosing from $5 \cdot d$ to $10 \cdot d$ (number of dimension). F and CR lies in the range of $[0.4, 1.0]$ and $[0.1, 1.0]$ respectively. The procedure of DEA is summarized in Fig.1.

For each individual i in the population, initialize $X_i(0)$ randomly, $g = 0$

Repeat until a stopping criterion is satisfied:

Mutation step:

$$v_{i,j} = x_{i,j}(g) + \lambda(x_{best,j}(g) - x_{i,j}(g)) + F(x_{r1,j}(g) - x_{r2,j}(g)), \\ \forall i \leq N \text{ and } \forall j \leq d .$$

Crossover step:

$$u_{i,j} = \begin{cases} v_{i,j} & \text{with probability } CR \\ x_{i,j}(g) & \text{with probability } 1 - CR \end{cases}, \forall i \leq N \text{ and } \forall j \leq d .$$

Selection step:

$$\text{if } f(U_i) < f(X_i(g)) \text{ then } x_i(g+1) = u_i \text{ else } x_i(g+1) = x_i(g), \forall i \leq N$$

$$g = g + 1.$$

Fig. 1. Procedure of DEA algorithm

3.2 NN-Based Local Search

In addition, to further improve the solution quality and convergence speed of DEA, a NN, trained with the DEA results, is used to guide the DEA search for a better optimum value in the subsequent generations. In this paper, we uses a three-layer feed-forward NN with hyperbolic tangent sigmoid transfer function and Powell-Beale's learning algorithm, which has shown to be very efficient in improving the convergence of the IDEA.

4 Intelligent DEA (IDEA)

By hybridizing DEA and NN, a two-phased iterative strategy named intelligent DEA (IDEA) is proposed, in which DEA operators are used for exploration by updating individuals so as to maintain the diversity of population and speedup the search process, and a multi-layer feed-forward neural network is employed for local exploitation to avoid being trapped in local optima and improve the convergence of the IDEA. The procedure of IDEA is illustrated in Fig.2.

It can be seen that DEA operators are used for exploration by updating in the population, and NN-based local search is applied for exploitation by locally modified the best individual resulted by DEA.

Step 1: Randomly initialize the population of individual for DEA, where each individual contains n variables.

Step 2: Evaluate the objective values of all individuals, and determine X_{best} which has the best objective value.

Step 3: Perform mutation operation for each individual according to Eq. (6) in order to obtain each individual's mutant counterpart.

Step 4: Perform crossover operation between each individual and its corresponding mutant counterpart according to Eq. (7) in order to obtain each individual's trial individual.

Step 5: Evaluate the objective values of the trial individuals.

Step 6: Perform selection operation between each individual and its corresponding trial counterpart according to Eq. (8) so as to generate the new individual for the next generation.

Step 7: Determine the best individual of the current new population with the best objective value. If the objective value is better than the objective value of X_{best} , then update X_{best} and its objective value with the value and objective value of the current best individual.

Step 8: Train a neural network with the objective function values of each individual (as an input) and the individual (as an output).

Step 9: Predict a combination of the variables which could possibly produce an objective function value slightly better than the X_{best} objective value by the above trained network.

Step 10: Replace the worst individual's by the objective value and the corresponding variables found in step 9.

Step 11: Update the X_{best} and its corresponding value.

Step 12: If a stopping criterion is met, then output X_{best} and its objective value; otherwise go back to Step (3).

Fig. 2. Procedure of IDEA algorithm

5 Numerical Simulations and Comparisons

5.1 Comparisons of IDEA, DEA, GA, and PSO on Benchmarks

To test the performance of the proposed algorithms, four famous benchmark optimization problems [3] are used, which are described in the Appendix. Firstly, we compare the IDEA with the standard DEA [5], PSO [8], and GA [9]. In both DEA and IDEA, $NP = 20$, $CR = 0.5$, $\lambda = F = 0.8$. In PSO, the population size is 20, c_1 and c_2 are set to 2.0, and v_{max} is clamped to be 15% of the search space, and uses a linearly varying inertia weight over the generations, varying from 1.2 at the beginning of the search to 0.2 at the end. The GA [9] with a population of 40 is real-valued with random initialization, tournament selection with tournament size four, a 2-element elitist strategy, arithmetic crossover with random weight, Gaussian mutation with

distribution $N(0,a) = \frac{1}{\sqrt{2\pi a}} \exp(-x^2/2a)$, where a is linearly decreasing from 1 to

0. Crossover and mutation probabilities are set as 0.8 and $1/n$ respectively, where n is the dimension of the problem. Fixed the total number of function evaluation as 2000, Table 1 lists the average best function value and the standard deviation of 50 independent runs.

Table 1. Comparisons of IDEA, DEA, GA, and PSO

F	IDEA	DEA	GA	PSO
F ₁	0 ± 0	0 ± 0	6.31E-7 ± 6.41E-13	1.27E-3 ± 4E-3
F ₂	2.14E-172 ± 0	1.97E-104 ± 6.00E-104	2.39E-4 ± 6.56E-8	3.83E-7 ± 6.77E-7
F ₃	1.97E-31 ± 1.94E-61	1.51E-30 ± 3.32E-30	0.132 ± 0.0089	0.257 ± 0.346
F ₄	680.63 ± 1.50E-8	680.63 ± 0.005	684.78 ± 3.277	684.44 ± 2.056

From Table 1, it can be seen that the results of IDEA are much closer to the theoretical optima, and IDEA is superior to DEA, GA, and PSO in term of searching quality and derivation of the results. So, it is concluded that IDEA is more effective and it is more robust on initial conditions.

5.2 Optimal Designing of the Water Right Trading System

In this section we use the proposed IDEA to design the trading-ratio system for water market, which is to maximize the social benefits of all traders. For the model proposed in Section 2, the parameters are set as follows: the number of water users is four; V_0 is equal to 150; $a_i = (30, 35, 55, 45)$; $b_i = (0.18, 0.28, 0.14, 0.22)$; $R_i = (0.5, 0.4, 0.3, 0.5)$; $F_i = (40, 35, 30, 20)$ ($i=1, 2, 3, 4$). By using our proposed IDEA, the maximum benefits of all traders is $\sum B_i(q_i) = 385.325$ under the decision variables $q_i = (31.49, 90.32, 27.40, 81.76)$.

6 Conclusions

To our knowledge, this is the first report of hybridizing differential evolution algorithm and neural network-based search to propose an effective and efficient optimization algorithm for numerical functions and optimal designing of optimal designing of trading-ratio system for water market. The proposed approach not only performs exploration by using the population-based evolutionary searching ability of DEA, but also performs exploitation by using the NN-based local searching behavior. The proposed IDEA is superior in term of searching quality, efficiency and robustness on initial conditions. The future work is to apply the IDEA for some real-life optimization problems.

Acknowledgement

This work is supported by National Natural Science Foundation of China (Grant No. 60204008, 60374060 and 60574072) and National 973 Program (Grant No. 2002CB312200).

References

1. Tank, D.W., Hopfield, J.J.: Simple 'Neural' Optimization Network: An A/D Converter, Signal Decision Circuit and a Linear Programming Circuit. *IEEE Trans. Circuits and System* **33** (1986) 533-541
2. Storn, R., Price, K.: Differential Evolution - A Simple and Efficient Heuristic for Global Optimization Over Continuous Spaces. *J Global Optim* **11** (1997) 341-59
3. Wang, L.: *Intelligent Optimization Algorithms with Applications*. 4th edn. Tsinghua University & Springer Press, Beijing (2001)
4. Liu, B., Wang, L., Jin, Y.H., Huang, D.X.: Advances in Particle Swarm Optimization Algorithm. *Control and Instruments in Chemical Industry* **32** (2005) 1-6
5. Price, K., Storn, R.: Differential Evolution Homepage. The URL of which is: <http://www.ICSI.Berkeley.edu/~storn/code.html>
6. Hung, M.F., Shaw, D.: A Trading-Ratio System for Trading Water Pollution Discharge Permits. *Journal of Environmental Economics and Management* (in press)
7. Shaw, D., Liu, Y., Hong, M.F.: A Trading-ratio System for Water Market. *Economic Research* **39** (2004) 69-77
8. Kennedy, J., Eberhart, R.C., Shi, Y.: *Swarm Intelligence*. Morgan Kaufmann Publishers, San Francisco (2001)
9. Goldberg, D.E.: *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley, MA (1989)

Appendix: Four Benchmark Optimization Problems Used

F_1 : *Rastrigin's function* ($n=2$)

$$F_1(x) = 10n + \sum_{i=1}^n (x_i^2 - 10\cos(2\pi x_i)), \quad -5.12 \leq x_i \leq 5.12.$$

The global minimum is equal to 0 and the minimum point is (0, 0).

F_2 : *Sphere function* ($n=5$)

$$F_2 = \sum_{i=1}^n x_i^2, \quad -5 \leq x_i \leq 5.$$

The global minimum is equal to 0 and the minimum point is (0, 0, 0, 0, 0).

F_3 : *Rosenbrock's function* ($n=3$)

$$F_3 = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2], \quad -5.12 \leq x_i \leq 5.12.$$

The global minimum is equal to 0 and the minimum point is (1, 1, 1).
 F_4 : *Constrained function* ($n=7$)

$$F_4 = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 \\ + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7.$$

Subject to nonlinear constraints:

$$127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 \geq 0;$$

$$282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \geq 0;$$

$$196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 \geq 0;$$

$$-4x_1^2 - 3x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \geq 0;$$

$$-10 \leq x_i \leq 10.$$

The global minimum is equal to 680.6300573 and the minimum point is (2.330499, 1.951372, -0.4775414, 4.365726, -0.624487, 1.038131, 1.594227).

Neural Network-Based H_∞ Filtering for Nonlinear Jump Systems*

Xiaoli Luan and Fei Liu

Institute of Automation, Southern Yangtze University
Wuxi 214122, P.R. China
fliu@thmz.com

Abstract. This paper addresses the problem of designing a Markovian H_∞ filter for a class of nonlinear stochastic Markovian jump systems. Firstly, neural networks are employed to approximate the nonlinearities in the different jump modes. Secondly, the overall system is represented by the mode-dependent linear difference inclusion (LDI). Then, a neural network-based Markovian H_∞ filter is developed using the stochastic Lyapunov-Krasovskii stability theory under some linear matrix inequality (LMI) constraints. Finally, a numerical example is worked out to show the usefulness of the theoretical results.

1 Introduction

Stochastic Markovian jump systems (MJS), which are appropriate to describe dynamic systems with structures and parameters varying abruptly in a random way, have gained a great deal of attention [1]. Stochastic MJS can be regarded as a special class of hybrid systems with many operation modes, which are determined by a Markov chain taking values in a finite set [2, 3]. In the past decade, a lot of control issues have been investigated for linear MJS, which cover a large variety of problems such as H_∞ control [4], output feedback control [5], passive control [6], guaranteed cost control [7], etc. As for the filtering problems concerning linear MJS, a number of results are also available. For example, the Kalman filtering problem was studied in [8] and [9], where the results were given in terms of coupled Riccati equations. Based on linear matrix inequality (LMI) approach, a robust H_∞ filter was designed in [10] and reduced-order H_∞ filtering problem was tackled in [11] with matrix rank constraints. It is worth noticing that all above results are limited to linear MJS. In practice, however, a nonlinear MJS may be more reasonable to account for the nonlinear structural changes. Due to its complexity, until now, the filtering problems commence gaining initial attention, see e.g. [12]. Compared with linear cases, there have been few papers dealing with filter design problems for general nonlinear MJS. This situation motivates us to study the filtering problem for a class of nonlinear MJS.

* This work is supported by The National Natural Science Foundation of China (NSFC: 60574001) and by Program for New Century Excellent Talents in University (NCET-05-0485).

In recent years, neural network has been widely used in nonlinear area owing to its universal approximation capability. Details concerning neural networks and their relations to the deterministic linear and nonlinear systems can be found everywhere [13, 14]. In spite of these successes, there are many basic issues remain to be addressed. One of them is how to achieve a systematic design that guarantees closed-loop stability and performance [15]. Recently, a class of multi-layer neural networks that admit a linear difference inclusion (LDI) state-space representation to approximate a deterministic nonlinear system has been proposed in [16, 17]. Based on LDI model, some systematic model-based neural network control design methodologies have been developed [18].

This paper contributes to develop a Markovian H_∞ filter for a class of nonlinear stochastic MJS based on neural networks. The problem we address is the design of Markovian H_∞ filter such that the resulting error system achieves asymptotical stability and the H_∞ gain from the noise signal to the filtering error remains bounded by a prescribed value. To solve this problem, an LMI approach is developed and sufficient conditions for the solvability are obtained. The desired filter can be constructed through a convex optimization problem, which can be efficiently handled by using standard numerical algorithm.

2 Problem Formulation and Analysis

Consider a class of nonlinear MJS described as follows:

$$\begin{aligned} \dot{x}(t) &= A(r_t)x(t) + B(r_t)w(t) + f(x(t), r_t), \\ y(t) &= C(r_t)x(t) \\ z(t) &= L(r_t)x(t) \end{aligned} \tag{1}$$

where $x(t) \in \mathbf{R}^n$ is the state vector, $y(t) \in \mathbf{R}^m$ is measured output vector, $z(t) \in \mathbf{R}^p$ is the signal to be estimated, $w(t) \in \mathbf{L}_2^n[0, \infty)$ is the disturbance input vector, $f(\cdot) : \mathbf{R}^n \mapsto \mathbf{R}^n$ is continuous nonlinear mapping. $A(r_t)$, $B(r_t)$, $C(r_t)$, $L(r_t)$ are mode-dependent matrices with appropriate dimensions, r_t represents a continuous-time discrete-state Markov process with values in a finite set $S = \{1, 2, \dots, s\}$ with transition probability matrix $\Pi = [\pi_{ij}]$ given by

$$\pi_{ij} = P(r_{t+\Delta t} = j | r_t = i) = \begin{cases} \pi_{ij}\Delta t + o(\Delta t), & \text{if } i \neq j \\ 1 + \pi_{ii}\Delta t + o(\Delta t), & \text{if } i = j \end{cases} \tag{2}$$

where π_{ij} is the transition rate from mode i to j , and $\Delta t > 0$. For $i \neq j$,

$$\pi_{ij} \geq 0, \quad -\pi_{ii} = \sum_{j=1, j \neq i}^s \pi_{ij}$$

For presentation convenience, when $r_t = i$, we denote $f(x(t), r_t)$, $A(r_t)$, $B(r_t)$, $C(r_t)$ and $L(r_t)$ as $f_i(x(t))$, A_i , B_i , C_i and L_i respectively.

For each mode i , the nonlinear function $f_i(x(t))$ can be approximated arbitrarily well on a compact interval by single hidden layer neural networks $N_i(x(t), W_{i1}, W_{i2})$ described as following

$$N_i(x(t), W_{i1}, W_{i2}) = \Psi_{i2}[W_{i2}\Psi_{i1}[W_{i1}x(t)]] \tag{3}$$

where for $r = 1, 2$, the activation function vector is defined as

$$\Psi_{ir}[v] = [\varphi_{ir}(v_1), \varphi_{ir}(v_2), \dots, \varphi_{ir}(v_{n_i})]^T$$

with

$$\varphi_{ir}(v) = \lambda \left(\frac{1 - e^{-v/q}}{1 + e^{-v/q}} \right) \quad q, \lambda > 0$$

and W_{ir} denotes the connecting weights matrices need to be trained by back propagation learning algorithm, n_i is the neurons of the i th layer. According to the neural network theory, for a set of error upper bounds $\rho_i > 0$, there exist optimal approximation weights defined as W_{i1}^*, W_{i2}^* such that

$$\|f_i(x(t)) - N_i(x, W_{i1}^*, W_{i2}^*)\| \leq \rho_i \|x(t)\| \tag{4}$$

While the maximum and minimum values $S_i(0, \varphi_{ir})$ and $S_i(1, \varphi_{ir})$ of activation function $\varphi'_{ir}(v)$ are defined, respectively, as follows:

$$S(k, \varphi_{ir}) = \begin{cases} \min_v \frac{\partial \varphi_j(v)}{\partial (v)}, & k = 0 \\ \max_v \frac{\partial \varphi_j(v)}{\partial (v)}, & k = 1 \end{cases} \tag{5}$$

It is direct to represent $\varphi_{ir}(v)$ as

$$\varphi_{ir}(v) = h_i(0)S_i(0, \varphi_{ir}) + h_i(1)S_i(1, \varphi_{ir}) \tag{6}$$

where $h_i(k), k = 0, 1$ is a set of positive real number associated with φ_{ir} satisfying $h_i(k) > 0$ and $h_i(0) + h_i(1) = 1$.

Denote a set of n_i dimensional index vectors of the hidden layer as

$$\gamma_{n_i} = \gamma_{n_i}(\delta) = \{\delta \in R^{n_i} | \delta_h \in \{0, 1\}, h = n_1, n_2\}$$

where δ is used as a binary indicator. Obviously, the i th layer with n_i neurons has 2^{n_i} combinations of binary indicator with $k = 0, 1$ and the elements of index vectors for two-layers neural network have $2^{n_2} \times 2^{n_1}$ combinations in the set

$$\Omega = \gamma_{n_2} \oplus \gamma_{n_1}$$

By using (5), the neural network $N_i(x(t), W_{i1}^*, W_{i2}^*)$ can be expressed as

$$N_i(x(t), W_{i1}^*, W_{i2}^*) = \sum_{\sigma \in \gamma_{n_2} \oplus \gamma_{n_1}} \mu_{i\sigma} A_{i\sigma}(\sigma, \Psi_i, W_i^*)x(t) \tag{7}$$

where

$$A_\sigma = \text{diag}[S_{i2k}(v_{i2k}, \varphi_{i2k})]W_{i2}^* \text{diag}[S_{i1k}(v_{i1k}, \varphi_{i1k})]W_{i1}^* \tag{8}$$

which is a constant matrix whose coefficients depend on optimal weights W_i^* , activation functions Ψ_{ir} and all index vectors δ . Further, $\mu_{i\sigma}$ is the product of the real number $h_i(k)$ of all neurons in whole neural network, and for $\sigma \in \Omega$,

$$\mu_{i\sigma} > 0, \quad \sum_{\sigma \in \gamma_{n_2} \oplus \gamma_{n_1}} \mu_{i\sigma} = 1.$$

Thus, by means of neural networks, nonlinear MJS (1) is translated into a group of linear differential inclusions, in which the different inclusion is powered by stochastic Markovian process, i.e.

$$\begin{aligned} \dot{x}(t) &= \sum_{\sigma \in \Omega} \mu_{i\sigma} (A_{i\sigma} + A_i)x(t) + B_i w(t) + \Delta f_i(x(t)), \\ y(t) &= C_i x(t) \\ z(t) &= L_i x(t) \end{aligned} \tag{9}$$

where

$$\Delta f_i(x(t)) = f_i(x(t)) - \sum_{\sigma \in \Omega} \mu_{i\sigma} A_{i\sigma} x(t) \leq \rho_i \|x(t)\| \tag{10}$$

3 H_∞ Filter Design

The intention of the H_∞ filtering problem is to obtain an estimate $\hat{z}(t)$ of the signal $z(t)$ such that a guaranteed performance criterion is minimized in an estimated error sense. Consider the following linear Markovian full-order filter for which the jumping process $\{r_t\}$ is available for $t \geq 0$:

$$\begin{aligned} \dot{\hat{x}}(t) &= K_{i1}\hat{x}(t) + K_{i2}y(t) \\ \hat{z}(t) &= K_{i3}\hat{x}(t) \end{aligned} \tag{11}$$

where the matrices K_{i1} , K_{i2} and K_{i3} are to be determined in the course of the design.

In terms of the state error $e(t) = x(t) - \hat{x}(t)$, it follows from system (9) and filter (11) that the state error dynamics has the form

$$\dot{e}(t) = (A'_i - K_{i2}C_i - K_{i1})x(t) + K_{i1}e(t) + B_i w(t) + \Delta f_i(x(t)) \tag{12}$$

where $A'_i = \sum_{\sigma \in \Omega} \mu_{i\sigma} (A_{i\sigma} + A_i)$.

A state-space augmented model of the filtering error, $\tilde{z}(t) = z(t) - \hat{z}(t)$, can then be constructed using (9), (11) and (12)

$$\begin{aligned} \dot{\tilde{x}}(t) &= \begin{bmatrix} \dot{\tilde{x}} \\ \dot{\tilde{e}} \end{bmatrix} = \tilde{A}_i \tilde{x}(t) + \tilde{B}_i w(t) + \tilde{E}_i \Delta f_i(\tilde{x}), \\ \tilde{z}(t) &= \tilde{C}_i \tilde{x}(t) \end{aligned} \tag{13}$$

where

$$\begin{aligned} \tilde{A}_i &= \begin{bmatrix} A'_i & 0 \\ A'_i - K_{i2}C_i - K_{i1} & K_{i1} \end{bmatrix} \\ \tilde{B}_i &= \begin{bmatrix} B_i \\ B_i \end{bmatrix} \\ \tilde{E}_i &= \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \\ \tilde{C}_i &= [L_i - K_{i3} \quad K_{i3}] \\ \Delta f_i(\tilde{x}) &= \begin{bmatrix} \Delta f_i(x) \\ \Delta f_i(e) \end{bmatrix} \end{aligned}$$

The primary objective of this paper is to provide a practical design procedure for a Markovian H_∞ filter for the nonlinear MJS (1). In other words, we shall design the filter parameter K_{i1} , K_{i2} and K_{i3} such that the filtering error systems (13) is stochastically stable and for all non-zero $w \in L_2[0, \infty)$

$$E\left\{ \int_0^\infty \tilde{z}^T(t)\tilde{z}(t)dt - \gamma^2 \int_0^\infty w^T(t)w(t)dt \right\} < 0$$

Lemma 1 [19]. let $x \in R^n$, $y \in R^n$ and $\varepsilon > 0$, then

$$x^T y + y^T x \leq \varepsilon x x^T + \varepsilon^{-1} y^T y$$

Theorem 1. Consider the nonlinear MJS (1) and given $\gamma > 0$, if there exist positive definite symmetric matrices P_{i1} , P_{i2} , matrices X_i , Y_i , K_{i3} and a set of positive real number ε_i such that

$$\begin{bmatrix} E_i & O_i \\ O_i^T & X_i \end{bmatrix} < 0 \tag{14}$$

where

$$\begin{aligned} E_i &= \begin{bmatrix} P_{i1}A''_i + (A''_i)^T P_{i1} + \varepsilon_i \rho_i^2 I + \sum_{j=1}^s \pi_{ij} P_{j1} & (A''_i)^T P_{i2} - C_i^T Y_i^T - X_i^T \\ * & X_i + X_i^T + \varepsilon_i \rho_i^2 I + \sum_{j=1}^s \pi_{ij} P_{j2} \end{bmatrix} \\ O_i &= \begin{bmatrix} P_{i1}B_i & L_i^T - K_{i3}^T & P_{i1} & 0 \\ P_{i2}B_i & K_{i3}^T & P_{i2} & 0 \end{bmatrix} \\ X_i &= -diag[\gamma^2 I \quad I \quad \varepsilon_i I \quad \varepsilon_i I] \\ A''_i &= A_{i\sigma} + A_i \end{aligned}$$

then the H_∞ filtering problem for system (1) is solvable. Moreover, the matrices of the filter is given by

$$\begin{aligned} K_{i1} &= P_{i2}^{-1} X_i \\ K_{i2} &= P_{i2}^{-1} Y_i \end{aligned}$$

Proof. For $T > 0$, introduce the following cost function for system (13)

$$J(T) = E\left\{ \int_0^T \tilde{z}^T(t)\tilde{z}(t)dt - \gamma^2 \int_0^T w^T(t)w(t)dt \right\} \tag{15}$$

with concerned stochastic Lyapunov function

$$V(\tilde{x}(t), r_t = i) = V(\tilde{x}, i) = \tilde{x}^T P_i \tilde{x} \tag{16}$$

where P_i is mode-dependent positive definite symmetric matrix for each i .

Then the weak infinitesimal operator \hat{A} of (13) is given by

$$\begin{aligned} \hat{A}V(\tilde{x}, i) &= \lim_{\Delta t \rightarrow 0} \frac{1}{\Delta t} [E\{V(\tilde{x}(t + \Delta t), r_{t+\Delta t}) | \tilde{x}(t), r_t\} - V(\tilde{x}(t), r_t)] \\ &= 2\tilde{x}^T P_i [\tilde{A}_i \tilde{x} + \tilde{B}_i w + \tilde{E}_i \Delta f_i(\tilde{x})] + \tilde{x}^T \sum_{j=1}^s \pi_{ij} P_j \tilde{x} \end{aligned} \tag{17}$$

Applying Lemma 1 and noticing (10), it follows that

$$\hat{A}V(\tilde{x}, i) \leq \tilde{x}^T \Lambda_i \tilde{x} + 2\tilde{x}^T P_i \tilde{B}_i w$$

where

$$\Lambda_i = P_i \tilde{A}_i + \tilde{A}_i^T P_i + \varepsilon_i^{-1} P_i \tilde{E}_i \tilde{E}_i^T P_i + \varepsilon_i \rho_i^2 I + \sum_{j=1}^s \pi_{ij} P_j$$

In zero initial condition, the index $J(T)$ can be rewritten as

$$\begin{aligned} J(T) &= E\left\{ \int_0^T [\|\tilde{z}(t)\|^2 - \gamma^2 \|w(t)\|^2 + \hat{A}V(\tilde{x}, i)] dt \right\} - V(\tilde{x}, i) \\ &\leq E\left\{ \int_0^T [\tilde{x}^T (\Lambda_i + \tilde{C}_i^T \tilde{C}_i) \tilde{x} + 2\tilde{x}^T P_i \tilde{B}_i w - \gamma^2 w^T w] dt \right\} - V(\tilde{x}, i) \\ &= E\left\{ \int_0^T \begin{bmatrix} \tilde{x} \\ w \end{bmatrix}^T \begin{bmatrix} \Lambda_i + \tilde{C}_i^T \tilde{C}_i & P_i \tilde{B}_i \\ \tilde{B}_i^T P_i & -\gamma^2 I \end{bmatrix} \begin{bmatrix} \tilde{x} \\ w \end{bmatrix} dt \right\} - V(\tilde{x}, i) \end{aligned}$$

Let

$$\Theta = \begin{bmatrix} \Lambda_i + \tilde{C}_i^T \tilde{C}_i & P_i \tilde{B}_i \\ \tilde{B}_i^T P_i & -\gamma^2 I \end{bmatrix}.$$

If $\Theta < 0$, then for $w(t) = 0$,

$$P_i \tilde{A}_i + \tilde{A}_i^T P_i + \sum_{j=1}^s \pi_{ij} P_j < 0$$

which guarantees the stability of filtering error dynamics (13). On the other hand, for $T \rightarrow \infty$, $\Theta < 0$ yields

$$J(\infty) < -V(\infty) < 0$$

i.e.

$$E\left\{\int_0^\infty \tilde{z}^T(t)\tilde{z}(t)dt - \gamma^2 \int_0^\infty w^T(t)w(t)dt\right\} < 0$$

Applying Schur complements, condition $\Theta < 0$ is equivalent to following inequality:

$$\begin{bmatrix} A_i & P_i \tilde{B}_i & \tilde{C}_i^T & P_i \tilde{E}_i \\ * & -\gamma^2 I & 0 & 0 \\ * & * & -I & 0 \\ * & * & * & -\varepsilon_i I \end{bmatrix} < 0 \tag{18}$$

Now defining $P_i = \begin{bmatrix} P_{i1} & 0 \\ 0 & P_{i2} \end{bmatrix}$, where $P_{i1} \in R^{n \times n}$, $P_{i2} \in R^{n \times n}$, the above inequality can be rewritten as

$$\begin{bmatrix} \Sigma_i & O_i \\ O_i^T & X_i \end{bmatrix} < 0 \tag{19}$$

where

$$\Sigma_i = \begin{bmatrix} H_i & (A'_i)^T P_{i2} - C_i^T K_{i2}^T P_{i2} - K_{i1}^T P_{i2} \\ * & P_{i2} K_{i1} + K_{i1}^T P_{i2} + \varepsilon_i \rho_i^2 I + \sum_{j=1}^s \pi_{ij} P_{j2} \end{bmatrix}$$

$$H_i = P_{i1} A'_i + (A'_i)^T P_{i1} + \varepsilon_i \rho_i^2 I + \sum_{j=1}^s \pi_{ij} P_{j1}$$

Introduce a class of variables

$$X_i = P_{i2} K_{i1}, \quad Y_i = P_{i2} K_{i2},$$

then LMI (14) can be obtained from (19) using $\sum_{\sigma \in \Omega} \mu_{i\sigma} = 1$. This concludes the proof.

4 Numerical Example

Consider the nonlinear MJS (1) with parameters given by Mode 1:

$$A_1 = \begin{bmatrix} -1 & -3 \\ 0 & -5 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0 \\ 3 \end{bmatrix}, \quad C_1 = [2 \quad 3]$$

$$L_1 = [0.1 \quad 0], \quad f_1(x) = \begin{bmatrix} 0 \\ \sin(x_1(t)) \end{bmatrix}$$

Mode 2:

$$A_2 = \begin{bmatrix} 0 & -2 \\ 0 & -3 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad C_2 = [1 \quad 1.8]$$

$$L_2 = [0.2 \quad 0.1], \quad f_2(x) = \begin{bmatrix} 0 \\ \sin(0.3x_1(t)) \end{bmatrix}$$

Mode 3:

$$A_3 = \begin{bmatrix} 1 & -3 \\ 0 & -4 \end{bmatrix}, \quad B_3 = \begin{bmatrix} 0 \\ 1.7 \end{bmatrix}, \quad C_3 = [1.2 \quad 1.5]$$

$$L_3 = [0 \quad 0.1], \quad f_3(x) = \begin{bmatrix} 0 \\ \sin(0.5x_1(t)) \end{bmatrix}$$

the transition rate matrix is defined by

$$\Pi = \begin{bmatrix} -3 & 1.8 & 1.2 \\ 0.3 & -2 & 1.7 \\ 0.3 & 0.7 & -1 \end{bmatrix}$$

In this example, the same three single hidden layer neural networks with $h = 2$ are chosen to respectively approximate three nonlinear functions $f_i(x(t))$ for each mode, and three LDI are obtained as

$$A_{11} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, A_{12} = \begin{bmatrix} 0 & 0 \\ 0.515 & 0 \end{bmatrix}, A_{13} = \begin{bmatrix} 0 & 0 \\ -0.225 & 0 \end{bmatrix}, A_{14} = \begin{bmatrix} 0 & 0 \\ 0.74 & 0 \end{bmatrix}$$

$$A_{21} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, A_{22} = \begin{bmatrix} 0 & 0 \\ 1.248 & 0 \end{bmatrix}, A_{23} = \begin{bmatrix} 0 & 0 \\ 4.43 & 0 \end{bmatrix}, A_{24} = \begin{bmatrix} 0 & 0 \\ -3.182 & 0 \end{bmatrix}$$

$$A_{31} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}, A_{32} = \begin{bmatrix} 0 & 0 \\ 1.366 & 0 \end{bmatrix}, A_{33} = \begin{bmatrix} 0 & 0 \\ 2.61 & 0 \end{bmatrix}, A_{34} = \begin{bmatrix} 0 & 0 \\ -2.607 & 0 \end{bmatrix}$$

The upper bounds of approximation errors are $\rho_1 = 0.2, \rho_2 = 0.19, \rho_3 = 0.42$, respectively.

By solving LMI (14), the Markovian H_∞ filter is obtained with the parameters described by

$$K_{11} = \begin{bmatrix} 2.0668 & 1.6107 \\ -34.9113 & -49.448 \end{bmatrix}, K_{12} = \begin{bmatrix} -1.4649 \\ 17.6371 \end{bmatrix}, K_{13} = \begin{bmatrix} 0.0972 \\ -0.0015 \end{bmatrix}^T$$

$$K_{21} = \begin{bmatrix} -92.3674 & -141.8372 \\ 45.2211 & 60.6040 \end{bmatrix}, K_{22} = \begin{bmatrix} 93.8452 \\ -45.9772 \end{bmatrix}, K_{23} = \begin{bmatrix} 0.1983 \\ 0.0897 \end{bmatrix}^T$$

$$K_{31} = \begin{bmatrix} -51.9729 & -40.5190 \\ 17.2689 & -0.5486 \end{bmatrix}, K_{32} = \begin{bmatrix} 44.448 \\ -15.5181 \end{bmatrix}, K_{33} = \begin{bmatrix} -0.0017 \\ 0.2663 \end{bmatrix}^T$$

For addressed stochastic nonlinear MJS, the simulation results of the state response of the system are given in Fig.1, where the initial condition is $[1.5 \ 1]^T$, and $w(t)$ is chosen from a normal disturbance with mean zero and variance one.

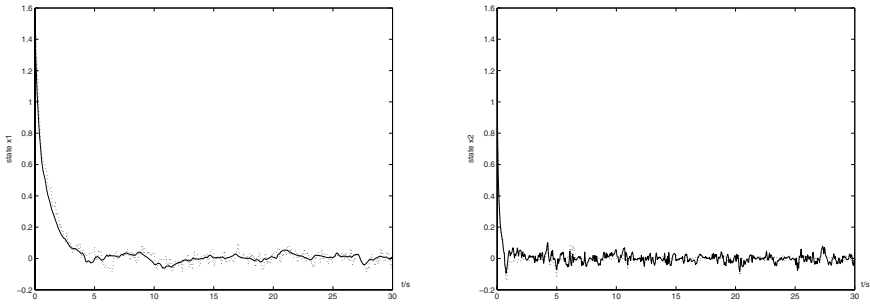


Fig. 1. Real state response $x(t)$ and estimated state $\hat{x}(t)$

5 Conclusions

In this paper, we develop a methodology for designing Markovian H_∞ filter that ensures asymptotically stable for the filtering error system and a prescribed bound on the H_∞ gain from the noise signals to the estimation error. By applying matrix transformation and variable substitution, the main results are provided by LMI form. Moreover, further research can be extended to the general nonlinear MJS with time-delays and parameter uncertainties.

References

1. Feng, X., Loparo, K. A., Ji, Y., et al.: Stochastic stability properties of jump linear systems. *IEEE Trans. Automat. Contr.* **37** (1992) 1141-1146
2. Wang, Z., Lam, J., Liu, X.: Exponential filtering for uncertain Markovian jump time-delay systems with nonlinear disturbances. *IEEE Transactions on Circuits and Systems - Part II* **51** (2004) 262-268
3. Wang, Z., Qiao H., Burnham, K.: On stabilization of bilinear uncertain time-delay stochastic systems with Markovian jumping parameters. *IEEE Transactions on Automatic Control* **47** (2002) 640-646
4. Cao, Y. Y., Lam, J.: Robust H_∞ control of uncertain Markovian jump systems with time-delay. *IEEE Trans. Automat. Contr.* **45** (2000) 77-83

5. Farias, D. P., Geromel, J. C., Val, J., et al.: Output feedback control of Markov jump linear systems in continuous-time. *IEEE Trans. Automat. Contr.* **45** (2000) 944-949
6. Liu, F., Cai, Y.: Passive analysis and synthesis of Markovian jump systems with norm bounded uncertainty and unknown delay. *Dynamics of Continuous Discrete and Impulsive System - Series A - Mathematical Analysis* **13** (2006) 157-166
7. Chen, W. H., Xu, J. X., Guan, Z. H.: Guaranteed cost control for uncertain Markovian jump systems with mode-dependent time-delays. *IEEE Trans. Automat. Contr.* **48** (2003) 2270-2276
8. Shi, P., Boukas, E. K.: Kalman filtering for continuous-time uncertain systems with Markovian jumping parameters. *IEEE Trans. Automat. Contr.* **44** (1999) 1592-1597
9. Mahmond, M. S., Shi, P.: Robust Kalman filtering for continuous time-lag systems with Markovian jump parameters. *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.* **50** (2003) 98-105
10. Souza, C. E., Fragoso, M. D.: H_∞ filtering for Markovian jump linear systems. *Int. J. Syst. Sci.* **33** (2002) 909-915
11. Sun, F., Liu, K. H., Sun, Z.: Reduced-order H_∞ filtering for linear systems with Markovian jump parameters. *Syst. Control Lett.* **54** (2005) 739-746
12. Wang, Z., Lam, J., Liu, X.: Nonlinear filtering for state delayed systems with Markovian switching. *IEEE Trans. Signal Process* **51** (2003) 2321-2328
13. Chen, F. C., Khalil, H. K.: Adaptive control of nonlinear systems using neural networks. *International Journal of Control* **55** (1992), 1299-1317
14. Liu, C. C., Chen, F. C.: Adaptive control of nonlinear continuous time systems using neural networks-general relative degree and MI MO cases. *International Journal of Control* **58** (1993) 317-335
15. Bass, E., Lee, K. Y.: Robust control of nonlinear system using norm-bounded neural networks. *IEEE Int. Conf. on neural networks* **4** (1994) 2524-2529
16. Tanaka, K.: An approach to stability criteria of neural-network control systems. *IEEE Trans. Neural Networks* **7** (1996) 629-642
17. Limanond, S., Si, J.: Neural-network-based control design: an LMI approach. *IEEE Trans. on Neural Networks* **9** (1998) 1422-1429
18. Lin, C. L., Lin, T. Y.: An H_∞ design approach for neural net-based control schemes. *IEEE Trans. Autom. Control* **46** (2001) 1599-1605
19. Wang, Z., Huang, B., Unbehauen, H.: Robust H_∞ observer design of linear time-delay systems with parametric uncertainty. *Systems & Control Letters* **42** (2001) 303-312

A Novel Off-Line Signature Verification Based on Adaptive Multi-resolution Wavelet Zero-Crossing and One-Class-One-Network

Zhiqiang Ma¹, Xiaoyun Zeng², Lei Zhang², Meng Li², and Chunguang Zhou^{1,*}

¹College of Computer Science and Technology, Jilin University, Changchun, Jilin Province, China

²Computer School, Northeast Normal University, Changchun, Jilin Province, China
{mazq, zengxy863}@nenu.edu.cn

Abstract. This paper proposes a novel off-line signature verification method based on adaptive multi-resolution wavelet zero-crossing and one-class-one-network classification. First, the horizontal, vertical, 45 degree direction and the 135 degree direction projections of the binarized signature images are calculated, respectively. The curvature data of the projections are decomposed into multi-resolution signals using wavelet transforms. Then the zero-crossings corresponding to the curvature data are extracted as features for verification. At last, one-class-one-network classifier is used to verify the signatures. The signature verification system was experimented on real data sets and the results show the system is very effective.

1 Introduction

Handwritten signature is one of the most widely accepted personal attributes for identity verification. As a symbol of consent and authorization, especially in the prevalence of credit cards and bank cheques, handwritten signature has long been the target of fraudulence. Therefore, with the growing demand for processing of individual identification faster and more accurately, the design of an automatic signature system faces a real challenge.

Handwritten signature verification can be divided into on-line (or dynamic) and off-line (or static) verification. On-line verification refers to a process that the signer uses a special pen called a stylus to create his or her signature, producing the pen location, speeds and pressures, while off-line verification just deals with signature images acquired by a scanner or a digital camera.

During the last few years, researchers have made great efforts on off-line signature verification. Ammar et al. [1] proposed parametric and reference-pattern based features to verify skillful simulated handwritten signatures. Qi and Hunt [2] adopted a multi-resolution approach for off-line signature verification. They only made use of lowpass data to extract some geometrical and statistical features. Highpass data,

* Corresponding author.

which are believed to have discriminating power, were not used at all. To decompose a curvature-based signature into a multi-resolution format, wavelet theory [3, 4] is introduced. Wavelet theory has broad applications in image analysis [5, 6]. In order to build an efficient off-line handwritten signature verification system, we propose a new representation scheme for a signature and then use wavelet transforms to decompose and analyze the transformed signals. In the first stage of the system, the horizontal, vertical, 45 degree direction and the 135 degree direction projections of the binarized signature images are calculated, respectively. The four pieces of one-dimensional data are transformed into another space using Daubechies Wavelets transform, respectively. Then the zero-crossings corresponding to the curvature data are extracted as features for verification. At last, One-class-one-network classifier is employed to identify the signature images.

The remainder of the paper is organized as follows: Section 2 introduces the database used in our study and the preprocessing stage, Section 3 describes the features selected, Section 4 provides a brief introduction to one-class-one-network classifier, In Section 5 the verification strategies and experimental results are presented, Section 6 presents the conclusion and future work.

2 The Signature Database and Preprocessing

2.1 Database

The signature database consists of 672 signature images, scanned at a resolution of 300 dpi, 8-bit gray-scale. They are organized into 21 sets; each set corresponds to one signature enrollment. There are 12 genuine signatures and 20 forgery signatures in each set. Each volunteer was asked to sign his or her own signature on a white paper with base-lines 12 times. After this process had been done, we invited some people who are good at imitating other's handwritings. Before formal collection, they could practice any times they wanted when imitating. Each person's name was imitated 20 times in total. Fig. 1 shows some samples in the database.

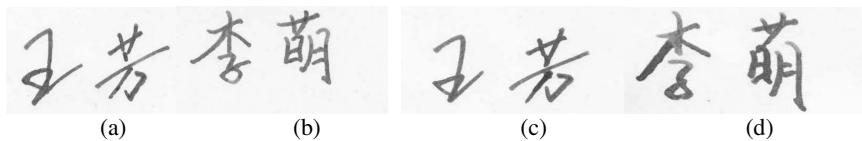


Fig. 1. (a), (b) Genuine signature. (c), (d) Forgery signature.

2.2 Preprocessing

The preprocessing stage is divided into four different parts: noise removal, image binarization, data area cropping, and width normalization.

Standard noise reduction and isolated peak noise removal techniques, such as median-filtering and average filtering [7] are used to clean the initial image.

Morphological operations are applied to fill small holes and to remove small connected components mostly generated by the noisy background.

The signature area is separated from the background by using the well-known segmentation method of vertical and horizontal projections [7]. Thus, the white space surrounding the signature is discarded. Fig. 2 shows the image binarization.



Fig. 2. (a), (b) Image binarization

3 Feature Extraction

For each preprocessed signature image, the projections of the horizontal, vertical, 45 degree direction and the 135 degree direction are calculated, respectively. Then these data are represented by one-dimensional signal, respectively (see Fig. 3). So we can obtain four one-dimensional signals from a signature image, the next step is to extract features from these signals data and use them as the bases for signature verification.

Basically, the features to be extracted should be stable and should retain the characteristics of the original pattern. At this stage, we use Daubechies wavelets transform to perform feature extraction. In general, the multi-resolution wavelet transform can decompose a signal into lowpass and highpass information [3, 4]. The lowpass (i.e., low frequency) information represents the main body of the original data while the highpass (i.e., high frequency) information usually represents features that contain sharper variations. The zero-crossings of the transformed highpass data naturally indicate sharper variation points.

In our study, for every zero-crossing point, three attributes associated with the zero-crossing point will be extracted. They are: (1) the abscissa of the zero-crossings [8], (2) the variance of the abscissa calculated by formula (1) and (2), and (3) the left-hand side wavelet integral between the current zero-crossing and the previous one [4].

$$\mu = \frac{\sum_{i=1}^n F(i)}{n} . \quad (1)$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^n (F(i) - \mu)^2}{n}} . \quad (2)$$

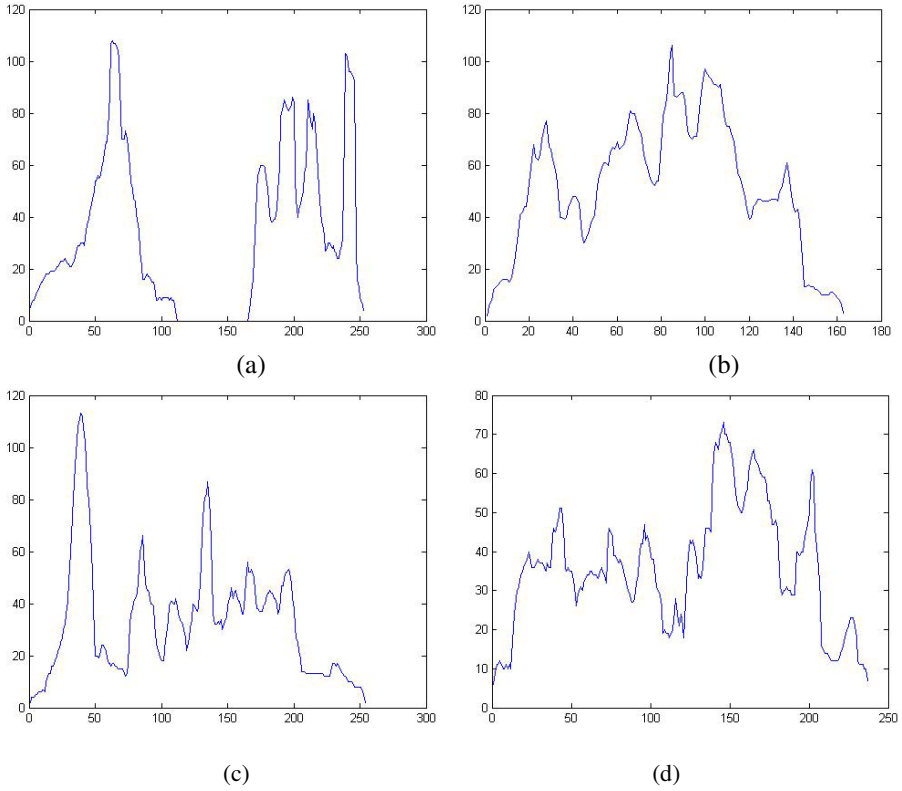


Fig. 3. (a), (b), (c) and (d) The four one-dimensional signals of the horizontal projection, vertical projection, 45 degree direction projection and the 135 degree direction projection of a preprocessed signature, respectively

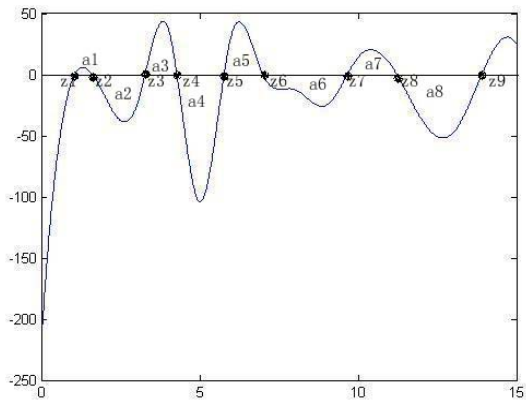


Fig. 4. Illustration showing the meaning of wavelet integrals between zero-crossings

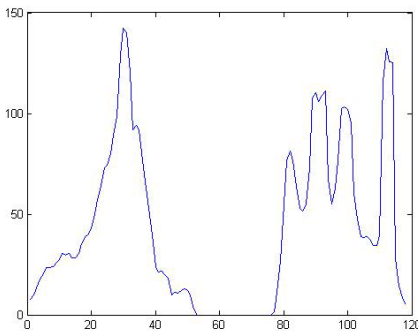
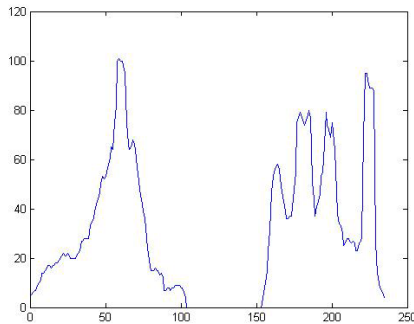
Fig. 4 is a typical example showing how the attributes are calculated. Let z_5 be the zero-crossing point under consideration. Its abscissa is close to 6.0. The third attribute associated with z_5 is the wavelet integral bound by z_5 and z_4 (i.e., the area of a_4).

When decomposing the four one-dimensional signals, we restrict the dyadic scale to 2^j in order to obtain a complete and stable representation (detailed in [3, 4]) of the four projections. Fig. 5 show the wavelet transformed one-dimensional signals of the horizontal projection of a preprocessed signature. From top to bottom in the figure, the signals on the left-hand side are the lowpass data with resolutions of 2^0 (the original signal), 2^{-1} , 2^{-2} , 2^{-3} and 2^{-4} , respectively, and the signals on the right-hand side are the highpass data with resolutions of 2^{-1} , 2^{-2} , 2^{-3} and 2^{-4} , respectively.

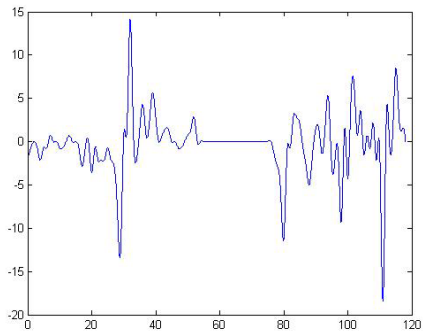
As above, the other three one-dimensional signals of the vertical, 45 degree direction and the 135 degree direction projections are wavelet transformed in proper resolution level, respectively. As mentioned previously, the three attributes associated with the zero-crossing point of the transformed highpass data are extracted as features to verify.

Using wavelet transform in alterable resolution levels to extract the features of the projections can get better effect. We can select different resolution levels for different signature images to obtain the best verification result. The experiments show that the

Lowpass data



Highpass data



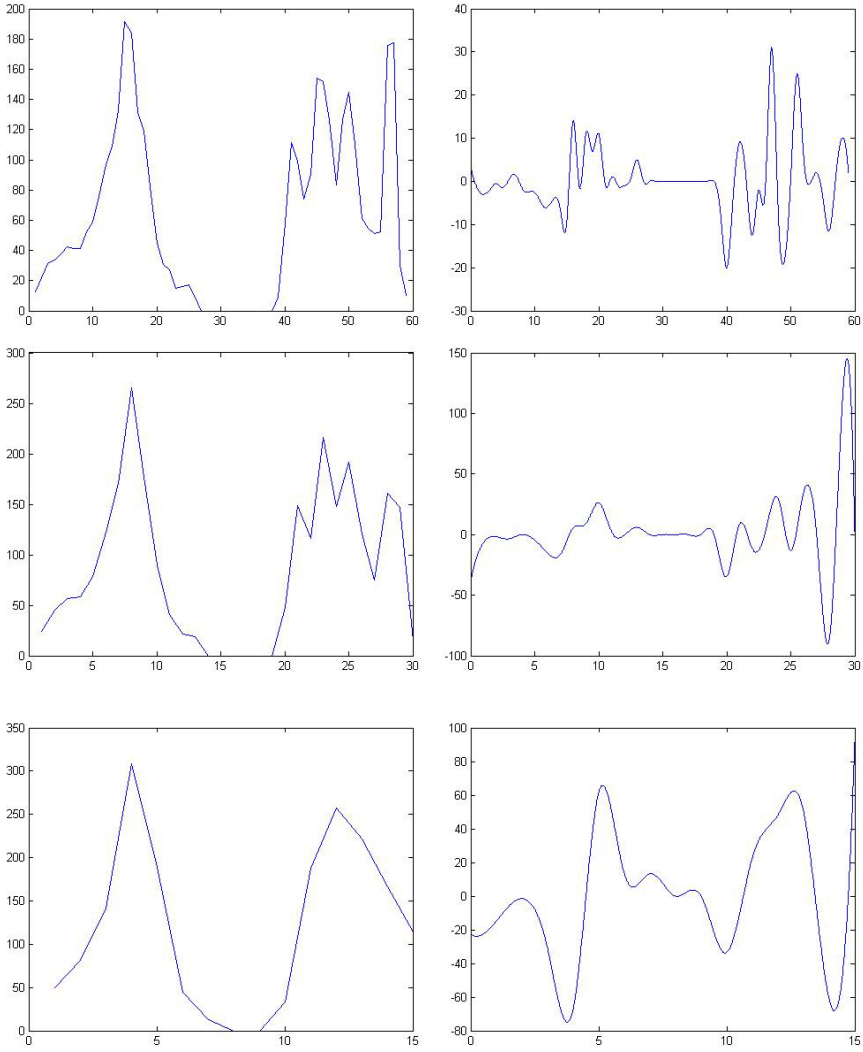


Fig. 5. The multi-resolution wavelet transformed signals of the horizontal projection of a pre-processed signature. The resolutions shown are from 2^0 to 2^{-4} , respectively, from top to bottom. The left column represents the lowpass data, and the right column shows the highpass data.

number of the zero-crossings which applied to the same signature image are diverse in different resolution levels. And even signatures of the same person may vary in the number of zero-crossings in the same resolution level. Therefore, we consider if the variance of the number of the zero-crossings which belong to genuine signatures in the same resolution level is smaller and in reasonable bound, the resolution level is better. So for a special enrollment, the optimal resolution level is corresponding to the smallest variance. Here, we make an example to show how to decide the optimal resolution for an enrollment. There are 8 genuine signatures of training samples for

every person. In this paper, we use $L=1, 2, 3 \dots$ to denote the resolutions of highpass data is $2^{-1}, 2^{-2}, 2^{-3} \dots$ respectively. Table 1 shows the variance of the number of the zero-crossings in the highpass data of the horizontal projection signal, and the signal data are wavelet transformed in different resolution levels.

Table 1. The results showing the variance of the number of the zero-crossings in different resolution levels

L	Genuine signature i								Variance
	1	2	3	4	5	6	7	8	
1	42	46	49	46	33	49	42	43	0.81
2	30	24	22	26	23	23	22	32	0.57
3	14	16	22	20	18	19	12	15	0.65
4	9	9	8	6	10	12	8	6	0.47
5	4	5	6	3	4	8	3	4	0.61
6	1	1	2	1	2	1	2	1	0.19

The variance showed in table 1 are calculated by the formula: (1), (2) and (3).

$$\sigma / \mu . \quad (3)$$

From the table 1, we can see that when $L=4$, the variance is the smallest except the $L=6$. Although $L=6$ the variance is the smallest, the zero-crossings corresponding to the signals data are make no sense. So for this enrollment, the optimal resolution is 4 when transforming the horizontal projection signals.

4 One-Class-One-Network

Neural network presents a computational paradigm for constructing classifiers that can perform as accurately as conventional techniques [9]. In our system we use a completely connected feedforward neural network with the classical backpropagation learning algorithm, more simply known as the Backpropagation Network (BPN) which is described in detail in many textbooks [10], [11].

We have understood that for a signature verification system to be functional in practical applications, the ability to easily add/remove signatures from new/obsolete owners to its database must be inherent. Our approach towards this goal is to implement the structure of the neural network classifier is a one-class-one-network scheme. That is for each signature owner an individual classifier is being implemented. Each time the signatures from a new owner are added to the signature verification database, only a small, fixed-size, neural-network-based classifier must be trained.

5 Experiments

In this section, we shall report some experimental results. For training the system we randomly selected 420 signature images from the database. There are 21 small,

fixed-size, neural-network-based classifiers corresponding to the 21 sets, each set includes 8 genuine signatures and 12 forgery signatures. To verify the performance of our approach, the other 252 signatures include 84 genuine signatures and 168 forgery signatures are used to test the system.

In Section 3, we described the detailed feature extraction process. The next step is to use these features to verify whether a signature is genuine or forgery. Due to different personal writing styles, it is impossible to uniquely determine a resolution level to fit all writers. The only thing that we can do is to get training samples from every writer and then determine his/her own resolution level. First of all, we shall show how L , the chosen resolution, affects the performance. Table 2 shows the false reject rate (FRR), false accept rate (FAR) and the average rate error (ARR), respectively. The first six rows correspond, respectively, to the six cases with L fixed at from 1 to 6 for all writers. The bottom row of Table 1, on the other hand, corresponds to errors determined by using the optimal L value dynamically for each writer. From the data shown in Table 2, it is obvious that the average error generated by dynamic selection of L is smaller than that of any of the other six cases, in which a fixed resolution for signature verification was used.

Table 2. The results showing how resolution selection affects system performance

Resolution	FRR	FAR	ARR
1. $L=1$	6.4%	7.9%	7.2%
2. $L=2$	6.2%	5.9%	6.1%
3. $L=3$	5.4%	6.1%	5.8%
4. $L=4$	7.1%	6%	6.6%
5. $L=5$	4.9%	6.5%	5.3%
6. $L=6$	5%	7.3%	6.2%
7. Optimal dynamic L	4%	2.4%	3.2%

Thereinto, the experiment results showed in table 2 are obtained by using the structure of one-class-one-network to verify.

Using only one BP network and the optimal L value dynamically for each writer is also implemented in our experiment. The results showed in table 3.

Table 3. Results of using only one network to verify

FRR	FAR	ARR
11.8%	9.6%	10.7%

Compared with the structure that using only one network, the one-class-one-network classifier can gain a higher accuracy. As we know, for English handwritten signature verification system designed for skilled forgeries, the ever-reported accuracy rate is

less than 80 % (Mizukmi et al., 1999; Yoshiki et al., 2002). Of course, they are based on different databases. So the approach proposed in this paper is very effective and accurate.

6 Conclusion and Future Work

Off-line signature verification is a difficult two-class pattern recognition problem. A new approach based on wavelet and one-class-one-network classifier has been proposed in this paper. A major contribution of this work is using four one-dimensional signals to represent a signature and using a wavelet-based feature extractor to extract complete features from multi-resolution signals. Synchronously, the one-class-one-network classifier is employed in the field of signature verification. Despite the fact that the approach shown in this paper seems to be effective, it should be validated on a large signature database where several types of signatures can be taken into account (North American, European, Arabic, etc.).

Further perspectives and attractive challenges for future research lie in three aspects: how to extract more effective features, how to combine one-class-one-network classifier with other signature verification methods and how to devise an effective wavelet-based approach to detect forgeries by projecting only genuine signature. This is one of the most difficult types of forged signatures to detect.

Acknowledgement

This paper is supported by the National Nature Science Foundation of China (Grant No. 60433020, 60673099), the Key Laboratory for Symbol Computation and Knowledge Engineering of the National Education Ministry of China, and the Project '985': Science Technique Innovation Platform of Computation and Software Science.

References

1. Ammar, M., Yoshida, Y., Fukumura, T.: Structural Description and Classification of Signature Images. *Pattern Recognition* **23**(7) (1990) 697–710
2. Qi, Y., Hunt, B. R.: A Multiresolution Approach to Computer Verification of Handwritten Signature. *IEEE Trans. Image Process* **4** (1995) 870–874
3. Mallat, S. G.: A Theory for Multiresolution Signal Decomposition: The Wavelet Representation. *IEEE Trans. Pattern Anal. Mach. Intell.* **11**(7) (1989) 674–693
4. Mallat, S. G.: Zero-crossings of a Wavelet Transform, *IEEE Trans. Inform. Theory* **37**(4) (1991) 1019–1033
5. Hsieh, J.W., Mark Liao, H.Y., Ko, M. T., Fan, K. C.: Wavelet-based Shape from Shading, *Graph. Models Image Process* **57** (1995) 343–362
6. Hsieh, J. W., Mark Liao, H. Y., Fan, K. C., Ko, M. T., Huang, Y. P.: A New Edge-based Technique for Image Registration. *Comput. Vision Image Understand* **67** (1997) 112–130
7. Gonzalez, C., Wintz, P.: 1987. *Digital Image Processing*. 2nd edition. Addison-Wesley, MA.

8. Logan, B.: Information in the Zero-crossings of Bandpass Signals. *Bell System Tech. J.* **56**, (1977) 510.
9. Burr, D. J.: Experiments on Neural Net Recognition of Spoken and Written Text. *IEEE Trans Acoust. Speech SigProcess* **36** (1988) 1162-1168
10. Wasserman, P.D.: *Neural Computing: Theory and Practice*. Van Nostrand Reinhold, New York (1989)
11. Hecht-Nielsen, R.: *Neurocomputing*. Addison-Wesley, New York (1990).

Combining News and Technical Indicators in Daily Stock Price Trends Prediction

Yuzheng 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, saman}@unimelb.edu.au

Abstract. Stock market prediction has always been one of the hottest topics in research, as well as a great challenge due to its complex and volatile nature. However, most of the existing methods neglect the impact from mass media that will greatly affect the behavior of investors. In this paper we present a system that combines the information from both related news releases and technical indicators to enhance the predictability of the daily stock price trends. The performance shows that this system can achieve higher accuracy and return than a single source system.

1 Introduction

Stock market prediction has always been one of the hottest topics in research, as well as a great challenge due to its complex and volatile nature. Research suggests that the financial time series do not exhibit random behavior and the stock price is predictable [28]. Numerous publications have attempted to construct an accurate model for the stock market. Most of these works focus on time series prediction with various AI models, such as Artificial Neural Networks [1, 17], Genetic Algorithm [10], Fuzzy System [18], Hidden Markov Model [9] or some hybrid combinations [26, 29], as well as statistical techniques, such as moving average [6]. However, these methods inevitably have their own limitations. Back-propagation neural network for example, suffers from the risk of over-fitting and large number of parameters. More importantly, they have neglected other source of information such as mass media that will greatly affect the behavior of investors.

The entities listed on the Australian stock exchange are required to fully inform the investors at all times so that investment decision can be made with rich and timely information. The materials include negotiations of purchase, director appointment/resignation and divestitures of businesses. Since most of this information can be obtained from the news articles, major financial newspapers become a good source of information in assisting the traders. Mitchell and Mulherin [13] studied the influence of public information reported by Dow Jones and concluded that a direct relation does exist between released news articles and stock market activities. News release provides abundant information regarding the activities that companies are involved in and it may produce speculations among traders that results in movements of the stock prices. NofSinger [16] showed that in some cases, investors tend to buy

after positive news which results in buying pressure and push the price higher; and sell after negative news which results in a drop in price. While there is no doubt that news releases create expectations among investors, there are only a few researches conducted recently in predicting the price movement using this information. Mittermayer [14] proposed a trading system to predict stock price trends immediately after the release of a news article through text mining techniques. They found the system significantly outperforms a random trader. However, only news that is directly related to the stock is included in their study while NofSinger's study suggests that both the firm specific and the general economics news affect trading behaviors [16].

However, using investors' expectations caused by news alone as a trading strategy is inadequate, as concluded by Brown and Cliff [23]. Therefore, this paper proposes to combine the information from both the news release and technical indicators to enhance the predictability of the daily stock price trends. The news articles used are composed of both company specific and relevant market sector news to reflect the overall impact of the media. Text mining techniques are employed to encode the news articles by forming and extracting important concepts. While support vector machine (SVM) [2], a supervised learning method, is applied as classifier. The resulting system is shown to be more accurate than the one that uses only single source of information.

The rest of the paper is organized as follow. Section 2 briefly introduces SVM and its application in financial and text mining. Section 3 presents the architecture of the system and design of various components. Research data and experiment results are detailed in Section 4. Conclusion and future research directions are given in Section 5.

2 Support Vector Machine

The SVM, originated from the work of Vapnik [22], is now being widely used to solve classification and prediction problems. SVM performs classification by constructing a hyperplane that separates the input space into two classes. It attempts to find the maximum margin hyperplane so that the separation between the decision classes is maximized. The input vectors that define the width of maximum margin are called support vectors and all the other points are not important in defining the separation. SVM maps the original input feature space into a higher dimension so that it can be separated by a linear model in the high dimensional space.

However, not all the problems can be separated by this *hard margin*. Therefore, in case of non-separable feature spaces, a *soft margin* is applied to allow some points to be misclassified. It chooses a hyperplane that separates the inputs as cleanly as possible, while still maximizing the distance between the support vectors. a penalty parameter of the error term C , ($C > 0$) can be set to be the upper bound in order to control the amount of deviation to be tolerated.

SVM has several advantages: it has little control parameters that need to be selected, over-fitting is unlikely to occur and it does not get trapped in a local optimum. It has been used in both areas of financial forecasting and text mining [8, 11]. There is a number of researches that has apply SVM to financial time series predictions and showed that SVM outperforms back-propagation networks [11, 20]. Dumais et al and Joachims demonstrated the applicability of SVM to text clustering applications and also suggest that SVM has outperformed others [3, 8].

3 System Architecture and Design

An overview of the system architecture is shown in Figure 1. The final prediction is determined base on two factors: the forecasting of price movement based on the technical indicators extracted from historical data and the combined impact of direct and indirect news articles related to the stock.

3.1 Price Forecast

Seven technical indicators [11, 24] are selected and computed for each trading day from the prices in the past five days. Table 1 presents the formula for each feature.

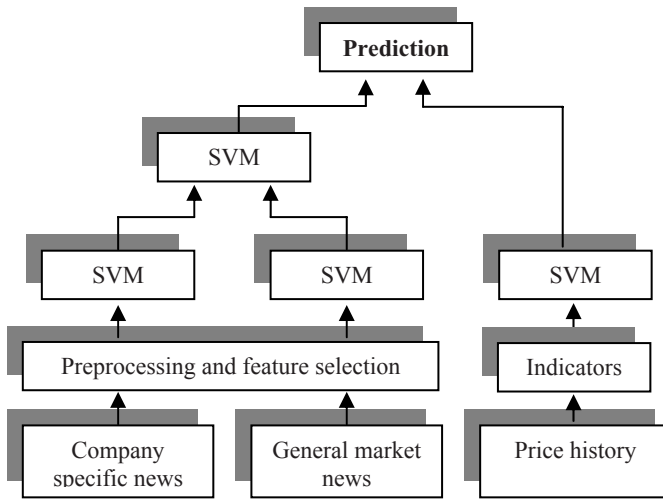


Fig. 1. Overview of the system architecture

Table 1. Summary of price features

Feature	Formula	Feature	Formula
Stochastic %K	$\frac{C_t - LL_n}{HH_n - LL_n}$	William's %R	$\frac{H_n - C_t}{H_n - L_n} \times 100$
Stochastic %D	$\frac{\sum_{i=0}^{n-1} \%K_{t-i}}{n}$	A/D Oscillator	$\frac{H_t - C_{t-1}}{H_t - L_t}$
Momentum	$C_t - C_{t-4}$	Disparity 5	$\frac{C_t}{MA_5} \times 100$
Rate of Change	$\frac{C_t}{C_{t-n}} \times 100$		

C_t is the closing price at day t , L_t is the lowest price at day t , H_t is the highest price at day t , MA_n is the moving average of the past n days, LL_n and HH_n is the lowest low and highest high in the past n days, respectively.

The direction of next day's price movement is categorized into two classes: '1' or '-1'. '1' indicates the next day's closing price is higher or equal to today's closing price, and '-1' indicate a drop in share price.

3.2 News Prediction

There are two groups of news releases: the ones that directly relate to the stock and the ones that relate to the general market. Each group is trained separately as they may have different measure of influence on the stock. The categorization of the output classes is the same as in price forecasting. News is assumed to have valid influence on the stock only on the same day it is published.

The vector space model [19] is a commonly used technique in text mining, which has been successfully used in document categorization [12]. This model is employed in this study to represent the document in high dimensional space. It represents each document as binary vectors where each element is a word from a vocabulary. The elements will have a value of 1 if the corresponding word is present within the document or have a value of zero otherwise. A weight can be associated with each element to reflect their relative importance.

The preprocessing stage starts by first removing common stop words (such as "the", "a", etc) from each documents, and then the remaining words are tagged with their corresponding Part-Of-Speech (POS) tag. Instead of using each word directly, a background thesaurus WordNet [4] is used to replace words by higher level concepts [7]. WordNet is a semantic network that can give hierarchical hypernyms and hyponyms relations between words. The use of concepts increases the flexibility of the system to be able to account for vocabulary changes, as well as reduces the dimensions of feature space. The POS tag generated earlier is utilized to help disambiguating the word when assigned to WordNet.

The concepts are weighted by the conventional multiplicative combination of term frequency (TF) and the inverse document frequency (IDF), so that terms occur more often in a document and/or rarer in other documents will be given a higher weight. Moreover, those concepts that only occur in one class but not in the other are given a higher weight (multiplied by an arbitrary constant, 2 in this case) to help better distinguish between classes. Examples of some the unique concepts from documents that are considered as "good news" are: establishment, accumulation, growth, etc; while the ones from "bad news" are: separate, discharge, impair, etc. The feature space is then reduced to be the top 30 concepts with the highest weights, which are used to code each document.

The two groups of news are trained and classified using SVM and their results feed into another SVM to produce the combined prediction of price trends.

3.3 SVM

In this study, Gaussian RBF kernel and polynomial kernel are used for SVM. The only controlling parameters are the upper bound C and the feature width σ in case of BRF or the power d in case of polynomial. These parameters are varied to ensure the optimal values are selected. Table 2 below presents an example of the performance of SVM with different parameters on the price data set.

Table 2. Prediction performance with various parameter values

Parameter value	Hit ratio (%)
<i>C</i> = 10	
$\sigma = 1$	35.5
$\sigma = 3$	58.8
$\sigma = 5$	50.0
$\sigma = 7$	50.0
$\sigma = 3$	
<i>C</i> = 1	50.0
<i>C</i> = 20	61.7
<i>C</i> = 50	52.9
<i>C</i> = 10	
<i>d</i> = 2	57.6
<i>d</i> = 3	52.9
<i>d</i> = 4	51.9
<i>d</i> = 2	
<i>C</i> = 1	38.8
<i>C</i> = 20	47.1
<i>C</i> = 50	44.1

The results obtained conforms to Tay and Cao's findings where the prediction performance deteriorates when the value of *C* and σ are either too small or too large [20]. Finally, the value of *C* is set to be 20, σ to be 3 and *d* to be 3.

Gaussian RBF kernel is used most of the time as it performs slightly better and takes less time. However when classifying the news articles, it often reaches 100% classification rate for one class, yet 0% for the other. Therefore, polynomial kernel is used instead in this case as it is less biased toward one class.

4 Experiment and Results

The research data used in this study is the daily prices (open, high, low, close) of BHP Billiton Ltd. (BHP.AX) of Australian Stock Exchange between March 1st, 2005 and May 31st, 2006. As well as the news articles related to BHP and its market sector in the same period that are published on Australian Financial Review, a major newspaper on business, finance and investment news in Australia. BHP is chosen because not only it has a large trading volume (20 Million on average) so it can be assumed that the transactions can take place whenever required, but also due to its popularity that attracts a lot of media attentions. Since BHP mainly involves in material mining, articles that concerns the directions of the metal market are included as general economic news. However, any press releases that only report the outcome of the day before are specifically excluded as it gives no new information. The data points in the first 12 months were used as training set, while the remaining two months serves as validation set. There are 286 training data and 34 holdout data from the price section. And for the news section, there are 120 training data and 28 holdout

data within the direct news category, while 53 training data and 15 holdout data are in the indirect news category. Table 3 below shows the prediction accuracy for using different data sets as inputs.

Table 3. Prediction accuracy

Data sets	Accuracy (%)
Price	58.8
Direct news	62.5
Indirect news	50.0
Combined news	64.7
Price & News	70.1

While the prediction accuracy is comparable to that obtained by Kim [11] using technical indicators only (on a different stock index), it is clear that with the combined information from both time series and textual data, the performance of stock trend prediction is noticeably improved.

It is also observed that when the predictions made from the numerical data are in conflict with the predictions from news articles, the later is more accurate most of the time. In the training data set, there are 39 inconsistent predictions, and the combined news predictions are correct for all of them and in the testing data, the accuracy is four out of six. Thus, news predictions in this case are considered more important than the data predictions.

4.1 Market Simulation

A market simulation is conducted to evaluate the profitability of the system under real life conditions. It is assumed that the initial investment is \$10,000 and each transaction (buy/sell) incurs a fee of \$20. The two months validation data (April and May 2006) used in previous section are employed in this test. Figure 2 below shows the price movements of BHP during those two months, which is a reasonably representative example. Further more, in order to avoid excessive transaction charges that will result from frequent operations, each day is limited to one transaction (buy/sell) only.

Three sets of similar strategies are used in this study for different input sets. If the prediction is based on past prices only, then the strategy for buy, sell and hold follows five simple rules:

- If the predicted trend is positive and the share has been bought, then hold.
- If the prediction is negative and the share hasn't been bought, then do nothing.
- If the prediction is positive and the share hasn't been bought, then buy.
- If the prediction is negative and the share has been bought, then sell.
- All transaction take place at the end of each trading day, thus the closing price is assumed to be the trading price.

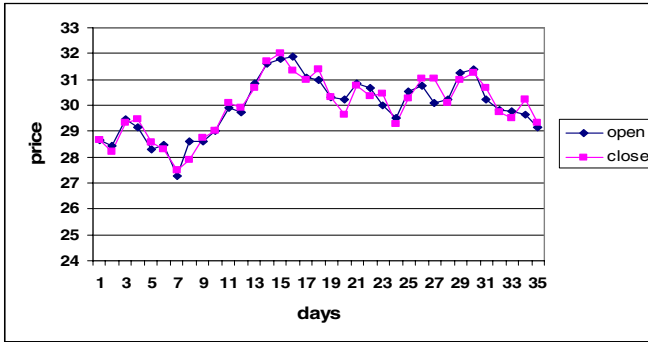


Fig. 2. Plot of closing/opening prices vs. date of BHP.ax during April and May, 2006

If the prediction is based on news only, then the above tactics are no longer applicable. This is due to the fact that news articles for each day can only be obtained in the morning before stock market opens. Therefore, changes are made to accommodate for the late arrival of news input and it is summarized as follows:

- Assuming the overall trend of the stock is rising (this assumption certainly may not be valid and it is solely used to simplify the experiment)
- If the news prediction is positive and the share hasn't been bought, then buy at opening price.
- If the prediction is positive and the share has been bought, then hold.
- If the prediction is negative and share hasn't been bought, buy at closing price.
- If the prediction is negative and the share has been bought, sale at opening price.
- If the news prediction is absent, then buy at closing price.

A recent study on the relationship between public announcements and stock volatility from Australian Stock Exchange suggests that the non-trading period overnight acts as a barrier for information to be reflected on the stock price. Therefore, the difference between yesterday's closing price and today's opening prices is normally negligible [25]. It can also be seen from Figure 2 that the opening price follows the trace of the closing price really closely. This justifies the above strategy to operate at the opening price instead of the closing price, when the news releases take place during non-trading hours.

If information from both the news releases and past prices are combined together, then the decision follows the predictions made from technical indicators at a day's close and it is revised at the next day's open based on the news predictions (if present). If the two predictions are contradictory, then the outcome of the news always supersedes, as discussed earlier. It operates as follows:

- If news releases are absent, the strategy is unchanged from strategy one above.
- If the share has been bought and the news prediction is negative for that day, the share will be sold at the opening price

- If the share has been bought and the news prediction is positive, do nothing
- If the share has been sold and the news prediction is negative, do nothing.
- If the share has been sold and the news prediction is positive, buy at the opening price.

Table 4 below displays the net compound profit of the system in two month with different information input. A trading system that employs random strategy that has approximately the same number of transactions is used as the benchmark for comparison.

It can be seen that by supplementing conventional technical indicators with the influences of news releases, the proposed system demonstrates promising results under real life situation. Since the stock market is an extremely complicated system, richer information source will be able to provide a better model.

Table 4. Compound net profit for different inputs

System	No. of trades	Net profit
Random	8	\$-54
Price Only	9	\$284
News Only	7	\$275
Price and News	11	\$511

5 Conclusions

In this paper, news paper releases are combined with the technical indicators to predict daily direction of a stock price using SVM. The case study results showed that both the prediction performance and the profitability of the system are enhanced.

However, the current system only categorizes the output into simple rise/fall without specifying the level of change. Therefore, future research efforts will focus on refining the prediction of price trends. Furthermore, the general applicability of the system also needs to be examined further by applying it to other stocks in different sectors.

References

1. Choi, J.H., Lee, M.K., Rhee, M.W.: Trading S& P 500 Stock Index Futures Using a Neural Network. Proc. of the Annual Int. Conference on Artificial Intelligence Applications on Wall Street, New York (1995) 63–72
2. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press, Cambridge (2000)
3. Dumais, S., Platt, J., Heckerman, D.: Inductive Learning Algorithms and Representations for Text Categorization. Proc. of the 7th Int. Conf. on Information and Knowledge Management, ACM Press (1998) 148-155

4. Fellbaum, C.(Ed.), WordNet.: An Electronic Lexical Database. Cambridge, Massachusetts: MIT Press (1998)
5. Fung, G.P.C., Yu, J.X., Lam, W.: News Sensitive Stock Trend Prediction. Proc. of 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Taiwan (2002) 289–296
6. Hellstrom, T., Holmstrom, K.: Predicting the Stock Market. Technical Report Series IMA-TOM-1997-07 (1998)
7. Hotho, A., Stumme, G.: Conceptual Clustering of Text Clusters. Proc. Fachgruppentreffen Maschinelles Lernen Hannover (2002) 37-45
8. Joachims, T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Proc. of the 10th European Conference on Machine Learning, Springer, Heidelberg (1998) 137-142
9. Hassan, M.R., Nath, B.: Stock Market Forecasting Using Hidden Markov Model: a new approach. Proc. of 5th Int. Conf.on intelligent systems design and applications (2005)
10. Kim, K., Han, I.: Genetic Algorithms Approach to Feature Discretization in Artificial Neural Networks for the Prediction of Stock Price Index. Expert Syst. Appl. **19** (2) (2000) 125–132
11. Kim, K.J.: Financial Time Series Forecasting Using Support Vector Machines. Neurocomputing **55** (2003) 307-319
12. Rosso, P., Ferretti, E., Jimenez, D., Vidal, V.: Text Categorization and Information Retrieval Using WordNet Senses. CICLing 2004, LNCS **2945** (2004)
13. Mitchell, M.L., Mulherin, J.H.: The Impact of Public Information on the Stock Market. Journal of Finance **49** (3) (1994) 923–950
14. Mittermayer, M.: Forecasting Intraday Stock Price Trends with Text Mining techniques. Proc. of the 37th Hawaii International Conference on System Sciences, Hawaii (2004)
15. Mukherjee, S., Osuna, E., Girosi, F.: Nonlinear Prediction of Chaotic Time Series Using Support Vector Machines. Proc. of the IEEE Workshop on Neural Networks for Signal Processing, Amelia Island, FL (1997) 511–520
16. Nofsinger, J.R.: The Impact of Public Information on Investors. Journal of Banking & Finance **25** (2001)1339-1366
17. Quah, T.S., Srinivasan, B.: Improving Returns on Stock Investment through Neural Network Selection. Expert Syst. Appl. **17** (1999) 295–301
18. Romahi, Y., Shen, Q.: Dynamic Financial Forecasting with Automatically Induced Fuzzy Associations. Proc. of the 9th Inter. Conf. on Fuzzy systems (2000) 493-498
19. Salton, G., McGill, M.J.: Introduction to Modern Information Retrieval. McGraw-Hill, New York (1983)
20. Tay, F.E.H., Cao, L.: Application of Support Vector Machines in Financial Time Series Forecasting. Omega **29** (2001) 309–317
21. Thissen, U., Brakel, R. van, Weijer, A.P.de, Melssen, W.J., Buydens, L.M.C.: Using Support Vector Machines for Time Series Prediction. Chemom. Intell. Lab. Syst. **69** (2003) 35–49
22. Vapnik, V.N.: Statistical Learning Theory, Wiley, New York (1998)
23. Brown, G.W., Cliff, M.T.: Investor Sentiment and the Near-Term Stock Market. Journal of Empirical Finance **11** (2004) 1-27
24. Achelis, S.B.: Technical Analysis from A to Z. Probus Publishing, Chicago (1995)
25. Kalev, P.S., Liu, W.M., Pham, P.K., Jarnecic, E.: Public Information Arrival and Volatility of Intraday Stock Returns. Journal of Banking and Finance **28** (2004) 1441-1467

26. Leigh, W., Purvis, R., Ragusa, J.M.: Forecasting the NYSE Composite Index with Technical Analysis, Pattern Recognizer, Neural Network, and Genetic Algorithm: a Case Studying Decision Support. *Decision Support Systems* **32** (2002) 361–377
27. Lee, R.S.T.: iJADE Stock Advisor: an Intelligent Agent Based Stock Prediction System Using Hybrid RBF Recurrent Network. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans* **34** (3) (2004) 421-428
28. Abdullah, M.H.L.b, Ganapathy, V.: Neural Network Ensemble for Financial Trend Prediction. *Proc. TENCON* **3** (2000) 157-161
29. Abraham, A., Nath, B., Mahanti, P.K.: Hybrid Intelligent Systems for Stock Market Analysis. *Proc. of the Inter. Conf. on Computational Science* (2001) 337-345

Project-Based Artificial Neural Networks Development Software and Applications

Xiaofeng Lin, Shaojian Song, Chunling Song, Qingbao Huang, and Xiao xiao Song

College of Electrical Engineering, Guangxi University,
530004, Nanning, Guangxi, P.R. China
gxulinx@163.com

Abstract. We have designed a kind of practical artificial neural network development software for ordinary engineering technicians. This software, with graphic interface, not only supports multiple types and algorithms of artificial neural networks, but also supports the IEC 61131-3 International Standard. This article, through three application examples of artificial neural networks, shows the feasibility and the easy implementation of this development software, as well as the realization of artificial neural networks in IEC 61131-3 Standard-based software. It also shows the application value of artificial neural networks development tool and the realistic significance of applying artificial neural networks control in the projects.

1 Introduction

Artificial neural networks have massive parallel processing and distributed information storage capacity, and they have good adaptation, self-organization, strong learning function, association function and fault tolerance function. Thanks to these advantages, artificial neural networks have been widely applied in industrialized countries. But in China, because of some engineering technicians' lack of in-depth understanding of the artificial neural networks design, and the complexity of training and parameter adjustment in practical application of artificial neural networks, it is not easy for them to apply artificial neural networks in regular industrial control equipment such as PLC, industrial PC, and DCS etc. In order to popularize the applications of artificial neural networks, we have developed a kind of artificial neural networks development software NNDS. NNDS can effectively combine with industrial equipment and industrial control software, and it can also operate integrally with such industrial hardware equipment as PLC, industrial PC, DCS etc., IEC 61131-3 International Standard-based industrial control software, as well as configuration software. NNDS has promoted the applications of artificial neural networks.

2 Composition of NNDS

Based on the demand analysis and functional requirements of artificial neural network-controlled development software, and abided by the modular approach, we

have designed the general structure of NNDS, which mainly includes artificial neural networks, design of training and algorithms, as well as data interface (see Figure 1).

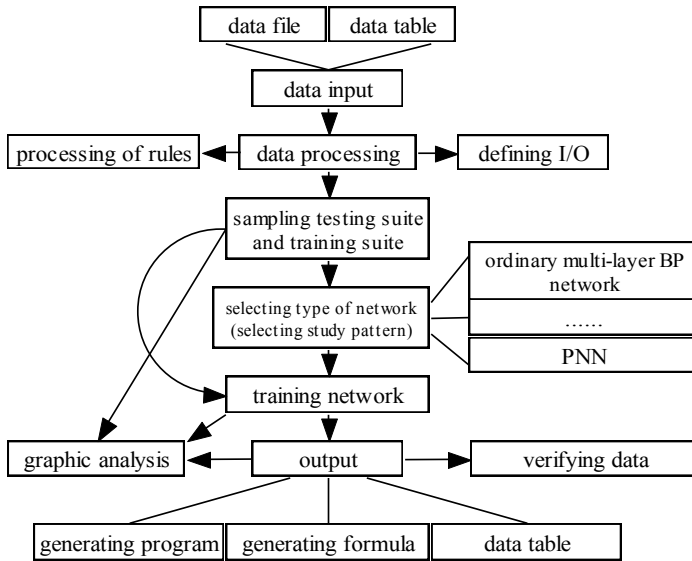


Fig. 1. Basic Structure of NNDS

3 Functions of Each Component

3.1 Design of Type, Training and Algorithms of Artificial Neural Networks[2][3]

The whole artificial neural networks software is designed with modular approach, which includes data input, data preprocessing, type-selecting, training networks, networks output, etc. Ordinary data file (e.g. txt file) and data table file (e.g. Excel file) can be indirectly input into the data input module.

Data preprocessing module: Define the input and output variables, and set the maximum values and minimum values for input and output.

Type-selecting module: With the adoption of multiple types of artificial neural networks, this software can be applied to the following types of artificial neural networks. Appropriate type of network is selected in real control based on different objects:

- (1) Standard Linkage Multi-layer Forward Artificial Neural Networks;
- (2) Double Parallel Forward Artificial Neural Networks;
- (3) Elman Forward Artificial Neural Networks;
- (4) Ward Forward Artificial Neural Networks;
- (5) Fuzzy Artificial Neural Networks;
- (6) Kohonen Networks;
- (7) Normal Regression Networks.

Different class definitions are adopted for different artificial neural networks, and different training methods are adopted according to different networks. We can use the figure to monitor the decrease curve of error of training suite during the whole training process. The networks result can not only be input directly by the data output module as table file or output as C function, but also be generated as a controller program.

3.2 Data Interface

3.2.1 OPC Mode

OPC Communication Mode: OPC (OLE for Process Control) is an industrial technical code and standard to resolve the communication problems between application software and various device drivers. By following the OPC standard and designing the OPC server and client, NNDS can communicate with all OPC standard-based industrial control equipment and software.

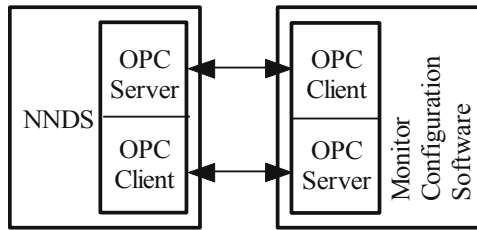


Fig. 2. OPC Mode

3.2.2 ODBC Mode

ODBC Communication Mode: ODBC (Open Database Connectivity), a standard brought forward by Microsoft, aims to realize the interconnection among heterogeneous database. NNDS, through open ODBC interface, can operate integrally with a variety of configuration software, such as Chinese configuration software, Force Control SCADA software, Configuration Software KingView, MCGS, Century star Configuration Software, etc. But the real-time capability of this communication mode is not as good as others, and therefore it is not appropriate to be applied in the occasion with high real-time requirements.

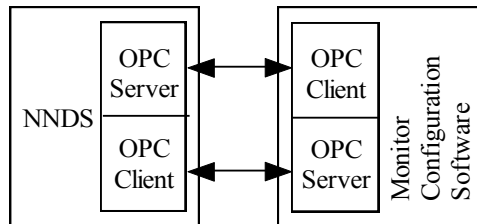


Fig. 3. ODBC Mode

3.2.3 Embedded Mode[4]

The IEC 61131-3 Standard, a PLC standardization programming standard, was issued by International Electrotechnical Commission (IEC) in 1993. The IEC 61131-3 Standard has prescribed five kinds of programming languages. With many advantages of the IEC 61131-3 automation programming language, it has become an international standard with extensive application bases in the automation industry. The standard is not only applied in PLC, but also widely used in distributed control systems, industrial control computers, numerical control systems, remote terminal units and so on. Many of the world leading automation equipment manufacturers have manufactured products under IEC 61131-3 Standard. IEC 61131-3 Standard allows users by themselves to define function blocks, so the users can easily create advanced control algorithms and special functions into function blocks and add them into the function block database within the software.

NNDS provides an interface for IEC 61131-3 Standard-based software. Artificial neural networks developed by NNDS, after certain processing, can be altered into a function block of IEC 61131-3 Standard-based software. These kinds of software include OpenPCS of infoteam in Germany, the Paradym-31 used by ADAM5510 of ADVANTECH and OMC of Mirle, etc.

Applications of NNDS in IEC 61131-3 Standard-based software are indicated in Figure 4.

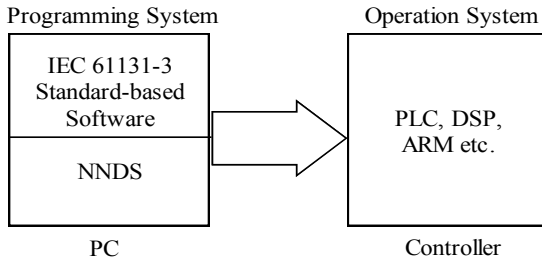


Fig. 4. Embedded Mode

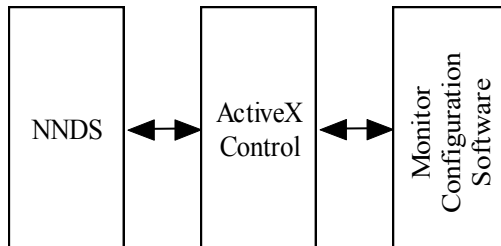


Fig. 5. ActiveX Control Mode

3.2.4 ActiveX Control Mode

Some kinds of industrial monitor configuration software provide ActiveX control to communicate with third-party software, as shown in Figure 5. NNDS, through these ActiveX control, is able to read and write the real time data of configuration software, such as ForceControl.

4 Application Examples

4.1 Designing a PID Artificial Neural Networks Controller with NNDS to Control Three-Tank Liquid Level Control Apparatus

As is shown in figure 6, the structure of a PID artificial neural networks controller is a 3-layer networks and includes the input layer, the hidden layer, and the output layer, with their structure being 2-3-1. There are two neurons in the input layer, inputting respectively the given output $r(t)$ and the output value of controlled object $y(t)$; there are three neurons in the hidden layer, and each output function of the neurons is different, referring respectively to proportion, integral and differential; the output of output layer is the control volume of the controlled system. By applying the PID artificial neural networks as a controller, the effective control of the controlled object can be achieved without needing to identify the complication.

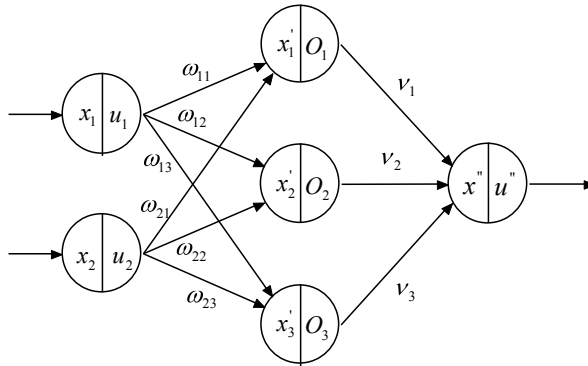


Fig. 6. Structure of PID Artificial Neural Networks

The controlled object of this experiment is a two-tank object that is composed of an experimental apparatus controlled by three-tank liquid level (we use LT1 and LT2 in the experiment), and the control objective is the liquid level of LT2, as illustrated in figure.

The control trend graph attained from the level control experiment is shown in figure 8. In the figure, the abscissa refers to the running time of monitor configuration software, and the ordinate refers to the controlled liquid level (unit: mm). As we can

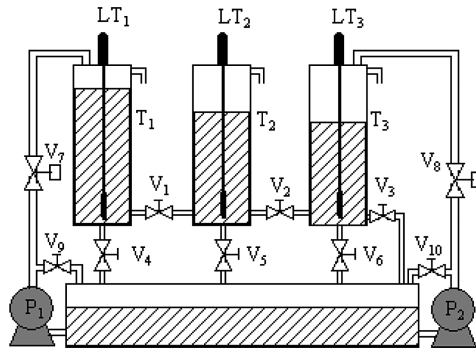


Fig. 7. Three-tank Liquid Level

see from the figure, the overshoot of PID artificial neural networks control system is about 8%, and the control accuracy is within $\pm 1\%$, attaining relatively good control effect. Therefore, the PID artificial neural networks controller developed by NNDS can well control the two-tank object.

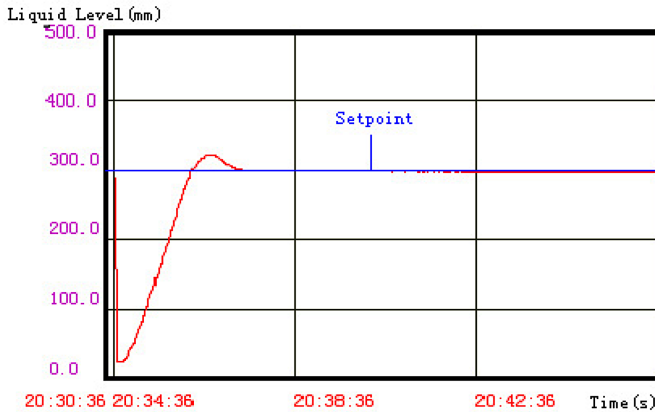


Fig. 8. Graph of Control Results

4.2 Application of NNDS in Short-Term Load Forecast in Power Systems

Short-term load forecast in power system is the basis of optimizing the operation of power systems and is one of the important tasks for power companies. We use artificial neural networks to forecast short-term load 24 hours on a particular day. The structure of artificial neural networks includes seven nodes in the input layer: $I(n-14, j)$, $I(n-7, j)$, $I(n-2, j)$, $I(n-1, j)$, $I(n, j-2)$, $I(n, j-1)$ and $T(n)$, and one node in the output layer: $I(n, j)$.

Of which, $I(n, j)$ refers to the load at the j th hour on the n th day; $I(n-14, j)$ refers to the load at the j th hour on the $n-14$ th day; $I(n-7, j)$ refers to the load at the j th hour on the $n-7$ th day; $I(n-2, j)$ refers to the load at the j th hour on the $n-2$ th day; $I(n-1, j)$ refers to the load at the j th hour on the $n-1$ th day; $I(n, j-2)$ refers to the load at the $j-2$ th

hour on the n th day; $I(n, j-1)$ refers to the load at the $j-1$ th hour on the n th day; $T(n)$ refers to the average temperature on the n th day.

Table 1. Forecast Results and Error of Multi-layer Forward Artificial Neural Networks Based on Different Types of Networks

Time (h)	Actual (MW)	Net1 (MW)	Error1 (%)	Net2 (MW)	Error2 (%)	Net3 (MW)	Error3 (%)	Net4 (MW)	Error4 (%)
1	2158	2176	0.8	2163	0.3	2195	1.7	2118	-1.9
2	2127	2123	-0.2	2112	-0.7	2176	2.3	2164	1.7
3	2119	2076	-2	2082	-1.7	2098	-1.0	2024	-4.5
4	2109	2115	0.3	2118	0.4	2141	1.5	2076	-1.6
5	2108	2109	0.05	2113	0.2	2120	0.6	2075	-1.6
6	2197	2243	2.1	2250	2.4	2230	1.5	2245	2.2
7	2437	2448	0.5	2443	0.2	2449	0.5	2338	-4.1
8	2497	2536	1.6	2545	1.9	2533	1.4	2469	-1.1
9	2672	2658	-0.5	2650	-0.8	2657	-0.6	2629	-1.6
10	2691	2710	0.7	2705	0.5	2715	0.9	2717	1.0
11	2761	2844	3.1	2868	3.9	2843	3.0	2844	3.0
12	2668	2706	1.4	2702	1.3	2770	1.2	2709	1.5
13	2382	2498	4.9	2490	4.5	2540	6.6	2478	4.0
14	2541	2531	-0.4	2544	0.1	2589	1.9	2506	-1.4
15	2610	2722	4.3	2728	4.5	2700	3.4	2728	4.9
16	2642	2622	-0.8	2662	0.8	2665	0.9	2665	0.9
17	2848	2833	-0.5	2866	0.6	2875	0.9	2799	-1.7
18	2967	2833	-4.5	3008	1.4	2999	1.1	2956	-0.4
19	3205	3291	2.7	3296	2.8	3270	2.0	3277	2.3
20	3128	3174	1.5	3156	0.9	3146	0.6	3130	0.06
21	2984	3011	0.9	3000	0.5	2993	0.3	2990	0.2
22	2740	2763	0.8	2753	0.5	2787	1.7	2771	1.1
23	2436	2466	1.2	2466	1.2	2464	0.3	2412	1.0
24	2188	2227	1.8	2236	2.2	2310	5.6	2155	-1.5

Different types of multi-layer forward artificial neural networks were trained, and then the well trained networks were applied to forecast the load at each integral point of time (24h). The forecast results are shown in the table (In order to save space, the sampled data are not herein listed). Within the table, Actual refers to the actual load value at a corresponding period; Net1 refers to the forecast value of typical and standard-linkage three-layer forward networks, and Error1 refers to its relative error; Net2 refers to the forecast value of standard-linkage four-layer forward networks, and Error2 refers to its relative error; Net3 refers to the forecast value of double parallel forward networks, and Error3 is its relative error; Net4 refers to the forecast value of Elman forward networks, and Error4 is its relative error.

4.3 Implementation of IEC 61131-3 Standard-Based Artificial Neural Networks Control System[5]

The control system is composed of industry PC, ADVANTECH IO Board, OpenPCS (a kind of IEC 61131-3 Standard-based development software, it is developed by infoteam in Germany and provides user-defined function block interface programmed in standard C language), as well as the controlled object. The controlled object is a vertical experimental furnace, within which the temperature field is irregularly distributed and keeps changing as the time changes, as shown in Figure 9. In this experiment, NNDS is applied to create a single neuron adaptive PSD (proportion, summation and differential) intelligent control algorithm function block, then the function block is embedded into the IEC 61131-3 Standard-based control software platform to realize the control of furnace.

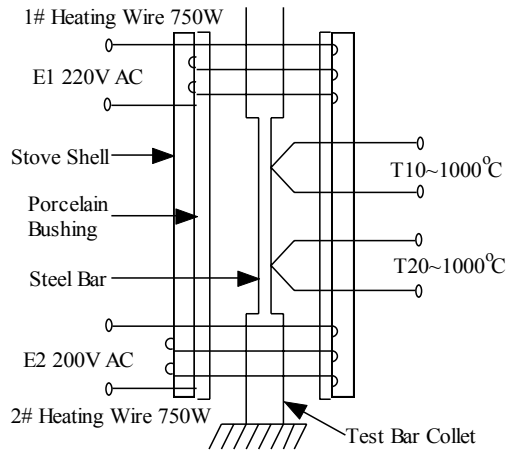


Fig. 9. Structure Plan of Vertical Furnace

The structure of single neuron adaptive PSD intelligent controller is shown in Figure 10.

In Figure 10, Σ refers to the neuron; y_r refers to the set point of temperature; y refers to the measured temperature; and u refers to the control volume.

Within the NNDS, artificial neural networks module created based on the adaptive PSD algorithm, ADVANTECH board driver module, and the sampling period timing function block are programmed into user-defined function blocks in standard C language and then added into the function block database of OpenPCS. The control program programmed by FBD function block diagram language is shown in Figure 11. The control software is able to directly read data and output control volume through an ADVANTECH board card. The control result illustrated by ForceControl SCADA is indicated in Figure 12.

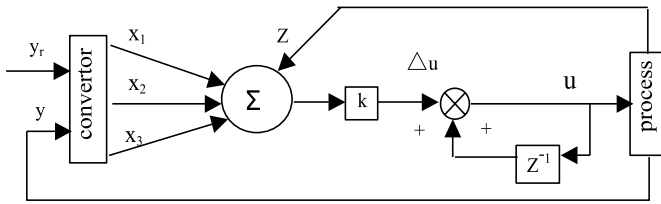


Fig. 10. Control Diagram of Single Neuron Adaptive PSD

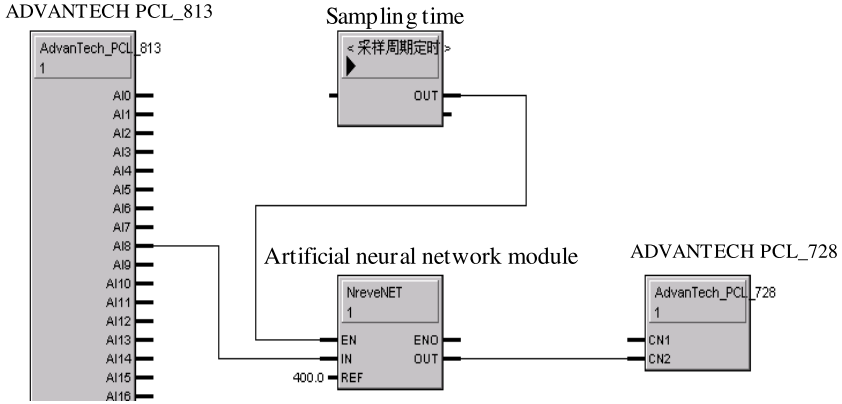


Fig. 11. FBD Control Program

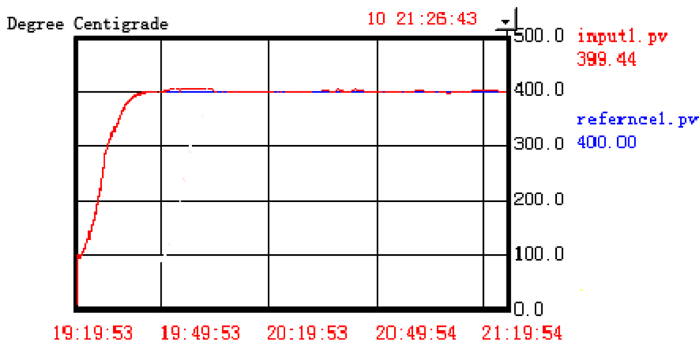


Fig. 12. Control Result Graph of Single Neuron Adaptive PSD

5 Conclusion

Artificial neural networks development software NNDS, with graphic display, is able to accomplish the design and training process of artificial neural networks, and it can operate integrally with configuration software and the IEC 61131-3 Standard-based software. NNDS makes it easier to develop and improve the artificial neural networks system and it reduces greatly the debugging time in practical application and

improves the production efficiency, therefore, it plays significant and positive roles in applying artificial neural networks to the projects.

References

1. Widrow, B., et al.: Neural Networks Application in Industry. Business and Science. Communication of the ACM **37** (1994) 93-105 <http://www.neuroshell.com>
2. Antsaklis: Neural Networks in Control System[J]. **12** (2) (1992) 8-10
3. Lewis, R.W.: Programming Industrial Control System Using IEC 1131-3. IEE Control Engineering. The Institution of Electrical Engineers (1998)
4. Hassapis, George: Soft-testing of Industrial Control Systems Programmed in IEC 1131-3 languages. ISA Transactions **39** (2000) 345-355

Effects of Salinity on Measurement of Water Volume Fraction and Correction Method Based on RBF Neural Networks

Chunguo Jing^{1,2}, Guangzhong Xing¹, Bin Liu¹, and Qiuguo Bai²

¹ Yanshan University, Qinhuangdao 066004, China
{jingchunguo}@163.com

² Northeastern University at Qinhuangdao, 066004, Qinhuangdao, China

Abstract. The gamma ray dual modality densitometry was presented to measure salinity independent of water volume fraction in pipe flows. The simulation geometries of the dual modality densitometry were built using Monte Carlo software Geant4. Computer simulations were carried out with different types of salt and various salinity. The results show that type of salt and salinity have significant effects on the water volume fraction measured by dual modality densitometry. By means of measuring attenuation of transmitted and scattered radiation of dual modality densitometry, the information about the salinity changes can be obtained. But it is difficult to calculate WVF and salinity from dual modality densitometry models. The RBF neural networks were used to predict salinity and water volume fraction. The results show that the predicting values fit true values well. It was demonstrated that the water volume fraction measuring errors caused by salinity can be reduced by using RBF neural networks.

1 Introduction

The petroleum industry has a need for accurate measurement gas volume fraction (GVF) and water volume fraction (WVF) in the pipe flows [1]. Gamma ray densitometries are often used to measure GVF and WVF due to their robustness and non-intrusive character. The gamma ray densitometry consists of a source and detectors, and its basic principle is to measure ray beams attenuation. The attenuation of gamma ray beams depends on the composition of the flow, the photon energy and the diameter of the pipe. Since both the photon energy and the pipe diameter are constant, the gamma ray attenuation will be influenced only by the change of the composition of the flow. According to this feature, gamma ray densitometry can be used to measure phase volume fraction of oil water and gas mixture flow.

Dual-energy and single-energy gamma ray densitometry are two main types of gamma densitometry. Dual-energy gamma densitometry can measure oil, water and gas three components of volume fraction of mixture flow [2]. The single-energy densitometry was used in situations where there is no gas in pipeline [3]. Because single-energy densitometry can set up exact models, it was widely used in petroleum industry. But the water from oil wells could be the mixture of formation water,

injected water and cleaning well water, all of which have different composition and often change. The dominating composition in water is salt of various types. The typical salt includes positive ions such as Na^+ , Mg^{2+} , Ca^{2+} etc. and negative ions such as Cl^- , HCO_3^- , SO_4^- etc. These ions are equal to new phase adding in mixture flows. For these reasons, the precision of measurement WVF using single-energy gamma ray densitometry will be affected by the salinity.

Dual modality densitometry (DMD) is another type of gamma densitometry, it was primarily used to measure oil water gas three-phase flow [4]. G. A. Johansen etc. have studied salinity independent measurement of gas volume fraction in oil water gas pipe flows using DMD and carried out captive test in their paper [5]. M. B. Holstad etc. have measured salinity and the types of salt in the produced water of offshore oil wells using DMD [6]. In this study, the effects of salinity on measuring WVF were analyzed using Geant4 Monte Carlo simulation package and DMD was used to measure WVF. The methods of correcting salinity effects were presented for measuring WVF based on RBF neural networks.

2 The Principle of Dual Modality Densitometry

The DMD consists one single-energy source and two detectors, one is used to detect transmitted radiation intensity and another is used to detect scattered radiation intensity. The scatter detector is positioned at the vertical direction of source and transmitted detector line. At this position, the scattering radiation energy is obviously different from the incident radiation energy and it is easy to design the measurement machinery. Figure 1 gives a simplified measurement sketch of DMD.

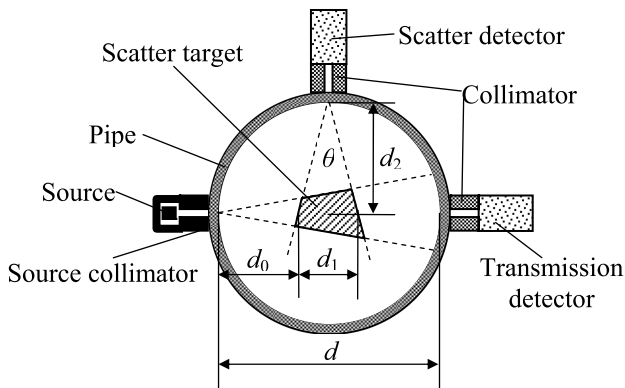


Fig. 1. Sketch of the horizontal cross-section of the simulation geometries

In figure 1, the diameter of pipe is d . The solid angle as which the scattered detector faces to the scattered target is θ . The average width of scattered target is d_1 . The distance from the center of scattered target to scattered detector is d_2 . The distance from the scattered target to the window of gamma source is d_0 . According to

the theory of gamma ray attenuation, the intensities detected by the transmitted detector and the scattered detector are [10]

$$I_t = I_0 e^{-\mu d} \quad (1)$$

$$I_s = BI_0 \frac{\mu_\sigma}{\mu} e^{-\mu d_0} (1 - e^{-\mu d_1}) e^{-\mu' d_2} \quad (2)$$

where I_0 is the intensity detected by transmission detector as the pipe is vacuum, μ and μ' are the linear attenuation coefficients of the mixture flow at the energy of incident and scattered radiation respectively. μ_σ is the attenuation coefficient of Compton scattering. B is a ratio constant between scattered intensity counted by scattered detector and all scattered intensity. The radiation energy decreased in the Compton scattering process, and the attenuation coefficient of scattered radiation is thus higher. The attenuation process of scattering is more complex than transmission from (1) and (2), it relates to the attenuation from source to scatter target, the volume of scatter target, the attenuation from scatter target to scatter detector and so on. So it is difficult to directly solve WVF and salinity from (1) and (2). In this study, a RBF neural network was used to predict WVF as salinity changing.

3 Simulation Geometries

Nuclear experiment is difficult to carry out for the reasons of ionizing radiation, measurement structure complexity and many existing radiation source types. The best way to study feasibility of nuclear measurement is to use simulation software. There are many particle simulation software such as MNC, EGS4 and Geant4 [7] using Monte Carlo methods. These softwares can be used to design instruments and to get simulation data [8]. Geant4 is an open source toolkit for the simulation of the passage of particles through matter. Its areas of application include high energy, nuclear physics, as well as studies in medical and space science. In this paper the simulation geometries was developed as a tool for the study of detector responding to different salinities. Figure 2 shows a 3D simulation geometry constructed by the Geant4 software.

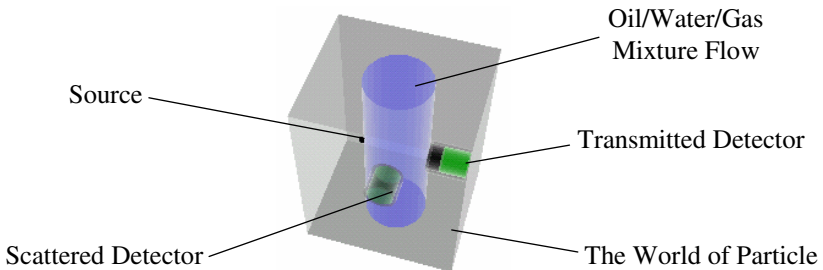


Fig. 2. The 3D simulation geometry constructed by Geant4

In figure 2, the inner diameter of pipe is 100mm. The 59.5keV gamma radiation energy produced by ^{241}Am source is used for these simulation geometries. The

detectors are the $\Phi 40\text{mm}\times 40\text{mm}$ NaI crystal. In front of the detectors and source, there are lead collimators. Instead of using crude oil in the simulation, the cetene (molecular formula, $\text{C}_{16}\text{H}_{34}$, density, 0.7733g/cm^3) was employed. The simulation events are 100,000. The threshold value for transmission detector is set at 50keV and for scatter detector it is set at 45keV to eliminates multiple scattering effects.

4 Simulation Results

The simulations were done with the geometries shown in figure 2. Four different types of salt were used in simulations, they are NaCl, MgCl_2 , CaCl_2 and NaHCO_3 . The salts were added into water and get 2%, 4%, 6%, 8% and 10% solutions of NaCl, MgCl_2 , CaCl_2 .and NaHCO_3 . All percentages are weight percent (% w/w) of salt in water.

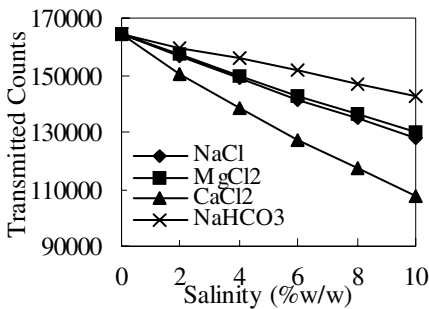


Fig. 3. The counts of transmitted detector against salinity in the simulations

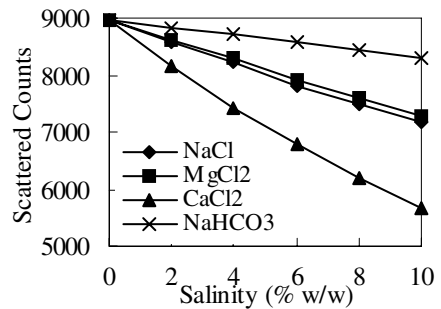


Fig. 4. The counts of scattered detector against salinity in the simulation

Figures 3 and 4 show that the counts in both detectors depend on both salinity and type of salt and at the same time the WVF keeps at 0.8. As the photons hit the detectors and its energy above the threshold values, it was counted. The results show a more linear decrease as salinity increases in figure 3 than in figure 4. This is because the transmitted radiation is only related to the components of mixed liquid and the scattered radiation is not only related to the components of mixture material but to the scattered radiation intensity, energy and the position of scattered detector. Also the scattered count is lower than transmitted count, so the statistical uncertainty becomes more significant.

As seen from the figures 3 and 4, the curve of NaCl is similar to that of MgCl_2 and has large difference to that of CaCl_2 . This is because the atomic number of natrium element approximates magnesium element, and the atomic number of calcium element is bigger than natrium and magnesium, so that the attenuation of radiation in CaCl_2 solution is larger than that in NaCl and MgCl_2 . Because the elements of NaHCO_3 are similar to those of water and oil and the atomic number of natrium

element is lower, the curve of NaHCO_3 decreases more slowly than others. From figure 3 and 4 it can be seen that the different types of salt have different attenuation for both transmitted and scattered intensities. This is equal to adding various salt phases in oil and water mixture flows, so it is difficult to calculate salinity and WVF using DMD models. In this study, the RBF neural networks were used to predict WVF and salinity.

5 Neural Networks and Results Analyze

It has been demonstrated that a neural network can approach any nonlinear function based on inputs and outputs. It can be applied to the recognition and function approximation of a wide range of situation, pattern and individual features of different systems. Bishop ect. [9] and Jing Chunguo [10] have shown that it is possible to determine oil, water and gas fraction in multiphase pipelines using neural networks based on dual-energy densitometry and DMD. The aim of this study was to apply neural network technique to measure WVF and salinity of flow using DMD.

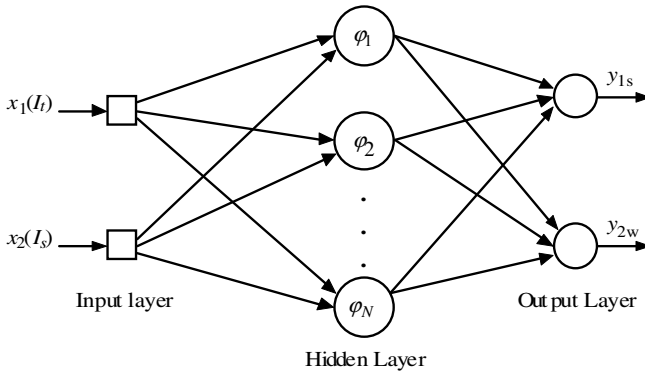


Fig. 5. A architecture of a RBF neural network

The back-propagation (BP) neural network is the most widely applied neural network technique. In general, the BP neural network encounters local minimum, slow convergence speed and convergence instability. The radial basis function (RBF) neural network is similar to the BP. It can be trained very quickly because the algorithm uses a fixed Gaussian function [11]. In this paper RBF neural network was used. A RBF neural network architecture for measuring WVF with salt in flows is shown in figure 5. The input layer consists of two neurons which are the intensities of transmitted and scattered. The output layer is two neurons predicting WVF and salinity. The hidden layer nodes has ten neurons and was called RBF units, determined by a parameter vector called center and a scalar called width. The Gaussian density function is used in the hidden layer as an activation function. The overall input-output mapping is as follows:

$$Y = \sum_{i=1}^N w_i \exp\left(-\frac{\|X - c_i\|^2}{2\sigma_i^2}\right) \tag{3}$$

where $X=[x_1(I_t), x_2(I_s)]^T$ is the input vector, c_j is the i^{th} center of RBF unit in the hidden layer, N is the number of RBF units, w_i is the weight between the hidden and output layers, respectively, $Y=[y_{1s}, y_{2w}]^T$ is output vector.

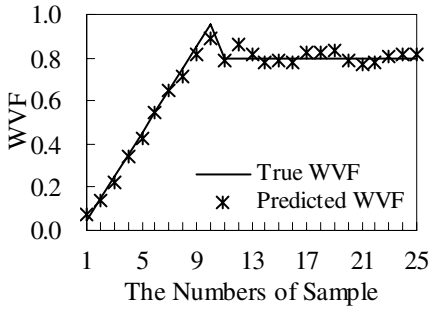


Fig. 6. The comparisons of the predicting WVF to true WVF. Form 12 sample to 25, three different types of salt with salinity from 1% to 9% in step 2%.

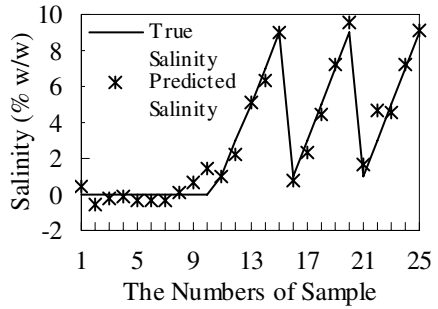


Fig. 7. The comparisons of the predicting salinity to true salinity. Form 1 sample to 10, the change of WVF was from 0.05 to 0.95 in step 0.1.

The training set and test set were produced by computer simulation. There are 26 samples in training set, in 11 samples, there is no salt in water and WVF is from 0 to 1 in step 0.1. the rest are NaCl, MgCl₂ and CaCl₂ brine whose salinity is from 2% to 10% in step 2% and at the same time WVF is 0.8. The test set corresponds to training set and has 25 samples, but WVF and salinity are different to those of the training set. The WVF is from 0.05 to 0.95 in step 0.1 with no salt in water and salinity is from 1% to 9% in step 2%. Figure 6 is the comparison of the predicting WVF to true WVF. From figure 6 it can be seen that the predicting WVF are close to true WVF. The predicting salinity fit true salinity well in figure 7. Table 1 is statistical results of predicting WVF and salinity. Although the maximal absolute error of WVF is 0.06 between predicting WVF and true WVF, the mean square error (MSE) of WVF is very low. The MSE of Salinity is 10 times lower than MSE of WVF.

Table 1. The statistical results of predicting WVF and salinity

Performance	WVF	Salinity
MSE	0.000685439	6.48106E-05
Min Abs Error	0.001114083	0.0242663
Max Abs Error	0.062112194	1.7159279

5 Conclusions

When salinity and type of salt change, the attenuations of transmitted radiation and scattered radiation detected by transmitted and scattered detectors in DMD are obvious. It can be used to measure WVF in salinity changing situation. But it is difficult to find out the algorithm from DMD models to calculate salinity and WVF. In order to verify the feasibility of RBF neural networks predicting WVF and salinity, the training and test set were produced by Geant4. The test results show that predicting WVF is close to true WVF as salinity changes. All these demonstrate that it is feasible to eliminate the effects of salinity in calculating WVF using DMD.

References

1. Thorn, R., Johansen, G.A., Hammer, E.A.: Recent Developments in Three-phase Flow Measurement. *Measurement Science and Technology* **8**(7) (1997) 691-701
2. Li, D.H., Wu Y.X., Li Z.B., Zhong, X.F.: Volumetric Fraction Measurement in Oil-water-gas Multiphase Flow with Dual Energy Gamma-ray System. *Journal of Zhejiang University SCIENCE* **6A**(12) (2005) 1405-1411.
3. Bai, Q.G., Zhang, X.B., and Jing, C.G.: Measurement Instrument of Lower Percentage of Water Content in Petroleum. *Nuclear Electronics and Detection Technology* **20**(4) (2000) 269-271
4. Tjugum, S.A., Johansen, G.A., Holstad, M.B.: The Use of Gamma Radiation in Fluid Flow Measurements. *Radiation Physics and Chemistry* **61**(3-6) (2001) 797-798
5. Johansen, G.A., Jackson, P.: Salinity Independent Measurement of Gas Volume Fraction in Oil/Gas/Water Pipe Flows. *Applied Radiation and Isotopes* **53**(4-5) (2000) 595-601
6. Holstad, M.B., Johansen, G.A.: Produced Water Characterization by Dual Modality Gamma-ray Measurements. *Measurement Science and Technology* **16**(4) (2005) 1007-1013
7. GEANT4 Collaboration. GEANT4 User's Guide. <http://geant4.web.cern.ch/geant4/>
8. Ouardi, A., Benchekroun, D., Hoummada, A., Alami, R.: Geant Simulation of the Gamma Nuclear Gauge. *IEEE Trans. Nuclear Science* **50**(4) (2003) 1257-1270
9. Bishop, C.M., James, G.D.: Analysis of Multiphase Flows Using Dual-energy Gamma Densitometry and Neural Networks. *Nuclear Instruments and Methods in Physics Research* **A327** (1993) 580-593
10. Jing, C.G., Xing, G.Z., Liu, B., Bai, Q.G.: Determination of Gas and Water Volume Fraction in Oil Water Gas Pipe Flow Using Neural Networks Based on Dual Modality Densitometry. In: Wang, J., Yi, Z. et al(eds.): *Advances in Neural Networks. Lecture Notes in Computer Science*, Vol. **3973**. Springer-Verlag, Berlin Heidelberg New York(2006) 1248-1253.
11. Laurentiu, A., Tarca, B., Grandjean, P.A., Larachi, F.: Designing Supervised Classifiers for Multiphase Flow Data Classification. *Chemical Engineering Science* **59** (16) (2004) 3303-3313

Implementation of Brillouin-Active Fiber for Low Threshold Optical Logic and Memory Based Neural Networks in Smart Structures

Yong-Kab Kim¹, Woo-Soon Kim², Yue Soon Choi¹, and Jong Goo Park¹

¹ Schools of Electrical Electronics & Information Engineering, Wonkwang University,
570-749, Iksan, Korea
ykim@wonkwang.ac.kr
<http://www.for.wonkwang.ac.kr>

² Schools of Mechanical Engineering, Wonkwang University,
570-749, Iksan, Korea
ncatcello@hanmail.net

Abstract. In this paper research and development are ongoing in the implementations of Brillouin-active fiber based, highly versatile active optical device for optical communication, sensing and computation. An active device in general requires the employment of nonlinearity, and possibly feedback for increased efficiency in device function. However, the presence of nonlinearity together with intrinsic delayed feedback has been repeatedly demonstrated to lead to instabilities and ultimate optical chaos. Our effort is then to exploit device function and suppress instabilities by simulation, and design for optimization based on neural networks in smart structures. Instabilities are unavoidable in optical fiber due to its intrinsic nonlinearity and feedback instabilities. The suppression of such instabilities is devoted to the exploitation of them for memory capacity. These memories can be estimated as an optical logic function used for all-optic in-line switching, channel selection, oscillation, optical logic elements in optical computation with neural network application.

1 Introduction

Optical fibers based on neural networks and hardware implementation has been extensively used in optical systems [1], [2]. Recent interest has been also focused on using optical fibers as sensors since fiber parameters are sensitive to the fiber immediate environment [3]. Specially, in the case of stimulated Brillouin scattering (sBs) sensor, the backward scattering nature of scattering has long been viewed as an ultimate intrinsic loss mechanism in long haul fibers since Brillouin threshold decreases with increasing effective fiber length. On the other hand, the very backscattering nature of this process and the existence of a threshold, provide potential optical device functions, such as optical switching, channel selection, amplification, sensing, arithmetic and neural functions in optical signal processing, and neural network applications and hardware implementation. The theoretical and

physical background of this nonlinear process has been well explained [4],[5]. The backward scattering scheme based on neural networks in optical fiber is shown in Figure 1.

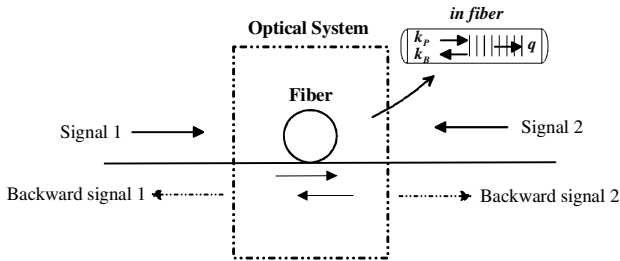


Fig. 1. Brillouin-active fiber with forward/backward scattering schemes in neural networks

Active device in optical systems generally require the employment of nonlinearity, and possibly feedback for increased device efficiency. The presence of nonlinearity together with intrinsic delayed feedback has been repeatedly demonstrated to lead to instabilities and optical chaos [6], [7]. This phenomenon has extensively investigated by us for its potential detrimental effect to the Brillouin fiber sensor [8], [9].

Such a smart sensor system would implement a massively parallel computational architecture with its attendant reduction in processing time while managing the complexity of the system, i.e. the sensing/actuation grid. Our sBs network would learn the correct "algorithms" by example during training and have the ability to generalize to untrained inputs after training is completed. The inputs to the network are the fiber optic sensor signal outputs, and the network outputs are the control signals for actuation controls. The true advantage of this system for application to smart sensor structures lies both in its capability to analyze complex sensor signal patterns and its speed in generating the appropriate control signal for the actuators. The key lies in the implementation of a neuron operation using sBs in optical fibers.

2 SBS Optical Logic

An artificial neuron, used in neural network research, can be thought of as a device with multiple inputs and single or multiple outputs in hardware implementations. The inputs to a neuron are weighted signals. Neuron-type operations can be performed by an optoelectronic system that uses sBs for the weighted summation required in a neuron. Weighting can be achieved by optical summation and subtraction, conveniently carried out in an optical fiber using sBs. Weighted additions and subtractions are needed in many situations. For example, a neuron performs weighted summation of the incoming signals. The performance of such a device will enhance if it operates optically. We propose to study a system that can perform the practical implementation of a Brillouin-active fiber for optical neural net, neural function by exploiting the acousto-optic nature of the sBs process [7].

Nonlinear effects in optical fibers, specifically sBs has emerged as a versatile approach to the construction of active optical devices for all-optic in-line switching, channel selection, amplifiers and oscillators in optical sensing, and optical communications[2], [3]. The backward scattering nature of Brillouin scattering, which is the light reflection by laser induced acoustic wave in the fiber, has long been viewed as an ultimate intrinsic loss mechanism in long haul fibers, since Brillouin threshold decreases with increasing effective fiber length[5], [6]. The very backscattering nature of this nonlinear process and the existence of a threshold provide potential optical device functions, such as optical switching, arithmetic and neural functions in networks. An artificial neuron, used in neural network research, can be thought of as a device with multiple inputs and a single or multiple outputs. The inputs to a neuron are weighted signals. The neuron adds the weighted signals, compares the result with a preset value, and activates if the sum exceeds threshold. In the nonlinear optical phenomenon, the system combined weighted signals also produces an output if the weighted sum is greater than the threshold. A theoretical sBs based neural network, utilizing SBS threshold sensing with an embedded sensor is seen in Figure 2.

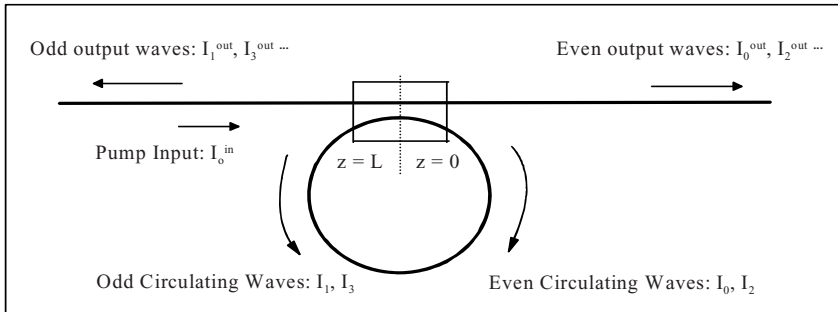


Fig. 2. Hardware implementations based Brillouin-active fiber with forward/backward scattering schemes

The arithmetic building block of energy addition and subtraction, as in Fig.2, can conceivably be accomplished by the sBs process, which involves energy transfer between waves. Thus, if two waves at a frequency difference equal to the stokes shift of the fiber propagate in the fiber in opposite directions, then energy is “subtracted” from the higher frequency wave and “added” to the lower frequency wave. If three waves are present in a fiber with equal stokes shifts, then the wave at the middle frequency will receive energy from the higher frequency wave and lose energy to the lower frequency wave. Practical implementation of this scheme calls for all the waves to be generated by the same laser.

3 SBS Based Threshold Logic Implementation

A practical sBs logic implementation of theoretical neuron calls for all the waves to be generated by the same laser. We are very familiar with this method and a scheme is devised in Figure 3.

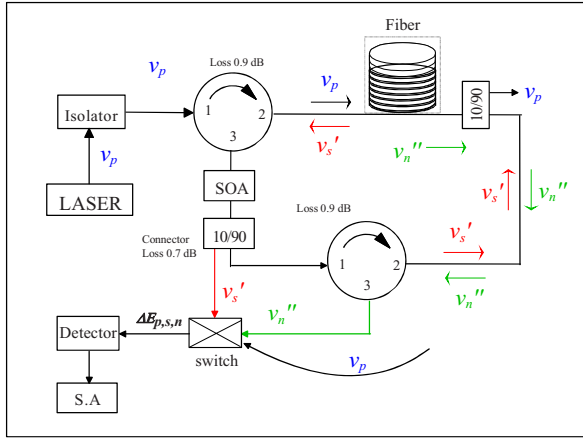


Fig. 3. Practical hardware implementation sBs based threshold logic. Optical fibers are used for neuron operation as the medium for providing sBs gain to the stokes wave.

We assume that two input waves at frequencies v_p and v_n are launched in a fiber of length L and a third wave, called a sensor signal v_s , is launched from the other end of the fiber. The sensor signal wave will act as a Stokes wave for the v_p signal and as a pump wave for the v_n when $v_p - v_s \approx \Delta v_B$, and $v_s - v_n \approx \Delta v_B$, where Δv_B is Brillouin shift. The frequencies are such that the difference $v_p - v_n \approx 2\Delta v_B$. This spacing is chosen to eliminate any interference due to sBs. The intensity level of each wave is below the Brillouin threshold, ($I_{th} = 2I/g_B L_{eff}$) in order to avoid the generation of backward Stokes waves from spontaneous scattering. Here L_{eff} is the effective length. The energy can be added to and/or subtracted from the sensor wave by its interaction with v_p and v_n signal via sBs. The addition of energy to the sensor wave is proportional to the pump intensity; while the subtraction is proportional to the pump intensity v_n . The weighted summation can be performed by properly setting the intensity of each pump beam. An optical amplifier can be used for this purpose. The energy transfer can take both positive and negative values. Generally, positive weights of energy can be easily accomplished in optics field. However, the negative weighting is not so direct. It can be achieved via stimulated Brillouin scattering process, which requires the shifted frequency of the signal in optical fiber. This will be shown that energy can be added to or subtracted from an optical signal via sBs optical fiber-ring based on logic principles. The arithmetic building block of energy addition and subtraction can conceivably be accomplished by the sBs process (see Figure 4), which involves energy transfer between waves. Thus, if two waves at a frequency difference equal to the Stokes shift of the fiber propagate in the fiber in opposite directions, then energy is “subtracted” from the higher frequency wave and “added” to the lower frequency wave.

If three waves are present in a fiber with equal Stokes shifts, then the wave at the middle frequency will receive energy from the higher frequency wave and lose energy to the lower frequency wave. Intensity wave subtraction is unique to the sBs process since the negative value was not so direct accomplished in optics field.

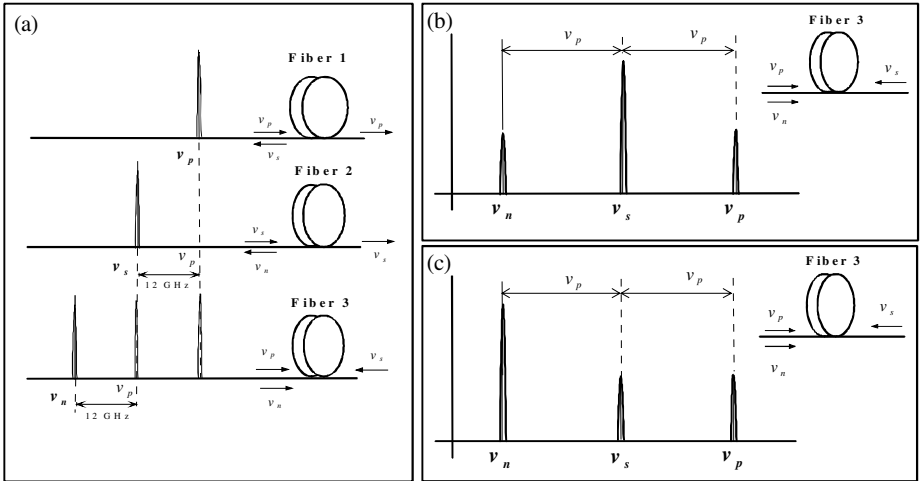


Fig. 4. (a) sBs oscillator amplification scheme with no mixing in fiber 3 (b) addition = energy from v_p added to v_s (c) subtraction = energy from v_s added to v_n

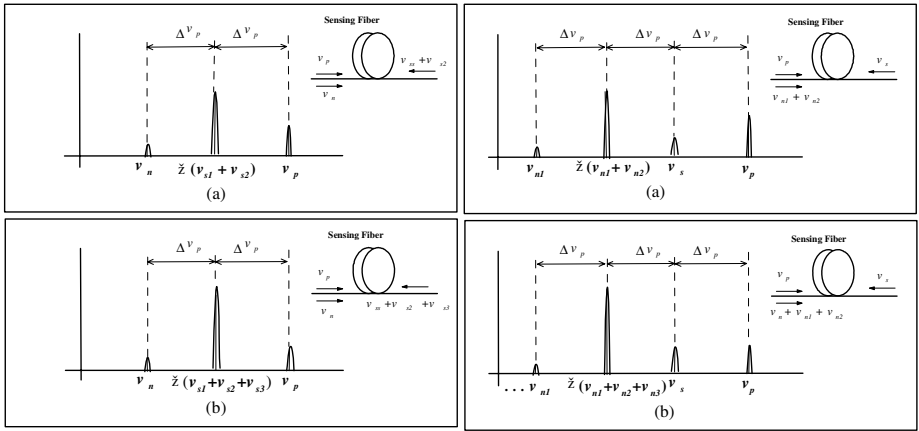


Fig. 5. (a) addition: energy v_p added to $v_{s1} + v_{s2}$ (b) addition: energy v_p added to $v_{s1} + v_{s2} + v_{s3}$

Fig. 6. (a) subtraction: energy v_s added $v_{n1} + v_{n2}$ (b) subtraction: energy v_s added $v_{n1} + v_{n2} + v_{n3}$

In the neuron building block-3 wave optical addition and subtraction, the arithmetic building block of optical addition and subtraction in the form of energy can conceivably be readily accomplished by the sBs process, which involves energy transfer between waves. A practical implementation scheme, using the 4-wave sBs based threshold logic studied as shown Figure 5 and Figure 6.

4 Conclusions

We studied that the ability of sBs to perform both sensing and optical arithmetic render such a scheme as the simplest building block for neural network based smart structures. The potential of extremely low generation threshold for multiple Stokes lines with ready energy exchange among them lends the process to optical addition and subtractions, simulating neuron functions. The building block for energy addition and subtraction can conceivably be accomplished by such sBs process, which involves energy transfer between waves. The experiments show the sBs instabilities with periodic and chaotic dynamics that are in good agreement with results. In addition to application, a theoretical possibility of applying multistability as a memory device for complicated information has been discussed for sBs logic element.

Acknowledgement. This work was financially supported by MOCIE (I-2004-0-074-0-00) through EIRC program.

References

1. Gregory, D., Vanwiggeren, Roy, R.: Communication with Chaotic Lasers. *Science* **279** (1998) 1198-1200
2. Hotate, K., Sean, S., L, Ong.: Distributed Dynamic Strain Measurement Using a Correlation-Based Brillouin Sensing System. *IEEE. Photonics Letters* **15** (2003) 272-274
3. Bernini, R., Minardo, A., Zeni. L.: Stimulated Brillouin Scattering Frequency-Domain Analysis in a Single-mode Optical Fiber for Distributed Sensing. *Opt. Lett.* **29** (2004) 1977-1979
4. Koyamada, Y., Sato, S., Nakamura, S., Sotobayashi, H., Chujo. W.: Simulating and Designing Brillouin Gain Spectrum in Single-Mode Fibers. *J. of Lightwave Tech.* **22** (2004) 631-639
5. Tanemura, T., Takyshima, Y., Kikuchi. K.: Narrowband Optical Filter, with a Variable Transmission Spectrum, using Stimulated Brillouin Scattering in Optical Fiber. *Opt. Lett.* **27** (2002) 1552-1554
6. Harrison, R.G., Yu, D., Lu, W., and Ripley, P. M.: Chaotic Stimulated Brillouin Scattering Theory and Experiment. *Physica D* **86** (1995) 182-188
7. Yu, D., Lu, W., and Harrison, R.G.: Physical Origin of Dynamical Stimulated Brillouin Scattering in Optical Fibers with Feedback. *Physical Review A* **51** (1995) 669-674.
8. Kim, Y.K., Kim, S., Lim, S., Kim, D.: Neuron Operation Using Controlled Chaotic Instabilities in Brillouin-Active Fiber based Neural Network. *LNCS* **3612** (2005) 1045-1051.
9. Kim, Y.K., Lim, S., Kim, D.H.: Effect of steady and relaxation oscillation using controlled chaotic instabilities in Brillouin fibers based neural network. *LNCS* **4222** (2006) 880-883.

Hydro Plant Dispatch Using Artificial Neural Network and Genetic Algorithm

Po-Hung Chen

St. John's University, Department of Electrical Engineering,
Taipei, Taiwan, 25135, R.O.C.
phchen@mail.sju.edu.tw

Abstract. This paper presents a novel approach to solve the hydro plant dispatch problem based on the artificial neural network (ANN) and genetic algorithm (GA). In this work, the difficult water balance constraints are embedded and satisfied throughout the proposed encoding and decoding algorithms. The ANN is used as a pre-dispatch tool to generate raw hydro output for each hour temporarily ignoring time-dependent constraints. Then, the proposed decoding algorithm decodes the raw schedule of each plant into a feasible one. Finally, a GA is used to find the optimal schedule. The proposed approach is applied to an actual utility system of four hydro plants and 22 thermal units with great success. Results show that the new approach obtains a more highly optimal solution than the conventional dynamic programming method.

1 Introduction

Hydro plants play important roles within the power system security due to their fast response characteristics. Hydro plant dispatch is a difficult task in the hydro-thermal coordination. The problem is mainly concerned with both hydro plants scheduling and thermal units dispatching. Both electric and hydraulic couplings create a multi-dimensional, non-linear programming problem. To solve such a complex problem, several methods have been proposed in the literature [1-6]. Among these methods, the dynamic programming (DP) method [1, 2, 5, 6] has gained much popularity. In our previous work [6], a hydro plant dispatch software using the DP method was completed and applied to the actual Taipower system of Taiwan. Since the DP technique belongs to the class of "greedy search" algorithms, the solution cost usually got stuck at a local optimum rather than at the global optimum. However, the optimality of solution is very important to the utility. Even a small reduction in percentage production cost may lead to a large money saving. Obviously, a complete and efficient algorithm for solving the hydro plant dispatch is still in demand.

In recent years, biologically artificial intelligence techniques, such as artificial neural network (ANN) and genetic algorithm (GA) have emerged as candidates for the optimization problem [7-8]. ANN is intended to model the behavior of biological neural network. It is modeled as a massively parallel interconnected network of elementary neurons. GA is a stochastic searching algorithm combining an artificial

survival of the fittest with genetic operators that is suitable for a variety of optimization problems. In this paper, a new approach based on the ANN and GA is developed for solving the 24-hour ahead hydro plant dispatch. The ANN is used as a pre-dispatch tool to generate raw hydro schedule for each hour temporarily ignoring time-dependent constraints. Then, the proposed decoding algorithm decodes the raw schedule of each plant into a feasible one. The GA is finally used to find the optimal schedule. Comparative studies on the actual Taipower system show that the new approach obtains lower solution cost than the conventional DP method.

2 Problem Description and Formulation

2.1 List of Symbols

P_{hj}^t : power generation of hydro plant j in hour t

P_{si}^t : power generation of thermal unit i in hour t

$F_i^t (P_{si}^t)$: production cost for P_{si}^t

T : number of scheduling hours

N_h : number of hydro plants

N_s : number of thermal units

P_L^t : system load demand in hour t

P_{loss}^t : system transmission network losses in hour t

V_j^t : water volume of reservoir j at the ending of hour t

I_j^t : natural inflow into reservoir j in hour t

Q_j^t : water discharge of hydro plant j in hour t

S_j^t : water spillage of hydro plant j in hour t

τ_{j-1} : water travel time from plant $j-1$ to plant j

UR_{si} : up ramp rate limit of thermal unit i

DR_{si} : down ramp rate limit of thermal unit i

$R_{si}^t (P_{si}^t)$: spinning reserve contribution of unit i for P_{si}^t

$R_{hj}^t (P_{hj}^t)$: spinning reserve contribution of plant j for P_{hj}^t

R_{req}^t : system spinning reserve requirement in hour t

2.2 Equivalent Hydro Plant Model

The number of hydro units is usually much greater than the number of hydro plants. Therefore, in practical hydro plant dispatch, it is advantageous to model hydro generation at the plant (or reservoir) level to reduce the problem size. The equivalent plant model can be obtained using an off-line mathematical procedure which maximizes the total plant generation output under different water discharge rates [1]. The generation output of an equivalent hydro plant is a function of the water discharge through the turbine and the net head (or the content of reservoir). The general form is expressed by:

$$P_{hj}^t = f(Q_j^t, V_j^{t-1}). \tag{1}$$

The quadratic discharge-generation function to be used in this paper as a good approximation of the hydro plant generation characteristics, considering the head effect, is given below:

$$P_{hj}^t = \alpha_j^{t-1} Q_j^{t^2} + \beta_j^{t-1} Q_j^t + \gamma_j^{t-1}, \tag{2}$$

where coefficients α_j^{t-1} , β_j^{t-1} , and γ_j^{t-1} depend on the content of reservoir j at the ending of hour $t-1$. In this work, the read-in data include five groups of α , β , γ coefficients that relate to different storage volumes, from minimum to maximum, for each reservoir. Then, the corresponding coefficients for any reservoir volume are calculated by using a linear interpolation between the two closest volumes.

2.3 Objective Function and Constraints

The hydro plant dispatch deals with the problem of obtaining the optimal generations for both hydro plants and thermal units. It aims to minimize the production costs of thermal units while satisfying various constraints. With discretization of the total scheduling time into a set of shorter time intervals (say, one hour as one time interval), the hydro plant dispatch can be mathematically formulated as a constrained nonlinear optimization problem as follows:

Problem: Minimize $\sum_{t=1}^T \sum_{i=1}^{N_s} F_i^t (P_{si}^t)$. (3)

Subject to the following constraints:

o System power balance

$$\sum_{i=1}^{N_s} P_{si}^t + \sum_{j=1}^{N_h} P_{hj}^t - P_L^t - P_{loss}^t = 0. \tag{4}$$

o Water dynamic balance with travel time

$$V_j^t = V_j^{t-1} + I_j^t + Q_{j-1}^{t-\tau_{j-1}} + S_{j-1}^{t-\tau_{j-1}} - Q_j^t - S_j^t. \tag{5}$$

o Thermal generation and ramp rate limits

$$\text{Max}(\underline{P}_{si}, P_{si}^{t-1} - DR_{si}) \leq P_{si}^t \leq \text{Min}(\overline{P}_{si}, P_{si}^{t-1} + UR_{si}). \tag{6}$$

o Hydro discharge (generation) limits

$$\underline{Q}_j \leq Q_j^t \leq \overline{Q}_j. \tag{7}$$

o Reservoir limits and specified final volume

$$\underline{V}_j \leq V_j^t \leq \overline{V}_j. \tag{8}$$

o System spinning reserve requirement

$$\sum_{i=1}^{N_s} R_{si}^t (P_{si}^t) + \sum_{j=1}^{N_h} R_{hj}^t (P_{hj}^t) \geq R_{req}^t. \tag{9}$$

3 Solution Methodology

The basic conception of ANN is intended to model the behavior of biological neural functions. The original desire for the development of ANN is intended to take advantage of parallel processors computing than traditional serial computation. The GA is essentially a search algorithm combining solution evaluation with randomized, structured exchanges of information between solutions to obtain optimality. The power of this algorithm comes from its ability to exploit historical information structures from previous solution guesses in an attempt to increase the performance of future solution structures. In every generation, a new set of artificial strings is created using bits and pieces of the fittest of the old ones. The three prime operators associated with the GA are matching, crossover, and mutation.

In this work, a novel approach based on the ANN and GA is developed for solving the hydro plant dispatch within the daily hydro-thermal coordination. The ANN is used as a pre-dispatch tool to generate raw hydro schedule for each hour ignoring time-dependent constraints temporarily. Then, the proposed decoding algorithm decodes the raw schedule of each plant into a feasible one. The GA is finally used to find the optimal schedule. The solution methodology for solving the hydro plant dispatch by the proposed approach is outlined in the flowchart in Fig. 1 and will be described in detail later.

From the literature survey, several models and learning algorithms of ANN have been proposed for solving combinatorial optimization problems [9]. In this work, we establish a triple-layer feed-forward back-propagation neural network (BPNN), as shown in Fig. 2, for solving the hydro plant dispatch problem. The number of output layer neurons is set at N, where N is the number of hydro plants. The number of hidden

layer neurons is set at 24 according to experimental results. The input data for the BPNN include hourly load demand, hourly remaining thermal load, storage of reservoir, and a heuristic control variable. The output layer neurons generate raw unit combinations for each hour temporarily ignoring time-dependent constraints. In this work, the transfer function used in the hidden layer is a hyperbolic tangent function [11], as shown in Fig. 3. The transfer function used in the output layer is a sigmoid function [11] as shown in Fig. 4.

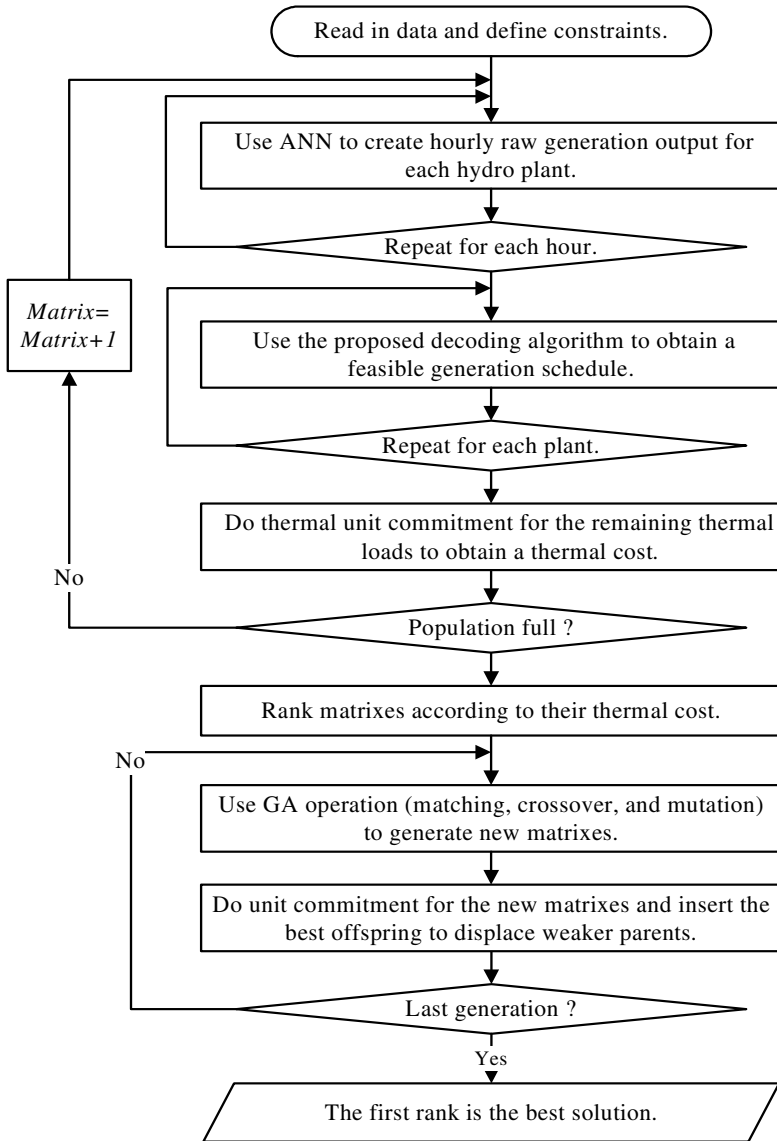


Fig. 1. General flow chart of the proposed approach

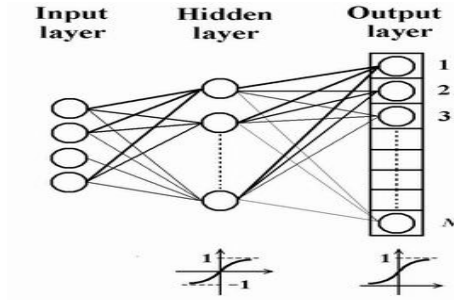


Fig. 2. Triple-layer feed-forward BPNN

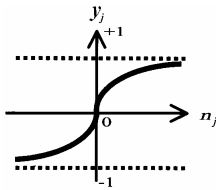


Fig. 3. Hyperbolic tangent function

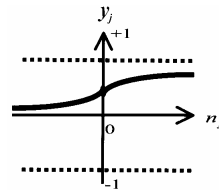


Fig. 4. Sigmoid function

3.1 Learning Algorithm

In the training procedure, A faster off-line back-propagation learning algorithm named “RPROP algorithm” is used as the learning rule. Riedmiller and Braun [10] showed that both convergence speed and memory requirement of the RPROP algorithm are better than traditional gradient-descent learning algorithms. In the RPROP algorithm, the update-values for each weight are modified according to the behavior of the sequence of signs of the partial derivatives in each dimension of the weight space, not according to the gradient value. The modified procedure of a weight of the RPROP algorithm can be mathematically formulated as follows:

$$\Delta_{ij}(t) = \begin{cases} \eta^+ \times \Delta_{ij}(t-1) & , \text{ if } \frac{\partial E}{\partial W_{ij}}(t-1) \times \frac{\partial E}{\partial W_{ij}}(t) > 0 \\ \eta^- \times \Delta_{ij}(t-1) & , \text{ if } \frac{\partial E}{\partial W_{ij}}(t-1) \times \frac{\partial E}{\partial W_{ij}}(t) < 0 \\ \Delta_{ij}(t-1) & , \text{ else} \end{cases} \quad (10)$$

$$\Delta W_{ij}(t) = \begin{cases} -\Delta_{ij}(t) & , \text{ if } \frac{\partial E}{\partial w_{ij}}(t) > 0 \\ +\Delta_{ij}(t) & , \text{ if } \frac{\partial E}{\partial w_{ij}}(t) < 0 \\ 0 & , \text{ else} \end{cases} \quad (11)$$

$$W_{ij}(t+1) = W_{ij}(t) + \Delta W_{ij}(t) \tag{12}$$

where

$W_{ij}(t)$: weight back propagated from neuron j to neuron i

η^+ / η^- : learning velocity, where $0 < \eta^- < 1 < \eta^+$

$E(t)$: error function

$\Delta_{ij}(t)$: update value of $W_{ij}(t)$

3.2 Encoding and Decoding

The encoding must be carefully designed to efficiently transfer information between encoding strings and objective function of a problem. The encoding scheme that translates the encoded parameter-water discharges of each hydro plant into their binary representation is shown in Fig. 5. Using a plant's water discharge, instead of the plant's generation output, the encoded parameter is more beneficial for dealing with the difficult water balance constraints. Each string contains 4×24 digital bits to represent a solution for a 24-hour discharge schedule. Each hour is assigned the same number of four bits to represent a normalized water discharge q_j^t . The resolution is equal to $1/2^4$ of the discharge difference from minimum to maximum.

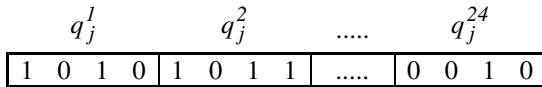


Fig. 5. The proposed encoding scheme

Evaluation of a hydro schedule is accomplished by decoding the encoded string and computing the schedule's corresponding thermal cost using the decoded parameter. The detailed decoding procedure is summarized in the following stages:

1. Decode each hour of the string to obtain the normalized discharge q_j^t in decimal values:

$$q_j^t = \sum_{i=1}^4 (b_i \times 2^{-i}) \quad b_i \in \{0,1\}$$

q_j^t			
b ₁	b ₂	b ₃	b ₄
×	×	×	×
2^{-1}	2^{-2}	2^{-3}	2^{-4}

2. Calculate the upper and lower boundaries of the discharge:

$$\overline{Q}_j^t = \text{Min}[\overline{Q}_j, (V_j^{t-1} + Q_{j-1}^{t-\tau_{j-1}} + I_j^t - \underline{V}_j), (-V_{j+1}^{t-1} - I_{j+1}^t + \overline{Q}_{j+1} + \overline{V}_{j+1})], \tag{13}$$

$$\underline{Q}_j^t = \text{Max}[\underline{Q}_j, (V_j^{t-1} + Q_{j-1}^{t-\tau_{j-1}} + I_j^t - \overline{V}_j), (-V_{j+1}^{t-1} - I_{j+1}^t + \underline{Q}_{j+1} + \underline{V}_{j+1})], \tag{14}$$

where \overline{Q}_j^t and \underline{Q}_j^t denote, respectively, the upper and lower bounds of Q_j^t .

3. Translate the normalized value q_j^t to the actual value Q_j^t :

$$Q_j^t = \underline{Q}_j^t + q_j^t (\overline{Q}_j^t - \underline{Q}_j^t). \tag{15}$$

4. Calculate the hydro generation output P_{hj}^t using (2).
5. Continue the computation for each plant, and for hour 1 to hour 24.
6. Calculate the remaining thermal load profile:

$$P_{rm}^t = P_L^t - \sum_{j=1}^{N_h} P_{hj}^t \quad t = 1, 2, \dots, 24. \tag{16}$$

where P_{rm}^t is the remaining thermal load in hour t .

7. Do thermal unit commitment for the remaining thermal load profile, and return the corresponding thermal cost.

In the proposed approach, the thermal subproblem can be solved entirely independently. Each string represents a complete discharge schedule of a hydro plant. The UC package was executed to calculate the corresponding thermal cost of this discharge schedule. Then, the genetic iterations proceed to search the optimal string (discharge schedule) which has the lowest thermal cost.

4 Test Results

The proposed approach was implemented into a software and tested on the actual Taipower generation system, which consists of 22 thermal units and four hydro plants in cascade. Detailed characteristic data of the four plants are given in Table 1. Besides the common constraints listed in Section III, the Taipower system has the following unique characteristics that increase the difficulty of the problem.

1. Chin-Shan is a must-run unit, because it is connected to Taipower's Automatic Generation Control (AGC) system. But, it has only a small reservoir.
2. There is a one-hour water travel time delay between the Chin-Shan and Ku-Kuan plants.
3. The 300MW system spinning reserve requirement must be satisfied.
4. The large load fluctuation at the noon break hours can not be completely handled by thermal units due to their ramp rate limits.

The proposed approach is tested on a summer weekday whose load profile, as shown in Fig. 6, is obtained by subtracting the expected generation output of other hydro plants and base load units from the actual system load profile. Throughout the study, the DP method [6] is then used as the main benchmark of comparison for the proposed approach. Fig. 7 shows the total generation profile of the four hydro plants created by

Table 1. Characteristic of hydro plants

Plant	Maximal Output (MW)	Minimal Output (MW)	Maximal Discharge (m ³ /s)	Maximal Storage (k x m ³)	Natural Inflow (m ³ /s)
Te-Chi	234	60	177	254420	58
Chin-Shan	360	40	150	427	-
Ku-Kuan	180	30	113	7960	6
Tien-Lun	90	30	62	395	-
Total	864 MW				

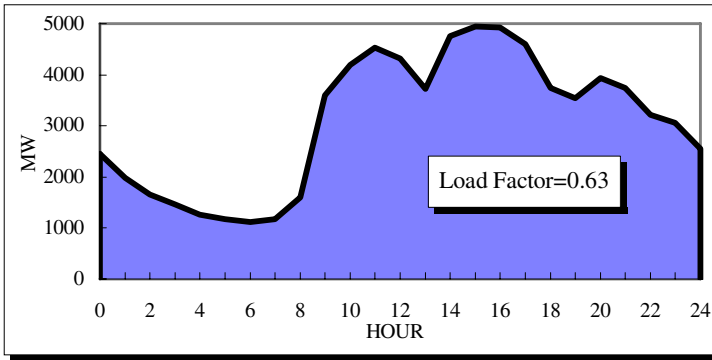


Fig. 6. A summer weekday load profile

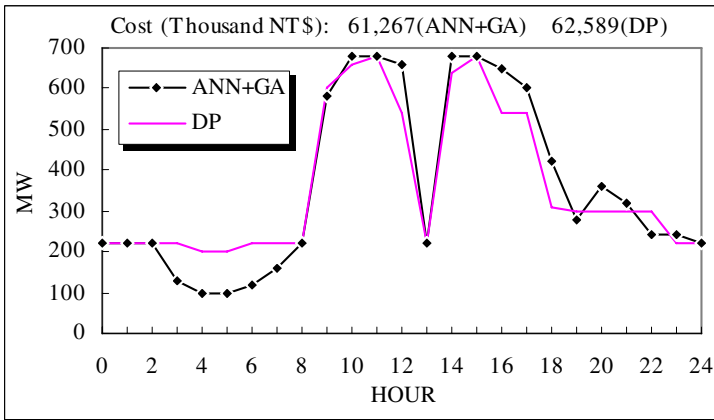


Fig. 7. Total generation profile of the four hydro plants

the proposed approach, and the one created by the DP method. From the test results, two interesting and important observations can be summarized as follows:

1. Both generation profiles basically follow the load fluctuation that is consistent with our economic expectation. However, an additional cost saving of 1,322 thousand NT dollars has been realized by the proposed approach.
2. The reason the hydro plants are not generating to their maximum in peak-hours is due to the system spinning reserve requirement.

5 Conclusions

This paper presents a new solution method based on the ANN and GA for solving the hydro plant dispatch problem. The difficult water balance constraints due to hydraulic coupling are embedded in the encoding string and are satisfied throughout the proposed decoding algorithm. To make the dispatching results more practical, the effects of net head and water travel time are also taken into account. Numerical results from an actual utility system indicate the attractive properties of the proposed approach in practical application, namely, a highly optimal solution cost and more robust convergence behavior.

Acknowledgments

Support for this research by the National Science Council of the Republic of China, Taiwan, under grant NSC 90-2213-E-129-005 is greatly appreciated.

References

1. Wood, A.J., Wollenberg, B.F.: *Power Generation, Operation, and Control*. 2nd edn. John Wiley & Sons, New York (1996)
2. El-Hawary, M.E.: An Overview of Scheduling Functions in Hydro-Thermal Electric Power Systems. *Proceedings of the 1991 IASTED International Conference* (1991) 31-37
3. Guan, X., Luh, P.B.: Nonlinear Approximation Method in Lagrangian Relaxation-Based Algorithm for Hydrothermal Scheduling. *IEEE Trans. Power Systems* **10** (1995) 772-778
4. Rux, L.M.: An Incremental Economic Dispatch Method for Cascaded Hydroelectric Power Plants. *IEEE Trans. Power Systems* **8** (1993) 1266-1273
5. Yang, J.S., Chen, N.: Short Term Hydrothermal Coordination using Multi-Pass Dynamic Programming. *IEEE Trans. Power Systems* **4** (1989) 1050-1056
6. Chen, P.H.: *Generation Scheduling of Hydraulically Coupled Plants*. Master Thesis in Electrical Engineering, National Tsing-Hua University (1989)
7. Sasaki, H., Watanabe, M., Yokoyama, R.: A Solution Method of Unit Commitment by Artificial Neural Network. *IEEE Trans. Power Systems* **7** (1992) 974-981
8. Chang, H.C., Chen, P.H.: A Genetic Algorithm for Solving the Unit Commitment Problem. *Proceedings of the International Power Engineering Conference* (1997) 831-836
9. Wasserman, P.D.: *Neural Computing, Theory and Practice*. Van Nostrand Reinhold (1989)
10. Riedmiller, M., Braun, H.: A Direct Adaptive Method for Faster Back Propagation Learning- The RPROP Algorithm. *IEEE Int. Conf. Neural Networks* **1** (1993) 586-591
11. Wasserman, P.D.: *Neural Computing, Theory and Practice*. Van Nostrand Reinhold (1989)

Application of Wavelet Packet Neural Network on Relay Protection Testing of Power System

Xin-Wei Du, Yuan Li, and Di-Chen Liu

School of Electrical Engineering, Wuhan University,
Wuhan 430072, China
xiaobei0724@sina.com.cn

Abstract. The paper presents a wavelet packet neural network (WPNN) approach for solving the waveform distortion problem of protective relaying testing instrument. With its excellent time-frequency localization property and approximation ability, WPNN is used to establish an identification model of the non-linear amplifier of the protective relaying testing instrument. The fault data to be put into the instrument is compensated by an adjusting function getting from the identification model, which makes the whole instrumentation system show linear performance so that the distortion of the output waveform is constrained greatly. Simulation results indicate the feasibility and validity of the proposed approach, and a prototype has been put into practical operation.

1 Introduction

The continuous expansion of the modern electric network's scale and complication of its configuration requires higher reliability of protection relays in power system, and testing protection relays with fault recoding data amplified by instrument before putting into operation is an effective way for improving their performance [1], [2]. Traditional protective relaying testing instruments could realize such testing function, but they used to adopt analog amplifier, which is a typical non-linear system, to realize power amplifying. So the non-linear distortion of output waveform inevitably becomes a serious problem for the relay protection testing. In the paper, a WPNN approach is presented for resolving this problem.

WPNN is a combination of wavelet packet theory and conventional neural network, which not only possesses good localization property and feature extraction ability of wavelet packet, but also inherits most merits of neural network such as self-study, adaptability and high fault-tolerant [3], [4]. It selects wavelet packet basis as its neuron's activation function and has normative design procedures and solid academic foundation, so WPNN has been widely applied in many technical fields [5], [6], [7].

In this study, WPNN is adopted to establish an identification model of the non-linear amplifier of the protective relaying testing instrument. And by comparing the identification model's output with idea output, an adjusting function is generated to guide adaptive adjustment of fault data before to be put into the instrument, which makes the whole instrumentation system show linear performance so that the distortion of the output waveform is constrained greatly. A simulation using fault recording

data is carried out, whose results demonstrate the feasibility and validity of application of WPNN on relay protection testing of power system, and a prototype with the proposed approach has been put into practical operation.

2 Construction of WPNN

WPNN is the development of wavelet neural network (WNN). WNN can be viewed as the combination of reconstructions using wavelet basis of orthogonal wavelet spaces of $L^2(R)$ based on multi-resolution analysis (MRA) [8], [9], [10]. As everyone knows, wavelet space can be decomposed further using wavelet packet, so signals can be decomposed in more frequency bands to increase frequency resolution than by MRA. Therefore, selecting best wavelet packet basis to be network neuron’s activation function will obtain better time-frequency localization property and approximation ability for the network. So WPNN utilizes wavelet packet basis extracting feature of input signal and neural network in WPNN takes charge of information identification, i.e., WPNN can be divided into two parts: wavelet packet feature extraction and neural network information identification, which is shown in Fig.1.

Throughout the paper, Z denotes the set of all integers. Let $\psi(\bullet)$ and $\{u_n(\bullet)\}_{n \in Z}$ denote wavelet basis and wavelet packet generated from $\psi(\bullet)$ respectively. The structure design of WPNN consists of following three primary steps:

Step 1. Calculating scale range: Using $[t_1, t_2]$ and $[t_{\min}, t_{\max}]$ to denote the time extent of $\psi(\bullet)$ and the goal system $f(\bullet)$, their energy concentrating areas of frequency extent can be estimated with training data, which are expressed as $[f_1, f_2]$ and $[f_{\min}, f_{\max}]$ separately. According to the properties of Fourier transform, with the increase of the wavelet scale j , frequency extent will expand by 2^j , i.e., frequency extent of $\psi_j(\bullet)$ is $[2^j f_1, 2^j f_2]$. Therefore the wavelet scale j contains a finite range for covering $[f_{\min}, f_{\max}]$, and it can be calculated by below:

$$J = \{j\} = [\text{int}_{-\infty}(\log_2 \frac{f_{\min}}{f_1}), \text{int}_{+\infty}(\log_2 \frac{f_{\max}}{f_2})] \tag{1}$$

Where $\text{int}_{-\infty}$ and $\text{int}_{+\infty}$ denote choosing smaller or bigger integer value nearby respectively.

Step 2. Selecting best wavelet packet basis: Shannon entropy criterion is introduced to calculating the entropies for the set of coefficients of each node in scale range getting in step1. Then, replace the parent nodes by the two children nodes directly below it if the sum of children’s entropies is less than that of parent. In this method, we can uncover the set of minimum entropy basis, which can be denoted as follows:

$$U = \{u_{n_e, j_e}\}, 1 \leq e \leq E, e \in Z \tag{2}$$

Where E is the number of best wavelet packet basis.

Step 3. Determination of number of nodes: This step is also can be seen as determination of translation factor k for each wavelet scale j . It is known as that the time

extent of wavelet packet $\{u_n(\bullet)\}_{n \in Z}$ is invariable with n changes, so the time extent of wavelet packet basis $u_{n,j}(\bullet)$ can be expressed as $[2^j(t_1+k), 2^j(t_2+k)]$. With the increase or decrease of k , the extent slides on the time axis. For covering the time area $[t_{\min}, t_{\max}]$ of $f(\bullet)$, range of k is determined as:

$$K = \{k\} = [\text{int}_{-\infty}(2^j \times t_{\min} - t_1), \text{int}_{+\infty}(2^j \times t_{\max} - t_2)] \tag{3}$$

By the three steps above, the structure and parameters of first part of WPNN (feature extraction) can be definitely determined. So the second part (information identification) can be viewed as a simple three-layered neural network with known input value, whose connection rights $w(n, j, k)$ are also that of WPNN. The whole structure of WPNN is thus of the following form, and is illustrated in Fig.1.

$$f(t) \approx \sum_{(n,j) \in U} \sum_{k \in K} w(n, j, k) \times u_n(2^j \times t - k) \tag{4}$$

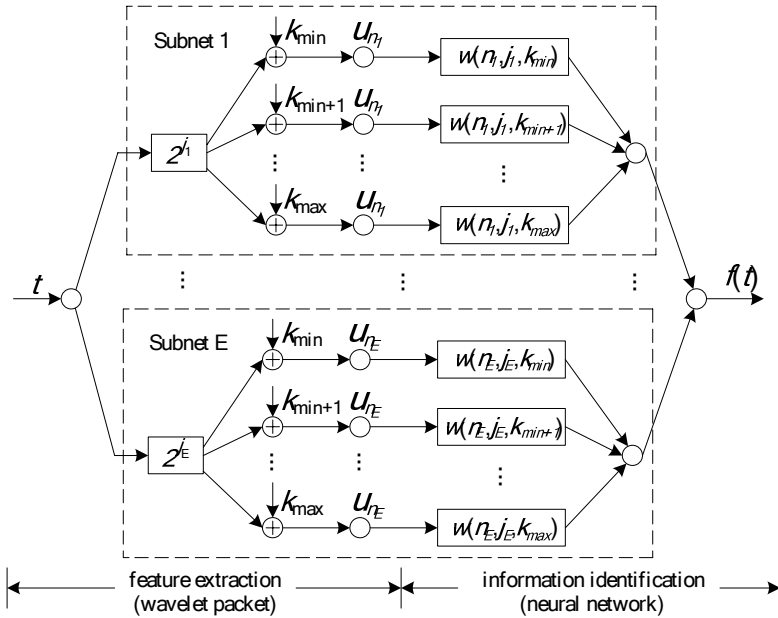


Fig. 1. The structure of WPNN

3 Overall Scheme of Relay Protection Testing Instrument

As referred in introduction, the non-linear distortion of output waveform is the most serious problem for relay protection testing. Aiming at this problem, a new scheme of closed-loop relay protection testing instrument is proposed as shown in Fig.2.

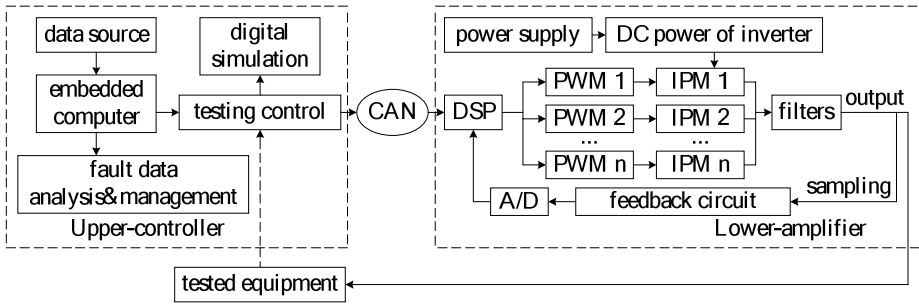


Fig. 2. Overall scheme of relay protection testing instrument

Double CPUs configuration including upper-controller and lower-amplifier is applied in this system.

Upper-controller adopts high-performance portable computer or embedded computer as its core, which realizes data acquisition, fault analysis and integrated control. Besides, it can also adjust sampling frequency, value, releasing speed or harmonic content of the input data according to the requirements of testing. And a suit of protection testing digital simulation software is successfully embedded into upper-controller of the instrument. It can simulate the testing process before analog testing on the digital platform, which improves the flexibility and repeatability and avoids potential harm to the tested equipments [11].

Lower-amplifier mainly consists of Digital Signal Processing (DSP) chip, array of Intelligent Power Modules (IPM), and feedback circuit. DSP receives data form upper-controller computer through CAN bus and generates PWM (Pulse Width Modulation) pulse by regular sampling method, and IPM is drove by the PWM pulse to realize power amplification. Feedback circuit is designed to sample the output signals to compose closed-loop configuration, which mainly takes charge of the transformation of amplitude and polarity.

For eliminating non-linear distortion, an algorithm of digital closed-loop modification is used based on the proposed hardware [12], which can be described as follow: Identify the lower-amplifier part with training data and establish an input-output model for the instrumentation system. By comparing the identification model's output with idea output, an adjusting function is generated to guide adaptive adjustment of fault data in numeric area before being to be input to the instrument, so that the output waveform can furthest approach to ideal value. It is clear that accurate identification of system is of great importance in the algorithm, and WPNN can be applied to complete this task because of its excellent time-frequency localization property and approximation ability.

4 Procedure of the Algorithm with WPNN

The procedure of digital closed-loop modification with WPNN is shown in Fig.3, which can be explained like that: Some random sampling points within the effective range are input to the actual instrument with proposed configuration and the output

waveform is recorded using the feedback circuit. The group composing by the sampling data and their corresponding feedback is regarded as training data set. An identification model $f_{id}(\bullet)$ is established by the training data set as substitute of unknown non-linear performance $f(\bullet)$ of instrument's amplifier in the algorithm. And then compare the output of $f_{id}(\bullet)$ to the idea output, and construct an adjusting function to compensate the initial to be put into the instrumentation system for realizing the goal of constraining distortion of output waveform greatly.

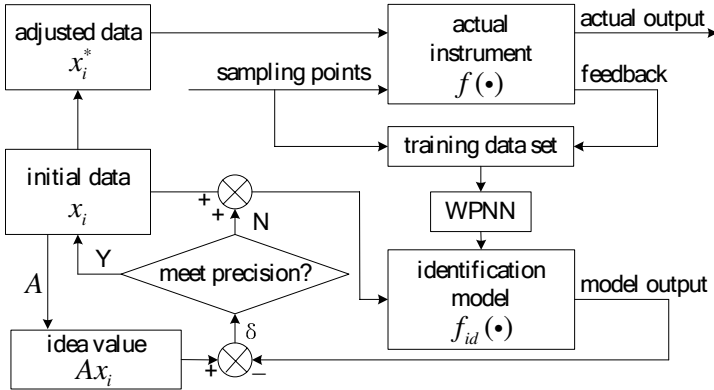


Fig. 3. Procedure of digital closed-loop modification with WPNN

Accurate system identification and acquirement of adjusting function are two the key points of the algorithm.

With its excellent time-frequency localization property and approximation ability, WPNN is used to establish the identification model for the system. Select a suitable mother wavelet function and estimate the frequency domain of the non-linear performance $f(\bullet)$ with training data set. Network structure and neurons number of WPNN can be determined by the method proposed in the second section, and the connection weights of WPNN can be trained by some optimization algorithm, e.g., back propagation (BP), genetic algorithm (GA), and etc.

And the adjusting function is obtained by the method of iterative modification. As shown in Fig.3, x_i denotes a certain data point of the fault data to be input to the instrument and $f_{id}(x_i)$ is its output amplified by the identified model $f_{id}(\bullet)$. The difference δ of $f_{id}(x_i)$ and idea amplifying value Ax_i , where A is the idea amplification factor, is used to adjust the original data x_i to x_i^* . And then setting x_i^* as initial point, repeat the process above until δ meets the precision requirement. The last δ is recorded into adjusting value form and the last x_i^* will be input to the testing instrument to realize fault waveform amplification.

This algorithm is essentially a compensating method for the non-linear performance of the amplifier, which makes the instrumentation system show linear characteristics on the whole, so that the non-linear error of output waveform can be greatly reduced.

5 Simulation Results

To testify the effectiveness of applying WPNN on relay protection testing of power system, a simulation experiment is carried out using actual fault data recorded in a certain region of Jiangxi Province.

Following the procedure mentioned above, an identification model is established using WPNN based on the training data and the compensating value related to each sampling data can be calculated by close-loop modifying, which is draw out in Fig.4. The results show that the identification model can accurately approximate the simulated non-linear performance and its tracking error is within 0.1%.

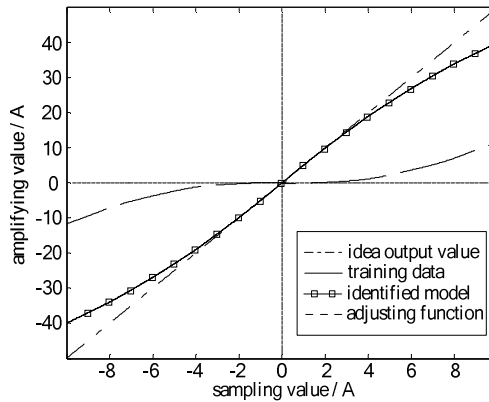


Fig. 4. System identification

Fig.5 displays a segment of initial input data of the simulation and its adjustment process by the compensating value. The initial data is a phase current of a current oscillation fault, whose maximum reaches up to 10A. And in peak or vale points, input data has bigger compensating value because of more serious non-linear attenuation.

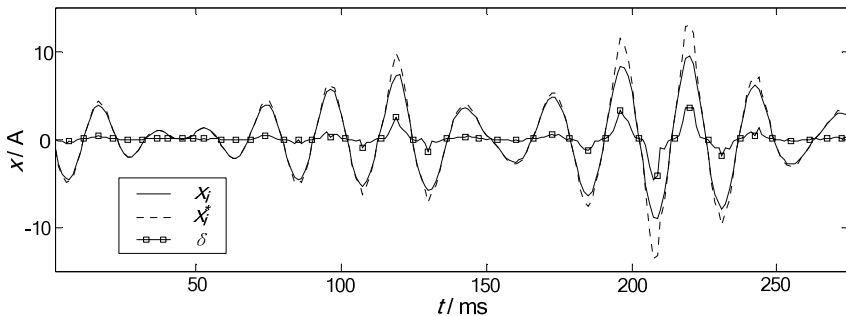


Fig. 5. Initial input data of the simulation and its adjustment

The comparison of output waveform with and without the method proposed in the paper is shown as Fig.6. Results from the analysis of the waveforms indicate that 1)

Because of non-linear performance of amplifier, the distortion will inevitably come into being in the output waveform which possibly leads to false relay protection testing conclusions; 2) By using system identification and close-loop modification, the root mean square error of output waveform reduces from 2.09 to 0.76. The distortion is constrained so greatly that the output waveform could simulate the power fault exactly, 3) and the compensation function is most remarkable especially at the points near peak or vale value.

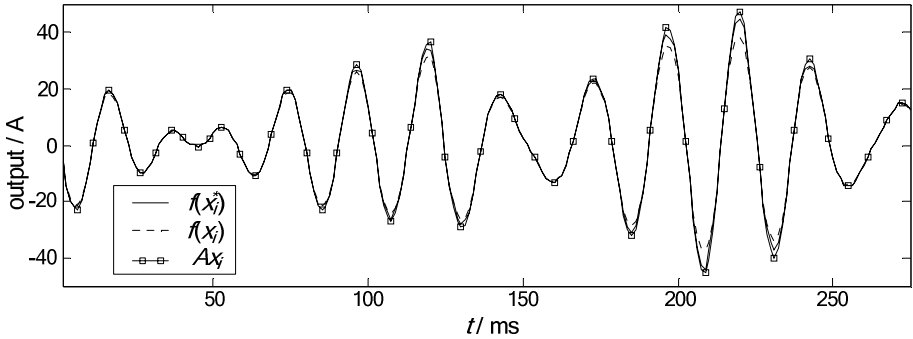


Fig. 6. Simulation results

6 Conclusion

(1) A novel neural networks, WPNN, with best wavelet packet basis as neuron's activation function is introduced in the paper, which has normative procedures of structure design and accurate system approximation performance.

(2) In this study, WPNN is applied to resolve the output waveform's distortion problem of protective relaying testing instrument. The simulation results prove its feasibility and validity and a prototype with the proposed algorithm has now put into practical operation.

(3) WPNN has excellent capability of approximating the complex nonlinear system, so it can also be applied to other modeling or optimizing problems in power system such as pattern recognition, fault diagnosis, load forecasting and data compress.

References

1. Jodice, J.A.: Relay Performance Testing: A Power System Relaying Committee Publication. IEEE Transactions on Power Delivery **12** (1997) 169-171
2. Sachdev, M.S., Sidhu, T.S., McLaren, P.G.: Issues and Opportunities for Testing Numerical Relays. IEEE Power Engineering Society Summer Meeting **2** (2000) 1185-1190
3. Benediktsson, J.A., Sveinsson, J.R., Ersoy, O.K., Swain, P.H.: Wavelet Packet Parallel Consensual Neural Networks. Intelligent Engineering Systems Through Artificial Neural Networks **5** (1995) 5-13

4. Avci, E., Turkoglu, I., Poyraz, M.: Intelligent Target Recognition Based on Wavelet Packet Neural Network. *Expert Systems with Applications* **29**(1) (2005) 175-182
5. Zhou, Z.J., Hu, C.H., Han, X.X., Chen, G.J.: Adaptive Wavelet Packet Neural Network Based Fault Diagnosis for Missile's Amplifier. *Second International Symposium on Neural Networks Proceedings* (2005) 591-596
6. Wang, L., Teo, K.K., Lin, Z.: Predicting Time Series with Wavelet Packet Neural Networks. *Proceedings of the International Joint Conference on Neural Networks* **3** (2001) 1593-1597
7. Schuck Jr., A., Guimaraes, L.V., Wisbeck, J.O.: Dysphonic Voice Classification using Wavelet Packet Transform and Artificial Neural Network. *Annual International Conference of the IEEE Engineering in Medicine and Biology* **3** (2003) 2958-2961
8. Zhang, Q.: Benveniste, A.: Wavelet Network. *IEEE Trans on Neural Networks* **3**(6) (1992) 889-898
9. Gao, X.P.: A Comparative Research on Wavelet Neural Networks. *Proceedings of the 9th International Conference on Neural Information Processing* **4** (2002) 1699-1703
10. Zhao, X.Z., Ye, B.Y.: Identification of Vibrating Noise Signals of Electromotor Using Adaptive Wavelet Neural Network. *Third International Symposium on Neural Networks* **2** (2006) 727-734
11. Meliopoulos, A.P.S., Cokkinides, G.J.: A Virtual Environment for Protective Relaying Evaluation and Testing. *IEEE Transactions on Power Systems* **19** (2004) 104-111
12. Sun, X.M., Du, X.W., Liu D.C., Cai, X.: The Obstacle Recurrence and Amplification Device Based on Digital Closed-loop Modification Technology. *Automation of Electric Power Systems* **28**(4) (2004) 49-53

Robust Stabilization of Uncertain Nonlinear Differential-Algebraic Subsystem Using ANN with Application to Power Systems

Qiang Zang and Xianzhong Dai

School of Automation, Southeast University,
Nanjing 210096, China
qzang@seu.edu.cn

Abstract. The controlled system is an uncertain nonlinear differential-algebraic subsystem (DASs) in a large-scale system. The problem of robust stabilization for such class of uncertain nonlinear DASs is considered in this paper. The robust stabilization controller is proposed based on backstepping approach using two-layer Artificial Neural Networks (ANN) whose weights are updated on-line. The closed-loop error systems are uniformly ultimately bounded (UUB) and the error of convergence can be made arbitrarily small. Finally, using the design scheme proposed in this paper, a governor controller is designed for one synchronous generator in a multi-machine power systems. The simulation results demonstrate the effectiveness of the proposed scheme.

1 Introduction

Many physical systems are described by differential-algebraic equation systems (DAS). Compared to ordinary differential equation (ODE) systems description, the differential-algebraic system description is a more general form of system description. Now the theoretic system of linear DAS has in principle formed parallel to that of the linear ODE systems^[1]. Some great progress has been made recently for nonlinear DAS. The sufficient conditions are presented for the stability of nonlinear DAS in [2]. The concept of controlled invariant distribution is introduced into nonlinear DAS in [3]. The problem of exact linearization for nonlinear DAS is considered in [4,5].

However, the controlled system in many practical engineering applications is often a nonlinear differential-algebraic subsystem (DASs) within a large-scale system. There exists constraint between the controlled DASs and the rest of the large-scale system, which constraint arises naturally from the point of physics. The controlled DASs is influenced by the rest of the large-scale systems. A so-called power systems component structural model formulated in [6] just falls into this category. As far as the writer knows, the research for nonlinear DASs has seldom been found.

Owing to its excellent ability to approximate a nonlinear function with satisfactory accuracy, Artificial Neural Network (ANN) has been applied to system identification or identification-based control^[7]. The goal of this paper is to investigate the robust stabilization problem for uncertain nonlinear DASs. Firstly an equivalent system is

achieved through a local diffeomorphism and a feedback. Then the robust stabilization controller is proposed for the equivalent system based on backstepping approach^[8] using ANN whose weights are tuned on- line. At last a governor controller is designed for one synchronous generator in multi-machine power systems to show the effectiveness of the proposed scheme in this paper.

2 System Description and Problem Formulation

We consider the following uncertain nonlinear DASs within large-scale systems [6]:

$$\begin{aligned} \dot{x} &= f_1(x, z) + g(x, z)u + H(x, z, \bar{v}), \\ 0 &= f_2(x, z, \bar{v}), \\ y &= h(x, z), \end{aligned} \tag{1}$$

where $x \in R^n$ is the vector of differential variables, $z \in R^m$ is the vector of algebraic variables, $\bar{v} \in R^s$ is the interconnection input acted on (1) by the rest of the large-scale systems, $u \in R$ is the control input, $y \in R$ is the control output, f_1, g, H, f_2, h are all sufficiently smooth vector fields. $H(\cdot)$ contains both parametric and nonparametric uncertainties. The interconnection input \bar{v} and its sufficient order time derivatives are local measurable and bounded.

Denote $\Omega = \{(x, z, \bar{v}) \in R^n \times R^m \times R^s : f_2(x, z, \bar{v}) = 0\}$. Without loss of generality, we assume that the origin is the isolated equilibrium of (1) and the zero equilibrium is not affected by the uncertainties.

The objective of this paper is to find a controller defined on a neighborhood of the origin such that the closed-loop system (1) has the following properties:

- 1) for $\forall \bar{v}$ there exists a unique solution $(x(t, x_0, z_0, \bar{v}_0), z(t, x_0, z_0, \bar{v}_0))$ with $(x(0), z(0), \bar{v}(0)) = (x_0, z_0, \bar{v}_0)$, where (x_0, z_0, \bar{v}_0) is the compatible initial condition^[21],
- 2) the state of the closed-loop system (1) can stay around the origin equilibrium as close as possible.

3 Main Results

Throughout this paper, the following assumptions are made for (1):

Assumption 1. The Jacobi matrix $\frac{\partial f_2}{\partial z}$ of f_2 with respect to z is nonsingular on Ω .

For convenience, we give the following notation

$$F(x, z, \bar{v}) = \begin{bmatrix} I_n \\ -\left(\frac{\partial f_2}{\partial z}\right)^{-1} \frac{\partial f_2}{\partial x} \end{bmatrix} \tag{2}$$

Definition. The DASs (1) is said to have uniform relative degree $r(1 \leq r \leq n)$ at the origin, if on a neighborhood $U_0 \in \Omega$ of the origin the following conditions hold:

$$\begin{aligned} L_{F_g} L_{F_{f_1}}^k h(x, z) &= 0, k = 0, \dots, r-2, \forall (x, z) \in U_0 \\ L_{F_g} L_{F_{f_1}}^{r-1} h(0, 0) &\neq 0, \end{aligned} \tag{3}$$

Assumption 2. The DASs (1) has uniform relative degree n at the origin.

Theorem 1. If the Assumptions 1 and 2 hold for (1), then on U_0 there exist a local diffeomorphism and a state feedback control such that the DASs (1) can be equivalently transformed into the following form:

$$\begin{aligned} \dot{\xi}_1 &= \xi_2 + \phi_1(\xi, \bar{v}, \dot{\bar{v}}), \\ &\vdots \\ \dot{\xi}_{n-1} &= \xi_n + \phi_{n-1}(\xi, \bar{v}, \dot{\bar{v}}), \\ \dot{\xi}_n &= v + \phi_n(\xi, \bar{v}, \dot{\bar{v}}), \\ \chi &= f_2(x, z, \bar{v}) = 0, \\ y &= \xi_1 \end{aligned} \tag{4}$$

where $\xi = (\xi_1, \dots, \xi_n)^T, \phi_i = \left(\frac{\partial L_{F_{f_1}}^{i-1} h}{\partial x} \quad \frac{\partial L_{F_{f_1}}^{i-1} h}{\partial z} \right) FH - \frac{\partial L_{F_{f_1}}^{i-1} h}{\partial z} \left(\frac{\partial f_2}{\partial z} \right)^{-1} \frac{\partial f_2}{\partial \bar{v}} \dot{\bar{v}}, i = 1, \dots, n.$

Proof. Define the following nonlinear transformation for (1) on U_0

$$\begin{bmatrix} \xi \\ \dots \\ \chi \end{bmatrix} = \begin{bmatrix} T(x, z, \bar{v}) \\ \dots \\ f_2(x, z, \bar{v}) \end{bmatrix} \tag{5}$$

where $\xi_1 = T_1(x, z) = h(x, z), \dots, \xi_n = T_n(x, z, \bar{v}) = L_{F_{f_1}}^{n-1} h(x, z).$ Similar to [5,9], it can be verified that (6) is a diffeomorphism. Then from (1) and (6), we can obtain that

$$\begin{aligned} \dot{\xi}_i &= \xi_{i+1} + \phi_i, 1 \leq i \leq n-1 \\ \dot{\xi}_n &= L_{F_{f_1}}^n h + L_{F_g} L_{F_{f_1}}^{n-1} hu + \phi_n \end{aligned}$$

where $\phi_i, 1 \leq i \leq n$ as indicated in (5). Then with diffeomorphism (6) and the following feedback

$$u = \frac{1}{L_{F_g} L_{F_{f_1}}^{n-1} h} (v - L_{F_{f_1}}^n h), \tag{6}$$

the nonlinear DASs (1) is equivalently transformed into (5) where v is the new control input to be designed later. The proof is completed.

Theorem 2. If Assumption 1 and 2 hold for (1) and the following conditions

$$\phi_i(\xi, \bar{v}, \dot{\bar{v}}) = \phi_i(\xi_1, \dots, \xi_i, \bar{v}, \dot{\bar{v}}), i = 1, \dots, n \tag{7}$$

are satisfied for (5), then the DASs (1) is robust stabilizable.

Proof. Suppose each ϕ_i is approximated by a two-layer ANN for some ideal constant weights $W_i, i = 1, \dots, n$, i.e.,

$$\phi_i = W_i^T \psi_i + \varepsilon_i, |\varepsilon_i| \leq c = \text{cons}, i = 1, \dots, n \tag{8}$$

where ψ_i is radial basis function. Then the estimate of ϕ_i can be chosen as

$$\hat{\phi}_i = \hat{W}_i^T \psi_i \tag{9}$$

where \hat{W}_i is the current ANN weight estimate. In this note, \hat{W}_i is updated on-line by following adaptive algorithm

$$\dot{\hat{W}}_i = \gamma_i \psi_i e_i - \sigma_i \hat{W}_i, i = 1, \dots, n \tag{10}$$

where $\gamma_i, \sigma_i > 0$ are design parameters. Then define the following error variables

$\tilde{W}_i^T = W_i^T - \hat{W}_i^T$ and $e_1 = \xi_1, e_i = \xi_i - \alpha_{i-1}, i = 2, \dots, n$, where the virtual controller α_i is

chosen as $\alpha_i = -c_i e_i - e_{i-1} - \hat{\phi}_i + \sum_{j=1}^i \frac{\partial \alpha_{i-1}}{\partial e_j} \dot{e}_j + \sum_{j=1}^i \frac{\partial \alpha_{i-1}}{\partial \bar{v}^{(j-1)}} \bar{v}^{(j)}$ and the actual controller

v is $v = -c_n e_n - e_{n-1} - \hat{\phi}_n + \sum_{j=1}^n \frac{\partial \alpha_{n-1}}{\partial e_j} \dot{e}_j + \sum_{j=1}^n \frac{\partial \alpha_{n-1}}{\partial \bar{v}^{(j-1)}} \bar{v}^{(j)}$ with $c_i > 0$ is the design

parameter. The Lyapunov function for the whole closed-loop error system is picked as

$$V = \frac{1}{2} \sum_{i=1}^n e_i^2 + \frac{1}{2} \sum_{i=1}^n \frac{1}{\gamma_i} \tilde{W}_i^T \tilde{W}_i \tag{11}$$

It can be verified that with v and (7), the time derivative of V along the whole closed-loop error system satisfies the following inequality

$$\dot{V} \leq -kV + \sum_{i=1}^n \left(\frac{\sigma_i}{2\gamma_i} M + \frac{1}{2c_i} c \right) \tag{12}$$

where $k = \min(c_i, \sigma_i), i = 1, \dots, n$ and M is the upper bound of W_i . Then by virtue of Assumption 1 and Theorem 2 in [2], the closed-loop DASs (1) is uniformly ultimately bounded (UUB). Furthermore, all states of the closed-loop systems converge to a compact residual set which can be made arbitrarily small if c_i, σ_i are chosen large enough. The proof is completed.

4 Application to Power Systems

Based on the proposed control scheme, the speed governor control of turbo-generator set G1 (DASs) in the two-area four-machine large power systems (Fig.1) is studied

[10]. Based on MATLAB, simulation is conducted under the following operating status: at the beginning the system operates with double lines in a stable state; at 0.5s a three-phase symmetrical short-circuit to ground on one of the lines occurs at point $k = 0.1$ (see Fig.1); at 0.65s the fault is cleared, and the system operates in two-line mode.

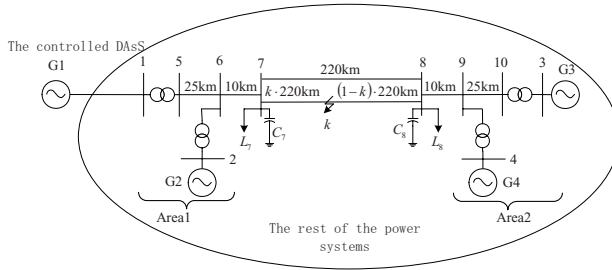


Fig. 1. Two-area four-machine power systems

The DAS model of the generator G1 is^[6]:

$$\begin{aligned}
 \dot{\delta} &= \omega - \omega_0, \\
 \dot{\omega} &= \frac{\omega_0}{H} \{ P_H + C_{ML} P_{m0} - \frac{D}{\omega_0} (\omega - \omega_0) \\
 &\quad - \frac{\omega_0}{H} [E'_q + (x_q - x'_d) I_d] I_q \}, \\
 \dot{P}_H &= \frac{1}{T_{H\Sigma}} (-P_H + C_H P_{m0} + C_H U_c), \\
 0 = f_2^1(\cdot) &= I_d - \frac{I_t (Q_t + x_q I_t^2)}{\sqrt{(Q_t + x_q I_t^2)^2 + (P_t + r_a I_t^2)^2}}, \\
 0 = f_2^2(\cdot) &= I_q - \frac{I_t (P_t + r_a I_t^2)}{\sqrt{(Q_t + x_q I_t^2)^2 + (P_t + r_a I_t^2)^2}}, \\
 0 = f_2^3(\cdot) &= E'_q - \frac{(P_t + r_a I_t^2)^2 + (Q_t + x'_d I_t^2)(Q_t + x_q I_t^2)}{I_t \sqrt{(Q_t + x_q I_t^2)^2 + (P_t + r_a I_t^2)^2}}, \\
 0 = f_2^4(\cdot) &= \theta_U - \delta - \text{arc ctg} \frac{x_q I_q - r_a I_d}{E'_q - x'_d I_d - r_a I_q}
 \end{aligned}
 \tag{13}$$

$$\tag{14}$$

where vector of differential variables $(\delta, \omega, P_H)^T$ are relative power angle between G1 and G4, rotate speed deviation of G1 and the high pressure mechanical power respectively, the vector of algebraic variables $(I_d, I_q, Q_t, \theta_U)^T$ are the d-axis current, the q-axis current, the reactive power and the angle of voltage respectively, and the

interconnection input (I_r, P_r) are the generator stator current and active power. The governor position U_c is the control input, and the output of G1 is chosen as $y = \delta$.

We can verify that the conditions in Theorem 2 hold. According to the design scheme proposed in Theorem 2, the simulation results show as follows:

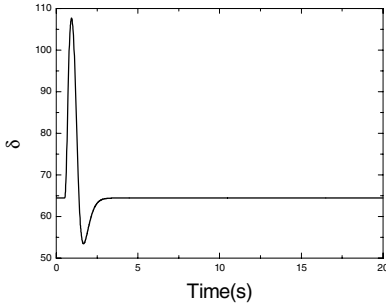


Fig. 2. Relative power angle between G1 and G4

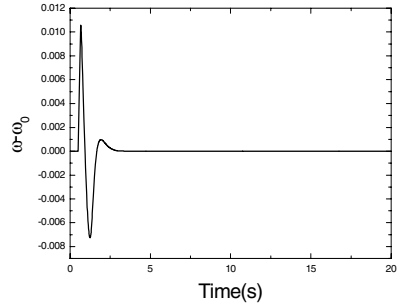


Fig. 3. Rotate speed deviation of G1

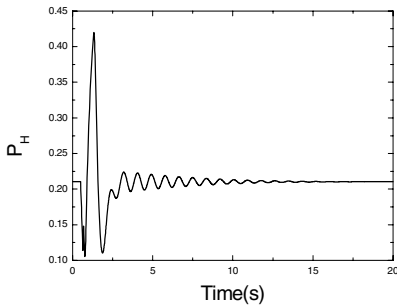


Fig. 4. The high pressure mechanical power

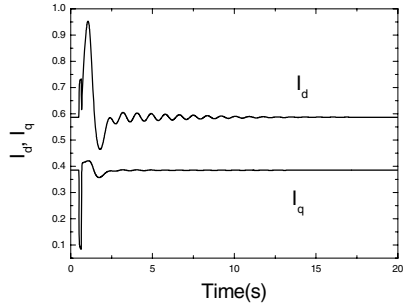


Fig. 5. The d-axis current and the q-axis current

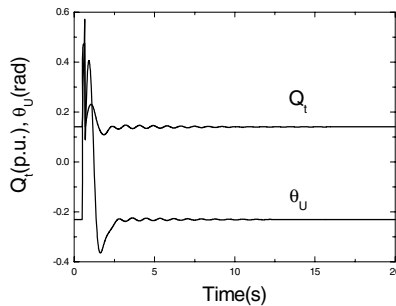


Fig. 6. The reactive power and the angle of voltage

Apparently, all the states of G1 have good performance of convergence.

5 Conclusion

For a class of uncertain nonlinear DASs, the robust stabilization controller design scheme is proposed in this paper based on backstepping approach using ANN whose weights are updated on-line.

Acknowledgements. This work is supported by National Natural Science Foundation of China (50507002) and National Basic Research Program of China under Grant (2002CB312204).

References

1. S. L. Campbell, N. Nichols, W. J. Terrell.: Duality. Observability and Controllability for Linear Time-Varying Descriptor Systems. *Circuits, Systems, Signal process* **10** (3) (1991) 455-470
2. D. J. Hill, I. M. Y. Mareels.: Stability Theory for Differential/Algebraic Systems with Application to Power Systems. *IEEE Trans. Circuits and Systems* **37**(11) (1990)1416-1423
3. Wang Wen-Tao, Liu Xiao-Ping, Zhao Jun.: Controlled Invariant Distributions of Nonlinear Singular Systems and Their Invariant Properties. *ACTA Automatica Sinica* **30** (6) (2004) 911-919
4. J Wang, C. Chen.: Exact Linearization of Nonlinear Differential Algebraic Systems International Conference on info-tech and info-net **4** (2001) 284-290
5. Zhu, J., Cheng, Z.: Exact Linearization for a Class of Nonlinear Differential-Algebraic Systems. Proceedings in the 4th World Congress on Intelligent Control and Automation (2002) 211-214
6. Zhang Kai Feng.: Decentralized Nonlinear Control Method of Power Systems Based on Interface Concept. [Doctor Dissertation]. Nanjing Southeast University. (2004)
7. Kwan, C., Lewis, L.F.L.: Robust Backstepping Control of Nonlinear Systems Using Neural Networks. *IEEE Trans. Systems, Man and Cybernetics-Part A: Systems and Humans* **30** (6) (2000) 753-766
8. Kanellakopoulos, M. K., Kokotovic, P.V.: *Nonlinear and Adaptive Control Design*. New York: Wiley (1995)
9. Byrnes, C.I., Isidori, A.: Asymptotic Stabilization of Minimum Phase Nonlinear Systems. *IEEE Trans. Autom.Control* **36** (10) (1991) 1122-1137
10. Kundur, P.: *Power System Stability and Control*. New York: McGraw-Hill Inc (1994)

A Novel Residual Capacity Estimation Method Based on Extension Neural Network for Lead-Acid Batteries

Kuei-Hsiang Chao¹, Meng-Hui Wang¹, and Chia-Chang Hsu²

¹ National Chin-Yi University of Technology, Department of Electrical Engineering, 35, 215 Lane, Sec. 1, Chung Shan Road, Taiping, Taichung, Taiwan
{chaokh, wangmh}@ncut.edu.tw

² National Chin-Yi Institute of Technology, Taichung, Institute of Information and Electrical Energy, 35, 215 Lane, Sec. 1, Chung Shan Road, Taiping, Taichung, Taiwan
change0413@hotmail.com

Abstract. This paper presents a state-of-charge (SOC) estimation method based on extension neural network (ENN) theory for lead-acid batteries. First, a constant electric current discharging experiment with an electronic load for lead-acid batteries is made to measure and record the internal resistance and open-circuit-voltage by utilizing internal resistance tester. Then, the experimental data are adopted to construct an estimation method based on ENN for recognizing the residual capacity of the lead-acid battery. The simulated results indicate that the proposed estimation method can determine the residual capacity of lead-acid batteries rapidly and accurately with less time and memory consumption.

1 Introduction

There are currently many methods for estimating the residual capacity of lead-acid batteries, including electrolyte specific gravity, open circuit voltage, internal resistance, coulometric measurement, and loaded voltage [1-4]. Among these, the open circuit voltage and internal resistance methods are the most commonly used. The open circuit voltage method uses the linear relation between voltage and the residual capacity of the lead-acid batteries to carry out SOC estimation. However, it requires between half an hour to one hour, which is the recovery time of the open circuit voltage of the battery after moving charge state to discharge state, for a lead-acid battery to achieve a stable state. Therefore, the open circuit voltage of a battery is used during this period to estimate the residual capacity, there will be sizable errors. Moreover, during this period, the open circuit voltage with respect to remnant capacity is not linear but quasi-linear relation. The internal resistance method is adopted to estimate the residual capacity of lead-acid battery by using the change of internal resistance during battery discharging. However, mistakes may occur when the value is not sufficiently accurate enough due to the small variety of resistance at the early discharge period.

This paper presents a residual capacity estimation method based on ENN for SOC estimation of the lead-acid battery, which uses a combination of neural networks and

extension theory. The extension theory [5-6] provides a novel distance measurement for classification processes, while the neural network can embed the salient features of parallel computation power and learning capability [7]. The proposed residual capacity estimation method will first create a set of remant capacity matter-elements of the lead-acid battery, and then a regular extended correction function can directly estimate the residual capacity of the battery by calculating the degrees of extended correction. According to the results, the proposed capacity estimation method can discriminate the residual capacity exactly, and thus make the most efficient use of the lead-acid battery energy.

2 Lead-Acid Battery

Figure 1 shows the equivalent circuit of a lead-acid battery [8]. Here R_{in1} is the equivalent resistance between the electrode and the electrolyte of a battery, and R_{in2} is the interface resistance of the battery electrode and the battery electrolyte. C is the variety quantity of voltage during discharging, which is the electricity capacity through the space charge between the active substrate and the electrolyte interface. I is the discharge current. The equation of the battery voltage can be obtained from Fig.1:

$$V_b = V_{oc} - (R_{in1} + R_{in2})I + R_{in2}Ie^{\left(\frac{-t}{CR_{in1}}\right)} - V_C e^{\left(\frac{-t}{CR_{in2}}\right)} \tag{1}$$

If electric circuit is in a steady state, equation (1) can be simplified as equation (2)

$$V_b = V_{oc} - I(R_{in1} + R_{in2}) \tag{2}$$

Then the internal resistance can be derived as equation (3)

$$(R_{in1} + R_{in2}) = \Delta V / I \tag{3}$$

where $\Delta V \equiv V_{oc} - V_b$

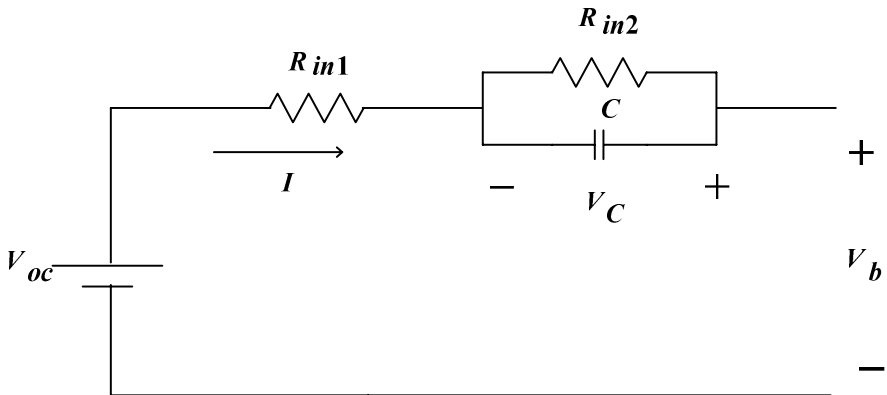


Fig. 1. Equivalent circuit of lead-acid battery

3 The Summary of Extension Theory

Based on extension theory, we can define matter, called N , whose characteristic vector is C , and V is the value vector related to C [5]. The multi-dimensional matter-element can be expressed as

$$R = (N, C, V) = \begin{bmatrix} R_1 \\ R_2 \\ \dots \\ R_n \end{bmatrix} = \begin{bmatrix} N, & c_1, & v_1 \\ & c_2, & v_2 \\ & \dots & \dots \\ & c_n, & v_n \end{bmatrix} \tag{4}$$

Based on the matter-element model, a new mathematical concept can be established to characterize the relationship between the quality and quantity of a matter by matter-element model. The extension set extends the fuzzy set from $[0, 1]$ to $(-\infty, \infty)$ [5]. An extension set is composed of the following two definitions.

Definition 1. Let U be a space of objects and x a generic element of U , then an extension set \tilde{E} in U is defined as a set of ordered pairs as follows:

$$\tilde{E} = \{(x, y) | x \in U, y = K(x) \in (-\infty, \infty)\} \tag{5}$$

where $y=K(x)$ is called the correlation function for extension set \tilde{E} . The $K(x)$ maps each element of U to a membership grade between $-\infty$ and ∞ . An extension set \tilde{E} in U can be denoted by:

$$\tilde{E} = E^+ \cup Z_o \cup E^- \tag{6}$$

where

$$E^+ = \{(x, y) | x \in U, y = K(x) > 0\} \tag{7}$$

$$Z_o = \{(x, y) | x \in U, y = K(x) = 0\} \tag{8}$$

$$E^- = \{(x, y) | x \in U, y = K(x) < 0\} \tag{9}$$

In Eqs. (7) to (9), E^+ , E^- and Z_o are called the positive field, negative field and zero boundary in \tilde{E} , respectively.

Definition 2. If $X_o = \langle a, b \rangle$ and $X = \langle f, g \rangle$ are two intervals in the real number field, and $X_o \subset X$, where X_o and X are the classical (concerned) and neighborhood domains, respectively. The correlation function in the extension theory can be defined as follows:

$$K(x) = \begin{cases} -\rho(x, X_o) & x \in X_o \\ \frac{\rho(x, X_o)}{\rho(x, X) - \rho(x, X_o)} & x \notin X_o \end{cases} \tag{10}$$

where

$$\rho(x, X_o) = \left| x - \frac{a+b}{2} \right| - \frac{b-a}{2} \tag{11}$$

$$\rho(x, X) = \left| x - \frac{f + g}{2} \right| - \frac{g - f}{2} \tag{12}$$

The correlation function can be used to calculate the membership grade between x and X_o .

4 Extension Neural Network

4.1 The Structure of ENN

The schematic structure of the ENN is depicted in Fig. 2. It includes both the input layer and the output layer. The nodes in the input layer receive an input feature pattern and use a set of weighted parameters to generate an image of the input pattern. In this network, there are two connection values (weights) between input nodes and output nodes, one connection represents the lower bound for this classical domain of the features, and the other connection represents the upper bound. The connections between the j -th input node and the k -th output node are w_{kj}^L and w_{kj}^U . This image is further enhanced in the process characterized by the output layer. Only one output node in the output layer remains active to indicate a classification of the input pattern. The operation mode of the proposed ENN can be separated into the learning phase and the operation phase. The learning algorithm of the ENN is discussed in the next section.

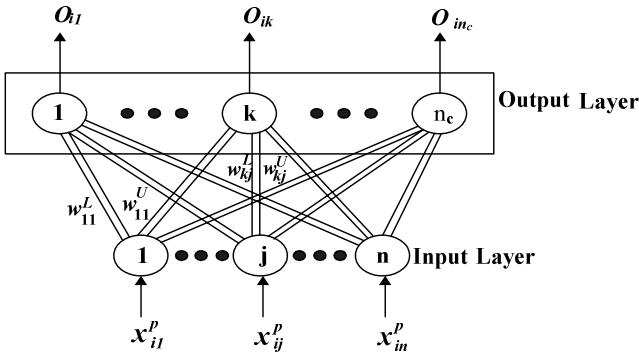


Fig. 2. The structure of extension neural network

4.2 Learning Algorithm of the ENN

The learning of the ENN can be seen as supervised learning, and its purpose is to tune the weights of the ENN to achieve good clustering performance or to minimize the clustering error. Before the learning, several variables have to be defined. Let training pattern set be $X \equiv \{X_1, X_2, \dots, X_{N_p}\}$, where N_p is the total number of training patterns.

The i -th pattern is $X_i^p \equiv \{x_{i1}^p, x_{i2}^p, \dots, x_{in}^p\}$, where n is the total number of the feature of

patterns, and the category of the i -th pattern is p . To evaluate the clustering performance, the total error number is set as N_m , and the total error rate E_τ is defined below:

$$E_\tau = \frac{N_m}{N \cdot p} \tag{13}$$

The detailed supervised learning algorithm can be described as follows:

Step 1: Set the connection weights between input nodes and output nodes. The range of classical domains can be either directly obtained from the previous requirement, or determined from training data as follows:

$$w_{kj}^L = \min_{i \in N} \{x_{ij}^k\} \tag{14}$$

$$w_{kj}^U = \max_{i \in N} \{x_{ij}^k\} \tag{15}$$

Step 2: Calculate the initial cluster center of every cluster.

$$Z_k = \{z_{k1}, z_{k2}, \dots, z_{kn}\} \tag{16}$$

$$z_{kj} = (w_{kj}^L + w_{kj}^U) / 2 \tag{17}$$

for $k = 1, 2, \dots, n_c$; $j = 1, 2, \dots, n$

Step 3: Read the i -th training pattern and its cluster number p .

$$X_i^p = \{x_{i1}^p, x_{i2}^p, \dots, x_{in}^p\}, p \in n_c \tag{18}$$

Step 4: Use the proposed extension distance (ED) to calculate the distance between the training pattern X_i^p and the k -th cluster, as follows:

$$ED_{ik} = \sum_{j=1}^n \left[\frac{|x_{ij}^p - z_{kj}| - (w_{kj}^U - w_{kj}^L) / 2}{|(w_{kj}^U - w_{kj}^L) / 2|} + 1 \right] \tag{19}$$

$k = 1, 2, \dots, n_c$

The proposed distance is a modification of extension distance [5], and it can be graphically presented as in Fig. 3. It can describe the distance between the x and a range $\langle w^L, w^U \rangle$. Figure 3 shows that different ranges of classical domains can arrive at different distances due to different sensitivities. This is a significant advantage in classification applications. Usually, if the feature covers a large range, the data should be fuzzy or less sensitive to distance. On the other hand, if the feature covers a small range, the data should be precise or highly sensitive to distance.

Step 5: Find the k^* , such that $ED_{ik^*} = \min\{ED_{ik}\}$. If $k^* = p$ then go to Step 7, otherwise Step 6.

Step 6: Update the weights of the p -th and the k^* -th clusters as follows:

(a) Update the centers of the p -th and the k^* -th clusters.

$$z_{pj}^{new} = z_{pj}^{old} + \eta (x_{ij}^p - z_{pj}^{old}) \tag{20}$$

$$z_{k^*j}^{new} = z_{k^*j}^{old} - \eta (x_{ij}^p - z_{k^*j}^{old}) \tag{21}$$

(b) Update the weights of the p -th and the k^* -th clusters.

$$\begin{cases} W_{pj}^{L(new)} = W_{pj}^{L(old)} + \eta(x_{ij}^p - z_{pj}^{old}) \\ W_{pj}^{U(new)} = W_{pj}^{U(old)} + \eta(x_{ij}^p - z_{pj}^{old}) \end{cases} \quad (22)$$

$$\begin{cases} W_{k^*j}^{L(new)} = W_{k^*j}^{L(old)} - \eta(x_{ij}^p - z_{k^*j}^{old}) \\ W_{k^*j}^{U(new)} = W_{k^*j}^{U(old)} - \eta(x_{ij}^p - z_{k^*j}^{old}) \end{cases} \quad (23)$$

where η is a learning rate. The result of tuning two clusters' weights shown in Fig. 4, which clearly indicates the change of ED_A and ED_B . The cluster of pattern x_{ij} is changed from cluster A to B because $ED_A > ED_B$. From this step, we can clearly see that the learning process is only to adjust the weights of the p-th and the k^* -th clusters. Therefore, the proposed method has a rapid speed advantage over other supervised learning algorithms and can quickly adapt to new and important information.

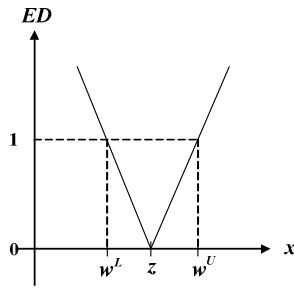


Fig. 3. The proposed extension distance

Step 7: Repeat Step 3 to Step 6, and if all patterns have been classified then a learning epoch is finished.

Step 8: Stop if the clustering process has converged or the total error rate E_τ has arrived at a preset value; otherwise, return to Step 3.

It should be noted that the proposed ENN can take input from human expertise before the learning, and it can also produce meaningful output after the learning, because the classified boundaries of the features are clearly determined.

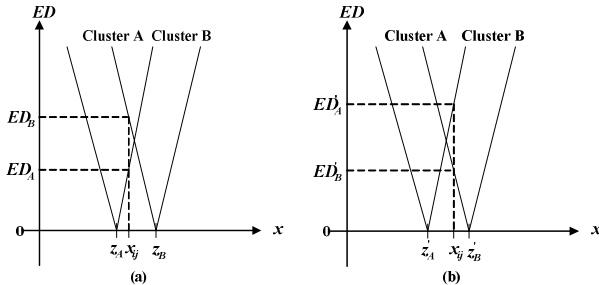


Fig. 4. The results of tuning cluster weights: (a) original condition; (b) after Tuning

4.3 Operation Process of ENN

There can be recognition or sorting when the ENN completes a learning procedure, and its operation procedure is summarized as follows:

Step 1: Read the weight matrix of ENN.

Step 2: Calculate the initial cluster centers of every cluster by using equation (16) and equation (17).

Step 3: Read the tested pattern.

$$X_t = \{x_{t1}, x_{t2}, \dots, x_{tm}\} \tag{24}$$

Step 4: Use the proposed extension distance (ED) to calculate the distance between the tested pattern and every existing cluster by equation (19).

Step 5: Find the k^* , such that $ED_{ik^*} = \min\{ED_{ik}\}$, and set the $O_{ik^*} = 1$ to indicate the cluster of the tested pattern.

Step 6: Stop, if all the tested patterns have been classified, otherwise go to Step 3.

5 Proposed SOC Estimation Method

5.1 Operation Phase of the Proposed SOC Estimation Method

Classify the SOC of lead-acid batteries into ten types according to the possible range of open circuit voltage, internal resistance and short circuit current (which defined as V_{oc} / R_m). The definitions of these ten types are listed in Table 1. Fig. 5 shows the flowchart of the operation process for the proposed SOC estimation method based on ENN. The operation procedure is shown as follows:

Step 1: Input the training data and implement the training process of the ENN in section 4.3 until it matches the setting recognized rate of 0.01%.

Step 2: Input test data.

Step 3: Carrying on the SOC estimation of the battery by using the ENN which has been trained.

Step 4: Return to step 2 and carry on next test data for SOC estimation, until all test data have been completed.

Table 1. The ten types definition of the residual capacity for lead-acid batteries

K1	Definition of the residual capacity is 90%	K6	Definition of the residual capacity is 40%
K2	Definition of the residual capacity is 80%	K7	Definition of the residual capacity is 30%
K3	Definition of the residual capacity is 70%	K8	Definition of the residual capacity is 20%
K4	Definition of the residual capacity is 60%	K9	Definition of the residual capacity is 10%
K5	Definition of the residual capacity is 50%	K0	Definition of the residual capacity is 0%

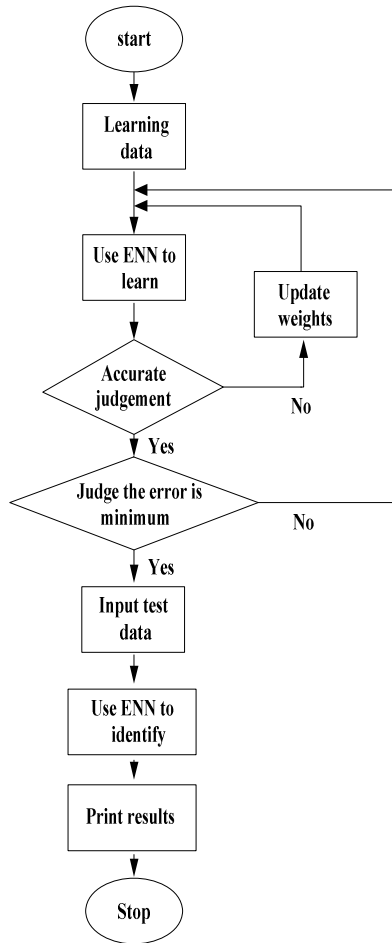


Fig. 5. Flowchart of the operation process for the proposed SOC estimation method based on ENN

5.2 Simulation Results

In order to confirm the effectiveness of the proposed SOC estimation method based on ENN, 100 random sample data with known SOC are selected for testing by using the MATLAB software package. The simulated results for the SOC estimate based on extension theory and on ENN are shown in Fig. 6 and Fig. 7 respectively, which indicates that there are four errors caused by extension theory, but only two errors occur when using the proposed ENN. This shows that the recognition rate based on extension theory estimation is 96%, whereas it is 98% for ENN.

Table 2 compares the results of SOC estimation by using ENN and by using extension theory, showing that the recognition rate of ENN is better than that of only extension theory since the weight can be adjusted by using ENN.

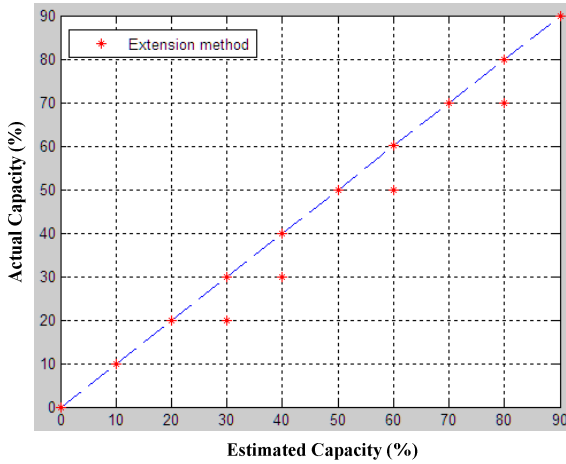


Fig. 6. The recognition results of extension method

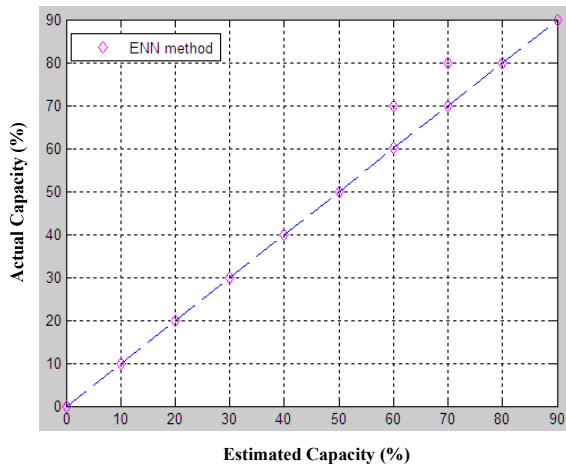


Fig. 7. The recognition results of ENN method

Table 2. The comparison of SOC estimation by using ENN and extension theory

Method	Training set		Testing set	
	Error/Total	Accuracy	Error/Total	Accuracy
Extension	0/100	100%	4/100	96%
ENN	0/100	100%	2/100	98%

6 Conclusions

In this paper, an SOC estimation method for lead-acid batteries based on the ENN was proposed. The proposed novel ENN method is based on the extension theory and

neural networks. First, the SOC data of batteries are measured by charging and discharging experiments. Secondly, the recognition rate of the proposed ENN are compared with extension theory. The simulated results show that the proposed ENN estimation method can recognize the residual capacity of lead-acid batteries accurately and rapidly. The proposed ENN method combines the neural network learning process, and the fast recognition characteristic of the extension theory, so it has the advantages of less learning time, higher accuracy and less memory consumption. The use of less training data and a higher identification rate are the advantages of the proposed ENN method. When the capacity of the lead-acid batteries increases, only a small portion of the data should be modified, so the update interval may be significantly reduced.

References

1. Aylor, J.H., et al.: A Battery State-of-charge Indicator for Electric Wheelchairs. *IEEE Transactions on Industrial Electronics* (1992) 398-409
2. Hlavac, M.J., Feder, D.: VRLA Battery Monitoring Using Conductance Technology. *International Telecommunications Energy Conference, INTELEC* (1995) 284-291
3. Yanagihara, T., Kawamura, A.: Residual Capacity Estimation of Sealed Lead-acid Batteries for Electric Vehicles. *Power Conversion Conference* (1997) 943-946
4. Caumont, O., et al.: Energy Gauge for Lead-acid Batteries in Electric Vehicles. *IEEE Transactions on Energy Conversion* **15**(3) (2000) 354-360
5. Cai, W.: The Extension Set and Incompatibility Problem. *Journal of Scientific Exploration* **1** (1983) 81-93
6. Wang, M.H., Chen, H.C.: Application of Extension Theory on the Fault Diagnosis of Power Transformers. In: *Proceeding of the 22nd Symposium on Electrical Power Engineering*. Taiwan (2001) 797-800
7. Wang, M.H., Hung, C.P.: Extension Neural Network. *Proceeding of the International Joint Conference on Neural Network* **1** (2003) 399-403.
8. Sato, S., Kawamura, A.: A New Estimation Method of Charge Using Terminal Voltage and Internal Resistance for Lead-acid Battery. *Power Conversion Conference* **2** (2002) 565-570

A Genetic Algorithm-Based Artificial Neural Network Approach for Parameter Selection in the Production of Tailor-Welded Blanks

Yun Liu, De Xu, Xiuqing Wang, Min Tan, and Yongqian Zhang

The Key Laboratory of Complex Systems
and Intelligent Science, Institute of Automation,
Chinese Academy of Sciences, Beijing, 100080, P.R. China
{Yun.Liu, De.Xu, Xiuqing.Wang, Min.Tan}@ia.ac.cn

Abstract. Laser cutting and welding is an efficient way to produce Tailor-Welded Blanks (TWBs). A genetic algorithm (GA)-based artificial neural network (ANN) approach is designed for parameter selection of laser cutting and welding to produce TWBs. These parameters include laser power for cutting and welding, speed for cutting and welding, and pressure of assistant gas. Experimental results demonstrate that the proposed parameter selection approach combines the merits of GA and ANN, and solves the problem of local optimum in ANN and low convergence speed in GA. As a result, it tackles the difficulty in parameter selection of laser cutting and welding and paves the way for TWBs' production.

1 Introduction

Tailor-Welded Blanks (TWBs) are composed of the materials with different characteristics and dimension. The materials are welded together by many kinds of arts. TWBs have their great advantages in automobile's design and manufacture [1] [2] [3]. They enhance the precision of part by the procedure of welding their components together before they are stamped. The left material tailored from the large blocks can be reused to form TWBs. The number of dies used in stamping is decreased and the assembly process is simplified accordingly. In addition, the flexibility of components' design is increased as the type of materials used in TWBs can be added as needed. TWBs have been widely used in automobile industry since 1980s. Experiments from laboratories and automobile manufactures have verified that TWBs could be successfully and reliably used to produce automobile components and parts [4] [5] [6]. Laser cutting and welding is an efficient art to produce TWBs with high quality, in which, the steel blanks are cut first, and then are welded together without any filling material along the gap formed by the two cuts. The parameter selection for laser cutting and welding is a difficult but very important step in the course of TWBs' manufacturing for it directly affects the quality of TWBs' seam. The factors affecting the quality of laser cutting are numerous and complicated. J. Fieret et al. think the

number of the factors is up to 50 [7]. In the early time, the good parameters for cutting and welding can be obtained by experiences from a certain number of experiments, as the working mode of laser is single and processing on the target is simple. However, because of the improvement of laser's quality and the high requirement from the processing target, the parameters can't be simply acquired from experiences as before. In [8] [9], an analytical expression of processing quality about cutting and welding parameters is established by the study on energy balance, and heat balance, etc, in the course of cutting and welding. However, obtaining the analytical expression demands a lot of hard work, which is obviously unfeasible for the practical manufacturing. In [10], an artificial neural network (ANN) has been employed to analyze the experimental data, and to predict the processing parameters. It provides a new insight for parameter selection. The back propagation (BP) algorithm is suitable for the optimization of objects with undetermined models [11] [12], so it enjoys wide application in industry. However, the training algorithm of BP demands the conversion from an object with a group of inputs and outputs into an optimization of nonlinear problem without constraints. Usually, there are many locally optimal solutions for the weights connecting the network because of high dimension brought up by the conversion.

If the conventional gradient descent search (GDS) is used, the solution will be easily trapped in the local optimal locations. As a result, the object function cannot reach global optimal values. In order to overcome the local optimal, the genetic algorithm (GA) may be a desirable choice. However, low speed of GA's convergence retards it from wide application. The extended genetic algorithm (EGA) is also proposed to deal with the difficulty. Experiments show that EGA enhances the training speed and precision for ANN. For small and middle model ANN, EGA converges more slowly than GDS does. Therefore, the combination of EGA and GDS may be a good solution. In the trap of locally optimal points, EGA is used to escape away from the trapping. However, in the space outside of the traps, the converging speed is increased if GDS is employed. The combination of EGA and GDS is desirable in theory, however, its difficulty lies in finding the exchange points between EGA and GDS. A GA-based ANN approach is presented for parameter selection, in which, GA is used to train the weights connecting ANN.

2 A GA-Based Training Method for the ANN's Connecting Weights

GA has been used in training ANN's connecting weights and structure [12]. Here, GA is used for the former case. As the objective of ANN's learning course is to lower its energy function, the learning course can be seen as a course of minimizing the energy function with the connecting weights as its variables. So it is feasible to optimize the connecting weights with GA. The scheme of the course is shown in Fig.1. The structure of ANN can be determined by the number of inputs and outputs, complexity of the problem to be solved, designer's

experiences, etc. Usually, trial and error may be a good approach to determine the structure. The seed group will be generated when the binary codes for the weights connecting the net are encoded according to the precision requirement. The seeds are initialized randomly. Using the samples from experiments, the fitness is calculated for every seed in the group. If the precision requirement is met, the training course will be completed. Otherwise, the good seeds will be selected, and a new offspring will be produced after the operation of crossover and mutation. The course will be reiterated until a desirable result is obtained.

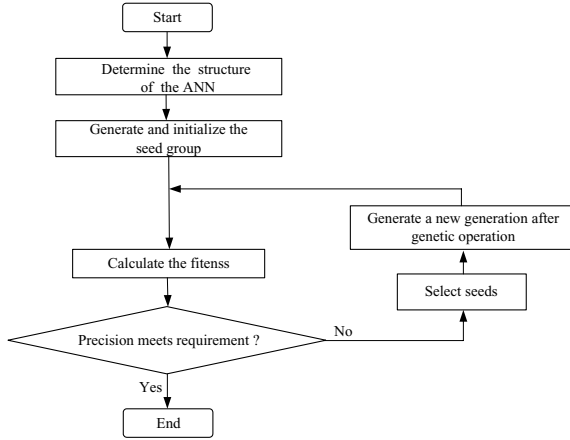


Fig. 1. The control flow of GA training for ANN's connecting weights

As is stated earlier, the disadvantage of GA lies in its low evolutionary speed. In EGA, the crossing and adaptive mutation for multiple points have been used to enhance training speed of BP algorithm. However, optimizing ANN is a new challenge to GA, as GA is built on the assumption that the problem to be optimized with high dimension can be expressed in binary codes. In conventional GA, the length of the solution for GA (i.e., the chromosome) is less than 50 bits, but the length of codes for ANN may be up to thousands of bits. Therefore, the arising problems are not only how GA can be applied in ANN, but also how to improve evolutionary speed and make it meet the required precision.

3 The ANN Model for Parameter Selection of Laser Cutting and Welding

According to the discussion in Section 1, there are numerous factors affecting the quality of laser cutting and welding. However, for a specified kind of steel blank, only laser power, pressure of assistant gas, thickness of steel, speed of cutting and welding are the main factors. For each kind of material, a sample group is built for it. The built model is shown in Fig. 2, which consists of an input

layer, a hidden layer and an output layer. The weights of the network are trained by samples. For different types of materials, different samples are used. In this model, the input parameters include the laser power for cutting and welding, assistant gas pressure for cutting and welding, the thickness of steel blanks, and the output parameters are the speed for cutting and welding. The hidden layer's structure of the ANN (i.e., six sub-layers and six neurons for each sub-layer) is determined by the results from our trial and error.

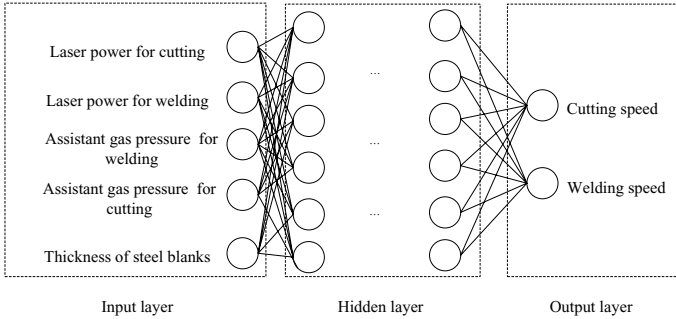


Fig. 2. The ANN model for parameter selection of laser cutting and welding

4 Parameter Selection Algorithm of GA-Based ANN

According to the rule of GA, the parameters to be optimized should be encoded first. In the parameter selection model, the connecting weights are the targets to be optimized. For convenience, we define a matrix M_1 with the dimension of 5×6 for the weights connecting the input layer with the sub-layer 1 of the hidden layer, $M_n (2 \leq n \leq 6)$ with the dimension of 6×6 for the weights connecting the sub-layer $n - 1$ with sub-layer n in the hidden layer, and M_o with the dimension of 2×6 for weights connecting the last sub-layer of the hidden layer with the output layer. So we have

$$\begin{aligned}
 M_1 &= \begin{bmatrix} W_{11}^{(1)} & \cdots & W_{11}^{(1)} \\ \vdots & \vdots & \vdots \\ W_{61}^{(1)} & \cdots & W_{65}^{(1)} \end{bmatrix}_{6 \times 5}, & M_2 &= \begin{bmatrix} W_{11}^{(2)} & \cdots & W_{16}^{(2)} \\ \vdots & \vdots & \vdots \\ W_{61}^{(2)} & \cdots & W_{66}^{(2)} \end{bmatrix}_{6 \times 6} \\
 \dots, M_6 &= \begin{bmatrix} W_{11}^{(6)} & \cdots & W_{16}^{(6)} \\ \vdots & \vdots & \vdots \\ W_{61}^{(6)} & \cdots & W_{66}^{(6)} \end{bmatrix}_{6 \times 6}.
 \end{aligned} \tag{1}$$

where $W_{ij}^{(1)}$ ($1 \leq i \leq 6, 1 \leq j \leq 5$) is the weight connecting the input neuron in row j of the input layer with the neuron in row i in sub-layer 1 of the hidden layer; $W_{kl}^{(m)}$ ($1 \leq k \leq 6, 1 \leq l \leq 6, 2 \leq m \leq 6$) is the weight connecting the

neuron in row k of sub-layer $m - 1$ with the neuron in row l in sub-layer m . The input matrix of the connecting weights for the output layer is defined as

$$M_o = \begin{bmatrix} W_{11}^{(o)} & \cdots & W_{16}^{(o)} \\ W_{21}^{(o)} & \cdots & W_{26}^{(o)} \end{bmatrix}_{2 \times 6}, \tag{2}$$

where $W_{ij}^{(o)}$ ($2 \leq i \leq 6, 1 \leq j \leq 6$) is the weight connecting the input neuron in row i of the sub-layer 6 in the hidden layer with the neuron in row j of output layer. In order to optimize the weights listed above, they should be encoded into binary codes. Generally, the weights range in the interval $[-100, 100]$ according to experiments. Here, the precision is set as $\delta = 0.0001$ to meet the precision requirement. According to the definition of precision, we have

$$\delta = \frac{U_{max} - U_{min}}{2^l - 1}, \tag{3}$$

where l is the length of the code for each weight, U_{max} and U_{min} are the maximum and minimum values that the weights can reach, respectively. After the simple transformation, the length of the code is given by

$$l = \left\lceil \log_2 \frac{U_{max} - U_{min}}{2^{l-1}} + 1 \right\rceil, \tag{4}$$

where the operator $[a]$ is to obtain the integer part of a .

The weights are encoded in the following order, $w_{11}^{(1)}, w_{12}^{(1)}, w_{13}^{(1)}, w_{14}^{(1)}, w_{15}^{(1)}, w_{16}^{(1)}, \dots, w_{61}^{(1)}, w_{62}^{(1)}, w_{63}^{(1)}, w_{64}^{(1)}, w_{65}^{(1)}, \dots, w_{61}^{(6)}, w_{62}^{(6)}, w_{63}^{(6)}, w_{64}^{(6)}, w_{65}^{(6)}, w_{66}^{(6)}$. One chain of the above binary code is called a gene, the length of the chromosome is

$$l_{ch} = l(6 \times 5 + 6 \times 6 \times 6 + 2 \times 6). \tag{5}$$

The fitness function is defined as

$$F_{fitness} = \sum_{i=1}^n \left[(y_{ci} - y_{pci})^2 + (y_{wi} - y_{pwi})^2 \right], \tag{6}$$

where y_{ci} and y_{pci} are the cutting speed from experiment and from prediction, respectively; y_{wi} and y_{pwi} are the welding speed from experiment and from prediction, respectively; and n is the number of samples.

When the weight training is completed, the weights will be input into the ANN built in Section 3. Using the ANN acquired, we could predict the unknown parameters (i.e., welding and cutting speed), according to the input parameters (i.e., laser power for cutting and welding, thickness of the steel blanks, pressure of the assistant gas for cutting and welding).

The prediction course goes as the following. Here we assume that the input vector for the network is $(x_1, x_2, x_3, x_4, x_5)^T$, composed of the input parameters,

and the output vector for the sub-layer 1 in hidden layer is $(y_1^{(1)}, y_2^{(1)}, y_3^{(1)}, y_4^{(1)}, y_5^{(1)}, y_6^{(1)})^T$, then

$$\begin{aligned} (y_1^{(1)}, y_2^{(1)}, \dots, y_6^{(1)})^T &= \left(\frac{1}{1+e^{-\lambda_{11}}}, \frac{1}{1+e^{-\lambda_{12}}}, \dots, \frac{1}{1+e^{-\lambda_{16}}} \right)^T, \\ (\lambda_{11}, \lambda_{12}, \dots, \lambda_{16})^T &= M_1(x_1, x_2, \dots, x_5)^T. \end{aligned} \tag{7}$$

As the obtained vector, $(y_1^{(1)}, y_2^{(1)}, y_3^{(1)}, y_4^{(1)}, y_5^{(1)}, y_6^{(1)})^T$, is the input of sub-layer 2 in the hidden layer, the output vector of sub-layer 2, $(y_1^{(2)}, y_2^{(2)}, y_3^{(2)}, y_4^{(2)}, y_5^{(2)}, y_6^{(2)})^T$, can be acquired. That is,

$$\begin{aligned} (y_1^{(2)}, y_2^{(2)}, \dots, y_6^{(2)})^T &= \left(\frac{1}{1+e^{-\lambda_{21}}}, \frac{1}{1+e^{-\lambda_{22}}}, \dots, \frac{1}{1+e^{-\lambda_{26}}} \right)^T, \\ (\lambda_{21}, \lambda_{22}, \dots, \lambda_{26})^T &= M_2(y_1^{(1)}, y_2^{(1)}, \dots, y_6^{(1)})^T. \end{aligned} \tag{8}$$

As the former layer's output is the input of the next layer, the outputs and inputs for every layer can be calculated in the similar way. In the end, we could obtain the output vector of the network

$$\begin{aligned} (y_1, y_2)^T &= \left(\frac{1}{1+e^{-\lambda_1}}, \frac{1}{1+e^{-\lambda_2}} \right)^T, \\ (\lambda_1, \lambda_2)^T &= M_o(y_1^{(6)}, y_2^{(6)}, y_3^{(6)}, y_4^{(6)}, y_5^{(6)}, y_6^{(6)})^T. \end{aligned} \tag{9}$$

where the vector, $(y_1^{(6)}, y_2^{(6)}, y_3^{(6)}, y_4^{(6)}, y_5^{(6)}, y_6^{(6)})^T$, is the output vector of sub-layer 6 in the hidden layer, and the vector $(y_1, y_2)^T$ is the final output vector of the network.

From the deducing course above, it can be found that the computation burden for this approach is heavy. Oftentimes, it needs to adjust the structure of the network such as the number of hidden layer, the number of neurons in each sub-layer in the hidden layer, the initial values of the weights and so on. All these adjustments usually are determined by the problem specified, and depend on the designer's experiences. So there is no universal formula for building the model. However, the proposed approach circumvents the learning rate and stability encountered in BP algorithm as they are replaced by crossover and mutation rates, which are set to be 0.8 and 0.15, respectively, in this case according to experiment.

5 Experiment

5.1 Experiment System

Laser Cutting and Welding Machine. The mechanical structure of laser cutting and welding is shown in Fig.3. The manipulator has two degrees of freedom (DOFs), the translations in the Z - and X -directions. The manipulator translates in the X -direction to cut and weld the blanks. The manipulator's translation in the Z -direction facilitates it to adjust the position of the laser's focus with respect to the steel blanks in the course of cutting and welding.

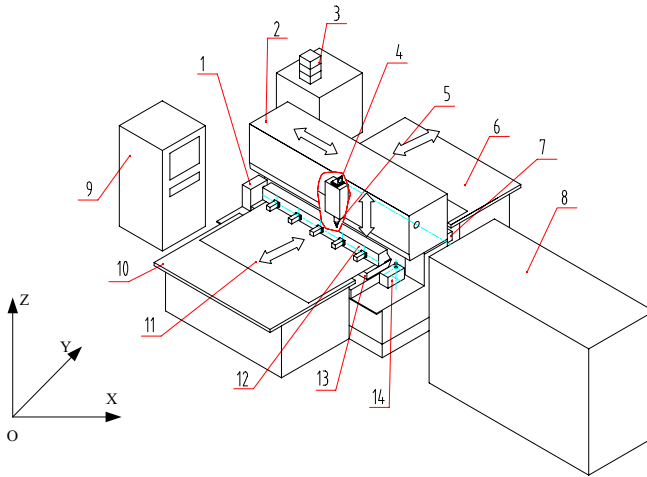


Fig. 3. The configuration of the system for laser cutting and welding. 1 stands for positioning machine, 2 for X-axis for manipulator’s translation, 3 for hydraulic control system, 4 for cutting and welding manipulator, 5 for cutting and welding head, 6 for right bracket for steel blank, 7 for right fixture holder, 8 for laser generator, 9 for computer control system, 10 for left bracket for the steel blanks, 11 for steel blank for cutting and welding. 12 for left fixture group, 13 for left fixture holder, and 14 for head exchanging machine.

The fixtures are divided into two groups: right and left fixture groups, which are employed to fix the blanks on the right and left fixture holders, respectively. The brackets on the right and left sides aid the fixtures to hold the blanks when the length of the blanks surpasses the maximum length of the blanks that the fixture holders can hold.

The manipulator cuts the steel blanks on the right and left fixture groups, separately. On the completion of cutting, the cutting head will be exchanged with the welding head by the head exchanging machine automatically. After the cuts of the two blanks are put together face to face, the manipulator will weld the blanks along the gap to produce TWBs.

Laser Generator. The cross current CO₂ laser generator with the power of 2KW, from PRC Company, U.S.A.

Materials. The IF steel and WLZn coated 08Al steel, whose composition is shown in Tab. 1, are made by Wuhan Steel Company. Other parameters: 0.75~1.2mm in thickness; and 2000mm×3000mm in width and length.

Table 1. Chemical composition of the material for experiment

Type of Steel	C	Si	Mn	P	S	Al
IF	0.003	0.012	0.11	0.006	0.003	/
WLZn	0.006	/	0.28	0.01	0.09	0.06

5.2 Experiment Steps

Train the Network with Experiment Data and Predict the Speed for Cutting and Welding. Firstly, train the network built for the materials listed in Tab.1 with two groups of samples for IF and WLZn, respectively. Each group contains two hundred samples from experiment. Secondly, input the network with the parameters, including thickness of steel blanks, laser power for cutting and welding, pressure of the assistant gas for cutting and welding, listed in Tab.2. Finally, using the trained ANN, we obtain the predicted speed for cutting and welding listed in Tab.2.

Table 2. The predicted parameters with the GA-based ANN

Type of Steel	Thickness (mm)	P_c (KW)	P_w (KW)	Pre_c (Bar)	Pre_w (Bar)	V_c (m/s)	V_w (m/s)
IF	0.75	1.8	1.8	5	1	4.50	3.52
IF	0.8	1.1	1.5	6	1	4.25	3.05
IF	0.85	1.8	1.8	6	1	4.33	2.92
IF	0.90	1.8	1.8	6	1	4.00	2.80
IF	0.95	1.6	1.8	4	1	3.67	2.60
IF	1.00	1.5	1.8	6	2	4.00	2.83
IF	1.1	1.9	1.9	8	1	4.40	3.20
WLZn	0.75	1.7	1.5	6	1	4.6	2.57
WLZn	0.8	1.3	1.9	6	2	3.82	3.32
WLZn	0.85	1.6	1.4	7	1	3.96	2.92
WLZn	0.90	1.5	1.8	6	1	3.65	3.68
WLZn	0.95	1.4	1.0	5	2	2.37	1.86
WLZn	1.00	1.8	1.8	5	1	3.58	3.06

Notes: P_c and P_w stand for the power of laser for cutting and welding, respectively; Pre_c and Pre_w stand for the pressure of the assistant gas for cutting and welding, respectively; V_c and V_w stand for the speed for cutting and welding, respectively.

Cut and Weld the Steel Blanks with the Predicted Parameters. The parameters used for laser cutting and welding are from row 3 in Tab. 2 for IF steel blanks. After the experiment, the Zn coat on the edge of the cut has not been found to be burned but 0.3 mm wide melted zone without being oxygenized. The roughness on the section of the cut is less than 0.248 μm , which ensures that the cuts of the two blanks can be put together face to face with maximum width of the gap less then 0.5 μm and can be welded together without any filling materials. No oxygenized film of the cuts guarantees the good quality of the welding seam. As a result, the cuts are welded together directly without any further processing.

The photos of the seam's structure are shown in Fig.4. From the photos, it can be seen that the structure of the seam is denser than that of its body material. Experiments demonstrate that the seam has good mechanical characteristics for component production in automobile industry.



Fig. 4. The photos for the cross section of the welding seams: (a) IF (X60) and (b) WLZn (X60)

6 Conclusion

In this paper, a GA-based ANN approach is presented for parameter selection, in which, GA is used to train the weights connecting the ANN. The network is trained by the samples from experiments. Experimental results demonstrate that this parameter selection approach combines the merits of GA and ANN, and solves the problem of the local optimum in ANN and the low convergence speed in GA. Given the parameters (i.e., blank's thickness, laser power for cutting and welding, pressure of the assistant gas for cutting welding and cutting), the proposed approach could predict the cutting and welding speed efficiently. Experiment verifies the correctness of the predicted parameters by the fact that the cuts have good quality for welding and the welding seam has the desirable mechanical characteristics. However, the construction of the network is closely related to the specified processing system and the developer's experiences, which manifests the limitation of the proposed approach.

References

1. Irving, B.: Welding Tailored Blanks Is Not an Issue for Automaker. *Welding Journal* **74**(8) (1995) 49-52
2. Westgate, S.A., Kimchi, M.: A New Process for Tailored Blanks Production. *Welding Journal* **74**(5) (1995) 45-53
3. Andrew, D.: Laser-Welded Blanks Gain Ground. *Manufacturing Engineering* **121** (1998) 76-82
4. Irving, B.: What's the Latest News on Laser Beam Cutting and Welding. *Welding Journal* **73** (1994) 31-35
5. Zhao, K.M., Chun, B.K., Lee, J.K.: Finite Element Analysis of Tailor-Welded Blanks. *Finite Elements in Analysis and Design* **37** (2001) 117-130
6. Kinsey, B., Song, N., Cao J.: Analysis of Clamping Mechanism for Tailor Welded Blank Forming. *Journal of Material and Manufacturing* **108** (1999) 1062-1068
7. Schuocker, D., Abel, W.: Material Removal Mechanism of Laser Cutting. *Proc., SPIE* **455** (1984) 880-895
8. Yibas B.S., Davies R., et al.: Investigation into Development of Liquid Layer and Formation of Surface Plasma During CO₂ Laser Cutting Process. *Proc. Inst. Mech. Eng. Part B: Journal of Mechanical Engineering Science* **208** (1994) 275-281
9. Miyamoto, I., Maruo, H.: The Mechanism of Laser Cutting. *Welding in the World* **29** (1991) 283-294

10. Yang, F.Y.: Artificial Neural Network-Based Parameter Selection and Processing Quality Prediction for Laser Processing. Dissertations for Master's Degree (1997), Huazhong University of Science and Technology, Wuhan
11. Baldi, P.: Gradient Descent Learning Algorithm Overview: a General Dynamical Systems Perspective. *IEEE Transactions on Neural Networks* **6** (1995) 182-195
12. Edgar, T.F., Himmelbau, D.M.: Optimization of Chemical Process. McGrawHill Book Company, London (1998) 123-151 and 190-239

Development of Artificial Neural Networks-Based In-Process Flash Monitoring (ANN-IPFM) System in Injection Molding

Joseph Chen¹, Mandara Savage², and Jie (James) Zhu³

¹Department of Agricultural and Biosystems Engineering,
Iowa State University, Ames, IA

²Department of Technology, Southern Illinois University Carbondale,
Carbondale, Illinois 62901 USA

³Research & Development
Natural Polymer International Corp. Plano, TX
msavage@engr.siu.edu

Abstract. This paper describes the development of an artificial neural networks-based in-process flash monitoring system (ANN-IPFM) in the injection molding process. This proposed system integrates two sub-systems. One is the vibration monitoring sub-system that utilizes an accelerometer sensor to collect and process vibration signals during the injection molding process. The other, a threshold prediction sub-system, predicts a control threshold based on the process parameter settings, thus allowing the system to adapt to changes in these settings. The integrated system compares the monitored vibration signals with the control threshold to predict whether or not flash will occur. The performance of the ANN-IPFM system was determined by using varying ratios of polystyrene (PS) and low-density polyethylene (LDPE) in the injection molding process, and comparing the number of actual occurrences of flash with the number of occurrences predicted by the system. After a 180 trials, results demonstrated that the ANN-IPFM system could predict flash with 92.7% accuracy.

1 Introduction

The plastic injection molding process is the most commonly used manufacturing process in the plastics industry due to its capability for mass production at a relatively low cost. In this relatively simple process, plastic is melted and then forced into the cavity of a closed mold under high pressure. After sufficient cooling time, the molten material solidifies into the desired shape, the mold is opened, and the part is removed. Then next injection cycle begins [1].

The demand for injection-molded products has grown tremendously in recent years. More and more plastics have been consumed and discarded, which has resulted in a shortage of petroleum, as well as waste disposal and pollution problems [2, 3]. Therefore, recycled plastics from defective parts, trimmings, and other manufacturing

scraps have been widely used in the injection molding process, ranging in use from very small-scale reprocessing in small companies to huge programs that utilize several tons of recycled materials per year [4].

Recycling plastics can often result in aggregate material with differing thermal and mechanical properties that can vary significantly from batch to batch. When mixed material is processed by injection molding, careful attention must be given to process parameters, regrind composition, and moisture content.

This study describes “mixed material” as a composition of plastic aggregate containing two different materials that differ in thermal and mechanical properties. Product defects may result from using mixed materials due to their different melting temperatures, resulting in inconsistent flow rates into the mold [5]. One of the more common defects is flash, which occurs when material flows outside of the edge of the mold cavity.

Machinists in an injection molding operation can control flash by setting proper process parameters. Some of those process parameters include proper material drying, mold clamp pressure, cylinder temperature, holding and injection pressure, and injection speed. However, even if parameters are controlled appropriately, flash may still occur. Therefore, a system that could monitor the occurrence of flash online would greatly improve process efficiency [6].

An in-process flash prediction system that can predict flash occurrence in real-time and online with a high degree of accuracy could prevent flash from occurring between routine inspection times. In recent years, considerable research has been conducted on in-process defect prediction systems in the injection molding process. For example, a few systems have been developed to monitor part weight or dimensions [7-9]. In these systems, when the part weight or dimensions are out of tolerance, the system notifies the operator that a defect has occurred and requires inspection immediately. Lee and Young [10] developed an on-line part shrinkage monitoring system to predict the shrinkage range of crystalline polymers and thus identify defective parts. Other systems have been developed [11-13] to allow effective setting and resetting of processing parameters based on the various part defects, such as flash, short shot, weld line, and cracking.

These systems were shown to work successfully in establishing process parameters and minimizing defects. Once these parameters are set to an optimal value, flash or other defects rarely happen if virgin or a homogeneous material is used. However, when recycled mixed materials are used in injection molding, flash often occurs, even at the optimum processing parameter settings. This research focuses on developing an in-process flash monitoring system (IPFMS) operating within optimum injection molding processing parameters.

This system consists of two major components. The first, the sensing mechanism, detects key characteristics of the injection molding process. The second, the decision-making mechanism, analyzes the sensor signal and performs monitoring functions.

To develop a real-time decision-making mechanism, this study employed artificial neural networks (ANN). These systems can model arbitrary input data by adjusting

their internal network connections. These adjustments systematically minimize the margin of error between the network output and the desired response. This process of supervised learning reduces the network error using a training set of matching input-output vectors [14].

Artificial neural networks have been widely used in plastics engineering to monitor part quality [7, 15] and shrinkage (10), as well as to control processing parameters [11, 12, 16]. For example, Choi, Lee, Chang, and Kim [11] used artificial neural networks to optimize processing parameters and predict injection molding part defects.

In summary, this research serves to develop an ANN-based in-process flash monitoring (ANN-IPFM) system in the injection molding process. The system has two capabilities: 1. to collect and analyze vibration signatures generated during the injection molding process; and 2. to determine and alert an operator when flash has occurred.

2 Structure of the ANN-IPFM System

The structure of the ANN-IPFM system (Figure 1) integrates two sub-systems, vibration monitoring and threshold prediction.

1. The vibration monitoring sub-system detects the occurrence of flash caused by mixed materials during the injection molding cycle. This system utilizes an accelerometer sensor to monitor the difference in the vibration signals between injection-molded specimens with flash and without flash during the last period of the injection filling stage. Using statistical analysis, a process characteristic indicator, γ_j , was calculated as the subsystem parameter for determining whether flash has occurred.
2. The threshold prediction sub-system predicts the control threshold value, based on training data, according to the current processing parameter settings. The significant processing parameters (injection speed, holding pressure, and melt temperature) are the inputs to this sub-system. The ANN training process incorporates results based on a combination of parameter settings within the optimum setting range.

The flash control threshold values were determined through a statistical process control (SPC) methodology. A component of the X-bar control charting procedure was followed when calculating the cutoff values. The upper and lower control limits were calculated, where the upper control limit represented the cutoff values for the flash control threshold. These values were generated from the data collected during cycles when no flash occurred (i.e., control material). In testing the ANN model, the output of the threshold prediction sub-system is the proposed flash control threshold value (θ_i). This value represents the maximum for which flash does not occur.

3 Methodology

This section describes the experimental setup, vibration monitoring sub-system development, and the threshold prediction sub-system development. The experimental setup consists of a *BOY 22M* injection molding machine outfitted with a *Procan MD* microprocessor control (Pennsylvania, USA); a Windows-based personal computer with *DaqView 8.0* from IOtech, Inc. (Ohio, USA) installed; a PCB Piezotronics model 356B08 3-axis accelerometer sensor (New York, USA); an IOtech model *DBK11A* screw terminal expansion card; and an IOtech *DaqBook 100* data acquisition system (Ohio, USA). The accelerometer sensor was installed on the top-center of the stationary mold-half with its Z- axis parallel to the travel of the movable platen. The X-axis was perpendicular-vertical and the Y-axis was perpendicular-horizontal to platen travel. A PCB model 480E09 (New York, USA) signal conditioner was used to power the accelerometer, amplify the signal, and filter noise before the signal is passed to the data acquisition system.

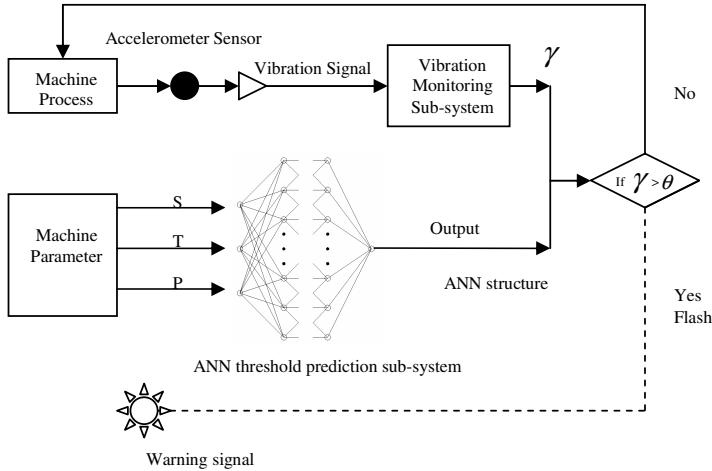


Fig. 1. The structure of the ANN-IPMFM system (*S* denotes injection speed, *T* denotes melting temperature, *P* denotes holding pressure.)

An example of the vibration signal collected during the injection molding cycle is shown in Figure 2. The first main signal peak is generated when the mold closes; the second main signal peak shows the beginning of the plastic injection filling stage.

3.1 Vibration Monitoring Sub-system Development

The vibration monitoring sub-system was developed by devising an experimental design to collect and analyze data, generate the process characteristic indicator for flash determination, and build the decision-making mechanism.

3.1.1 Experimental Design

The goal of the experimental design was to capture the difference in the vibration signals between the specimens with and without flash. The polymer materials used in this research were Polystyrene (PS) 147F manufactured by INEOS Styrenics and low-density polyethylene (LDPE) 2072 manufactured by the Huntsman Corporation.

PS was considered the control material, while a PS and LDPE mix was the treatment material.

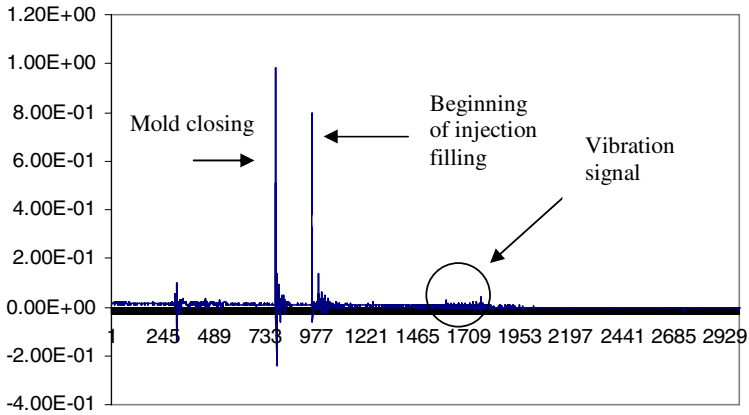


Fig. 2. An example of an injection molding processing vibration signal for a good specimen

The treatment material was a mixed material consisting of artificially mixed PS and LDPE (90% PS + 10% LDPE). This aggregate composition simulated aggregate compositions of virgin and regrind material often used in the injection molding process.

The specimen molded was a tensile bar measuring 4.95 x 0.5 inches. The specimen had a weight of 0.35 ounces and a volume of 0.62 in³. This research utilized processing parameter settings suggested by the material manufacturers to establish optimal processing ranges for the injection molding process in this study.

3.1.2 Processing Characteristic Indicator

Flash occurred in conjunction with stronger vibration signals during the last period of the filling stage (0.4 seconds circled in Figure 2). An approach for data treatment was then developed to compare the signals in the last period of the injection filling stage. The researchers utilized the following procedure to calculate a processing characteristic indicator (γ_j) to compare the signals in the last period of the filling stage:

Step 1: Starting from the point when the mold initiates closing, collect 3000 Z-axis vibration data points. This data collection covers vibration signals from the moment the machine initiates mold closing until the end of the injection filling stage (see Figures 2).

- Step 2: Locate the second peak of the Z-axis vibration signal, which represents the beginning of the injection filling stage.
- Step 3: Starting from the second peak point, collect 850 (1.7 seconds) Z-axis data points. ($Z_{ij}, i=1, 2, \dots, 850, j = 1, 2, \dots, 15$, where i denotes the data point and j denotes the specimen number used in this research.)
- Step 4: Find the maximum absolute peak value $Z_{j \max}$ within the last 200 data points:

$$Z_{j \max} = \text{Max}|Z_{ij}| = \text{Max}\{|Z_{651j}|, |Z_{652j}|, \dots, |Z_{850j}|\} \tag{1}$$

- Step 5: Calculate the average absolute peak value of the 200 (0.4 seconds) points:

$$\bar{Z}_j = \frac{\sum_{i=651}^{850} |Z_{ij}|}{200} \tag{2}$$

- Step 6: Calculate the ratio of the maximum peak value over the average peak value. This is called the max-avg. ratio γ_j , which has the following formula:

$$\gamma_j = \frac{Z_{j \max}}{\bar{Z}_j}, \text{ where } j \text{ is number of experiments} \tag{3}$$

- Step 7: Save the max-avg. ratio as γ_j .
- Step 8: Calculate the average of two consecutive max-avg. ratio data to generate the sub-group statistic $\bar{\gamma}_j$ from γ_j (sub-group size = 2):

$$\bar{\gamma}_j = \frac{\gamma_j + \gamma_{j+1}}{2}, \text{ where } j = 1, 2 \dots 15 \tag{4}$$

$\bar{\gamma}_j$ would then be considered as a processing characteristic indicator for monitoring the injection molding process.

3.1.3 Experimental Setup and Procedures

The researchers collected and recorded vibration data from fifteen consecutively molded specimens. Researchers reviewed each specimen visually for the presence of flash.

The injection filling time was set at 1.7 seconds. The data collection time was set to 6 seconds at a scanning frequency of 500 Hz, which was enough time to record activity during the mold closing and after the filling stage. Once the pilot data collection and analysis was complete, the monitoring sub-system processing characteristic indicator ($\bar{\gamma}_j$) was determined.

3.2 Threshold Prediction Sub-system Development

Backpropagation (BP), one of the most widely used and successfully applied supervised learning methods in many different neural network applications, was

applied in this study [17]. Backpropagation networks are usually layered, with each layer fully connected to surrounding layers by weighted connections.

The following four steps were used to develop the threshold prediction sub-system:

Step 1. Construct an experimental design to collect data for ANN training

Processing parameters of injection speed (S), melt temperature (T), and holding pressure (P) were recognized as significantly influencing the occurrence of flash and were used as inputs [5].

Table 1 lists the processing parameters of the experimental design at different treatment combinations. For the control (i.e., polystyrene) and treatment (i.e., 90% polystyrene + 10% low density polyethylene) material each experimental condition had a replication of fifteen consecutively produced specimens.

For each run, a flash control threshold was calculated based on the average-maximum ratio (γ_j). The flash control threshold (θ_i), also called the subgroup statistic, was calculated with size $n = 2$. The formula is defined as follows:

$$\text{Subgroup average: } \bar{\gamma}_j = \frac{\gamma_j + \gamma_{j+1}}{2}, \quad j = 1, 2, \dots, 15 \quad (5)$$

$$\text{Subgroup range: } R_j = |\gamma_{j+1} - \gamma_j|, \quad j = 1, 2, \dots, 15 \quad (6)$$

The upper and lower control limit can be calculated as:

$$UCL_\gamma = \bar{\gamma} + A_2 \bar{R}, \quad (7)$$

$$LCL_\gamma = \bar{\gamma} - A_2 \bar{R}, \quad (8)$$

where $\bar{\gamma}$ is the average of $\bar{\gamma}_j$, $j = 1, 2, \dots, 15$;

\bar{R} is the average of R_j , $j = 1, 2, \dots, 14$;

$A_2 = 1.88$ is the control chart coefficient if subgroup size is 2 [18].

The calculated upper control limit, UCL_γ , also called flash control threshold (θ_i), is used as the target output for the ANN training

Step 2. Assignment of input and output variable to form data sets

The input and output variables were next assigned to construct the threshold prediction sub-system. There were three input factors in this sub-system, which were injection speed (S), melt temperature (T), and holding pressure (P). The output factor was the flash control threshold (θ_i). The twelve runs of data collected for training were evaluated and broken into data sets. Each data set was expressed as:

$$[S_i, T_i, P_i; \theta_i], \quad i = 1 \text{ to } 12. \quad (9)$$

Step 3. Scale and prepare the data set before ANN training

After data scaling, the data set was expressed as:

$$\left[S'_i, P'_i, T'_i, \theta'_i \right], \quad i = 1 \text{ to } 12. \quad (10)$$

Step 4. Determine the optimal ANN model

The best configuration containing two hidden layers with seven hidden neurons in each layer was selected as the final ANN-threshold prediction sub-system model. Based on this, a 3-7-7-1 ANN-threshold prediction sub-system was developed (three input layers, two levels of seven hidden layers, and one output layer).

Table 1. Training data for the ANN-IPMFM sub-system

Run Number	Injection Speed (%)	Melt Temperature (°F)	Hold Pressure (psi)	Flash Control Threshold (θ_i)
1	95	450	1100	5.69
2	95	450	900	5.81
3	95	430	1100	5.57
4	95	430	900	5.63
5	90	450	1100	5.60
6	90	450	900	5.75
7	90	430	1100	5.32
8	90	430	900	5.48
9	85	450	1100	5.50
10	85	450	900	5.56
11	85	430	1100	5.24
12	85	430	900	5.28

4 ANN-IPMFM System Evaluation and Results

The ANN-IPMFM system evaluation was based on the experimental test conditions listed in Table 2. The three process parameters were randomized within 12 testing runs. Each test run combination had fifteen replications. The system decision-making mechanism was applied to determine whether or not flash occurred.

Test runs 1 through 6 were conducted using the control material. Each test run had 15 specimens, resulting in a total evaluation of 90 test specimens. For example, in test run 1, the injection speed (89%), the melt temperature (445° F), and the holding pressure (1020 psi), were employed as input to the ANN-IPMFM threshold prediction sub-system. The material condition was the control material, and the output (predicted flash control threshold) was calculated based on the threshold prediction sub-system as $\theta_i = 5.73$. The flash occurrence was then determined by comparing $\overline{\gamma_j}$ with θ_i .

As indicated in Table 2, when the control material was used, no flash was found among any of the 90 products. Six specimens were found to have higher $\overline{\gamma}_j$ than θ_i , indicating that flash had been predicted, but had not occurred. Test runs 7 through 12 were conducted using the treatment material; each test run had 15 specimens, for a total evaluation of 90 test specimens. All 90 specimens were identified as having flash. Seven products had lower $\overline{\gamma}_j$ than the flash threshold value θ_i , which indicated that no flash had occurred.

There were a total of 180 testing samples using these two material conditions. The accuracy of the ANN-IPFM system was calculated using the total number of errors made by the system divided by the total number of testing samples. As indicated by the results of this calculation, the ANN-IPFM system efficiently predicted flash with 92.7% accuracy.

Table 2. The testing results for the ANN-IPFM system

Test Run	S (%)	T (°F)	P (psi)	System Monitoring Result			Actual Result	
				Flash Thres hold θ_i	# of flash	# of Non-flash	# of flash	# of Non-flash
1	94	445	1020	5.73	1	14	0	15
2	94	445	920	5.55	2	13	0	15
3	94	435	1020	5.75	0	15	0	15
4	94	435	920	5.65	1	14	0	15
5	89	445	1020	5.61	0	15	0	15
6	89	445	920	5.72	2	13	0	15
7	94	445	1020	5.38	13	2	15	0
8	94	445	920	5.57	14	1	15	0
9	89	445	1020	5.58	14	1	15	0
10	89	445	920	5.60	15	0	15	0
11	84	435	1020	5.39	13	2	15	0
12	84	435	920	5.45	14	1	15	0
Total number of test runs = 180 FNN-IPFM system accuracy = 92.7%								

5 Conclusions

A new approach for a neural networks-based in-process flash monitoring (ANN-IPFM) system in the injection molding process was developed and evaluated in this study. The completed system was shown to be able to effectively monitor flash during the

injection molding operation. The main conclusions drawn from this research are summarized as follows:

1. A threshold prediction sub-system has been integrated with the vibration monitoring sub-system within optimum ranges of processing parameter settings.
2. The ANN approach used in the threshold prediction sub-system successfully predicted the flash control threshold under varying processing parameter settings.
3. The ANN-IPMFM system successfully predicted flash with 92.7% accuracy.

This research was limited to only two types of polymer (PS and LDPE) and one type of injection mold. Enlarging this system to include more materials and various types of workpiece molds could provide greater applicability to future automated machining processes and implementation in the plastics industry.

References

1. Strong, A.B.: *Plastics Materials and Processing*. Prentice-Hall, New Jersey (2000)
2. John, J., Tang, J., Bhattacharya, M.: Processing of Biodegradable Blends of Wheat Gluten and Modified Polycaprolactone. *Polymer* **39** (1998) 2883-2895
3. Zhong, Z., Sun, X.S.: Properties of Soy Protein Isolate/Polycaprolactone Blends Compatibilized by Methylene Diphenyl Diisocyanate. *Polymer* **42** (2001) 6961-6969
4. Richardson, T.L., Lokensgard, E.: *Current Status of the Plastics Industry. Industrial Plastics: Theory and Applications*, Delmar Learning, New York, (2003) 19-30
5. Osswald, T.A., Turng, L.S., Gramann, P.L.: *Injection Molding Handbook*. Hanser Publications Inc., Ohio (2001)
6. Bryce, D.M.: *Plastic Injection Molding Manufacturing Process Fundamentals*. Society of Manufacturing Engineers, Michigan (1996)
7. Smith, A.E.: Monitoring Product Quality with Backpropagation: a Thermoplastic Injection Molding Case Study. *Intl J Adv Manuf Technol* **8** (1993) 252-257
8. Xia, Z., Mallick, P.: Control of Dimensional Variability in Injection Molded Plastic Parts. In *SPE ANTEC Technical Papers* (1997)
9. Rewal, N., Toncich, D., Friedl, C.: Predicting Part Quality in Injection Molding Using Artificial Neural Networks. *J Inject Molding Technol* (1998)
10. Lee, S.C., Young, J.R.: Shrinkage Analysis of Molded Parts Using Neural Networks, *J Reinf Plast and Comp*, **18** (1999)
11. Choi, G.H., Lee, K.D., Chang, N., Kim, S.G.: Optimization of Process Parameters of Injection Molding with Neural Network Application in a Process Simulation Environment. *Annals of the CIRP* **43** (1994)
12. Petrova, T., Kazmer, D.: Hybrid Neural Models for Pressure Control in Injection Molding. *Adv in Polymer Technol*, **18** (1999)
13. He, W., Zhang, Y.F., Lee, K.S.: Development of a Fuzzy-Neural System for Parameter Resetting of Injection Molding. *J Manuf Sci and Eng*, **123** (2001) 110-118
14. Garvey, E.B.: *On-Line Quality Control of Injection Molding Using Neural Networks*. Masters Thesis, Royal Melbourne Institute of Technology (1997)

15. Woll, S.L.B., Cooper, D.J.: Online Pattern-Based Part Quality Monitoring of the Injection Molding Process. *Polymer Eng and Sci* **36** (1996)
16. Zhao, C., Furong, G.: Melt Temperature Profile Monitoring for Thermoplastic Injection Molding. *Polymer Eng and Sci* **39** (1999)
17. Lee, S.L., Chen, J.C.: On-Line Surface Roughness Recognition System Using Artificial Neural Networks System in Turning Operations. *Intl J Adv Manuf Technol*, **22** (2003) 498-509
18. Besterfield, D.H.: *Quality Control*. Prentice Hall, New Jersey (2004)

Reduce Feature Based NN for Transient Stability Analysis of Large-Scale Power Systems

Youping Fan, Min Xiong, Lu Liu, Jiguan Men, Cheng Tan, and Yunping Chen

Faculty of Electrical Engineering, Wuhan University
Hubei, Wuhan 430072, China
Fyopingnxinrong@yahoo.com.cn

Abstract. Aiming at the existence of relativity between repeat or similar samples and character parameters during diagnosis of character data, this paper presents an effective data analysis approach for character data compression from bi-direction, which can reduce the burden of learning machine without losing the connotative character knowledge of character data. At the first step of the algorithm, basing on the theory of component analysis, the paper adopt a principal component analysis approach to reduce the dimension of data horizontally, then after comparison of existing clustering algorithms, put forward an immune clustering algorithm based on similarity measurement of principle component core for vertical reduction by using related mechanism of clone selection as well as immune network self-stabilization in organism natural immune system for reference. Finally, to analyze machine behavior quantitatively, a pattern discrimination model based on a cerebellar model articulation controller neural network (NN) was developed. Simulation experiments proved the effectiveness of this algorithm.

1 Introduction

To monitor complex large-scale system effectively, we often need mass of process data in different working conditions and various fault states. This adds difficulty to utilize them efficiently while supplying available message. Now, more and more researchers are paying attention to this problem in fault diagnosis area based on knowledge. Clustering analysis is a basic method for data mining. The common clustering algorithms are as follows [1]: 1) K-means clustering algorithm (CM); 2) Fuzzy C- means clustering algorithm (FCM); 3) Clustering algorithm based on adaptive neural network, such as fuzzy adaptive resonance network (Fuzzy ART); 4) Optimal statistic analysis based on genetic algorithm (GAC). In the research, according to above theory, the redundancy data were managed to eliminate from two directions based on two possibilities of information redundancy.

The layout of the paper is as follows: Section 2 gives the model of multi-symptom characteristic knowledge. Dimension reduction of horizontal characteristic parameter based on principal component analysis will be presented in Section 3. Immune clustering-vertical reduce based on similarity measurement of principle component core will be put forward in Section 4. Finally, some simulations and experimental results are drawn in Section 5.

2 Model of Multi-symptom Domain’s Comprehensive Knowledge

The main problems for current intelligent fault diagnosis are as follows: The diagnosis system based on knowledge can’t solve the bottleneck of information acquisition. According to the above contradiction, the paper presents the concept of Comprehensive Feature Knowledge in Multi-Symptom Domains called M-K. The essential is to improve diagnosis ability through reasonable formation and diagnosis information and knowledge digging.

Definition 1. Multi-symptom comprehensive feature knowledge (M-K) is a set of three elements described as $\langle ID, S, Interact \rangle$.

Where, *ID* represents the symbol of current *M-K*. In a complicated system, the hierarchical knowledge structure is composed of many M-K. The element $S=\{S_1, S_2 \cdots S_i \cdots S_m\}$ indicates that the characteristic knowledge contains m entity of profiles all of which formed a solid knowledge structure. Each profile can be expressed by various knowledge expressions such as generation formula and semantic network. The operator $Interact(S_i, S_j)$ denotes the mutual excitation between the profiles of S_i and S_j , the result is that the content of message in knowledge increases explosively, and it brings amalgamation and increases of knowledge from different profiles and eliminates the contradiction between knowledge.

3 Dimension Reduction Based on PCA

Suppose that $M-K_i$ is corresponded to some subsystem to be diagnosed. Matrix X is the acquired symptom set of the j_{th} profile to describe this diagnosis subsystem. The row vector of X is the *m*-dimensional sample expressed as $[x_{i1}, x_{i2}, \cdots, x_{im}]$, while the list vector is the *n* group of detecting data. Here the bi-directional condensation of the sample data means that the matrix X is compressed from the direction of both row and list without loss of the main feature information contained.

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix}_{n \times m}, \quad \hat{X} = \begin{bmatrix} \hat{x}_{11} & \hat{x}_{12} & \cdots & \hat{x}_{1m'} \\ \hat{x}_{21} & \hat{x}_{22} & \cdots & \hat{x}_{2m'} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{x}_{n1} & \hat{x}_{n2} & \cdots & \hat{x}_{n'm'} \end{bmatrix}_{n' \times m'} \tag{1}$$

where $n' < n$, $m' < m$, the rate of condensation can be defined as: $\rho = nm - n'm' / nm$. The general strategies used in the bi-directional feature condensation are given here:

- 1) With the sequence from row to list, that is, we compress the number (n) of sample first, and then do it with m-dimension parameters of each sample.
- 2) With the sequence from list to row, that is the opposite of 1).

3) Compress by turns, the number (n'') of one part of sample is compressed first, then goes with the dimension (m'') of a part of sample parameters, then take turns until get the demand ratio of condensation or the algorithm converged.

In fact, the former method of 1) and 2) are more operative. The condensation strategy of bi-directional feature data in this paper is that, in the process of cutting down the dimension of sample parameter, a method of condensation was adopted based on principal component analysis (PCA) [1].

4 Clustering-Vertical Reduce Based on Similarity Measurement

A. Definition and Notion of Algorithm as well as Parameter Illustration

Definition 2. Immune shape space S is a L-dimensional matrix, which defines all possible immune operation in immune system between antibody and antigen (**Ab-Ag**) or between antibody(**Ab-Ab**). Each coordinate component in the space denotes the physical-chemistry measures property about the immune operation to shape variation of immune molecule.

The L-dimensional vector S_i is used to denote each element in shape space S , $s_i = [s_i^1, s_i^2, \dots, s_i^L]$, $s_i \in S, i = 1, 2, \dots, S_n, S_n$ is the number of samples in space S .

Definition 3. The affinity f between the immune molecule is the interaction among the sample vector in shape space S . There are two cases, the affinity of the **Ab-Ag** interaction is defined as the matching between antigenic epitope and antibody contraposition in shape space, the affinity of the **Ab-Ab** interaction is the inhibiting effect of immune network adjustment, which is expressed by comparability measure.

$Ab = \{Ab_1, Ab_2, \dots, Ab_N\}$ is a set of antibody, $Ab \in S^{N \times L}$, N is the number of antibody in the set; $Ag = \{Ag_1, Ag_2, \dots, Ag_M\}$ is the set of antigen, $Ag \in S^{M \times L}$, M is the number of antigen in space S ; $Ab_{(m)} \in S^{m \times L}$ denotes the set of memory antibody, $m \leq N$, m is the number of memory antibody; $f_{i,j}$ is the affinity between antibody Ab_i and antigen Ag_j , $f_{i,j} \propto d_{i,j}$, $d_{i,j}$ is the comparability distance between two points in space S ; $cAb = \{cAb_1, cAb_2, \dots, cAb_{N_c}\}$ represents the N_c entities of new antibody for clone selecting, $cAb \in S^{N_c \times L}$; $amcAb = \{amcAb_1, amcAb_2, \dots, amcAb_{N_c}\}$ indicates the clone antibody after the mutation of super-gene, maturation of affinity; M_j is the memory clone set of relative antigen Ag_j ; M_j^* is the memory clone set after inhibiting clone; ρ is the purified limit value of effective antigen; σ_d is the supposed natural death limit value, which can eliminate the antibody with lower affinity to antigen so as to improve the adaptability of whole selected clony. It can also control the level of latest network antibody; σ_s is the immune inhibiting limit, it

can dominate the number of latest antibody and reveal the capability of controlling network flexibility. The bigger the value is, the looser the memory antibody is. So the antibody is of more generality and the number of clustering types is less. Meanwhile, the smaller it is, the more compact the memory antibody is. The antibody is of more difference and the number of the clustering type is a bit more, finally the misclassification will occur if the number is overhigh.

B. Modified Immune Clustering Algorithm

The author makes great modification to the algorithm [2,3] from the project application point of view, presents a more valuable definition. The modification of the algorithm is as follows:

1) Increase operation of normalization of antigen data at beginning of algorithm.

The normalization of antigen data is to transform all original antigens to the area between lb and ub . We can eliminate the influence of dimension difference of the original data and on the other hand control the selection area of important parameter between 0 and 1 in the following algorithm, which brings the convenience of parameter selecting.

The value of normalization for any coordinate value s_i^j of any vector s_i in S can be expressed as: $\bar{s}_i^j = (s_i^j - \min_{n=1}^{S_n} s_n^j) \cdot (ub - lb) / (\max_{n=1}^{S_n} s_n^j - \min_{n=1}^{S_n} s_n^j) + lb$, $0 < lb < ub \leq 1$, in general, we take the value as $lb = 0.1$, $ub = 1.0$.

2) Increase the cleaning operation of effective antigen.

The operation is to eliminate accumulative duplicate or similar sample of reference antigen by computing the Euclid distance between samples. If $d_{i,j} < \rho$, then wipe off the sample labelled with i , which accelerates the convergence speed of the algorithm, ρ is often taken as 0.01.

3) Definition of appetency based on principal component core.

Here the author adopts an affinity definition based on the similarity measurement of principal component core, which take the reciprocal of the Euclid distance in various principal component spaces as affinity.

In mode identifying, the similarity measurement is a definition of certain distance in space. For a given set X of input sample, the similarity measurement $\delta(x_i, x_j)$ between any two samples x_i and x_j should satisfy the following requirements: the similarity measurement is nonnegative, that is $\delta(x_i, x_j) \geq 0$; the similarity measurement of the sample itself should be the largest; the similarity measurement must be symmetrical, namely $\delta(x_i, x_j) = \delta(x_j, x_i)$; in the case of compact pattern class, it is a monotone function of the distance between two points.

Definition 4. The degree of similarity between samples in principal component subspace can be measured by the Euclid distance between principal components directly, that is: $\Delta(s_i, s_j) = \sqrt{[(s_i - s_j) - U_j U_j^T (s_i - s_j)]^T [(s_i - s_j) - U_j U_j^T (s_i - s_j)]}$.

Definition 5. For the data transformed by principal component space, we can simplify the above expression using transformed Euclid distance similarly. Thus define the affinity between immune molecules in shape space as follows: $f_{i,j} = 1/\|Ab'_i - Ag'_j\|$. Where Ab'_i and Ag'_j should be the data in principal component subspace. We adopt the affinity definition in the form of principal components kernel all through the algorithm.

4) Select original antibody set utilizing experienced knowledge.

A key merit of clone selecting algorithm is that we need not to know the experienced knowledge of sample data distribution. But if we know a certain possible distribution to sample data in advance, we can select a representative sample as the original antibody set, which will improve the convergence and efficiency of the condensation algorithm.

5) The convergence condition of the algorithm.

We will take the following strategy to determine the end of iterative algorithm, set the iterative time g (it is of great reduction compared to the iterative time of optimal searching in inheritance algorithm) in ahead, provide the expression for condensation ratio of algorithm: $r_c = [Ab]/[Ag]$, the algorithm ends if $r_c \geq \delta_r$. $[Ab]$ is the number of antibody after condensation, $[Ag]$ is the number of antigen mapped from problem set. δ_r is the required ratio of condensation. The specified steps of the solution are as follows [1].

5 Simulation and Conclusion

The system is realized by adopting VC++6.0 based on Windows2000. Next it will be confirmed by analyzing a certain computing case in a simulating experiment. **Training samples** and **Measuring samples** are as follows[5].

A. Feature Reduction Using PCA and Immune Clustering

Though the number of data has been reduced greatly after the condensation of PCA, there are still quite a few redundancies between the samples. If the data are sent to the classification machine directly, the training number of samples will be very great. As the sort of the sample data is known, we can use the algorithm of immune clustering in each type of data directly to compress the data of sample.

After the process of the above data condensation from bi-direction, the last remaining character sample is a 64×7 dimensional data matrix. The condensation ratio of bi-direction can be expressed as: $\rho = 89.2\%$.

B. Monitoring Complex System Using Neural Network

Automated fault detection and diagnostic systems based on neural networks have been implemented for complex large-scale system in this paper. Briefly speaking, the cerebellar model articulation controller (CMAC) can be described as a computing device that accepts an input vector $S = (S_1, S_2, \dots, S_n)$ and produces an output vector $P = F(S)$. To compute the output vector P for a given input state S , pair mapping is performed, namely: $f: S \rightarrow A$; $g: A \rightarrow P$. A procedure for entering a function in CMAC is given in Reference [4,5].

Domain value selection: The net's stable index distributing figure responding for the training samples is combined with the stable pattern of training samples to determine the field value of unstable field, fuzzy field, stable field separately and then depend on them to determine the stable pattern of measuring samples. It's seen that unstable field, fuzzy field and stable field can be defined as $1 \leq y_i < 1.25$; $1.25 \leq y_i < 1.7$; $1.7 \leq y_i \leq 2.0$.

Measuring result: The net makes good effect on predicted contingency set under the known working state and possesses a certain deductive ability for predicted contingency set under the unknown working state, which shows that the function of the net depends on the covering range of the training samples on a large degree.

6 Conclusion

Aiming at the situation that there may exist two kind of information redundancy in constructing the multi-symptom domains character knowledge, namely, the relativity between the repeated or similar sample and character data, the author proposed an effective bi-directional condensation method of character data and implemented it. Compare the classification result of sample data pre-condensation to that after condensation, we will find that the sample data of the later keep the structure character of original data well, it eliminates lots of redundancy, has the ability of classification similarly. Finally, to analyze machine behavior quantitatively, a pattern discrimination model based on a cerebellar model articulation controller neural network was developed. Experiments proved the effectiveness of this algorithm.

Acknowledgements

This work is partially supported by the Provincial Natural Science Foundation of Hubei Grant #2005ABA289 to Y.P. Fan and the National Natural Science Foundation of China Grant #50477018 to Y.P. Chen.

References

1. Fan, Y. P., Chen, Y. P.: Feature Data Enriching Based on Optimizational Immune Clustering Approach. *Information and Control* **34(2)** (2005) 181-187
2. De Castro, L.N., Von Zuben, F.J.: The Clonal Selection Algorithm with Engineering Applications. In: Whitley D., Goldber D., Cantu-Paz (eds.). *Proceedings of the Genetic and Evolutionary Computation Conference*. Morgan Kaufmann Publishers (2000) 36-43
3. Hunt, J.E.: Learning Using an Artificial Immune System. *Journal of Network and Computer Applications* **19(2)** (1996) 189-212
4. Lee, J., Bruce, M.K.: Analysis of Machine Degradation Using a Neural Network Based Pattern Discrimination Model. *Journal of Manufacturing System* **12(5)** (1996) 379-388
5. Fan, Y.P., Chen, Y.P.: Contingency Screening of Power System Based on Rough Sets and Fuzzy ARTMA. *LNCS* **3498** (2005) 654-661

Semi-active Control of Eccentric Structures Based on Neural Networks

Hongnan Li, Linsheng Huo, and Tinghua Yi

State Key Laboratory of Coastal and Offshore Engineering,
Dalian University of Technology, Ganjingzi District, LingGong Road 2,
Dalian116023, P.R. China
hnli@dlut.edu.cn, {lshuo75, yitinghua}@163.com

Abstract. In this paper, the control approach to irregular structures excited by multi-dimensional ground motions is presented by using semi-active tuned liquid column damper (TLCD). A back propagation Artificial Neural Network (ANN) is used to predict the responses of structure due to two-dimensional seismic inputs. The semi-active control strategy is established and implemented based on ANN. The numerical examples have shown that it is an effective method presented for controlling the translational and rotational responses of irregular structures.

1 Introduction

The dynamic response of tall buildings due to earthquake is very important to civil engineers. These dynamic responses can result in uncomfortable, and even seriously dangerous circumstances for buildings. With the growing use of high-strength materials and modern construction techniques, buildings have become relatively light, flexible and lightly damped. The dynamic responses are often much more serious than those for earlier structures, resulting in increased discomfort to the occupants. The inclusion of vibration absorbers in tall buildings can be a successful method of mitigating the effects of these dynamic responses.

Vibration absorbers can be categorized as either active or passive. Tuned mass damper (TMD)[1], both passive and active, have been found to be effective in reducing the response of structures subjected to dynamics loads since 1970s. Tuned mass dampers have been installed in quite a few tall buildings and structures including the Sydney Tower and Chifley Tower in Sydney. The dampers that depend on liquid motion to absorb and dissipate vibration energy, such as tuned liquid dampers (TLD) and tuned liquid column dampers (TLCD) have also been proposed for suppressing structural vibration. Tuned liquid column damper (TLCD) as a passive control device can suppress the structural vibration by the motion of liquid in a column container. The potential advantages of liquid vibration absorbers include: low manufacturing and installation costs; the ability of the absorbers to be incorporated during the design stage of a structure, or to be retrofitted to serve a remedial role; relatively low maintenance requirements; and the availability of the liquid to be used for emergency purposes, or for the everyday function of the structure if fresh water is used [2]. Analytical and experimental research on this type of vibration has been conducted by Samali et al [3].

Viscous interaction between a liquid and solid boundary has been investigated and used to control vibration. The nonlinear mathematical description of the original TLCD was given by Sakai et al.[4]. Their experiments, defining the relationship between the coefficient of head loss (as well as its dependence on the orifice opening ratio) and the liquid damping, confirms the validity of their proposed equation of motion in describing liquid column relative motion under moderate excitation.

Yet, there are some problems in the present control ways, the Artificial Neural Network (ANN) gives an efficient approach to solve the above problems (Li et al., 1999) [5]. The ANN is the simulation of biology neural networks (Wang et al., 1995) [6-7]. It consists of a number of simple cells that are similar to the biology neural networks. The function of single cell is simple and limited, but the networks composed by numerous neural cells can finish some complex assignments. The ANN controller is an efficient method for vibration reduction, which is different from the conventional method in that it can learn control task. It is an adaptive controller with ability of learning and can compensate the time delay in the process of control.

In this paper, two tuned liquid column dampers (TLCDs) are set in orthogonal directions to control the translational and torsional responses of eccentric structure subjected to multi-dimensional earthquake excitations. The ANN is used to predict the response of structures to compensate the delay. The application of the ANN control method provides an effective way to solve these problems.

2 Equations of Motion for Control System

A model of multi-story eccentric structure is showed in figure1, in which O, S and M are the geometry center, stiffness center and mass center. Let u , v and θ denote the translational displacements of floor in x and y directions and rotational angle about the axis z , respectively.

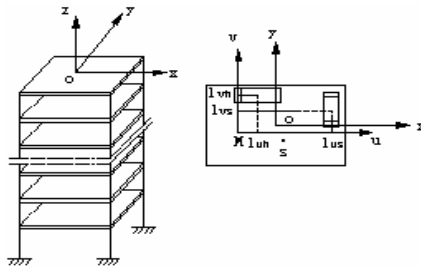


Fig. 1. Model of eccentric structure with TLCDs

The equation of motion for TLCD in x is derived as follow:

$$\rho A_h L_h \ddot{h} + \frac{1}{2} \rho A_h \xi_h |\dot{h}| \dot{h} + 2 \rho A_h g h = - \rho A_h B_h \left[\ddot{x}_l + \ddot{u}_g - l_{vh} (\ddot{\theta}_l + \ddot{\theta}_g) - l_{uh} (\dot{\theta}_l + \dot{\theta}_g)^2 \right]. \quad (1)$$

The equation of motion for TLCDC in y can also be derived through the same method:

$$\rho A_s L_s \ddot{s} + \frac{1}{2} \rho A_s \xi_s |\dot{s}| \dot{s} + 2 \rho A_s g s = -\rho A_s B_s \left[\ddot{y}_l + \ddot{v}_g + l_{us} (\ddot{\theta}_l + \ddot{\theta}_g) - l_{vs} (\dot{\theta}_l + \dot{\theta}_g)^2 \right]. \tag{2}$$

where h and s are the displacements of liquid in the TLCDCs of u and v directions, ρ is the density of the liquid in TLCDCs, ξ_h and ξ_s represent the damping ratios of TLCDCs, L_h and L_s , B_h and B_s , A_h and A_s mean the lengths, widths and cross section areas of liquid in the TLCDCs, l_{uh} and l_{vh} denote the position coordinates of TLCDC in the x direction, l_{us} and l_{vs} are the position coordinates of TLCDC in the y direction, \ddot{u}_g , \ddot{v}_g and $\ddot{\theta}_g$ are the seismic ground accelerations in u , v and θ directions, and \ddot{u}_l , \ddot{v}_l and $\ddot{\theta}_l$ are the accelerations of the l th floor in x , y and θ directions. The values of the fourth parts in the right brackets of Eqs. (1) and (2) are generally very small and can be ignored. The equations are simplified as

$$m_h \ddot{h} + c_h \dot{h} + k_h h = -\rho A_h B_h [\ddot{x}_l + \ddot{u}_g - l_{vh} (\ddot{\theta}_l + \ddot{\theta}_g)], \tag{3}$$

$$m_s \ddot{s} + c_s \dot{s} + k_s s = -\rho A_s B_s [\ddot{y}_l + \ddot{v}_g + l_{us} (\ddot{\theta}_l + \ddot{\theta}_g)], \tag{4}$$

where $m_h = \rho A_h L_h$, $m_s = \rho A_s L_s$, $c_h = \frac{1}{2} \rho A_h \xi_h |\dot{h}|$, $c_s = \frac{1}{2} \rho A_s \xi_s |\dot{s}|$, $k_h = 2 \rho A_h g$ and $k_s = 2 \rho A_s g$. The natural frequencies of TLCDCs are $\omega_h = \sqrt{k_h / m_h} = \sqrt{2g / L_h}$ and $\omega_s = \sqrt{2g / L_s}$, respectively. The control forces produced by TLCDCs can be expressed by

$$W_u = (m_h + m_s)(\ddot{u}_l + \ddot{u}_g) - (m_h l_{vh} + m_s l_{vs})(\ddot{\theta}_l + \ddot{\theta}_g) + \rho A_h B_h \ddot{h}, \tag{5}$$

$$W_v = (m_h + m_s)(\ddot{v}_l + \ddot{v}_g) + (m_h l_{uh} + m_s l_{us})(\ddot{\theta}_l + \ddot{\theta}_g) + \rho A_s B_s \ddot{s}, \tag{6}$$

$$W_\theta = -(m_h l_{vh} + m_s l_{vs})(\ddot{u}_l + \ddot{u}_g) + (m_h l_{vh}^2 + m_s l_{vs}^2)(\ddot{\theta}_l + \ddot{\theta}_g) - \rho A_h B_h l_{vh} \ddot{h} + (m_h l_{uh} + m_s l_{us})(\ddot{v}_l + \ddot{v}_g) + (m_h l_{uh}^2 + m_s l_{us}^2)(\ddot{\theta}_l + \ddot{\theta}_g) + \rho A_s B_s l_{us} \ddot{s}, \tag{7}$$

where W_u , W_v and W_θ denotes the control forces in x , y and θ directions, respectively.

The equation of motion for structural system can be derived as following

$$[m]\{\ddot{x}\} + [c]\{\dot{x}\} + [k]\{x\} = [m]\{\ddot{x}_g\} - \{W\}, \tag{8}$$

where $[m]=\text{diag}(m_1 \cdots m_n, m_1 \cdots m_n, J_1 \cdots J_n)$ and $\{x\}=\{u_1 \cdots u_n, v_1 \cdots v_n, \theta_1 \cdots \theta_n\}$ are the mass matrix and displacement vector of the structures, $[k]$ and $[c]$ is the stiffness and damping matrix of structure, $\{W\}=\{0 \cdots W_u \cdots 0 \cdots W_v \cdots 0 \cdots W_\theta \cdots\}^T$ is control force vector.

Assumed that $[c]$ is a classical damping matrix and then the equation of motion for the i th mode of the controlled structure is expressed as:

$$\ddot{\eta}_i + 2\xi_i \omega_i \dot{\eta}_i + \omega_i^2 \eta_i = F_i^* - U_i^*, \tag{9}$$

where

$$F_i^* = - \left[\sum_{k=1}^n (\phi_{k,i} m_k) \ddot{u}_g + \sum_{k=n+1}^{2n} (\phi_{k,i} m_k) \ddot{v}_g + \sum_{k=2n+1}^{3n} (\phi_{k,i} m_k) \ddot{\theta}_g \right] / m_i^*, \tag{10}$$

$$U_i^* = (\phi_{1,i} W_u + \phi_{n+1,i} W_v + \phi_{2n+1,i} W_\theta) / m_i^*, \tag{11}$$

$$m_i^* = \phi_i^T [m] \phi_i = \sum_{k=1}^{3n} \phi_{k,i}^2 m_k, \tag{12}$$

in which m_i^* is the i th generalized modal mass, m_k means the k th element in the diagonal of $[m]$ matrix, $\phi_{k,i}$ represents the k th element of the i th mode shape, F_i^* is the i th modal excitation and U_i^* is the i th modal control force.

Considering the first r models, the displacement of structure can be expressed as:

$$x = \sum_{i=1}^r (\phi_i \eta_i). \tag{13}$$

3 Control Strategy

The TLCD semi-active algorithm has been presented in reference (Abe and Fujino, 1996) [8]. The equation of motion for a TLCD/SDOF structure system is:

$$\begin{bmatrix} m_s + \rho A l & \rho A b \\ \rho A b & \rho A l \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{\zeta} \end{bmatrix} + \begin{bmatrix} c_s & 0 \\ 0 & c_T \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{\zeta} \end{bmatrix} + \begin{bmatrix} k_s & 0 \\ 0 & k_T \end{bmatrix} \begin{bmatrix} x \\ \zeta \end{bmatrix} = \begin{bmatrix} f(t) \\ m_s \\ 0 \end{bmatrix}, \tag{14}$$

where x is the displacement of structure, ζ is the displacement of the liquid in the column. m_s is the mass of structure, c_s the damping coefficient of the structure, k_s is the stiffness of structure, ρ is the mass density of the liquid, A is the cross sectional area of the column, b is the width of the column, l is the length of column, k is the head loss factor, $f(t)$ is the external force and g is the gravitational acceleration. Here, the orifice head loss is assumed to be proportional to the square of the velocity of the liquid flow.

To apply perturbation analysis, the following normalization of parameters and linearization of the head loss term are introduced: $\omega_s = \sqrt{K_s / M_s}$, $\omega_T = \sqrt{2g/l}$, $\zeta_s = C_s / 2M_s \omega_s$, $\mu = \rho A l / M_s$, $\gamma = \rho A b / \rho A l$.

The total hydraulic loss of half-cycle harmonic response of the liquid in the U type pipe is equal to the energy loss of the corresponding linear system. According to this condition, the equivalent damping coefficient of TLCD can be obtained as follows

$$\zeta_T = \frac{\kappa \omega_T^2 \xi_0}{3 \omega_s \pi g}, \tag{15}$$

where ξ_0 is the displacement amplitude of liquid when the velocity of the liquid is zero and k is the energy loss coefficient of liquid. It can be known the effect of control on the structural response is the most effective when the equivalent frequency, ω_T is tuned nearly to natural frequency ω_s of the structure, i.e.:

$$\omega_s \approx \omega_T \approx \omega_a = \omega_s / \sqrt{1 + \mu}, \tag{16}$$

where ω_a is the mean value of the frequencies of two mode shapes: $\omega_a = (\omega_1 + \omega_2) / 2$. The modal frequencies and modal damping ratios are obtained by disturbing technology as

$$\omega_{1,2} = \omega_a \left(1 \pm \frac{\text{Im} \beta}{2} \right), \tag{17}$$

$$\zeta_{1,2} = (\zeta_T \pm \text{Re} \beta) / 2. \tag{18}$$

The corresponding modes can be written by

$$\left\{ \begin{matrix} \phi_x \\ \phi_\xi \end{matrix} \right\}_{1,2} = \left\{ \begin{matrix} 1 \\ -i(\zeta_T \pm \beta) / (\mu \gamma) \end{matrix} \right\}, \tag{19}$$

where $\left\{ \phi_x \ \phi_\xi \right\}_{1,2}^T$ represents the two mode shape vectors, Im means the imaginary part, Re denotes the real part and $\beta = \sqrt{\zeta_T^2 - \mu \gamma^2}$. It can be known that the two damping factors may sufficiently take effect only when the damping ratios of the two modes are the same. Then, the optimal damping ratio can be expressed as follows

$$\zeta_T^{opt} = \gamma \sqrt{\mu}. \tag{20}$$

As the damping ratio ζ_T has relations with both cell-opening ratio and hydraulic energy loss coefficient, k can be controlled consecutively to keep damping ratio ζ_T have the optimal values. In order to make the application more convenient, the sloshing process of the liquid is divided into some parts. Only when the velocity of the liquid is zero, the area of orifice opening is adjusted swiftly. Hence, the loss coefficient is obtained by

$$\kappa = \frac{3\pi g \gamma \sqrt{\mu}}{\omega_a^2 \xi_0}. \tag{21}$$

The main aim of the adopted control criterion is only to excite the first mode and not the second mode by selecting rationally ζ_T , i.e., by adjusting the open-cell ratio to change the damping coefficient, ζ_T , of TLCD the participant value of the first mode at this time is the maximum. This process above can be realized by following equation

$$\frac{\xi}{\dot{x}} = -\frac{\zeta_T + \beta}{\omega \mu \gamma_a}. \tag{22}$$

At the moment of controlling the area of orifice opening of TLCD, which means at the moment merely after $\dot{\xi} = 0$, the response of structure can be approximately expressed as

$$\dot{x} = \text{Re} \left[\dot{x}_0 e^{i\omega_a t} \right], \tag{23}$$

and

$$\xi = \text{Re} \left[\xi_0 e^{i\omega_a t} \right], \tag{24}$$

where \dot{x}_0 is the displacement response of structure when the velocity of liquid is equal to zero. Substitution of Eqs. (17) and (18) into Eq. (16) yields:

$$\xi_0 = -\frac{(\zeta_T + \beta)\dot{x}_0}{\mu \gamma \omega_a}. \tag{25}$$

Due to $\zeta_T \gg \gamma \sqrt{\mu}$ and ζ_T being non-negative, Eq. (19) is simplified as:

$$\zeta_T = \left| \frac{\mu \gamma \omega_a \xi_0}{2\dot{x}_0} \right|. \tag{26}$$

Substituting Eq. (15) into Eq. (26) the following equation can be derived as

$$\kappa = \frac{3\omega_s \pi g}{2\omega_T^3} \left| \frac{\mu \gamma \omega_a}{\dot{x}_0} \right|. \tag{27}$$

4 Prediction of Structural Response

The Multi-layer forward BP network is an algorithm often used in ANN controller. It has the learning capability to learn control criterion and strategy. The trained ANN can predict the structural future response by exerting the control to the structure in advance, and then wipe out the “time lag”. Here, multi-layer forward ANN is applied

by means of the improved BP learning algorithm to accelerate the training speed. First, the ANN must be trained to learn the rule of structural vibration and save the information in the weights. The trained ANN can predict the structural response in later time.

A five-story eccentric structure is used to train the ANN. The mass of its every floor is $1.5 \times 10^6 \text{ N s}^2/\text{m}$, the moment of inertia to the mass center is $4.9 \times 10^5 \text{ Kg} \cdot \text{m}^2$ and the translational stiffnesses in x and y directions are $1.5 \times 10^8 \text{ N/m}$ and $6.6 \times 10^7 \text{ N/m}$, respectively. The torsional stiffness of each floor is $3.5 \times 10^{10} \text{ N} \cdot \text{m}/\text{Rad}$. The eccentricity in x is 3m and 2m in y. A three-layer ANN is used to predict the structural response. There are 22 nodes in input layer, which represent the translational structural responses of five stories in the foregoing two periods of time and the seismic input in x and y. There are two nodes in the output layer, which represent the structural translational response in the next period of time. The ANN is trained in the supervised way and the object error is 0.002.

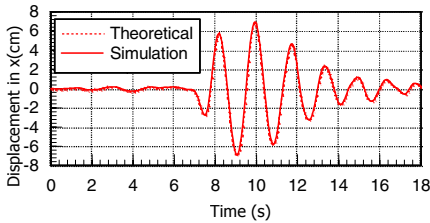


Fig. 2. Predicted response in x (Tianjin)

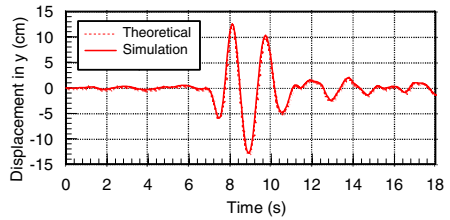


Fig. 3. Predicted response in y (Tianjin)

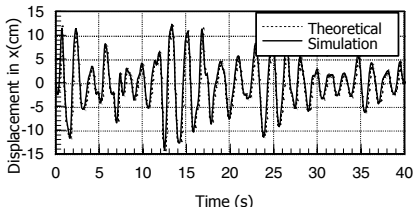


Fig. 4. Predicted response in x

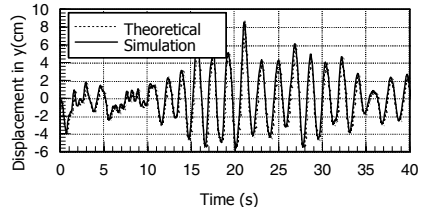


Fig. 5. Predicted response in y (El Centro)

The structural responses under Tianjin seismic excitation are used as sampling data. The one third of data is used to train the ANN and others are used to test the ANN. The results are showed in Fig.2 and Fig. 3. The structural responses to El Centro seismic record is also predicted by the ANN to verify the ability of generalization and the results are showed in Fig. 4 and Fig.5.

5 Numerical Example and Analysis

A five-story eccentric structure is as numerical example. Two TLCDS are installed on the top of structure in x and y directions respectively. The parameters of TLCDS in x

direction are: $\mu=0.01$, $\gamma=0.5$, $L=2.42\text{m}$. The parameters in y direction are: $\mu=0.01$, $\gamma=0.01$, $L=5.5\text{m}$. El Centro north-south and east-west waves are input to the two directions of the structure. The time history of structural displacements is showed in Fig.7 through Fig.9.

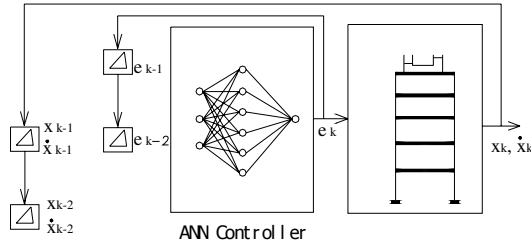


Fig. 6. Artificial neural Network

The artificial neural networks can trace and compute the required orifice-opening ratio of TLCD in real time, and the required damping ratio instead of numerous non-linear computations. The time history of orifice-opening ratio in x and in y is showed in Figure 10 and in Figure 11 respectively. The control effect of the top floor is showed in Table 1. Consequently, the effect of its control on seismic response of structures is very well.

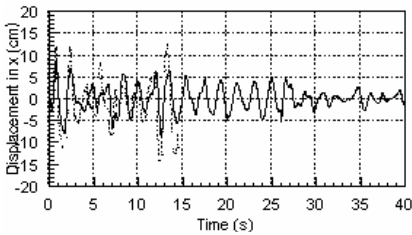


Fig. 7. History of displacements on top of structure in x direction (El Centro)

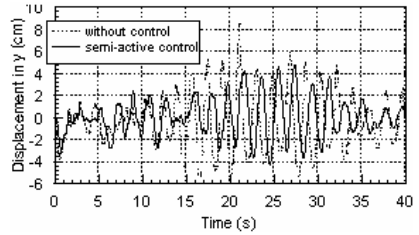


Fig. 8. History of displacement on top of structure in y direction (El Centro)

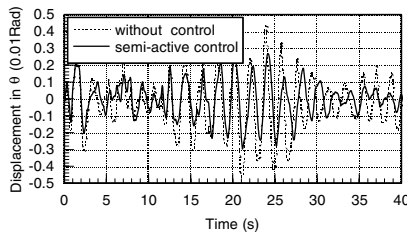


Fig. 9. History of displacement on top of structure in θ direction (El Centro)

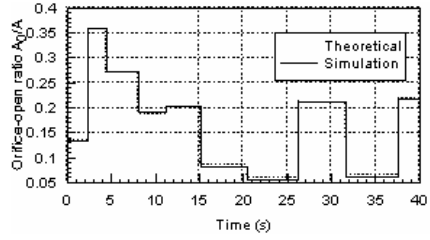
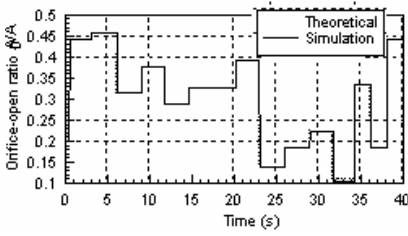


Fig. 10. Time history of orifice open ration in x **Fig. 11.** Time history of orifice open ratio; in y

Table 1. Control effect on the top of structure

	Translation			Rotation		
	x (cm)	y (cm)	θ (10^2 Rad)	a_x (cm/s^2)	a_y (cm/s^2)	a_θ (10^2 Rad/s2)
Without controlPeak	12.67	8.57	0.44	432.22	217.35	16.60
Passive control Peak	12.05	5.97	0.33	351.30	205.05	14.61
Control effect (%)	4.87	30.31	23.72	18.72	5.65	11.96
Semi-active control Peak	9.64	4.78	0.27	249.12	145.40	10.36
Control effect (%)	23.9	44.25	38.98	42.36	33.09	37.56

6 Conclusions

In this paper, the torsionally coupled control of eccentric structure using semi-active TLCD is presented. The following conclusions according to above studies can be drawn as:

- (1) The translational-torsional coupled vibration can be controlled with TLCDs installed in two directions of structures.
- (2) The structural response can be predicted using the ANN and compensate the time delay in the process of control.
- (3) It can avoid the complicated non-linear computation of TLCD and save the calculation time by replacing the control criterion with the ANN and the effectiveness of control is much better.

Acknowledgments

The research in this paper is supported by the Outstanding Youth Science Foundation of the National Natural Science Foundation of China, the education commission foundation of Liaoning Province and key project of Natural Science Foundation of Liaoning Province. These supports are greatly acknowledged.

References

1. Fujino, Y., Abe, M.: Design Formulas for Tuned Mass Dampers Based on a Perturbation Technique. *Journal of Earthquake Engineering and Structural Dynamics* **22** (1993) 833-854
2. Hitchcock, P.A., Kwok, K.C.S., Watkins, R.D.: Characteristics of Liquid Column Vibration Absorbers. *Engineering Structures* **2** (1997) 126-134

3. Samali, B.: Dynamic Response of Structures Equipped with Tuned Liquid Column Dampers. Prof. Inst. Aust. Vibration and Noise Conf. Melbourne (1990) 138-143
4. Sakai, F., Takaeda, S., Tamake, T.: Tuned Liquid Column Damper-new Type Device for Suppression of Building Vibrations. Proc. Int. Conf. On High-rise Buildings. Nanjing, China (1989) 926-931
5. Li, H.N., Yan S.: State-of-art of Review for Development of Control of Intelligent Structures. *Earthquake Engineering and Engineering Vibration* **19** (2) (1999) 29-36
6. Yan, S., Li, H.N., Lin, G.: Studies on Control Parameters of Adjustable Tuned Liquid Column Damper. *Earthquake Engineering and Engineering Vibration* **18** (4) (1998) 96-101
7. Yan, S., Li, H.N.: Performance of Vibration Control for U Type Water Tank with Variable Cross Section. *Earthquake Engineering and Engineering Vibration* **19** (1) (1999) 197-201
8. Abe, M., Kimura, S., Fujino, Y.: Control Laws for Semi-active Tuned Liquid Column Damper with Variable Orifice Openings. Proc. of 2nd International Workshop on Structural Control, Hong Kong (1996) 5-10

Development of the Multi-target Tracking Scheme Using Particle Filter

Yang Weon Lee

Department of Information and Communication Engineering, Honam University,
Seobongdong, Gwangsan-gu, Gwangju, 506-714, South Korea
ywlee@honam.ac.kr

Abstract. This paper introduces a particle filter algorithm determining the measurement-track association problem in multi-target tracking. This scheme is important in providing a computationally feasible alternative to complete enumeration of JPDA which is intractable. We have proved that given an artificial measurement and track's configuration, particle filter scheme converges to a proper plot in a finite number of iterations. Also, a proper plot which is not the global solution can be corrected by re-initializing one or more times. In this light, even if the performance is enhanced by using the particle filter, we also note that the difficulty in tuning the parameters of the particle filter is critical aspect of this scheme. The difficulty can, however, be overcome by developing suitable automatic instruments that will iteratively verify convergence as the network parameters vary.

1 Introduction

The primary purpose of a multi-target tracking(MTT) system is to provide an accurate estimate of the target position and velocity from the measurement data in a field of view. Naturally, the performance of this system is inherently limited by the measurement inaccuracy and source uncertainty which arises from the presence of missed detection, false alarms, emergence of new targets into the surveillance region and disappearance of old targets from the surveillance region. Therefore, it is difficult to determine precisely which target corresponds to each of the closely spaced measurements. Although trajectory estimation problems have been well studied in the past, much of this previous work assumes that the particular target corresponding to each observation is known. Recently, with the proliferation of surveillance systems and their increased sophistication, the tools for designing algorithms for data association have been announced.

Generally, there are three approaches in data association for MTT : non-Bayesian approach based on likelihood function [1], Bayesian approach [2,3,4], and neural network approach [5]. The major difference of the first two approaches is how treat the false alarms. The non-Bayesian approach calculates all the likelihood functions of all the possible tracks with given measurements and selects the track which gives the maximum value of the likelihood function. Meanwhile,

the tracking filter using Bayesian approach predicts the location of interest using *a posteriori* probability. These two approaches are inadequate for real time applications because the computational complexity is tremendous.

As an alternative approach, Sengupta and Iltis[5] suggested a Hopfield neural network probabilistic data association (HNPDA) to approximately compute *a posteriori* probability β_j^t , for the joint probabilities data association filter (JPDAF)[6] as a constrained minimization problem. This technique based on the use of neural networks was also started by comparison with the traveling salesman problem(TSP). In fact β_j^t is approximated by the output voltage X_j^t of a neuron in an $(m + 1) \times n$ array of neurons, where m is the number of measurements and n is the number of targets. Sengupta and Iltis[5] claimed that the performance of the HNPDA was close to that of the JPDAF in situations where the numbers of measurements and targets were in the ranges of 3 to 20 and 2 to 6, respectively. The success of the HNPDA in their examples was credited to the accurate emulation of all the properties of the JPDAF by the HNPDA.

However, the neural network developed in [5] has been shown the two problems. First, the neural network developed in [5] has been shown to have improper energy functions. Second, heuristic choices of the constant parameters in the energy function in [5] didn't guarantee the optimal data association.

2 CONDENSATION Algorithm

2.1 CONDENSATION Algorithm

The particle filter approach to track multi-target, also known as the condensation algorithm [9] and Monte Carlo localisation, uses a large number of particles to explore the state space. Each particle represents a hypothesised target location in state space. Initially the particles are uniformly randomly distributed across the state space, and each subsequent frame the algorithm cycles through the steps illustrated in Figure 1:

1. Deterministic drift: particles are moved according to a deterministic motion model (a damped constant velocity motion model was used).
2. Update probability density function (PDF): Determine the probability for every new particle location.
3. Resample particles: 90 % of the particles are resampled with replacement, such that the probability of choosing a particular sample is equal to the PDF at that point; the remaining 10 % of particles are distributed randomly throughout the state space.
4. Diffuse particles: particles are moved a small distance in state space under Brownian motion.

This results in particles congregating in regions of high probability and dispersing from other regions, thus the particle density indicates the most likely target states. See [9] for a comprehensive discussion of this method. The key

strengths of the particle filter approach to localisation and tracking are its scalability (computational requirement varies linearly with the number of particles), and its ability to deal with multiple hypotheses (and thus more readily recover from tracking errors). However, the particle filter was applied here for several additional reasons:

- it provides an efficient means of searching for a target in a multi-dimensional state space.
- reduces the search problem to a verification problem, ie. is a given hypothesis face-like according to the sensor information?
- allows fusion of cues running at different frequencies.

The last point is especially important for a system operating multiple cues with limited computational resources, as it facilitates running some cues slower than frame rate (with minimal computational expense) and incorporating the result from these cues when they become available. If a cue takes n frames to return a result, by the time the cue is ready, the particles will have moved from where they were n frames ago. To facilitate such cues the system keeps a record of every particle’s history over a specified number of frames k . The cue value determined for a particle nk frames ago can then be assigned to the children of that particle in the current frame, thus propagating forward the cues response to the current frame. Conversely, probabilities associated with particles that were not propagated are discarded.

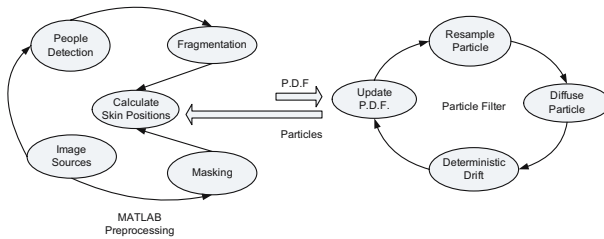


Fig. 1. Particle Filter Calculation Process

2.2 Application of CONDENSATION for the Multi-target Tracking

In order to apply the Condensation Algorithm to the multi-target tracking, we extend the methods described by Black and Jepson [10]. Specifically, a state at time t is described as a parameter vector: $s_t = (\mu, \phi^i, \alpha^i, \rho^i)$ where: μ is the integer index of the predictive model, ϕ^i indicates the current position in the model, α^i refers to an amplitudal scaling factor and ρ^i is a scale factor in the time dimension.

Initialization. The sample set is initialized with N samples distributed over possible starting states and each assigned a weight of $\frac{1}{N}$. Specifically, the initial parameters are picked uniformly according to:

$$\begin{aligned} \mu &\in [1, \mu_{max}] \\ \phi^i &= \frac{1 - \sqrt{y}}{\sqrt{y}}, y \in [0, 1] \\ \alpha^i &= [\alpha_{min}, \alpha_{max}] \\ \rho^i &\in [\rho_{min}, \rho_{max}] \end{aligned} \tag{1}$$

Prediction. In the prediction step, each parameter of a randomly sampled s_t is used to s_{t+1} determine based on the parameters of that particular s_t . Each old state, s_t , is randomly chosen from the sample set, based on the weight of each sample. That is, the weight of each sample determines the probability of its being chosen. This is done efficiently by creating a cumulative probability table, choosing a uniform random number on $[0, 1]$, and then using binary search to pull out a sample (see Isard and Blake for details[9]). The following equations are used to choose the new state :

$$\begin{aligned} \mu_{t+1} &= \mu_t \\ \phi_{t+1}^i &= \phi_t^i + \rho_t^i + N(\sigma_\phi) \\ \alpha_{t+1}^i &= \alpha_t^i + N(\sigma_\alpha) \\ \rho_{t+1} &= \rho_t^i + N(\sigma_\rho) \end{aligned} \tag{2}$$

where $N(\sigma_*)$ refers to a number chosen randomly according to the normal distribution with standard deviation σ_* . This adds an element of uncertainty to each prediction, which keeps the sample set diffuse enough to deal with noisy data. For a given drawn sample, predictions are generated until all of the parameters are within the accepted range. If, after, a set number of attempts it is still impossible to generate a valid prediction, a new sample is created according to the initialization procedure above. In addition, 10 percent of all samples in the new sample set are initialized randomly as in the initialization step above (with the exception that rather than having the phase parameter biased towards zero, it is biased towards the number of observations that have been made thus far). This ensures that local maxima can't completely take over the curve; new hypotheses are always given a chance to dominate.

Updating. After the Prediction step above, there exists a new set of N predicted samples which need to be assigned weights. The weight of each sample is a measure of its likelihood given the observed data $Z_t = (z_t, z_{t_1}, \dots)$. We define $Z_{t,i} = (z_{t,i}, z_{(t-1),i}, \dots)$ as a sequence of observations for the i th coefficient over time; specifically, let $Z_{(t,1)}, Z_{(t,2)}, Z_{(t,3)}, Z_{(t,4)}$ be the sequence of observations of the horizontal velocity of the left hand, the vertical velocity of the left hand, the horizontal velocity of the right hand, and the vertical velocity of the right hand

respectively. Extending Black and Jepson [10], we then calculate the weight by the following equation:

$$p(z_t|s_t) = \prod_{i=1}^4 p(Z_{t,i}|s_t) \tag{3}$$

where $p(z_{t,i}|s_t) = \frac{1}{\sqrt{2\pi}} \exp \frac{-\sum_{j=0}^{\omega-1} (z_{(t-j),i} - \alpha^* m_{(\phi^* - \rho^* j),i}^\mu)^2}{2(\omega-1)}$ and where ω is the size of a temporal window that spans back in time. Note that ϕ^* , α^* and ρ^* refer to the appropriate parameters of the model for the blob in question and that $\alpha^* m_{(\phi^* - \rho^* j),i}^\mu$ refers to the value given to the i th coefficient of the model μ interpolated at time $\phi^* - \rho^* j$ and scaled by α^* .

Classification. With this algorithm in place, all that remains is actually classifying the correct data association as one of the many observed measurements. Since the whole idea of Condensation is that the most likely hypothesis will dominate by the end, I chose to use the criterion of which measurement was deemed most likely at the end of the measurements sequence to determine the class of the entire track sequence. Determining the probability assigned to each target is a simple matter of summing the weights of each sample in the sample set at a given moment whose state refers to the model in question.

3 Development of the Energy Function

3.1 Representing Measurement-Target Relationship

Fig. 2 shows the overall scheme. This system consists of three blocks: acquisition, association, and prediction. The purpose of the acquisition is to determine the initial starting position of the tracking. After this stage, the association and prediction interactively determine the tracks. Our primary concern is the association part that must determine the actual measurement and target pairs, given the measurements and the predicted gate centers.

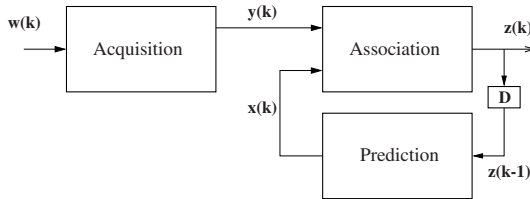


Fig. 2. An overall scheme for target tracking

Let m and n be the number of measurements and targets, respectively, in a surveillance region. Then, the relationships between the targets and measurements are efficiently represented by the *validation matrix* ω [6]:

$$\omega = \{\omega_{jt}|j \in [1, m], t \in [0, n]\}, \tag{4}$$

where the first column denotes clutter and always $\omega_{j0} = 1$. For the other columns, $\omega_{jt} = 1$ ($j \in [1, m], t \in [1, n]$), if the validation gate of target t contains the measurement j and $\omega_{jt} = 0$, otherwise.

Based on the validation matrix, we must find *hypothesis matrix* [6] $\hat{\omega}(= \{\hat{\omega}_{jt}|j \in [1, m], t \in [0, n]\})$ that must obey the data association hypothesis (or feasible events [6]):

$$\begin{cases} \sum_{t=0}^n \hat{\omega}_{jt} = 1 & \text{for } (j \in [1, m]), \\ \sum_{j=1}^m \hat{\omega}_{jt} \leq 1 & \text{for } (t \in [1, n]). \end{cases} \tag{5}$$

Here, $\hat{\omega}_{jt} = 1$ only if the measurement j is associated with clutter ($t = 0$) or target ($t \neq 0$). Generating all the hypothesis matrices leads to a combinatorial problem, where the number of data association hypothesis increases exponentially with the number of targets and measurements.

3.2 Relations Between Data Association and Energy Function

Suppose there are n targets and m measurements. The energy function for the data association is written below.

$$\begin{aligned} E_{DAP} = & \frac{A}{2} \sum_{j=0}^m \sum_{t=1}^n \sum_{\tau=1, \tau \neq t}^n X_j^t X_j^\tau + \frac{B}{2} \sum_{t=1}^n \sum_{j=0}^m \sum_{l=0, l \neq j}^m X_j^t X_l^t + \frac{C}{2} \sum_{t=1}^n (\sum_{j=0}^m X_j^t - 1)^2 \\ & + \frac{D}{2} \sum_{j=0}^m \sum_{t=1}^n (X_j^t - \rho_j^t)^2 + \frac{E}{2} \sum_{j=0}^m \sum_{t=1}^n \sum_{\tau=1, \tau \neq t}^n (X_j^t - \sum_{l=0, l \neq j}^m \rho_l^\tau)^2. \end{aligned} \tag{6}$$

X_j^t is the output voltage of a neuron in an $(m + 1) \times n$ array of neurons and is the approximation to the *a posteriori* probability β_j^t in the JPDAF [6]. This *a posteriori* probability, in the special case of the PDAF [6] when the probability P_G that the correct measurement falls inside the validation gate is unity, is denoted by ρ_j^t . Actually, P_G is very close to unity when the validation gate size is adequate. In (6), A,B,C,D, and E are constants.

In order to justify the first two terms of E_{DAP} in (6), we assumed that the dual assumptions of no two returns form the same target and no single return from two targets are consistent with the presence of a dominating X_j^t in each row and each column of the $(m + 1) \times n$ array of neurons. The third term of E_{DAP} is used to constrain the sum of the X_j^t 's in each column to unity *i.e.* $\sum_{j=0}^m X_j^t = 1$. This constraint is consistent with the requirement that $\sum_{j=0}^m \beta_j^t = 1$ in both the JPDAF and the PDAF. Therefore, this constraint, by itself, does not permit us to infer whether the β_j^t 's are from the JPDAF, or from the PDAF. The assumption used to set up the fourth term is that this term is small only if X_j^t is close to ρ_j^t , in which case the neural network simulates more closely the PDAF for each target rather than the intended JPDAF in the multitarget scenario. Finally, the fifth term is supposed to be minimized if X_j^t is not large unless for each $\tau \neq t$ there is a unique $l \neq j$ such that ρ_l^τ is large.

3.3 MAP Estimates for Data Association

The ultimate goal of this problem is to find the hypothesis matrix $\hat{\omega} = \{\hat{\omega}_{jt}|j \in [1, m], t \in [\theta, n]\}$, given the observation \mathbf{y} and \mathbf{x} , which must satisfy (5). From now on, let's associate the realizations- the gate center \mathbf{x} , the measurement \mathbf{y} , the validation matrix ω , and $\hat{\omega}$ - to the random processes- X, Y, Ω , and $\hat{\Omega}$.

Next, consider that $\hat{\Omega}$ is a parameter space and (Ω, Y, X) is an observation space. Then, *a posteriori* can be derived by the Bayes rule:

$$P(\hat{\omega}|\omega, \mathbf{y}, \mathbf{x}) = \frac{P(\omega|\hat{\omega})P(\mathbf{y}, \mathbf{x}|\hat{\omega})P(\hat{\omega})}{P(\omega, \mathbf{y}, \mathbf{x})}. \tag{7}$$

Here, we assumed that $P(\omega, \mathbf{y}, \mathbf{x}|\hat{\omega}) = P(\omega|\hat{\omega})P(\mathbf{y}, \mathbf{x}|\hat{\omega})$, since the two variables ω and (\mathbf{x}, \mathbf{y}) are separately observed. This assumption makes the problem more tractable as we shall see later. This relationship is illustrated in Fig. 3.

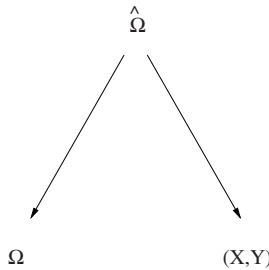


Fig. 3. The parameter space and the observation space

Given the parameter $\hat{\Omega}$, Ω and (X, Y) are observed. If the conditional probabilities describing the relationships between the parameter space and the observation spaces are available, one can obtain the MAP estimator:

$$\omega^* = \arg \max_{\hat{\omega}} P(\hat{\omega}|\omega, \mathbf{y}, \mathbf{x}). \tag{8}$$

3.4 Representing Constraints by Energy Function

As a system model, we assume that the conditional probabilities are all Gibbs distributions:

$$\begin{cases} P(\mathbf{y}, \mathbf{x}|\hat{\omega}) \triangleq \frac{1}{Z_1} \exp\{-E(\mathbf{y}, \mathbf{x}|\hat{\omega})\}, \\ P(\omega|\hat{\omega}) \triangleq \frac{1}{Z_2} \exp\{-E(\omega|\hat{\omega})\}, \\ P(\hat{\omega}) \triangleq \frac{1}{Z_3} \exp\{-E(\hat{\omega})\}, \end{cases} \tag{9}$$

where Z_s ($s \in [1, 2, 3]$) is called partition function:

$$Z_s = \int_{\hat{\omega} \in \mathcal{E}} \exp\{-E(\hat{\omega})\} d\hat{\omega}. \tag{10}$$

Here, E denotes the energy function. Substituting (9) into (7), (8) becomes

$$\hat{\omega}^* = \arg \min_{\hat{\omega}} [E(\mathbf{y}, \mathbf{x}|\hat{\omega}) + E(\omega|\hat{\omega}) + E(\hat{\omega})]. \tag{11}$$

Since the optimization is executed with respect to $\hat{\omega}$, the denominator in (7) is independent of $\hat{\omega}$ and therefore irrelevant for its minimization.

The energy functions are realizations of the constraints both for the target trajectories and the measurement-target relationships. For instance, the first term in (11) represents the distance between measurement and target and could be minimized approximately using the constraints in (5). The second term intends to suppress the measurements which are uncorrelated with the valid measurements. The third term denotes constraints of the validation matrix and it can be designed to represent the two restrictions as shown in (5). The energy equations of each term are defined respectively:

$$\begin{cases} E(\mathbf{y}, \mathbf{x}|\hat{\omega}) \triangleq \sum_{t=1}^n \sum_{j=1}^m r_{jt} \hat{w}_{jt}, \\ E(\omega|\hat{\omega}) \triangleq \sum_{t=1}^n \sum_{j=1}^m (\hat{w}_{jt} - w_{jt})^2, \\ E(\hat{\omega}) \triangleq \sum_{t=1}^n (\sum_{j=1}^m \hat{w}_{jt} - 1) + \sum_{j=1}^m (\sum_{t=0}^n \hat{w}_{jt} - 1). \end{cases} \tag{12}$$

Putting (12) into (11), one gets

$$\begin{aligned} \hat{\omega}^* = \arg \min_{\hat{\omega}} & \left[\alpha \sum_{t=1}^n \sum_{j=1}^m r_{jt}^2 \hat{w}_{jt} + \frac{\beta}{2} \sum_{t=1}^n \sum_{j=1}^m (\hat{w}_{jt} - w_{jt})^2 \right. \\ & \left. + \sum_{t=1}^n (\sum_{j=1}^m \hat{w}_{jt} - 1) + \sum_{j=1}^m (\sum_{t=0}^n \hat{w}_{jt} - 1) \right], \end{aligned} \tag{13}$$

where α and β are a coefficient of the weighted distance measure and the matching term respectively.

Using this scheme, the optimal solution is obtained by assigning observations to tracks in order to minimize the weighted total summed distance from all observations to the tracks to which they are assigned. This is thought of as a version of the well-known assignment problem for which optimal solutions have been developed with constraints (11).

4 Experimental Results

In this section, we present some results of the experiments comparing the performance of the proposed particle filter with that of the JPDA [6]. We just used the standard Kalman filter [12] for the estimation part once feasible matrix $\hat{\omega}$ is computed. The performance of the particle filter is tested in two separate cases in the simulation. In the first case, we consider two crossing and parallel targets for testing the track maintenance and accuracy in view of clutter density. In the second case, all the targets as listed in Table 1 are used for testing the multi-target tracking performance. The dynamic models for the targets have been used

by the Singer model developed in [13]. Target 8 and 9 in Table 1 were given acceleration $20m/sec^2$ and $10m/sec^2$ between 15 and 35 turn period, respectively.

Table 1. Initial Positions and Velocities of 10 targets

Target <i>i</i>	Position (<i>km</i>)		Velocity (<i>km/s</i>)	
	<i>x</i>	<i>y</i>	\dot{x}	\dot{y}
1	-4.0	1.0	0.2	-0.05
2	-4.0	1.0	0.2	0.05
3	-6.0	-5.0	0.0	0.3
4	-5.5	-5.0	0.0	0.3
5	8.0	-7.0	-0.4	0.0
6	-8.0	-8.0	0.4	0.0
7	-5.0	9.0	0.25	0.0
8	-5.0	8.9	0.25	0.0
9	0.5	-3.0	0.1	0.2
10	9.0	-9.0	0.01	0.2

Table 2 summarizes the rms position and velocity errors for each target in the second test. From Table 3, we note that MAPADA’s track maintenance capability is higher than JPDA, even if the rms error of each track is a little larger than the JPDA. This result comes from the choosing the course weighted distance measure. We also note that the general performance of the MAPADA is almost equivalent to that of JPDA. The MAPADA appears to be a alternative to the JPDA instead of HNPDA. Also, It could replace the sequential computations required for the JPDA with a parallel scheme. But the difficulty in adjusting the parameters still exist.

Table 2. RMS Errors in the case of ten targets

target <i>i</i>	Position error (<i>km</i>)		Velocity error (<i>km/s</i>)		Track maintenance (%)	
	JPDA	Particle	JPDA	Particle	JPDA	Particle
1	0.039	0.042	0.017	0.018	100	100
2	0.038	0.043	0.021	0.019	100	100
3	0.039	0.042	0.016	0.018	100	100
4	0.044	0.042	0.021	0.018	93	100
5	0.040	0.044	0.016	0.019	100	100
6	0.040	0.042	0.011	0.045	100	100
7	0.040	0.042	0.011	0.045	100	100
8	0.251	0.295	0.072	0.118	65	53
9	0.056	0.052	0.024	0.020	85	93
10	0.040	0.044	0.017	0.018	100	98

5 Conclusion

The purpose of this paper was to explore a particle filter for data association method as a tool for applying multi-target tracking. It was shown that it always yields consistent data association, in contrast to the JPDA, and that these associated data measurements are very effective for multi-target filters. Although the particle filter finds the convergence recursively, the particle filter is a general method for solving the data association problems in multi-target tracking.

Acknowledgements

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD, Basic Research Promotion Fund) (KRF-2006-D00356 (I00068)).

References

1. Alspach, D.L.: A Gaussian Sum Approach to the Multi-Target Identification Tracking Problem. *Automatica* **11** (1975) 285-296
2. Bar-Shalom, Y.: Extension of the Probabilistic Data Association Filter in Multi-Target Tracking. *Proceedings of the 5th Symposium on Nonlinear Estimation* (1974) 16-21
3. Reid, D.B.: An algorithm for tracking multiple targets. *IEEE Trans. on Automat. Contr.* **24** (1979) 843-854
4. Lee, Y.: Development of relaxation scheme for the Multiple Targets in a Cluttered Environment. *Lecture Notes in Computer Science*, Vol. 1281. Springer-Verlag, Berlin Heidelberg New York (2005) 434-445
5. D. Sengupta, D., Iltis, R.A.: Neural solution to the multitarget tracking data association problem. *IEEE Trans. on AES* **25**(1999) 96-108
6. Fortmann, T.E., Bar-Shalom, Y., Scheffe, M.: Sonar Tracking of Multiple Targets Using Joint Probabilistic Data Association. *IEEE J. Oceanic Engineering* **OE-8** (1983) 173-184
7. Lee, Y.: Adaptive Data Association for Multi-target Tracking using relaxation. *LNCS 35644*, (2005) 552-561
8. Lee, Y., Seo, J.H., Lee, J.G.: A Study on the TWS Tracking Filter for Multi-Target Tracking. *Journal of KIEE* **41**(4)(2004) 411-421
9. ISard, M., Blake, A.: CONDENSATION-conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision* **29**(1) (1998) 5-28
10. Black, M.J., Jepson, A.D.: A Probabilistic Framework for Matching Temporal Trajectories: Condensation-based Recognition of Gestures and Expressions. In *Proceedings 5th European Conf. Computer Vision* **1** (1998) 909-924
11. Cichocki, A., Unbenhauen, R.: *Neural networks for optimization and signal processing*. Wiley, New York, (1993) 520-526
12. Kalman, R.E.: A New Approach to Linear Filtering and Prediction Problems. *Trans. ASME (J. Basic Eng.)* **82** (1960) 34-45
13. Singer, R.A.: Estimating optimal tracking filter performance for manned maneuvering targets. *IEEE Trans. Aerospace and Electronic Systems* **6** (1970) 473-483

Author Index

- Abiyev, Rahib H. II-241
Acuña, Gonzalo I-311, I-1255, II-391
Afzulpurkar, Nitin V. III-252
Ahmad, Khurshid II-938
Ahn, Tae-Chon II-186
Ai, Lingmei II-1202
Akiduki, Takuma II-542
Al-Jumeily, Dhiya II-921
Al-shanableh, Tayseer II-241
Aliiev, R.A. II-307
Aliiev, R.R. II-307
Almeida, Anibal T. de I-138, III-73
Amari, Shun-ichi I-935
Anitha, R. I-546
Araújo, Ricardo de A. II-602
Aung, M.S.H. II-1177
- Bae, Hyeon III-641
Bae, JeMin I-1221
Baek, Gyeongdong III-641
Baek, Seong-Joon II-1240
Bai, Qiuguo III-1107
Bai, Rui II-362
Bai, Xuerui I-349
Bambang, Riyanto T. I-54
Bao, Zheng I-1303
Barua, Debjanee II-562
Bassi, Danilo II-391
Bastari, Alessandro III-783
Baten, A.K.M.A. II-1221
Bevilacqua, Vitoantonio II-1107
Bi, Jing I-609
Bin, Deng III-981
Bin, Liu III-981
Boumaiza, Slim I-582
- Cai, ManJun I-148
Cai, W.C. I-786
Cai, Wenchuan I-70
Cai, Zixing I-743
Caiyun, Chen III-657, III-803
Calster, B. Van II-1177
Canu, Stéphane III-486
Cao, Fenwen II-810
Cao, Jinde I-941, I-958, I-1025
- Cao, Shujuan II-680
Carpenter, Gail A. I-1094
Carvajal, Karina I-1255
Cecchi, Guillermo II-500, II-552
Cecchi, Stefania III-731, III-783
Celikoglu, Hilmi Berk I-562
Chacon M., Mario I. III-884
Chai, Lin I-222
Chai, Tianyou II-362
Chai, Yu-Mei I-1162
Chandra Sekhar, C. I-546
Chang, Bao Rong III-357
Chang, Bill II-1221
Chang, Guoliang II-1168
Chang, Hyung Jin III-506
Chang, Shengjiang II-457
Chang, T.K. II-432
Chang, Y.P. III-580
Chang, Zhengwei III-1015
Chao, Kuei-Hsiang III-1145
Che, Haijun I-480
Chen, Boshan III-123
Chen, Chaochao I-824
Chen, Dingguo I-183, I-193
Chen, Feng I-473, I-1303
Chen, Fuzan II-448
Chen, Gong II-1056
Chen, Huahong I-1069
Chen, Huawei I-1069
Chen, Hung-Cheng III-26
Chen, Jianxin I-1274, II-1159
Chen, Jie II-810
Chen, Jing I-1274
Chen, Jinhuan III-164
Chen, Joseph III-1165
Chen, Juan II-224
Chen, Lanfeng I-267
Chen, Le I-138
Chen, Li-Chao II-656
Chen, Lingling II-1291
Chen, Min-You I-528
Chen, Mou I-112
Chen, Mu-Song III-998
Chen, Ping III-426

- Chen, Po-Hung III-26, III-1120
 Chen, Qihong I-64
 Chen, Shuzhen III-454
 Chen, Tianping I-994, I-1034
 Chen, Ting-Yu II-336
 Chen, Wanming I-843
 Chen, Weisheng I-158
 Chen, Wen-hua I-112
 Chen, Xiaowei II-381
 Chen, Xin I-813
 Chen, Xiyuan III-41
 Chen, Ya zhu III-967
 Chen, Yen-wei II-979
 Chen, Ying III-311, III-973
 Chen, Yong I-1144, II-772
 Chen, Yuehui I-473, II-1211
 Chen, Yunping III-1176
 Chen, Zhi-Guo II-994, III-774
 Chen, Zhimei I-102
 Chen, Zhimin III-204
 Chen, Zhong I-776, III-914
 Chen, Zhongsong III-73
 Cheng, Gang I-231
 Cheng, Jian II-120
 Cheng, Zunshui I-1025
 Chi, Qinglei I-29
 Chi, Zheru I-626
 Chiu, Ming-Hui I-38
 Cho, Jae-Hyun III-923
 Cho, Sungzoon II-880
 Choi, Jeoung-Nae III-225
 Choi, Jin Young III-506
 Choi, Seongjin I-602
 Choi, Yue Soon III-1114
 Chu, Shu-Chuan II-905
 Chun-Guang, Zhou III-448
 Chung, Chung-Yu II-785
 Chung, TaeChoong I-704
 Cichocki, Andrzej II-1032, III-793
 Cruz, Francisco II-391
 Cruz-Meza, María Elena III-828
 Cuadros-Vargas, Ernesto II-620
 Cubillos, Francisco A. I-311, II-391
 Cui, Baotong I-935
 Cui, Baoxia II-160
 Cui, Peiling III-597

 da Silva Soares, Anderson III-1024
 Dai, Dao-Qing II-1081
 Dai, Jing III-607

 Dai, Ruwei I-1280
 Dai, Shaosheng II-640
 Dai, Xianzhong II-196, III-1138
 Dakuo, He III-330
 Davies, Anthony II-938
 Dell'Orco, Mauro I-562
 Deng, Fang'an I-796
 Deng, Qiuxiang II-575
 Deng, Shaojiang II-724
 Dengsheng, Zhu III-1043
 Dillon, Tharam S. II-965
 Ding, Gang III-66
 Ding, Mingli I-667, III-721
 Ding, Mingwei II-956
 Ding, Mingyong II-1048
 Ding, Xiao-qing III-1033
 Ding, Xiaoshuai III-117
 Ding, Xiaoyan II-40
 Dong, Jiyang I-776, III-914
 Dou, Fuping I-480
 Du, Ji-Xiang I-1153, II-793, II-819
 Du, Junping III-80
 Du, Lan I-1303
 Du, Wei I-652
 Du, Xin I-714
 Du, Xin-Wei III-1130
 Du, Yina III-9
 Du, Zhi-gang I-465
 Duan, Hua III-812
 Duan, Lijuan II-851
 Duan, Yong II-160
 Duan, Zhemin III-943
 Duan, Zhuohua I-743

 El-Bakry, Hazem M. III-764
 Eтчells, T.A. II-1177

 Fan, Fuling III-416
 Fan, Huaiyu II-457
 Fan, Liping II-1042
 Fan, Shao-hui II-994
 Fan, Yi-Zheng I-572
 Fan, Youping III-1176
 Fan, Yushun I-609
 Fang, Binxing I-1286
 Fang, Jiancheng III-597
 Fang, Shengle III-292
 Fang, Zhongjie III-237
 Fei, Minrui II-483
 Fei, Shumin I-81, I-222

- Feng, Chunbo III-261
 Feng, Deng-Chao III-869
 Feng, Hailiang III-933
 Feng, Jian III-715
 Feng, Xiaoyi II-135
 Feng, Yong II-947
 Feng, Yue I-424
 Ferreira, Tiago A.E. II-602
 Florez-Choque, Omar II-620
 Freeman, Walter J. I-685
 Fu, Chaojin III-123
 Fu, Jiakai II-346
 Fu, Jun I-685
 Fu, Lihua I-632
 Fu, Mingang III-204
 Fu, Pan II-293
 Fuli, Wang III-330
 Fyfe, Colin I-397
- Gan, Woonseng I-176
 Gao, Chao III-35
 Gao, Cunchen I-910
 Gao, Jian II-640
 Gao, Jinwu II-257
 Gao, Junbin II-680
 Gao, Liang III-204
 Gao, Liqun II-931, III-846
 Gao, Ming I-935
 Gao, Shaoxia III-35
 Gao, Song II-424
 Gao, Wen II-851
 Gao, Zengan III-741
 Gao, Zhi-Wei I-519
 Gao, Zhifeng I-875
 Gardner, Andrew B. II-1273
 Gasso, Gilles III-486
 Ge, Baoming I-138, III-73
 Ge, Junbo II-1125
 Geng, Guanggang I-1280
 Ghannouchi, Fadhel M. I-582
 Glantschnig, Paul II-1115
 Gong, Shenguang III-672
 Grossberg, Stephen I-1094
 Gu, Hong II-1
 Gu, Ying-kui I-553, II-275
 Guan, Peng I-449, II-671
 Guan, Zhi-Hong II-8, II-113
 Guirimov, B.G. II-307
 Guo, Chengan III-461
- Guo, Chenlei I-723
 Guo, Lei I-93, I-1054
 Guo, Li I-1286, II-931, III-846
 Guo, Ling III-434
 Guo, Peng III-633, III-950
 Guo, Ping II-474
 Guo, Qi I-904
 Guo, Wensheng III-80
 Guo, Xin II-1291
- Hadzic, Fedja II-965
 Haifeng, Sang III-330
 Halgamuge, Saman K II-801, II-1221,
 III-1087
 Hamaguchi, Kosuke I-926
 Han, Feng II-740
 Han, Fengqing I-1104
 Han, Jianda III-589
 Han, Jiu-qiang II-646
 Han, Min II-569
 Han, Mun-Sung I-1318
 Han, Pu III-545
 Han, Risheng II-705
 Han, SeungSoo III-246
 Hao, Yuelong I-102
 Hao, Zhifeng I-8
 Hardy, David II-801
 He, Fen III-973
 He, Guoping III-441, III-812
 He, Haibo I-413, I-441
 He, Huaifeng I-203
 He, Lihong I-267
 He, Naishuai II-772
 He, Qing III-336
 He, Tingting I-632
 He, Xin III-434
 He, Xuewen II-275
 He, Yigang III-570, III-860, III-1006
 He, Zhaoshui II-1032
 He, Zhenya III-374
 Heng, Yue III-561
 Hirasawa, Kotaro I-403
 Hoang, Minh-Tuan T. I-1077
 Hong, Chin-Ming I-45
 Hong, SangJeen III-246
 Hong, Xia II-516, II-699
 Hope, A.D. II-293
 Hou, Weizhen III-812
 Hou, Xia I-1247
 Hou, Zeng-Guang II-438

- Hsu, Arthur III-1087
 Hsu, Chia-Chang III-1145
 Hu, Chengquan I-652, II-1264
 Hu, Dewen I-1061
 Hu, Haifeng II-630
 Hu, Jing II-985
 Hu, Jinglu I-403
 Hu, Jingtao III-277
 Hu, Meng I-685
 Hu, Ruifen I-685
 Hu, Sanqing II-1273
 Hu, Shiqiang III-950
 Hu, Shou-Song I-1247
 Hu, Wei III-277
 Hu, Xiaolin III-194
 Hu, Xuelei I-1211
 Hu, Yun-an II-47
 Huaguang, Zhang III-561
 Huang, Benxiong I-1336, III-626
 Huang, D. II-1002
 Huang, Dexian III-219
 Huang, Fu-Kuo III-57
 Huang, Hong III-933
 Huang, Hong-Zhong III-267
 Huang, Jikun III-1058
 Huang, Kai I-1183
 Huang, Liangli III-407
 Huang, Liyu II-1202
 Huang, Peng II-593
 Huang, Qingbao III-1097
 Huang, Tingwen II-24
 Huang, Xinsheng III-853
 Huang, Xiyue II-772, III-553
 Huang, Yanxin II-1264
 Huang, Yuancan III-320
 Huang, Yuchun I-1336, III-626
 Huang, Zailu I-1336
 Huang, Zhen I-733, I-824
 Huffel, S. Van II-1177
 Huo, Linsheng III-1182
 Hussain, Abir Jaafar II-921
 Huynh, Hieu T. I-1077
 Hwang, Chi-Pan III-998
 Hwang, Seongseob II-880

 Imamura, Takashi II-542
 Irwin, George W. I-496
 Isahara, Hitoshi I-1310
 Islam, Md. Monirul II-562
 Iwamura, Kakuzo II-257

 Jarur, Mary Carmen II-1150
 Je, Sung-Kwan III-923
 Ji, Geng I-166
 Ji, Guori III-545
 Jia, Hongping I-257, I-642
 Jia, Huading III-497
 Jia, Peifa I-852, II-328
 Jia, Yunde II-896
 Jia, Zhao-Hong I-572
 Jian, Feng III-561
 Jian, Jigui II-143
 Jian, Shu III-147
 Jian-yu, Wang III-448
 Jiang, Chang-sheng I-112
 Jiang, Chenwei II-1133
 Jiang, Haijun I-1008
 Jiang, Minghui I-952, III-292
 Jiang, Nan III-1
 Jiang, Tiejun III-350
 Jiang, Yunfei II-474
 Jiang, Zhe III-589
 Jiao, Li-cheng II-120
 Jin, Bo II-510
 Jin, Huang II-151
 Jin, Xuexiang II-1022
 Jin, Yihui III-219, III-1058
 Jin-xin, Tian III-49
 Jing, Chunguo III-1107
 Jing, Zhongliang II-705
 Jinhai, Liu III-561
 JiuFen, Zhao III-834
 Jo, Taeho I-1201, II-871
 José Coelho, Clarimar III-1024
 Ju, Chunhua III-392
 Ju, Liang I-920, I-1054
 Ju, Minseong III-140
 Jun, Ma I-512
 Jun, Yang III-981
 Junfeng, Xu III-17
 Jung, Byung-Wook III-641
 Jung, Young-Giu I-1318

 Kanae, Shunshoku I-275, II-1194
 Kang, Jingli III-157
 Kang, Mei I-257
 Kang, Min-Jae I-1015
 Kang, Sangki II-1240
 Kang, Y. III-580
 Kao, Tzu-Ping II-336
 Kao, Yonggui I-910

- Kaynak, Okyay I-14
 Ke, Hai-Sen I-285
 Kelleher, Dermot II-938
 Kil, Rhee Man I-1117, I-1318
 Kim, Byungwhan I-602
 Kim, Dae Young III-368
 Kim, Dongjun II-1187
 Kim, DongSeop III-246
 Kim, Ho-Chan I-1015
 Kim, Ho-Joon II-715
 Kim, HyunKi II-206
 Kim, Jin Young II-1240
 Kim, Kwang-Baek II-756, III-923
 Kim, Kyeongseop II-1187
 Kim, Pyo Jae III-506
 Kim, Seoksoo II-1090, III-140
 Kim, Sungshin III-641
 Kim, Tai-hoon III-140
 Kim, Woo-Soon III-1114
 Kim, Yong-Kab III-1114
 Kim, Yountae III-641
 Ko, Hee-Sang I-1015
 Konakoglu, Ekrem I-14
 Koo, Imhoi I-1117
 Kozloski, James II-500, II-552
 Kumar, R. Pradeep II-1012
 Kurnaz, Sefer I-14

 Lai, Pei Ling I-397
 Lee, Ching-Hung I-38, II-317
 Lee, Geehyuk II-104
 Lee, InTae II-206
 Lee, Jeongwhan II-1187
 Lee, Jin-Young III-923
 Lee, Joseph S. II-715
 Lee, Junghoon I-1015
 Lee, Malrey I-1201, II-871
 Lee, Seok-Lae I-1045
 Lee, SeungGwan I-704
 Lee, Shie-Jue III-515
 Lee, SungJoon III-246
 Lee, Tsai-Sheng I-694
 Lee, Yang Weon III-1192
 Leu, Yih-Guang I-45
 Leung, Kwong-Sak II-371
 Li, Ang II-689
 Li, Bin I-767, I-1087
 Li, Chuandong II-24
 Li, Chun-hua III-382
 Li, Demin III-695
 Li, Guang I-685
 Li, Haibin I-994
 Li, Hailong III-9
 Li, Haisheng II-414
 Li, Hongnan III-1182
 Li, Hongru III-9
 Li, Ji III-686
 Li, Jianwei III-933
 Li, Jing II-47, II-656
 Li, Jiuxian II-889, III-392
 Li, Jun I-676
 Li, Jun-Bao II-905
 Li, Kang I-496, II-483
 Li, Li I-132
 Li, Liming III-407
 Li, Meng II-842, III-1077
 Li, Minqiang II-448
 Li, Ping II-33
 Li, Qing II-251
 Li, Qingdu II-96
 Li, Qingguo II-424
 Li, Qiudan I-1280
 Li, San-ping III-382
 Li, Shaoyuan I-505
 Li, Shutao III-407
 Li, Tao I-81, I-93, I-374, II-8
 Li, Weidong III-147
 Li, Weimin III-988
 Li, Xiao-Li I-87
 Li, Xiaodong I-176
 Li, Xiaomei II-170
 Li, Xiaou I-487, II-483
 Li, Xiuxiu I-796
 Li, Xue III-741
 Li, Yan II-1281
 Li, Yang I-1286
 Li, Yangmin I-757, I-813
 Li, Yanwen II-842
 Li, Yaobo II-1056
 Li, Yi II-612
 Li, Yibin I-1087
 Li, Yinghong I-424
 Li, Yong-Wei III-633
 Li, Yongming I-1
 Li, Yongwei III-950
 Li, Yuan III-1130
 Li, Yue I-414
 Li, Yufei I-424
 Li, Yunxia III-758
 Li, Zhengxue III-117

- Li, Zhiquan III-311
 Lian, Qiusheng III-454
 Lian, Shiguo II-79
 Liang, Dong I-572
 Liang, Hua I-618, I-920, III-399
 Liang, Huawei I-843
 Liang, Jinling II-33
 Liang, Rui II-257
 Liang, Yanchun I-8, I-652, II-1264
 Liang, Yong II-371
 Liao, Longtao I-505
 Liao, Wudai I-897, III-164
 Liao, X.H. I-70, I-786
 Liao, Xiaofeng I-1104, II-724
 Liao, Xiaoxin I-897, II-143,
 III-164, III-292
 Lim, Jun-Seok II-398, III-678
 Lin, Chuan Ku III-998
 Lin, Hong-Dar II-785
 Lin, ShiehShing III-231
 Lin, Sida I-968
 Lin, Xiaofeng III-1097
 Lin, Yaping II-1254
 Lin, Yu-Ching II-317
 Lin, Yunsong II-1048
 Lin, Zhiling I-380
 Ling, Zhuang III-213
 Linhui, Cai II-151
 Lisboa, P.J.G II-1177
 Liu, Benyong II-381
 Liu, Bin III-1107
 Liu, Bo III-219, III-1058
 Liu, Derong I-387, II-1299
 Liu, Di-Chen III-1130
 Liu, Dianting II-740
 Liu, Dongmei II-1231
 Liu, Fei III-1067
 Liu, Guangjun II-251
 Liu, Guohai I-257, I-642
 Liu, Hongwei I-1303
 Liu, Hongwu III-686
 Liu, Ji-Zhen II-179
 Liu, Jilin I-714
 Liu, Jin III-751
 Liu, JinCun I-148
 Liu, Jinguo I-767
 Liu, Ju II-1065
 Liu, Jun II-772
 Liu, Lu III-1176
 Liu, Meiqin I-968
 Liu, Meirong III-570
 Liu, Peipei I-1069
 Liu, Qiuge III-336
 Liu, Shuhui I-480
 Liu, Taijun I-582
 Liu, Wen II-57
 Liu, Wenhui III-721
 Liu, Xiang-Jie II-179
 Liu, Xiaohe I-176
 Liu, Xiaohua III-751
 Liu, Xiaomao II-680
 Liu, Yan-Kui II-267
 Liu, Ying II-267, III-1058
 Liu, Yinyin II-534, II-956
 Liu, Yongguo III-237
 Liu, Yun III-1155
 Liu, Yunfeng I-203
 Liu, Zengrong II-16
 Liu, Zhi-Qiang II-267
 Liu, Zhongxuan II-79
 Lloyd, Stephen R. II-1299
 Löfberg, Johan I-424
 Long, Aideen II-938
 Long, Fei I-292
 Long, Jinling I-1110
 Long, Ying III-1006
 Loosli, Gaëlle III-486
 López-Yáñez, Itzamá II-835, III-828
 Lu, Bao-Liang I-1310, III-525
 Lu, Bin II-224
 Lu, Congde II-1048
 Lu, Hong tao III-967
 Lu, Huiling I-796
 Lu, Wenlian I-1034
 Lu, Xiaoqing I-986, I-1193
 Lu, Xinguo II-1254
 Lu, Yinghua II-842
 Lu, Zhiwu I-986, I-1193
 Luan, Xiaoli III-1067
 Lum, Kok Siong II-346
 Luo, Qi I-170, II-301
 Luo, Siwei II-1281
 Luo, Wen III-434
 Luo, Yan III-302
 Luo, Yirong I-455
 Lv, Guofang I-618
 Ma, Chengwei III-973
 Ma, Enjie II-362
 Ma, Fumin I-658

- Ma, Honglian III-461
 Ma, Jiachen III-840, III-877
 Ma, Jianying II-1125
 Ma, Jieming II-1133
 Ma, Jinwen I-1183, I-1227
 Ma, Liyong III-840, III-877
 Ma, Shugen I-767
 Ma, Xiaohong II-40, III-751
 Ma, Xiaolong I-434
 Ma, Xiaomin III-1
 Ma, Yufeng III-672
 Ma, Zezhong III-933
 Ma, Zhiqiang III-1077
 Mahmood, Ashique II-562
 Majewski, Maciej III-1049
 Mamedov, Fakhreddin II-241
 Mao, Bing-yi I-867, III-454
 Marwala, Tshilidzi I-1237, I-1293
 Mastorakis, Nikos III-764
 Mastronardi, Giuseppe II-1107
 Matsuka, Toshihiko I-1135
 May, Gary S. III-246
 Mei, Tao I-843
 Mei, Xue II-889
 Mei, Xuehui I-1008
 Men, Jiguan III-1176
 Meng, Hongling II-88, III-821
 Meng, Max Q.-H. I-843
 Meng, Xiangping II-493
 Menolascina, Filippo II-1107
 Miao, Jun II-851
 Min, Lequan III-147
 Mingzeng, Dai III-663
 Miyake, Tetsuo II-542
 Mohler, Ronald R. I-183
 Mora, Marco II-1150
 Moreno, Vicente II-391
 Moreno-Armendariz, Marco I-487
 Musso, Cosimo G. de II-1107

 Na, Seung You II-1240
 Nagabhushan, P. II-1012
 Nai-peng, Hu III-49
 Nan, Dong I-1110
 Nan, Lu III-448
 Naval Jr., Prospero C. III-174
 Navalertporn, Thitipong III-252
 Nelwamondo, Fulufhelo V. I-1293
 Ng, S.C. II-664
 Ngo, Anh Vien I-704

 Nguyen, Hoang Viet I-704
 Nguyen, Minh Nhut II-346
 Ni, Junchao I-158
 Nian, Xiaoling I-1069
 Nie, Xiaobing I-958
 Nie, Yalin II-1254
 Niu, Lin I-465

 Oh, Sung-Kwun II-186, II-206, III-225
 Ong, Yew-Soon I-1327
 Ortiz, Floriberto I-487
 Ou, Fan II-740
 Ou, Zongying II-740

 Pan, Jeng-Shyang II-905
 Pan, Jianguo II-352
 Pan, Li-Hu II-656
 Pan, Quan II-424
 Pandey, A. III-246
 Pang, Zhongyu II-1299
 Park, Aaron II-1240
 Park, Cheol-Sun III-368
 Park, Cheonshu II-730
 Park, Choong-shik II-756
 Park, Dong-Chul III-105, III-111
 Park, Jong Goo III-1114
 Park, Sang Kyoon I-1318
 Park, Yongsu I-1045
 Pavesi, Leopoldo II-1150
 Peck, Charles II-500, II-552
 Pedone, Antonio II-1107
 Pedrycz, Witold II-206
 Pei, Wenjiang III-374
 Peng, Daogang I-302
 Peng, Jian-Xun II-483
 Peng, Jinzhu I-592, I-804
 Peng, Yulou III-860
 Pi, Yuzhen II-493
 Pian, Zhaoyu II-931, III-846
 Piazza, Francesco III-731, III-783
 Ping, Ling III-448
 Pizzileo, Barbara I-496
 Pu, Xiaorong III-237

 Qi, Juntong III-589
 Qian, Jian-sheng II-120
 Qian, Juying II-1125
 Qian, Yi II-689
 Qianhong, Lu III-981
 Qiao, Chen III-131
 Qiao, Qingli II-72

- Qiao, Xiao-Jun III-869
 Qing, Laiyun II-851
 Qingzhen, Li III-834
 Qiong, Bao I-536
 Qiu, Jianlong I-1025
 Qiu, Jiqing I-871
 Qiu, Zhiyong III-914
 Qu, Di III-117
 Quan, Gan II-151
 Quan, Jin I-64

 Rao, A. Ravishankar II-500, II-552
 Ren, Dianbo I-890
 Ren, Guanghui II-765, III-651
 Ren, Quansheng II-88, III-821
 Ren, Shi-jin II-216
 Ren, Zhen II-79
 Ren, Zhiliang II-1056
 Rivas P., Pablo III-884
 Roh, Seok-Beom II-186
 Rohatgi, A. III-246
 Román-Godínez, Israel II-835
 Ronghua, Li III-657
 Rosa, João Luís Garcia II-825
 Rossini, Michele III-731
 Rubio, Jose de Jesus I-1173
 Ryu, Joung Woo II-730

 Sakamoto, Yasuaki I-1135
 Sánchez-Garfias, Flavio Arturo III-828
 Sasakawa, Takafumi I-403
 Savage, Mandara III-1165
 Sbarbaro, Daniel II-1150
 Schikuta, Erich II-1115
 Senaratne, Rajinda II-801
 Seo, Ki-Sung III-225
 Serebinski, Franciszek III-85
 Shang, Li II-810
 Shang, Yan III-454
 Shao, Xinyu III-204
 Sharmin, Sadia II-562
 Shen, Jinyuan II-457
 Shen, Lincheng I-1061
 Shen, Yanjun I-904
 Shen, Yehu I-714
 Shen, Yi I-952, III-292, III-840, III-877
 Shen, Yue I-257, I-642
 Sheng, Li I-935
 Shi, Haoshan III-943
 Shi, Juan II-346
 Shi, Yanhui I-968

 Shi, Zhongzhi III-336
 Shiguang, Luo III-803
 Shin, Jung-Pil III-641
 Shin, Sung Hwan III-111
 Shunlan, Liu III-663
 Skaruz, Jaroslaw III-85
 Sohn, Joo-Chan II-730
 Song, Chunning III-1097
 Song, David Y. I-70
 Song, Dong Sung III-506
 Song, Jaegu II-1090
 Song, Jinya I-618, III-479
 Song, Joo-Seok I-1045
 Song, Kai I-671, III-721
 Song, Qiankun I-977
 Song, Shaojian III-1097
 Song, Wang-Cheol I-1015
 Song, Xiao xiao III-1097
 Song, Xuelei II-746
 Song, Y.D. I-786
 Song, Yong I-1087
 Song, Young-Soo III-105
 Song, Yue-Hua III-426
 Song, Zhuo II-1248
 Sousa, Robson P. de II-602
 Squartini, Stefano III-731, III-7783
 Starzyk, Janusz A. I-413, I-441,
 II-534, II-956
 Stead, Matt II-1273
 Stuart, Keith Douglas III-1049
 Sun, Bojiao I-1346
 Sun, Changcun II-1056
 Sun, Changyin I-618, I-920, III-479
 Sun, Fangxun I-652
 Sun, Fuchun I-132
 Sun, Haiqin I-319
 Sun, Jiande II-1065
 Sun, Lei I-843
 Sun, Lisha II-1168
 Sun, Pei-Gang II-234
 Sun, Qiuye III-607
 Sun, Rongrong II-284
 Sun, Shixin III-497
 Sun, Xinghua II-1065
 Sun, Youxian II-1097, II-1140
 Sun, Z. I-786
 Sung, KoengMo II-398

 Tan, Ah-Hwee I-1094
 Tan, Cheng III-1176

- Tan, Hongli III-853
 Tan, Min II-438, III-1155
 Tan, Yanghong III-570
 Tan, Ying III-705
 Tan, Yu-An II-301
 Tang, Guiji III-545
 Tang, GuoFeng II-465
 Tang, Jun I-572
 Tang, Lixin II-63
 Tang, Songyuan II-979
 Tang, Wansheng III-157
 Tang, Yuchun II-510
 Tang, Zheng II-465
 Tao, Liu I-512
 Tao, Ye III-267
 Testa, A.C. II-1177
 Tian, Fengzhan II-414
 Tian, GuangJun I-148
 Tian, Jin II-448
 Tian, Xingbin I-733
 Tian, Yudong I-213
 Tie, Ming I-609
 Timmerman, D. II-1177
 Tong, Ling II-1048
 Tong, Shaocheng I-1
 Tong, Weiming II-746
 Tsai, Hsiu Fen III-357
 Tsai, Hung-Hsu III-904
 Tsao, Teng-Fa I-694
 Tu, Zhi-Shou I-358

 Uchiyama, Masao I-1310
 Uyar, K. II-307

 Vairappan, Catherine II-465
 Valentin, L. II-1177
 Vanderaa, Bill II-801
 Veludo de Paiva, Maria Stela III-1024
 Vilakazi, Christina B. I-1237
 Vo, Nguyen H. I-1077
 Vogel, David II-534
 Volkov, Yuri II-938

 Wada, Kiyoshi I-275, II-1194
 Wan, Yuanyuan II-819
 Wang, Baoxian II-143
 Wang, Bin II-1133
 Wang, Bolin III-399
 Wang, C.C. III-580
 Wang, Chunheng I-1280
 Wang, Dacheng I-1104
 Wang, Dongyun I-897
 Wang, Fu-sheng I-241, I-249
 Wang, Fuli I-380
 Wang, Fuliang I-257
 Wang, Gaofeng I-632
 Wang, Grace S. III-57
 Wang, Guang-Jiang III-774
 Wang, Guoqiang II-740
 Wang, Haijun II-1254
 Wang, Haila II-79
 Wang, Hong I-93
 Wang, Hongbo I-733, I-824
 Wang, Honghui II-1097
 Wang, Hongrui I-749
 Wang, Hongwei II-1
 Wang, Jiacun III-695
 Wang, Jiahai III-184
 Wang, Jian III-821
 Wang, Jianzhong II-493
 Wang, Jingming I-1087
 Wang, Jue II-1202
 Wang, Jun III-95, III-194
 Wang, Jun-Song I-519
 Wang, Kai II-406
 Wang, Ke I-834
 Wang, Kuanquan II-583
 Wang, Kun II-931, III-846
 Wang, Lan III-416
 Wang, Le I-1183
 Wang, Lei I-29, III-497
 Wang, Liangliang I-1227
 Wang, Ling III-219, III-1058
 Wang, Lipo II-57
 Wang, Meng-Hui III-1145
 Wang, Nian I-572
 Wang, Qi III-721
 Wang, Qin II-689
 Wang, Qingren II-406
 Wang, Quandi I-528
 Wang, Rubin I-1127
 Wang, Ruijie III-80
 Wang, Sheng-jin III-1033
 Wang, Shiwei I-122
 Wang, Shoujue III-616
 Wang, Shoulin I-920
 Wang, Shufeng II-352
 Wang, Shuqin I-652
 Wang, Shuzong III-350
 Wang, Tian-Zhen II-985
 Wang, Tianmiao III-535

- Wang, Wei I-834, III-535
 Wang, Weiqi II-1125
 Wang, Xiang-ting II-120
 Wang, Xiaohong I-952
 Wang, Xiaohua III-860, III-1006
 Wang, Xihuai I-658
 Wang, Xin II-196
 Wang, Xiuhong II-72
 Wang, Xiumei I-652
 Wang, Xiuqing III-1155
 Wang, Xiuxiu II-612
 Wang, Xuelian II-1202
 Wang, Xuexia II-765
 Wang, XuGang II-465
 Wang, Yan I-652, II-1264
 Wang, Yaonan I-592, I-804, III-469
 Wang, Yen-Nien I-694
 Wang, Yong I-22
 Wang, Yongtian II-979
 Wang, Yuan I-852, II-328
 Wang, Yuan-Yuan II-284, II-819,
 II-1125, III-426
 Wang, Yuechao I-767
 Wang, Zeng-Fu I-1153
 Wang, Zhancheng III-988
 Wang, Zhaoxia II-1231
 Wang, Zhen-Yu III-633
 Wang, Zhihai II-414
 Wang, Zhiliang II-251
 Wang, Zuo II-301
 Wei, Guoliang III-695
 Wei, Hongxing III-535
 Wei, Miaomiao II-612
 Wei, Qinglai I-387
 Wei, Ruxiang III-350
 Wei, Wei I-292
 Wei, Xunkai I-424
 Wei, Yan Hao III-998
 Weiqi, Yuan III-330
 Wen, Bangchun I-29
 Wen, Cheng-Lin I-319, II-994, II-985,
 III-774
 Wen, Chuan-Bo III-774
 Wen, Lei III-284
 Wen, Shu-Huan I-863
 Wen, Yi-Min III-525
 Weng, Liguó I-74
 Weng, Shilie I-213
 Wenjun, Zhang I-512
 Wickramarachchi, Nalin II-1221
 Won, Yonggwan I-1077, II-1240
 Wong, Hau-San III-894
 Wong, Stephen T.C. II-1097, II-1140
 Woo, Dong-Min III-105
 Woo, Seungjin II-1187
 Woo, Young Woon II-756
 Worrell, Gregory A. II-1273
 Wu, Aiguo I-380
 Wu, Bao-Gui III-267
 Wu, Gengfeng II-352
 Wu, Jianbing I-642
 Wu, Jianhua I-267, II-931, III-846
 Wu, Kai-gui II-947
 Wu, Ke I-1310
 Wu, Lingyao I-1054
 Wu, Qiang I-473
 Wu, Qing-xian I-112
 Wu, Qingming I-231
 Wu, Si I-926
 Wu, TiHua I-148
 Wu, Wei I-1110, III-117
 Wu, Xianyong II-8, II-113
 Wu, Xingxing II-170
 Wu, You-shou III-1033
 Wu, Yunfeng II-664
 Wu, Yunhua III-1058
 Wu, Zhengping II-113
 Wu, Zhilu II-765, III-651
 Xi, Guangcheng I-1274, II-1159
 Xia, Jianjin I-64
 Xia, Liangzheng II-889, III-392
 Xia, Siyu III-392
 Xia, Youshen III-95
 Xian-Lun, Tang III-213
 Xiang, C. II-1002
 Xiang, Changcheng III-553
 Xiang, Hongjun I-941
 Xiang, Lan II-16
 Xiang, Yanping I-368
 Xiao, Deyun II-1072
 Xiao, Gang II-705
 Xiao, Jianmei I-658
 Xiao, Jinzhuang I-749
 Xiao, Min I-958
 Xiao, Qinkun II-424
 Xiaoli, Li III-1043
 Xiaoyan, Ma III-981
 Xie, Haibin I-1061
 Xie, Hongmei II-135

- Xing, Guangzhong III-1107
 Xing, Jie II-1072
 Xing, Yanwei I-1274
 Kingsheng, Gu I-536
 Xiong, Guangze III-1015
 Xiong, Min III-1176
 Xiong, RunQun II-465
 Xiong, Zhong-yang II-947
 Xu, Chi I-626
 Xu, De III-1155
 Xu, Dongpo III-117
 Xu, Guoqing III-988
 Xu, Hong I-285
 Xu, Hua I-852, II-328
 Xu, Huiling I-319
 Xu, Jian III-1033
 Xu, Jianguo I-807
 Xu, Jing I-358
 Xu, Jiu-Qiang II-234
 Xu, Ning-Shou I-519
 Xu, Qingsong I-757
 Xu, Qinzheng III-374
 Xu, Shuang II-896
 Xu, Shuhua I-1336, III-626
 Xu, Shuxiang I-1265
 Xu, Xiaoyun II-1291
 Xu, Xin I-455
 Xu, Xinhe I-267, II-160
 Xu, Xinzheng II-913
 Xu, Xu I-8
 Xu, Yang II-1042
 Xu, Yangsheng III-988
 Xu, Yulin III-164
 Xu, Zong-Ben II-371, III-131
 Xue, Xiaoping I-879
 Xue, Xin III-441
 Xurong, Zhang III-834

 Yan, Gangfeng I-968
 Yan, Hua II-1065
 Yan, Jianjun III-959
 Yan, Qingxu III-616
 Yáñez-Márquez, Cornelio II-835,
 III-828
 Yang, Guowei III-616
 Yang, Hongjiu I-871
 Yang, Hyun-Seung II-715
 Yang, Hong-yong I-241, I-249
 Yang, Jiaben I-183, I-193
 Yang, Jingming I-480

 Yang, Jiyun II-724
 Yang, Jun I-158
 Yang, Kuihe III-342
 Yang, Lei II-646
 Yang, Luxi III-374
 Yang, Ming II-842
 Yang, Peng II-1291
 Yang, Ping I-302
 Yang, Wei III-967
 Yang, Xiao-Song II-96
 Yang, Xiaogang I-203
 Yang, Xiaowei I-8
 Yang, Yingyu I-1211
 Yang, Yixian III-1
 Yang, Yongming I-528
 Yang, Yongqing II-33
 Yang, Zhao-Xuan III-869
 Yang, Zhen-Yu I-553
 Yang, Zhi II-630
 Yang, Zhi-Wu I-1162
 Yang, Zhuo II-1248
 Yang, Zi-Jiang I-275, II-1194
 Yang, Zuyuan III-553
 Yanxin, Zhang III-17
 Yao, Danya II-1022
 Ye, Bin II-656
 Ye, Chun-xiao II-947
 Ye, Mao III-741
 Ye, Meiyong II-127
 Ye, Yan I-582
 Ye, Zhiyuan I-986, I-1193
 Yeh, Chi-Yuan III-515
 Yi, Gwan-Su II-104
 Yi, Jianqiang I-349, I-358, I-368,
 I-374, I-1274
 Yi, Tinghua III-1182
 Yi, Yang I-93
 Yi, Zhang I-1001, II-526, III-758
 Yin, Fuliang III-751
 Yin, Jia II-569
 Yin, Yixin II-251
 Yin, Zhen-Yu II-234
 Yin, Zheng II-1097
 Yin-Guo, Li III-213
 Ying, Gao III-17
 Yongjun, Shen I-536
 Yu, Changrui III-302
 Yu, Chun-Chang II-336
 Yu, D.L. II-432
 Yu, D.W. II-432

- Yu, Ding-Li I-122, I-339
 Yu, Haocheng II-170
 Yu, Hongshan I-592, I-804
 Yu, Jian II-414
 Yu, Jiaxiang II-1072
 Yu, Jin-Hua III-426
 Yu, Jinxia I-743
 Yu, Miao II-724
 Yu, Wen I-487, I-1173, II-483
 Yu, Wen-Sheng I-358
 Yu, Xiao-Fang III-633
 Yu, Yaoliang I-449, II-671
 Yu, Zhiwen III-894
 Yuan, Chongtao III-461
 Yuan, Dong-Feng I-22
 Yuan, Hejin I-796
 Yuan, Quande II-493
 Yuan, Xiaofang III-469
 Yuan, Xudong III-35
 Yue, Dongxue III-853
 Yue, Feng II-583
 Yue, Heng III-715
 Yue, Hong I-329
 Yue, Shihong II-612
 Yusiong, John Paul T. III-174

 Zang, Qiang III-1138
 Zdunek, Rafal III-793
 Zeng, Qingtian III-812
 Zeng, Wenhua II-913
 Zeng, Xiaoyun III-1077
 Zeng, Zhigang II-575
 Zhai, Chuan-Min II-793, II-819
 Zhai, Yu-Jia I-339
 Zhai, Yuzheng III-1087
 Zhang, Biyin II-861
 Zhang, Bo II-40
 Zhang, Chao III-545
 Zhang, Chenggong II-526
 Zhang, Daibing I-1061
 Zhang, Daoqiang II-778
 Zhang, Dapeng I-380
 Zhang, David II-583
 Zhang, Guo-Jun I-1153
 Zhang, Hao I-302
 Zhang, Huaguang I-387, III-715
 Zhang, Jianhai I-968
 Zhang, Jinfang I-329
 Zhang, Jing II-381
 Zhang, Jingdan II-1081
 Zhang, Jingtang I-102
 Zhang, Jinhui I-871
 Zhang, Jiqi III-894
 Zhang, Jiye I-890
 Zhang, Jun II-680
 Zhang, Jun-Feng I-1247
 Zhang, Junxiong III-973
 Zhang, Junying I-776
 Zhang, Kanjian III-261
 Zhang, Keyue I-890
 Zhang, Kun II-861
 Zhang, Lei I-1001, III-1077
 Zhang, Lijing I-910
 Zhang, Liming I-449, I-723,
 II-671, II-1133
 Zhang, M.J. I-70, I-786
 Zhang, Meng I-632
 Zhang, Ming I-1265
 Zhang, Ning II-1248
 Zhang, Pan I-1144
 Zhang, Pinzheng III-374
 Zhang, Qi II-1125
 Zhang, Qian III-416
 Zhang, Qiang I-231
 Zhang, Qizhi I-176
 Zhang, Shaohong III-894
 Zhang, Si-ying I-241, I-249
 Zhang, Su III-967
 Zhang, Tao II-1248
 Zhang, Tengfei I-658
 Zhang, Tianping I-81
 Zhang, Tianqi II-640
 Zhang, Tianxu II-861
 Zhang, Tieyan III-715
 Zhang, Wei II-656
 Zhang, Xi-Yuan II-234
 Zhang, Xiao-Dan II-234
 Zhang, Xiao-guang II-216
 Zhang, Xing-gan II-216
 Zhang, XueJian I-148
 Zhang, Xueping II-1291
 Zhang, Xueqin II-1211
 Zhang, Yan-Qing II-510
 Zhang, Yanning I-796
 Zhang, Yanyan II-63
 Zhang, Yaoyao I-968
 Zhang, Yi II-1022
 Zhang, Ying-Jun II-656
 Zhang, Yongqian III-1155
 Zhang, You-Peng I-676

- Zhang, Yu II-810
 Zhang, Yu-sen III-382
 Zhang, Yuxiao I-8
 Zhang, Zhao III-967
 Zhang, Zhaozhi III-1
 Zhang, Zhikang I-1127
 Zhang, Zhiqiang II-465
 Zhang, Zhong II-542
 Zhao, Bing III-147
 Zhao, Chunyu I-29
 Zhao, Dongbin I-349, I-368,
 I-374, I-1274
 Zhao, Fan II-216
 Zhao, Feng III-382
 Zhao, Gang III-553
 Zhao, Hai II-234
 Zhao, Hongming III-535
 Zhao, Jian-guo I-465
 Zhao, Jianye II-88, III-821
 Zhao, Lingling III-342
 Zhao, Nan III-651
 Zhao, Shuying I-267
 Zhao, Xingang III-589
 Zhao, Yaou II-1211
 Zhao, Yaqin II-765, III-651
 Zhao, Youdong II-896
 Zhao, Zeng-Shun II-438
 Zhao, Zhiqiang II-810
 Zhao, Zuopeng II-913
 Zheng, Chaixin II-938
 Zheng, Hongying II-724
 Zheng, Huiru I-403
 Zheng, Jianrong III-959
 Zheng, Xia II-1140
 Zhiping, Yu I-512
 Zhon, Hong-Jian I-45
 Zhong, Jiang II-947
 Zhong, Shisheng III-66
 Zhong, Ying-Ji I-22
 Zhongsheng, Hou III-17
 Zhou, Chunguang I-652, II-842,
 II-1264, III-1077
 Zhou, Donghua I-1346
 Zhou, Huawei I-257, I-642
 Zhou, Jianting I-977
 Zhou, Jie III-695
 Zhou, Jin II-16
 Zhou, Qingdong I-667
 Zhou, Tao I-796
 Zhou, Wei III-943
 Zhou, Xianzhong III-434
 Zhou, Xiaobo II-1097, II-1140
 Zhou, Xin III-943
 Zhou, Yali I-176
 Zhu, Chongjun III-123
 Zhu, Xun-lin I-241, I-249
 Zhu, Jie II-593
 Zhu, Jie (James) III-1165
 Zhu, Lin I-904
 Zhu, Qiguang I-749, III-311
 Zhu, Qing I-81
 Zhu, Si-Yuan II-234
 Zhu, Xilin II-170
 Zhu, Zexuan I-1327
 Zhuang, Yan I-834
 Zimmerman S., Alejandro III-884
 Zong, Chi I-231
 Zong, Ning II-516, II-699
 Zou, An-Min II-438
 Zou, Qi II-1281
 Zou, Shuxue II-1264
 Zuo, Bin II-47
 Zuo, Wangmeng II-583
 Zurada, Jacek M. I-1015
 Zuyuan, Yang III-803