

LNCS 6792

Timo Honkela
Włodzisław Duch
Mark Girolami
Samuel Kaski (Eds.)

Artificial Neural Networks and Machine Learning – ICANN 2011

21st International Conference
on Artificial Neural Networks
Espoo, Finland, June 2011, Proceedings, Part II

2
Part II



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

Alfred Kobsa

University of California, Irvine, CA, 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

TU Dortmund University, Germany

Madhu Sudan

Microsoft Research, Cambridge, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Gerhard Weikum

Max Planck Institute for Informatics, Saarbruecken, Germany

Timo Honkela Włodzisław Duch
Mark Girolami Samuel Kaski (Eds.)

Artificial Neural Networks and Machine Learning – ICANN 2011

21st International Conference
on Artificial Neural Networks
Espoo, Finland, June 14-17, 2011
Proceedings, Part II

Volume Editors

Timo Honkela
Aalto University School of Science
Department of Information and Computer Science
P.O. Box 15400, 00076 Aalto, Finland
E-mail: timo.honkela@aalto.fi

Włodzisław Duch
Nicolaus Copernicus University
School of Physics, Astronomy and Informatics
Department of Informatics
ul. Grudziadzka 5, 87-100 Toruń, Poland
E-mail: wduch@is.umk.pl

Mark Girolami
University College London
Department of Statistical Science
1-19 Torrington Place, London, WC1E 7HB, UK
E-mail: girolami@stats.ucl.ac.uk

Samuel Kaski
Aalto University School of Science
Department of Information and Computer Science
P.O. Box 15400, 00076 Aalto, Finland
E-mail: samuel.kaski@aalto.fi

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-21737-1 e-ISBN 978-3-642-21738-8
DOI 10.1007/978-3-642-21738-8
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011929230

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

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

© Springer-Verlag Berlin Heidelberg 2011

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.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

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

Printed on acid-free paper

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

Preface

The International Conference on Artificial Neural Networks (ICANN) is the annual flagship conference of the European Neural Network Society (ENNS). The idea of ICANN is to bring together researchers from two worlds: information sciences and neurosciences. The scope is wide, ranging from machine learning algorithms to models of real nervous systems. The aim is to facilitate discussions and interactions in the effort toward developing more intelligent artificial systems and increasing our understanding of neural and cognitive processes in the brain.

In 2011, ICANN returned to its roots after 20 years (for more information, see the ENNS website www.e-nns.org). The very first ICANN in 1991 was organized on the premises of Helsinki University of Technology on its beautiful campus in Espoo, Finland. For ICANN 2011, we invited all neural network researchers worldwide to join us in celebrating this 20th anniversary of ICANN and to see the latest advancements in our fast progressing field.

ICANN 2011 had two basic tracks: brain-inspired computing and machine learning research, with Program Committee chairs from both worlds and a thorough reviewing system. The conference structure was built around plenary talks given by renowned scientists described briefly in the following section.

- Thomas Griffiths (University of California, Berkeley, USA) is the Director of the Computational Cognitive Science Lab and the Institute of Cognitive and Brain Sciences at the University of California, Berkeley. They develop, for instance, mathematical models of higher level cognition, including probabilistic reasoning, learning causal relationships, acquiring and using language, and inferring the structure of categories.
- Riitta Hari (Aalto University, Finland) is an internationally recognized and respected neuroscientist. She was newly appointed as Academician of Science, a title that can be held by only 12 scientists at a time in Finland. She has developed methods and applications of human brain imaging and contributed decisively to the progress of this branch of science. Prof. Hari's current focus is on the brain basis of social interaction.
- Geoffrey Hinton (University of Toronto, Canada), the first winner of the David E. Rumelhart Prize, has provided many influential contributions to the area of artificial neural networks and adaptive systems. A non-exhaustive list of the areas where he has contributed substantial inventions includes back-propagation algorithm, Boltzmann machines, distributed representations, time-delay neural networks, and mixtures of experts. Prof. Hinton was the founding director of the Gatsby Computational Neuroscience Unit at University College London.
- Aapo Hyvärinen (University of Helsinki, Finland) is widely known for his contributions to the theory and applications of independent component analysis. His recent work also includes research on natural image statistics. He has

published in the major journals in the areas of neural networks and machine learning and his books have been translated into Japanese and Chinese.

- John Shawe-Taylor (University College London, UK) is Head of Department of Computer Science, and the scientific coordinator of the Network of Excellence in Pattern Analysis, Statistical Modelling and Computational Learning (PASCAL). His main research area is statistical learning theory, but his contributions range from neural networks and machine learning to graph theory. He is the co-author of two very successful books on the theory of support vector machines and kernel methods.
- Joshua Tenenbaum (Massachusetts Institute of Technology, USA) is a prominent researcher in the area of computational cognitive science. With his research group, he explores topics such as learning concepts, judging similarity, inferring causal connections, forming perceptual representations, and inferring mental states of other people.

A special plenary talk, shared with the co-located WSOM 2011, Workshop on Self-Organizing Maps, was given by Teuvo Kohonen, Academician of Science. He has introduced several new concepts to neural computing including theories of distributed associative memory and optimal associative mappings, the learning subspace method, the self-organizing maps (SOM), the learning vector quantization (LVQ), the adaptive-subspace SOM (ASSOM) in which invariant-feature filters emerge. Academician Teuvo Kohonen was the initiator and Chair of the first ICANN conference in 1991.

The technical program of ICANN 2011 consisted of 106 oral or poster presentations that highlighted key advances in the areas of neural networks and statistical machine learning research. The overall quality of the contributions can be considered high, also due to the high rejection rate. Approximately only every fourth submission was accepted to be presented orally in the conference. In addition to the regular conference sessions, one day was devoted to five workshops on topics related to theory and applications of brain-inspired computing and statistical machine learning. Two of the workshops were related to special challenges. A mind reading competition on MEG data was sponsored by the PASCAL network of excellence. The META-NET network of excellence sponsored a workshop on the use of context in machine translation.

The organizers had a chance to welcome the participants to the new Aalto University School of Science. From the beginning of 2010, the 100-year-old Helsinki University of Technology changed its name and form. It merged with Helsinki School of Economics and University of Art and Design Helsinki into Aalto University, becoming the second largest university in Finland. The conference was organized at Aalto University School of Science and the nearby Dipoli Congress Center. Both are centrally located in Otaniemi, Espoo, 15 minutes west of Helsinki. Otaniemi features a unique mix of world-class research organizations, academic institutions and over 800 companies from start-ups to multinational corporations operating around a compact campus. Otaniemi has been twice selected by the EU as one of the most innovative regions in Europe. It

is a community of over 32,000 people with 16,000 students and 16,000 technology professionals.

We warmly thank all the authors of the contributed papers, workshop organizers and presenters. We also gratefully acknowledge the contribution of the plenary speakers whose presentations formed the backbone of the conference. We express our gratitude to the highly respected international Area Chairs and members of the Program Committee whose role was instrumental for the success of the conference. The Area Chairs, Program Committee members and the reviewers ensured a timely and thorough evaluation of the papers.

We are grateful to the members of the Executive Committee whose contributions were essential in ensuring the successful organization of the conference. Erkki Oja as the General Chair led the conference organizations with his great experience. Amaury Lendasse, the Local Chair, kept all details of the organization under control. Mari-Sanna Paukkeri committed a lot of work to compile the proceedings. Ilari Nieminen took care of numerous details in the arrangements, especially related to the review process and compilation of the proceedings. Laura Kainulainen efficiently handled the matters related to registrations in collaboration with the Aalto University staff. Jaakko Peltonen, the Publicity Chair, made sure that the conference was announced in all major forums. Alexander Ilin took good care of the workshop organizations. Francesco Corona as the Finance Chair ensured that the budget stayed in balance. Yoan Miche contributed in several practical areas in the arrangements including the Web. Ricardo Vigário was responsible for the social program and, in particular, the arrangements of the conference dinner. Tommi Vatanen organized the activities of the conference assistants and helped in preparing the evening program.

We are grateful to Microsoft Research whose representatives provided us with free access to their Conference Management Tool and helped in setting up the system. Last but not least, we would like to thank Springer for their co-operation in publishing the proceedings in the prestigious *Lecture Notes in Computer Science* series.

April 2011

Timo Honkela
Włodzisław Duch
Mark Girolami
Samuel Kaski

Organization

ICANN 2011 was held during June 14–17, 2011, organized by the Department of Information and Computer Science, Aalto University School of Science. The head of the department is Professor Olli Simula, who is a long-term member of the ENNS Executive Committee.

Executive Committee

General Chair	Erkki Oja (Aalto University, Finland)
Program Chairs	Włodzisław Duch (Nicolaus Copernicus University, Poland)
	Mark Girolami (University College London, UK)
	Timo Honkela (Aalto University, Finland)
	Samuel Kaski (Aalto University, Finland)
Local Chair	Amaury Lendasse (Aalto University, Finland)
Publicity Chair	Jaakko Peltonen (Aalto University, Finland)

Area Chairs

Peter Auer (Austria)	Jan Peters (Germany)
Christian Bauckhage (Germany)	Marios Polycarpou (Cyprus)
Wray Buntine (Australia)	Jose Principe (USA)
Vince Calhoun (USA)	Volker Roth (Switzerland)
Antonius Coolen (UK)	Craig Saunders (UK)
Fernando Morgado Dias (Portugal)	Masashi Sugiyama (Japan)
Barbara Hammer (Germany)	Alan Stocker (USA)
Giulio Jacucci (Finland)	Ron Sun (USA)
Kristian Kersting (Germany)	Alfred Ultsch (Germany)
Neil Lawrence (UK)	Peter Tino (UK)
Te-Won Lee (USA)	Koen Van Leemput (USA)
Mikko Kurimo (Finland)	Michel Verleysen (Belgium)
Hiroshi Mamitsuka (Japan)	Jean-Philippe Vert (France)
Klaus-Robert Müller (Germany)	Ole Winther (Denmark)
Klaus Obermayer (Germany)	Chang Yoo (Korea)
Cheng Soon Ong (Switzerland)	

Program Committee and Referees

- Luis Alexandre (Portugal)
Paul Aljabar (UK)
Tanel Alumae (Estonia)
Mauricio Álvarez (UK)
Elisabeth Andre (Germany)
Alessia Annibale (UK)
Ana Antunes (Portugal)
Bruno Apolloni (Italy)
Ebru Arisoy (Turkey)
Beghdadi Azeddine (France)
Vineeth Balasubramaniam (USA)
Peter Battaglia (USA)
Luba Benuskova (New Zealand)
Ulysses Bernardet (Spain)
Monica Bianchini (Italy)
Ginestra Bianconi (USA)
Michael Biehl (The Netherlands)
Benjamin Blankertz (Germany)
Jamie Blundel (UK)
Antoine Bordes (Canada)
Abdeslam Boularias (Canada)
Mikio Braun (Germany)
Sebastien Bubeck (France)
Samuel Bulo (Italy)
Guido Caldarelli (Italy)
Cristina Campi (Finland)
Tonatiuh Peña Centeno (UK)
Michal Cernansky (Slovakia)
Jung-Hsien Chiang (Taiwan)
Jens Christian Claussen (Germany)
James Cussens (UK)
Jesse David (Belgium)
Lieven De Lathauwer (Belgium)
Andrea De Martino (Italy)
Yannick Deville (France)
Dotan Di Castro (Israel)
Jan Drugowitsch (France)
Lan Du (Australia)
Carl Ek (Sweden)
Deniz Erdogmus (USA)
Igor Farkas (Slovakia)
Jason Farquhar (The Netherlands)
Santo Fortunato (Italy)
Gernot Fink (Germany)
Benoit Frenay (Belgium)
Artu Garcez (UK)
Phil Garner (Switzerland)
Shuzhi Sam Ge (Singapore)
Mohammad Ghavamzadeh (France)
Fausto Giunchiglia (Italy)
Tobias Glasmachers (Switzerland)
Geoff Goodhill (Australia)
Dan Grollman (Switzerland)
Moritz Grosse-Wentrup (Germany)
Hirotaka Hachiya (Japan)
David Hardoon (UK)
Kohei Hatano (Japan)
Sebastien Helie (USA)
Ricardo Henao (Denmark)
James Hensman (UK)
Tom Heskes (The Netherlands)
Antti Honkela (Finland)
Naira Hovakimyan (USA)
Matthew Howard (UK)
Guang-Bin Huang (Singapore)
Zakria Hussain (UK)
Alexander Ilin (Finland)
Robert Jacob (USA)
Matjaz Jogan (USA)
Shivaram Kalyanakrishnan (USA)
Kittipat Kampa (USA)
Bert Kappen (The Netherlands)
Juha Karhunen (Finland)
Matthias Kaschube (USA)
Dietrich Klakow (Germany)
Stefanos Kollias (Greece)
Vera Kurkova (Hungary)
Ville Kyrki (Finland)
Jorma Laaksonen (Finland)
Harri Lahdesmäki (Finland)
Stéphane Lallich (France)
Georg Langs (France)
Jan Larsen (Denmark)
Robert Leeb (Switzerland)
Amaury Lendasse (Finland)
Weifeng Liu (USA)

Jyrki Lötjönen (Finland)	Mikkel Schmidt (Denmark)
Mantas Lukosevicius (Germany)	Gerwin Schalk (USA)
Zhiyuan Luo (UK)	Markus Schläpfer (Switzerland)
Mathew Magimai Doss (Switzerland)	Benoit Scherrer (USA)
Amir Madany Mamlouk (Germany)	Mikkel Schmidt (Denmark)
Danilo Mandic (UK)	Gabriele Schweikert (Germany)
Naoki Masuda (Japan)	Sambu Seo (Germany)
Bjoern Menze (USA)	Sohan Seth (USA)
Alessio Micheli (Italy)	Vesa Siivola (Finland)
Kazushi Mimura (Japan)	Sören Sonnenburg (Germany)
Tetsuro Morimura (Japan)	Emilio Soria-Olivas (Spain)
Morten Mørup (Denmark)	Alessandro Sperduti (Italy)
Alexander Mozeika (UK)	Jochen Steil (Germany)
Christian Müller (Germany)	Marc Strickert (Germany)
Shinichi Nakajima (Japan)	Taiji Suzuki (Japan)
Sriram Natarajan (USA)	Csaba Szepesvari (Canada)
Tim Nattkemper (Germany)	Ichiro Takeuchi (Japan)
Yizhao Ni (UK)	Michael Tangermann (Germany)
Matthias Nickles (UK)	Franck Thollard (France)
Nikolay Nikolaev (UK)	Christian Thurau (Germany)
Ann Nowe (Belgium)	Michalis Titsias (UK)
Ronald Ortner (Austria)	Jussi Tohka (Finland)
Erhan Oztop (Japan)	Ryota Tomioka (Japan)
Sebastian Pannasch (Germany)	Antonios Tsourdos (UK)
Ulrich Paquet (UK)	Koji Tsuda (Japan)
Il Park (USA)	Laurens van der Maaten (The Netherlands)
Emilio Parrado (Spain)	Carmen Vidaurre (Germany)
Elzbieta Pekalska (UK)	Ricardo Vigário (Finland)
Jaakko Peltonen (Finland)	Silvia Villa (Italy)
Andrew Philippides (UK)	Nathalie Villa-Vialaneix (France)
Justus Piater (Austria)	Draguna Vrabie (USA)
Alex Pouget (USA)	Shinji Watanabe (Japan)
Novi Quadrianto (Australia)	Markus Weimer (Germany)
Sabrina Rabello (USA)	Stefan Wermter (Germany)
Achim Rettinger (USA)	Heiko Wersing (Germany)
Alexis Roche (Switzerland)	Daan Wierstra (Switzerland)
Michael Rohs (Germany)	Nick Wilson (UK)
Fabrice Rossi (France)	Zhao Xu (Germany)
Volker Roth (Switzerland)	Zenglin Xu (Germany)
Mert Sabuncu (USA)	Makoto Yamada (Japan)
Jun Sakuma (Japan)	Yoshihiro Yamanishi (Japan)
Scott Sanner (Australia)	Zhirong Yang (Finland)
Ignacio Santamaría (Spain)	Massimo Zancanaro (Italy)
Murat Saracclar (Turkey)	Xinhua Zhang (Canada)
Jagannathan Sarangapani (USA)	Shanfeng Zhu (China)

Table of Contents – Part II

A Markov Random Field Approach to Neural Encoding and Decoding	1
<i>Marcel A.J. van Gerven, Eric Maris, and Tom Heskes</i>	
Weakly Supervised Learning of Foreground-Background Segmentation Using Masked RBMs	9
<i>Nicolas Heess, Nicolas Le Roux, and John Winn</i>	
Recursive Multi-Way PLS for Adaptive Calibration of Brain Computer Interface System	17
<i>Andrey Eliseyev, Alim-Louis Benabid, and Tatiana Aksanova</i>	
Transformation of Edge Weights in a Graph Bipartitioning Problem	25
<i>Iakov M. Karandashev and Boris V. Kryzhanovsky</i>	
A Distributed Behavioral Model Using Neural Fields	32
<i>Mohamed Oubbati, Josef Frick, and Günther Palm</i>	
A Hypothetical Free Synaptic Energy Function and Related States of Synchrony	40
<i>Karim El-Laithy and Martin Bogdan</i>	
Observational Learning Based on Models of Overlapping Pathways	48
<i>Emmanouil Hourdakis and Panos Trahanias</i>	
On the Capacity of Transient Internal States in Liquid-State Machines	56
<i>Karim El-Laithy and Martin Bogdan</i>	
Hybrid Parallel Classifiers for Semantic Subspace Learning	64
<i>Nandita Tripathi, Michael Oakes, and Stefan Wermter</i>	
Temperature Prediction in Electric Arc Furnace with Neural Network Tree	71
<i>Miroslaw Kordos, Marcin Blachnik, and Tadeusz Wiecezorek</i>	
Optimizing Linear Discriminant Error Correcting Output Codes Using Particle Swarm Optimization	79
<i>Dimitrios Bouzas, Nikolaos Arvanitopoulos, and Anastasios Tefas</i>	
SOS-HMM: Self-Organizing Structure of Hidden Markov Model	87
<i>Rakia Jaziri, Mustapha Lebbah, Younès Bennani, and Jean-Hugues Chenot</i>	

Image Receptive Fields Neural Networks for Object Recognition	95
<i>Paméla Daum, Jean-Luc Buessler, and Jean-Philippe Urban</i>	
A Comparison of the Electric Potential through the Membranes of Ganglion Neurons and Neuroblastoma Cells	103
<i>Thiago M. Pinto, Roseli S. Wedemann, and Célia Cortez</i>	
SNPboost: Interaction Analysis and Risk Prediction on GWA Data	111
<i>Ingрид Brænne, Jeanette Erdmann, and Amir Madany Mamlouk</i>	
Binary Patterns Identification by Vector Neural Network with Measure of Proximity between Neuron States	119
<i>Vladimir Kryzhanovskiy</i>	
Emerging Bayesian Priors in a Self-Organizing Recurrent Network	127
<i>Andreea Lazar, Gordon Pipa, and Jochen Triesch</i>	
Momentum Acceleration of Least–Squares Support Vector Machines	135
<i>Jorge López, Álvaro Barbero, and José R. Dorronsoro</i>	
Fast Support Vector Training by Newton’s Method	143
<i>Shigeo Abe</i>	
Linear Operators and Stochastic Partial Differential Equations in Gaussian Process Regression	151
<i>Simo Särkkä</i>	
Learning from Multiple Annotators with Gaussian Processes	159
<i>Perry Groot, Adriana Birlutiu, and Tom Heskes</i>	
Estimation of the Number of Clusters Using Heterogeneous Multiple Classifier System	165
<i>Omar Ayad, Moamar Sayed-Mouchaweh, and Patrice Billaudel</i>	
A Distributed Self-adaptive Nonparametric Change-Detection Test for Sensor/Actuator Networks	173
<i>Cesare Alippi, Giacomo Boracchi, and Manuel Roveri</i>	
Weighted Mutual Information for Feature Selection	181
<i>Erik Schaffernicht and Horst-Michael Gross</i>	
Face Prediction from fMRI Data during Movie Stimulus: Strategies for Feature Selection	189
<i>Jukka-Pekka Kauppi, Heikki Huttunen, Heikki Korkala, Iiro P. Jääskeläinen, Mikko Sams, and Jussi Tohka</i>	
The Authentication System for Multi-modal Behavior Biometrics Using Concurrent Pareto Learning SOM	197
<i>Hiroshi Dozono, Shinsuke Ito, and Masanori Nakakuni</i>	

Hermite Polynomials and Measures of Non-gaussianity	205
<i>Jouni Puuronen and Aapo Hyvärinen</i>	
Complex-Valued Independent Component Analysis of Natural Images	213
<i>Valero Laparra, Michael U. Gutmann, Jesús Malo, and Aapo Hyvärinen</i>	
Improving Gaussian Process Value Function Approximation in Policy Gradient Algorithms	221
<i>Honor Jakab and Lehel Csató</i>	
Application of Nonlinear Neural Network Model for Self Sensing Characteristic in an Ionic Polymer Metal Composite (IPMC) Actuator	229
<i>Ngoc Chi Nam Doan, Kyoung Kwan Ahn, Quang Truong Dinh, and Jong Il Yoon</i>	
Optimal Control Using Functional Type SIRMs Fuzzy Reasoning Method	237
<i>Takashi Mitsuishi and Yasunari Shidama</i>	
High-Dimensional Surveillance	245
<i>Saylisse Dávila, George Runger, and Eugene Tuv</i>	
A One-Layer Dual Recurrent Neural Network with a Heaviside Step Activation Function for Linear Programming with Its Linear Assignment Application	253
<i>Qingshan Liu and Jun Wang</i>	
Neural Network Solution of Optimal Control Problem with Control and State Constraints	261
<i>Tibor Kmet</i>	
Singular Perturbation Approach with Matsuoka Oscillator and Synchronization Phenomena	269
<i>Yasuomi D. Sato, Kazuki Nakada, and Kiyotoshi Matsuoka</i>	
A RANSAC-Based ISOMAP for Filiform Manifolds in Nonlinear Dynamical Systems –An Application to Chaos in a Dripping Faucet–	277
<i>Hiromichi Suetani and Shotaro Akaho</i>	
Manifold Learning for Visualization of Vibrational States of a Rotating Machine	285
<i>Ignacio Díaz, Abel A. Cuadrado, Alberto B. Diez, and Manuel Domínguez</i>	

Bias of Importance Measures for Multi-valued Attributes and Solutions	293
<i>Houtao Deng, George Runger, and Eugene Tuv</i>	
A Computationally Efficient Information Estimator for Weighted Data	301
<i>Hideitsu Hino and Noboru Murata</i>	
Top-Down Induction of Reduced Ordered Decision Diagrams from Neural Networks	309
<i>Jan Chorowski and Jacek M. Zurada</i>	
A Framework for Application-Oriented Design of Large-Scale Neural Networks	317
<i>David Bouchain, Florian Hauser, and Günther Palm</i>	
A Dynamic Field Model of Ordinal and Timing Properties of Sequential Events	325
<i>Flora Ferreira, Wolfram Erlhagen, and Estela Bicho</i>	
Robot Trajectory Prediction and Recognition Based on a Computational Mirror Neurons Model	333
<i>Junpei Zhong, Cornelius Weber, and Stefan Wermter</i>	
A Sentence Generation Network That Learns Surface and Abstract Syntactic Structures	341
<i>Martin Takac, Lubica Benuskova, and Alistair Knott</i>	
A Perceptual Memory System for Affordance Learning in Humanoid Robots	349
<i>Marc Kammer, Marko Tscherepanow, Thomas Schack, and Yukie Nagai</i>	
Probabilistic Proactive Timeline Browser	357
<i>Antti Ajanki and Samuel Kaski</i>	
Person Tracking Based on a Hybrid Neural Probabilistic Model	365
<i>Wenjie Yan, Cornelius Weber, and Stefan Wermter</i>	
Gaze- and Speech-Enhanced Content-Based Image Retrieval in Image Tagging	373
<i>He Zhang, Teemu Ruokolainen, Jorma Laaksonen, Christina Hochleitner, and Rudolf Traunmüller</i>	
Modelling Hypothetical Wage Equation by Neural Networks	381
<i>Jaakko Talonen and Miki Sirola</i>	

On the Designing of Spikes Band-Pass Filters for FPGA	389
<i>Manuel Domínguez-Morales, Angel Jimenez-Fernandez, Elena Cerezuela-Escudero, Rafael Paz-Vicente, Alejandro Linares-Barranco, and Gabriel Jimenez</i>	
An Information Geometrical View of Stationary Subspace Analysis	397
<i>Motoaki Kawanabe, Wojciech Samek, Paul von Bünau, and Frank C. Meinecke</i>	
Forecasting Road Condition after Maintenance Works by Linear Methods and Radial Basis Function Networks	405
<i>Konsta Sirvio and Jaakko Hollmén</i>	
Multistart Strategy Using Delta Test for Variable Selection	413
<i>Dušan Sovilj</i>	
Speech Recognition Based on the Processing Solutions of Auditory Cortex	421
<i>Patrick J.C. May and Hannu Tiitinen</i>	
A Geometric Bio-inspired Model for Recognition of Low-Level Structures	429
<i>E. Ulises Moya-Sánchez and Eduardo Vázquez-Santacruz</i>	
View-Tuned Approximate Partial Matching Kernel from Hierarchical Growing Neural Gases	437
<i>Marco Kortkamp and Sven Wachsmuth</i>	
ANGE – Automatic Neural Generator	446
<i>Leonardo Reis, Luis Aguiar, Darío Baptista, and Fernando Morgado Dias</i>	
Uncertainty Sampling-Based Active Selection of Datasetoids for Meta-learning	454
<i>Ricardo B.C. Prudêncio, Carlos Soares, and Teresa B. Ludermir</i>	
Learning Scheme for Complex Neural Networks Using Simultaneous Perturbation	462
<i>Yutaka Maeda, Takahiro Yamada, Seiji Miyoshi, and Hiroomi Hikawa</i>	
Author Index	471

Table of Contents – Part I

Transformation Equivariant Boltzmann Machines	1
<i>Jyri J. Kivinen and Christopher K.I. Williams</i>	
Improved Learning of Gaussian-Bernoulli Restricted Boltzmann Machines	10
<i>KyungHyun Cho, Alexander Ilin, and Tapani Raiko</i>	
A Hierarchical Generative Model of Recurrent Object-Based Attention in the Visual Cortex.....	18
<i>David P. Reichert, Peggy Series, and Amos J. Storkey</i>	
ℓ_1 -Penalized Linear Mixed-Effects Models for BCI.....	26
<i>Siamac Fazli, Márton Danóczy, Jürg Schelldorfer, and Klaus Robert Müller</i>	
Slow Feature Analysis - A Tool for Extraction of Discriminating Event-Related Potentials in Brain-Computer Interfaces	36
<i>Sven Dähne, Johannes Höhne, Martijn Schreuder, and Michael Tangermann</i>	
Transforming Auto-Encoders	44
<i>Geoffrey E. Hinton, Alex Krizhevsky, and Sida D. Wang</i>	
Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction	52
<i>Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber</i>	
Error-Backpropagation in Networks of Fractionally Predictive Spiking Neurons	60
<i>Sander M. Bohte</i>	
ESN Intrinsic Plasticity versus Reservoir Stability	69
<i>Petia Koprinkova-Hristova and Guenther Palm</i>	
Adaptive Routing Strategies for Large Scale Spiking Neural Network Hardware Implementations	77
<i>Snaider Carrillo, Jim Harkin, Liam McDaid, Sandeep Pande, Seamus Cawley, and Fearghal Morgan</i>	
Self-Organizing Map for the Multi-Goal Path Planning with Polygonal Goals	85
<i>Jan Faigl and Libor Přeučil</i>	

Unlearning in the BCM Learning Rule for Plastic Self-organization in a Multi-modal Architecture	93
<i>Mathieu Lefort, Yann Boniface, and Bernard Girau</i>	
Neuronal Projections Can Be Sharpened by a Biologically Plausible Learning Mechanism	101
<i>Matthew Cook, Florian Jug, and Christoph Krautz</i>	
Explicit Class Structure by Weighted Cooperative Learning	109
<i>Ryotaro Kamimura</i>	
Unsupervised Data-Driven Partitioning of Multiclass Problems	117
<i>Hernán C. Ahumada, Guillermo L. Grinblat, and Pablo M. Granitto</i>	
Bounds for Approximate Solutions of Fredholm Integral Equations Using Kernel Networks	126
<i>Giorgio Gnecco, Věra Kůrková, and Marcello Sanguineti</i>	
An Improved Training Algorithm for the Linear Ranking Support Vector Machine	134
<i>Antti Airola, Tapio Pahikkala, and Tapio Salakoski</i>	
Extending Tree Kernels with Topological Information	142
<i>Fabio Aiolli, Giovanni Da San Martino, and Alessandro Sperduti</i>	
Accelerating Kernel Neural Gas	150
<i>Frank-Michael Schleif, Andrej Gisbrecht, and Barbara Hammer</i>	
State Prediction: A Constructive Method to Program Recurrent Neural Networks	159
<i>René Felix Reinhart and Jochen Jakob Steil</i>	
Cluster Self-organization of Known and Unknown Environmental Sounds Using Recurrent Neural Network	167
<i>Yang Zhang, Shun Nishide, Toru Takahashi, Hiroshi G. Okuno, and Tetsuya Ogata</i>	
Time-Dependent Series Variance Estimation via Recurrent Neural Networks	176
<i>Nikolay Nikolaev, Peter Tino, and Evgueni Smirnov</i>	
Historical Consistent Complex Valued Recurrent Neural Network	185
<i>Hans-Georg Zimmermann, Alexey Minin, and Victoria Kushnerbaeva</i>	
Sparse Spatio-Temporal Gaussian Processes with General Likelihoods	193
<i>Jouni Hartikainen, Jaakko Riihimäki, and Simo Särkkä</i>	
Learning Curves for Gaussian Processes via Numerical Cubature Integration	201
<i>Simo Särkkä</i>	

Cross-Species Translation of Multi-way Biomarkers	209
<i>Tommi Suvitaloval, Ilkka Huopaniemi, Matej Oresič, and Samuel Kaski</i>	
An Evaluation of the Image Recognition Method Using Pulse Coupled Neural Network	217
<i>Masato Yonekawa and Hiroaki Kurokawa</i>	
Using the Leader Algorithm with Support Vector Machines for Large Data Sets	225
<i>Enrique Romero</i>	
Automatic Seizure Detection Incorporating Structural Information	233
<i>Borbala Hunyadi, Maarten De Vos, Marco Signoretto, Johan A.K. Suykens, Wim Van Paesschen, and Sabine Van Huffel</i>	
The Grouped Author-Topic Model for Unsupervised Entity Resolution	241
<i>Andrew M. Dai and Amos J. Storkey</i>	
Kullback-Leibler Divergence for Nonnegative Matrix Factorization	250
<i>Zhirong Yang, He Zhang, Zhijian Yuan, and Erkki Oja</i>	
Distributed Deterministic Temporal Information Propagated by Feedforward Neural Networks	258
<i>Yoshiyuki Asai and Alessandro E.P. Villa</i>	
Chaotic Complex-Valued Multidirectional Associative Memory with Variable Scaling Factor	266
<i>Akio Yoshida and Yuko Osana</i>	
Predicting Reaction Times in Word Recognition by Unsupervised Learning of Morphology	275
<i>Sami Virpioja, Minna Lehtonen, Annika Hultén, Riitta Salmelin, and Krista Lagus</i>	
An Examination of the Dynamic Interaction within Metaphor Understanding Using a Model Simulation	283
<i>Asuka Terai, Saori Hirose, Naoko Kuriyama, and Masanori Nakagawa</i>	
Visual Pathways for Shape Abstraction	291
<i>Konstantinos A. Raftopoulos and Stefanos D. Kollias</i>	
Improving Articulatory Feature and Phoneme Recognition Using Multitask Learning	299
<i>Ramya Rasipuram and Mathew Magimai-Doss</i>	

OrBEAGLE: Integrating Orthography into a Holographic Model of the Lexicon	307
<i>George Kachergis, Gregory E. Cox, and Michael N. Jones</i>	
On the Problem of Finding the Least Number of Features by L1-Norm Minimisation	315
<i>Sascha Klement and Thomas Martinetz</i>	
Extracting Coactivated Features from Multiple Data Sets	323
<i>Michael U. Gutmann and Aapo Hyvärinen</i>	
Single Layer Complex Valued Neural Network with Entropic Cost Function	331
<i>Luís A. Alexandre</i>	
Batch Intrinsic Plasticity for Extreme Learning Machines	339
<i>Klaus Neumann and Jochen J. Steil</i>	
An Empirical Study on the Performance of Spectral Manifold Learning Techniques	347
<i>Peter Mysling, Søren Hauberg, and Kim Steenstrup Pedersen</i>	
Semi-supervised Learning for WLAN Positioning	355
<i>Teemu Pulkkinen, Teemu Roos, and Petri Myllymäki</i>	
Ensemble-Teacher Learning through a Perceptron Rule with a Margin	363
<i>Kazuyuki Hara and Seiji Miyoshi</i>	
Topic-Dependent Document Ranking: Citation Network Analysis by Analogy to Memory Retrieval in the Brain	371
<i>Hiroshi Okamoto</i>	
PADDLE: Proximal Algorithm for Dual Dictionaries LEarning	379
<i>Curzio Basso, Matteo Santoro, Alessandro Verri, and Silvia Villa</i>	
Author Index	387

A Markov Random Field Approach to Neural Encoding and Decoding

Marcel A.J. van Gerven^{1,2}, Eric Maris¹, and Tom Heskes^{1,2}

¹ Donders Institute for Brain, Cognition and Behaviour

² Institute for Computing and Information Sciences

Radboud University Nijmegen, Nijmegen, The Netherlands

m.vangerven@donders.ru.nl, e.maris@donders.ru.nl, t.heskes@science.ru.nl

Abstract. We introduce a new approach to neural encoding and decoding which makes use of sparse regression and Markov random fields. We show that interesting response functions were estimated from neuroimaging data acquired while a subject was watching checkerboard patterns and geometrical figures. Furthermore, we demonstrate that reconstructions of the original stimuli can be generated by loopy belief propagation in a Markov random field.

Keywords: Encoding, decoding, sparse regression, Markov random field.

1 Introduction

Neural encoding and decoding are two topics which are of key importance in cognitive neuroscience. Neural encoding refers to the representation of certain stimulus features by particular neuronal populations as reflected in measured brain activity. Conversely, neural decoding refers to the prediction of such stimulus features from measured brain activity. Encoding is a classical topic in (cognitive) neuroscience and can be tackled by reverse correlation methods [1]. Decoding has gained much recent popularity with the adoption of multivariate analysis methods by the cognitive neuroscience community [2]. While the first decoding studies have focused mainly on the prediction of discrete states such as stimulus orientation [3] or object category [2], more recent decoding studies have focused on the prediction of more complex stimulus properties, culminating in the reconstruction of the contents of visual scenes [4–6].

In this paper, we tackle the encoding problem using elastic net linear regression and show how the resulting parameter estimates can be used to solve the decoding problem. This is achieved by incorporating the estimated regression coefficients within a pairwise Markov random field (MRF) and by using loopy belief propagation [7] to approximate its MAP solution. We apply our methods to a neuroimaging dataset that has previously been used in [5] where subjects have been shown checkerboard patterns and simple geometrical figures. We analyzed the encoding models that have been learned by the employed sparse regression method and find interesting response functions. Furthermore, it is shown that a suitable stimulus prior leads to better stimulus reconstructions.

2 Methods

Let (s, r) denote a stimulus-response pair, say, an image $s = (s_1, \dots, s_I)^\top$, characterized by pixel values s_i , and its associated measured response vector $r = (r_1, \dots, r_K)$. The stimulus can be either discrete or continuous and the response is typically continuous, e.g., the BOLD response in multiple voxels.

2.1 Encoding

In an encoding analysis, both stimulus and response are observed and we are only interested in estimating the parameters $\hat{\theta}$ of the encoding distribution $p(r|s)$ given i.i.d. data $D = \{(s^n, r^n)\}_{n=1}^N$. This can be realized by taking the MAP estimate

$$\hat{\theta} = \arg \min_{\theta} \left\{ -\log p(\theta) - \sum_n \log p(r^n | s^n, \theta) \right\}. \quad (1)$$

We assume that the individual responses r_k are conditionally independent and given by a linear function of s with additive Gaussian noise, such that $r_k = \alpha_k + \beta_k^\top s + \epsilon_k$ where α_k is an offset term, β_k is a vector of regression coefficients for response k and ϵ_k is a zero mean Gaussian random variable with variance σ_k^2 . Thus, we have $p(r|s, \theta) = \mathcal{N}(r; \alpha + B^\top s, \Sigma)$ with $\theta = (\alpha, B, \Sigma)$ where $\alpha \equiv (\alpha_1, \dots, \alpha_K)^\top$, $B \equiv (\beta_1, \dots, \beta_K)$ and $\Sigma = \text{diag}(\sigma_1^2, \dots, \sigma_K^2)$. Also assuming that $p(\theta)$ factorizes accordingly, we obtain a set of minimization problems of the form

$$\hat{\theta}_k = \arg \min_{\alpha_k, \beta_k, \sigma_k^2} \left\{ -\log p(\alpha_k, \beta_k, \sigma_k^2) - \sum_n \log \mathcal{N}(r_k^n; \alpha_k + \beta_k^\top s^n, \sigma_k^2) \right\}. \quad (2)$$

We use the prior to express indifference about σ_k^2 as well as our preference for sparse regression vectors β_k containing just a small number of non-zero elements. This can be realized by choosing $-\log p(\alpha_k, \beta_k, \sigma_k^2) \propto R_{\lambda, \tau}(\beta_k)$ where

$$R_{\lambda, \tau}(\beta_k) = \lambda \sum_{k=1}^K \left\{ (1 - \tau) \frac{1}{2} \|\beta_k\|_2^2 + \tau \|\beta_k\|_1 \right\} \quad (3)$$

is the elastic net regularizer [8]. The parameter λ determines the amount of regularization and τ determines the balance between L_1 and L_2 regularization. Let $S_k(a, b) = \sum_n (r_k^n - a - b^\top s_k^n)^2$ denote the sum of squares error function. Minimization of Eq. (2) with respect to (α_k, β_k) then boils down to computing $(\hat{\alpha}_k, \hat{\beta}_k) = \arg \min_{\alpha_k, \beta_k} \frac{1}{2} S_k(\alpha_k, \beta_k) + R_{\lambda, \tau}(\beta_k)$ which can be achieved using an efficient coordinate gradient descent algorithm [9]. Minimization of Eq. (2) with respect to the variance yields $\hat{\sigma}_k^2 = \frac{1}{N} S_k(\hat{\alpha}_k, \hat{\beta}_k)$.

We use the elastic net algorithm to test how the responses are predicted by a small number of stimulus features. In our problem setting this amounts to probing how the responses r_k are encoded by which pixels via sparse vectors β_k . We refer to β_k as the response function of r_k . Encoding performance is quantified in terms of the coefficient of determination $R_k^2 = 1 - S_k(\hat{\alpha}_k, \hat{\beta}_k) / \sum_n (r_k^n - \bar{r}_k)^2$ where \bar{r}_k is the mean of the observed response data. We use R^2 to denote the average over all responses.

2.2 Decoding

Once the parameters (α, B, Σ) are estimated they may be used for decoding. That is, they are used to approximate the most likely configuration

$$s^* = \arg \max_s \{p(r|s)p(s)\} \quad (4)$$

given an observed response r . In the following, we show how s^* can be approximated by means of inference in a MRF. We start by assuming that the prior can be specified in terms of a MRF

$$p(s) = \frac{1}{Z} \prod_i \phi_i(s_i) \prod_{i \sim j} \phi_{i,j}(s_i, s_j) \quad (5)$$

where $\phi(s_i)$ and $\phi(s_i, s_j)$ are unary and pairwise potential functions, Z is the partition function and $i \sim j$ denotes (unordered) pairs (i, j) that are neighbors in some undirected graph G . The resulting model, together with the encoding of the responses r , is shown in Fig. 1.A.

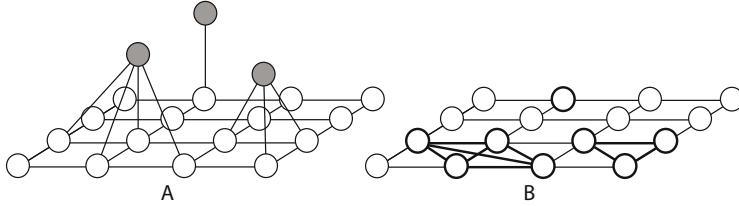


Fig. 1. Panel A: The correlation between stimulus elements s_i , represented by white discs, is given by a MRF. Measured responses r_k , represented by grey discs, are each assumed to be determined by a few stimulus elements. Panel B: Estimated regression coefficients can be absorbed into the MRF yielding a new, typically more dense, MRF. Fat lines indicate which node and edge potentials have changed due to this procedure.

Since $p(r|s) \propto \exp\left(-\frac{1}{2}r^\top \Sigma^{-1}r + r^\top \Sigma^{-1}\mu - \frac{1}{2}\mu^\top \Sigma^{-1}\mu\right)$, given a fixed response r , we can drop terms not depending on s and obtain

$$\begin{aligned} p(r | s) &\propto \exp\left(r^\top \Sigma^{-1}(\alpha + B^\top s) - \frac{1}{2}(\alpha + B^\top s)^\top \Sigma^{-1}(\alpha + B^\top s)\right) \\ &\propto \exp\left(r^\top \Sigma^{-1}B^\top s - \alpha^\top \Sigma^{-1}B^\top s - \frac{1}{2}s^\top B\Sigma^{-1}B^\top s\right) \\ &= \exp(c^\top s + s^\top Us) \end{aligned} \quad (6)$$

with $c^\top \equiv (r - \alpha)^\top \Sigma^{-1}B^\top$ and $U = -\frac{1}{2}B\Sigma^{-1}B^\top$. Equation (6) may also be written as a pairwise MRF $p(r|s) = \frac{1}{Z} \prod_i \psi_i(s_i) \prod_{i \sim j} \psi_{i,j}(s_i, s_j)$ where $\psi_i(s_i) = \exp(s_i \sum_k \frac{\beta_{ki}}{\sigma_k^2} (r_k - \alpha_k - \frac{1}{2}\beta_{ki}))$ encodes local information about the probability

of observing pixel i in state s_i and $\psi_{i,j}(s_i, s_j) = \exp(-s_i s_j \sum_k \frac{\beta_{k,i}}{\sigma_k^2} \beta_{k,j})$ couples pixels s_i and s_j whenever there exists a response r_k that is encoded (in part) by s_i and s_j (see Fig. 1.B). Since the prior is expressed in the same form, the decoding problem can be solved by approximating the mode of a pairwise Markov random field $p(s|r) = \frac{1}{Z} \prod_i (\phi_i(s_i) \psi_i(s_i)) \prod_{i \sim j} (\phi_{i,j}(s_i, s_j) \psi_{i,j}(s_i, s_j))$.

In order to decode the most likely stimulus given a response, we require the maximizer s^* of $p(s|r)$. In case of discrete stimuli, as used in this paper, we have no analytical expression for the mode and we need to rely on approximate inference methods in order to estimate it. We use loopy belief propagation as implemented in the UGM toolbox¹ to compute approximate marginals $q(s_i)$. These marginals are used to approximate the MAP estimate as $s^* \approx \arg \max_s \prod_i q(s_i)$. In order to quantify decoding performance, we use the Manhattan distance between the real stimulus s and its reconstruction s^* , separately averaged over N_0 inactive pixels and N_1 active pixels in order to penalize reconstruction errors of figure or background equally: $M(s, s^*) = \frac{1}{2} \sum_{j=0}^1 \frac{1}{N_j} \sum_{i: s_i=j} |s_i - s_i^*|$.

3 Experiments

3.1 Stimuli, Acquisition and Data Analysis

We are interested in the encoding and decoding of neural responses to simple visual stimuli. We make use of neuroimaging data for one subject which has been made available by the authors² and was used in [5]. Neuroimaging data was acquired while a subject was viewing checkerboard patterns or simple geometrical figures on a 10×10 grid. For the checkerboard patterns, 440 trials were presented which each lasted six seconds. For the geometrical figures, 120 trials were presented which each lasted twelve seconds. Geometrical figures consisted of six repetitions of twenty different patterns. Functional images were acquired with a 3T MRI scanner using an interleaved T2*-weighted gradient-echo echo-planar imaging scan which covered the entire occipital lobe (TR, 2000 ms; TE, 30 ms; flip angle, 80°; FOV, 192×192 mm; voxel size, $3 \times 3 \times 3$ mm; slice gap, 0 mm; number of slices, 30). Additionally, functional localizer scans were used to delineate the borders between visual cortical areas. Functional images were slice-timing corrected, motion corrected, coregistered with a high-resolution structural scan and reinterpolated to $3 \times 3 \times 3$ mm using SPM2 software. Data was linearly detrended, shifted by 3 volumes to take the HRF lag into account and standardized such that the BOLD response in each voxel had zero mean and unit variance. Finally, data was averaged over three consecutive volumes for the checkerboard patterns and six consecutive volumes for the geometrical figures.

3.2 Encoding Analysis

In the encoding analysis we were interested in examining how pixels within the 10×10 grid encode the BOLD response of individual voxels in visual areas

¹ <http://www.cs.ubc.ca/~schmidtm/Software/UGM>

² http://www.cns.atr.jp/~yoichi_m

V1, V2 and V3. To this end, we made use of the elastic net algorithm with $\tau = 0.99$ while varying λ . Encoding performance was computed by training on randomly selected trials (75% of the checkerboard pattern data) and testing on the remaining trials. The value of λ was selected by computing performance as a function of λ using an inner cross-validation and taking the λ with maximal performance to produce the results on the test data. The same procedure was followed using the geometric figure data. This allowed us to compare encoding performance and response functions between the two datasets.

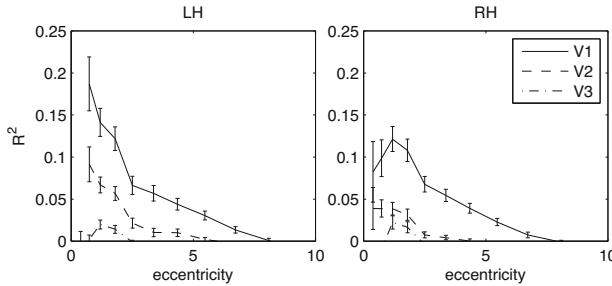


Fig. 2. Average R^2 as a function of visual field eccentricity for voxels in left (LH) and right (RH) hemispheres. Errorbars denote standard error of the mean.

Figure 2 shows average encoding performance of voxels in left and right hemisphere visual cortex while subjects were looking at checkerboard patterns. Voxels are ordered according to their visual field eccentricity as determined by the functional localizer. Maximal encoding performance ranged up to $R^2 = 0.75$, $R^2 = 0.53$ and $R^2 = 0.40$ for voxels that are exclusively assigned to area V1, V2 or V3, respectively. Encoding results degraded when moving to higher visual areas or to areas that code for pixels in the periphery. Spearman rank correlations between the R^2 values for responses to checkerboard patterns and geometric figures were $r = 0.35$, $r = 0.17$ and $r = 0.06$ for areas V1, V2 and V3, respectively.

Figure 3 depicts some of the properties of the estimated response functions. Figure 3.A shows the distribution of the number of pixels used in the response functions of voxels in primary visual areas when using either the checkerboard or the geometric data. Most voxels use no pixels whatsoever, indicating that their BOLD response to visual input is negligible or could not be properly detected. Most of the voxels which do show a response use just one pixel in the encoding. Some voxels, however, are dependent on a large number of pixels. Interestingly, the graphs for the geometric data have heavier tails than those for the checkerboard data. Hence, on average, more pixels are used when voxels are trained on structured images. We examined what the response functions look like for voxels that show large R^2 values for the checkerboard patterns and small R^2 values for the geometric figures (Fig. 3.B) and vice versa (Fig. 3.C). Figure 3.B shows that more diffuse encoding models are found using the geometric figure data for voxels that perform well on checkerboard pattern data. If we examine those voxels in Fig. 3.C which show good performance on the geometric figure

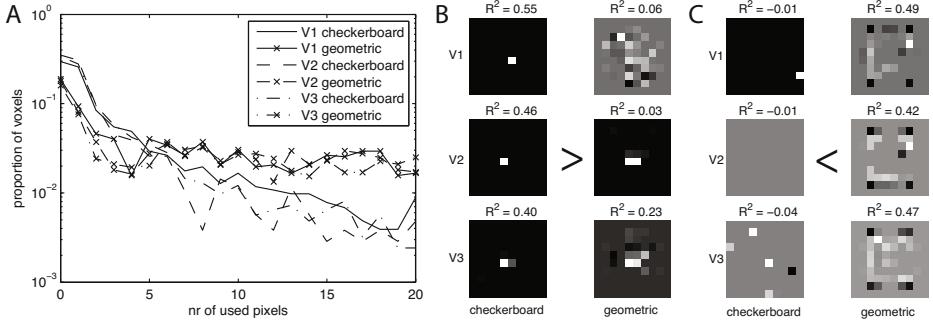


Fig. 3. Panel A: Distribution of the number of used pixels to encode the response for checkerboard and geometric data. Panel B: Examples of response functions $\hat{\beta}_k$ for voxels that show good encoding performance for the checkerboard patterns and bad performance for the geometric figures. Panel C: Examples of response functions $\hat{\beta}_k$ for voxels that show good encoding performance for the geometric figures and bad performance for the checkerboard patterns.

data and not on the checkerboard pattern data, we find quite striking response functions, resembling the presented geometric figures.

3.3 Decoding Analysis

In the decoding analysis, we were interested in reconstructing the stimuli which were most likely to have caused the observed responses by computing the most likely state of a pairwise MRF. We used the checkerboard pattern data to estimate the encoding distributions (training data) and to compute the residual variances (test data). Subsequently, we applied the constructed MRF to decode the geometric figure data. The rationale here is that we wanted to examine whether it is possible to build a generic decoder that is trained on random data. The decoding was achieved using voxels in areas V1, V2 and V3 which were sorted according to their R^2 values in decreasing order. Figure 4. A shows how

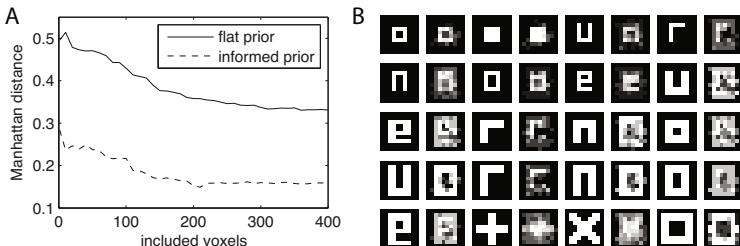


Fig. 4. Panel A shows the decrease in Manhattan distance between a stimulus and its reconstruction averaged over all trials as a function of the number of included voxels. Panel B shows the average reconstruction for each of the twenty structured patterns.

the distance between the stimuli and their reconstructions decreases as a function of the number of included voxels using a flat prior. A minimal average error of 0.33 was obtained by including the responses of 410 voxels.

In order to examine the effect of the prior, we have also constructed an informed prior. Specifically, we created a fully connected pairwise MRF and used the geometric figure data to estimate the MRF parameters. Figure 4.A shows that the structured prior gives much better decoding performance with a minimal error of 0.15 using 210 voxel responses. Figure 4.B depicts the average reconstructions obtained under this regime.

4 Conclusions

In this paper, we have shown that the elastic net model is capable of generating good predictions of the BOLD response as induced by the presentation of checkerboard patterns and geometric figures. The best predictions for checkerboard patterns were obtained by using single pixels in the central visual field. For the geometric figures a subset of the responsive voxels have complex response functions. By absorbing the estimated regression coefficients in a MRF over the stimulus pixels we were able to decode the presented stimuli to some degree.

The encoding results show that the response functions of certain voxels are quite complex (Fig. 3.C). Note however that the geometrical figures on which the encoding models were trained consisted of twenty figures which were repeated six times each. Furthermore, these patterns show strong correlations between pixels. This may lead one to believe that the response functions are just an artifact of these correlations and the voxels are in reality just sensitive to a small number of pixels. Note, however, that if this were the case, then (a) these voxels should also show good performance on the checkerboard patterns and (b) the sparseness constraint of the employed elastic net model would favor response functions which rely on a small number of voxels, even in the presence of strong correlations. In conclusion, we are led to believe that some voxels are truly responsive to complex inputs, although additional analyses are warranted.

Obtained reconstructions show that our MRF approach to decoding is feasible and the inclusion of an informed prior is shown to lead to better decoding performance as compared to the use of a flat prior. Still, single trial decoding results remain quite noisy, which can be due to various reasons. One important reason is that the encoding results actually show that the responses of voxels to checkerboard patterns and geometric figures are only weakly correlated. Hence, the encoding models learned from checkerboard pattern data will not generalize perfectly to the geometric figure data. Another observation is that the decoding results for the informed prior, shown in Fig. 4.B, are biased towards the reconstruction of ‘O’ shapes. This is due to the fact that many of the geometric figures share features with these shapes. This is taken into account by the prior and will bias the reconstructions towards such shapes. Decoding is also influenced by the employed approximate inference method. We used loopy belief propagation and estimated the MAP solution as a max product over the marginals. More sophisticated inference methods may further improve decoding performance.

Various extensions of the framework introduced in this paper are possible. One such extension is to explicitly incorporate the hemodynamic response within the model instead of collapsing over a number of successive measured volumes. Encoding could be improved by including latent variables that explicitly represent the complex features to which particular voxels are sensitive (cf. [10]). Finally, in this paper, we used Bayes' rule to decode using a (Markov random field) model that has been optimized for encoding. An alternative could be to train or at least fine-tune the model's parameters using discriminative training, specifically geared towards improving the decoding performance. We expect this to lead to better reconstructions, possibly at the expense of encoding.

Acknowledgements. The authors gratefully acknowledge the support of the Netherlands Organization for Scientific Research NWO (Vici grant 639.023.604) and the BrainGain Smart Mix Programme of the Netherlands Ministry of Economic Affairs and the Netherlands Ministry of Education, Culture and Science.

References

1. Ringach, D., Shapley, R.: Reverse correlation in neurophysiology. *Cogn. Sci.* 28, 147–166 (2004)
2. Haxby, J., Gobbini, M., Furey, M., Ishai, A., Schouten, J., Pietrini, P.: Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science* 293, 2425–2430 (2001)
3. Kamitani, Y., Tong, F.: Decoding the visual and subjective contents of the human brain. *Nat. Neurosci.* 8, 679–685 (2005)
4. Thirion, B., Duchesnay, E., Hubbard, E., Dubois, J., Poline, J., Lebihan, D., Dehaene, S.: Inverse retinotopy: inferring the visual content of images from brain activation patterns. *NeuroImage* 33, 1104–1116 (2006)
5. Miyawaki, Y., Uchida, H., Yamashita, O., Sato, M., Morito, Y., Tanabe, H.C., Sadato, N., Kamitani, Y.: Visual image reconstruction from human brain activity using a combination of multiscale local image decoders. *Neuron* 60, 915–929 (2008)
6. Naselaris, T., Prenger, R.J., Kay, K.N., Oliver, M., Gallant, J.L.: Bayesian reconstruction of natural images from human brain activity. *Neuron* 63, 902–915 (2009)
7. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Francisco (1988)
8. Zou, H., Hastie, T.: Regularization and variable selection via the elastic net. *J. Roy. Stat. Soc. Series B* 67, 301–320 (2005)
9. Friedman, J., Hastie, T., Tibshirani, R.: Regularization paths for generalized linear models via coordinate descent. *J. Stat. Softw.* 33, 1–22 (2010)
10. van Gerven, M.A.J., de Lange, F.P., Heskes, T.: Neural decoding with hierarchical generative models. *Neural Comput.* 22, 1–16 (2010)

Weakly Supervised Learning of Foreground-Background Segmentation Using Masked RBMs

Nicolas Heess¹, Nicolas Le Roux², and John Winn³

¹ University of Edinburgh, IANC, Edinburgh, UK

² INRIA, Sierra Team, Paris, France

³ Microsoft Research, Cambridge, UK

Abstract. We propose an extension of the Restricted Boltzmann Machine (RBM) that allows the joint shape and appearance of foreground objects in cluttered images to be modeled independently of the background. We present a learning scheme that learns this representation directly from cluttered images with only very weak supervision. The model generates plausible samples and performs foreground-background segmentation. We demonstrate that representing foreground objects independently of the background can be beneficial in recognition tasks.

Keywords: RBM, segmentation, weakly supervised learning.

1 Introduction

Learning generative models of natural images is a long-standing challenge. Recently, a new spectrum of approaches, loosely referred to as “deep learning” (DL), has led to advances in several AI-style learning tasks. At the heart of this framework is the use of RBMs for greedy learning of multi-layered representations such as Deep Belief Networks (DBN, [2]) and Deep Boltzmann Machines (DBM, [7]). However, despite interesting applications in vision (e.g. [3]), it has become apparent that the basic formulation of the RBM is too limited to model images well, and several alternative formulations have recently been proposed (e.g. [6]). One powerful notion from the computer vision literature is that of a layered representation. This allows images to be composed from several independent objects and can account for occlusion (e.g. [10,11]). In [4], we have proposed a model that introduces such a layered representation into the DL framework. In this model, the Masked RBM (MRBM), an image is composed of several regions, each of which is modeled in terms of its shape and appearance. The region shape determines where a region is visible, and the appearance determines the color or texture of the region, while overlapping regions occlude each other. We used this architecture to formulate a generative model of lower-level structure in *generic* images and therefore assumed that all regions were equivalent, i.e. all regions were governed by the same shape and appearance models, and that shape and appearance were independent. For higher-level tasks such as recognition, however, the different regions of an image are not equivalent and some

are more interesting than others. Here, we show that separating an image into *qualitatively different* layers and *jointly* modeling shape and appearance allows us to learn and represent specific objects or object categories *independently* of the background. As a result, the representation of the foreground is less affected by background clutter. Our model is able to perform foreground-background segmentation, and it can generate new instances of the foreground objects (shape and appearance). In particular, we show that learning is possible directly from cluttered images and requires only very weak supervision: inspired by [11], we bootstrap learning with an approximate model of the background. This is easily obtained by training on general natural images and sufficient for learning then to proceed without further supervision: foreground objects can be detected as outliers under the background model and a model of the foreground can thus be learned from the regularities of these outliers across training images. To our knowledge, foreground-background segmentation has not previously been addressed in the DL framework. Tang [8] proposes a model that is related to ours and applies it to the problem of recognition under occlusion, but considers a simpler scenario with binary images and fully supervised learning only.

2 Model

Our model extends the MRBM presented in [4]: instead of modeling general images that consist of generic and equivalent regions, it assumes that images contain a single foreground object in front of a cluttered background (which can and often will also contain parts of other objects, but these are not explicitly modeled). Foreground and background are assumed to be independent and the background is occluded by the foreground object. In the model, this is achieved by composing the *observed* image from two *latent* images: one for the background, and one for the foreground. The background image is visible only where the foreground is not, and the visibility of the foreground image is determined by a binary mask. Intuitively speaking, the foreground image determines the *appearance* of the foreground object, and the mask determines its *shape*. The model is a pixel-wise binary mixture with the mixture component for each pixel specified by the mask. Denoting the observed image by \mathbf{x} , the background image by \mathbf{v}^B , and the appearance and shape of the foreground by \mathbf{v}^F and \mathbf{m} respectively, the model can be written as

$$P(\mathbf{x}) = \sum_{\mathbf{m}} \int d\mathbf{v}^B \int d\mathbf{v}^F \left(\prod_i \delta[v_i^F = x_i]^{m_i} \delta[v_i^B = x_i]^{(1-m_i)} \right) p_{FG}(\mathbf{v}^F, \mathbf{m}) p_{BG}(\mathbf{v}^B) \quad (1)$$

where i is the pixel index, $m_i \in \{0, 1\}$, and the product of delta functions forces, for each pixel, one of the two latent images to take on the value of the observed image at that pixel. The mask determines which latent image is chosen. We formulate the priors over the background image and over foreground appearance and shape as RBMs. Assuming that pixels are continuous valued in $[0, 1]$, we choose for p_{BG} a special form of the Beta RBM with energy $E_{Beta}(\mathbf{v}^B, \mathbf{h})$ described in [4] so that $p_{BG}(\mathbf{v}^B) = 1/Z \sum_{\mathbf{h}} \exp\{-E_{Beta}(\mathbf{v}^B, \mathbf{h}; \Theta_{BG})\}$. Unlike the

Gaussian RBM with fixed variance commonly used for continuous data (e.g. [3]), the Beta RBM models mean *and* variance of the visible units.

The model of the foreground object defines a joint distribution over a continuous valued image and a binary mask. Here, we are interested in the case where shape and appearance are *dependent* and we will therefore model them jointly. This is in contrast with the approach taken in [4] where they were assumed to be independent, i.e. $p_{\text{FG}}(\mathbf{v}^F, \mathbf{m}) = p_{\text{FG}}(\mathbf{v}^F)p_{\text{FG}}(\mathbf{m})$. Thus, in this new model, we use a particular form of the RBM which has two sets of visible units, a set of binary units for the mask \mathbf{m} and a set of continuous valued units for the appearance image \mathbf{v}^F :

$$E_{\text{mixed}}(\mathbf{v}, \mathbf{m}, \mathbf{h}; \Theta) = E_{\text{Bin}}(\mathbf{m}, \mathbf{h}; \Theta^S) + E_{\text{Beta}}(\mathbf{v}, \mathbf{h}; \Theta^A) \quad , \quad (2)$$

where $E_{\text{Bin}}(\mathbf{m}, \mathbf{h}; \Theta) = \mathbf{m}^T \mathbf{W} \mathbf{h} + \mathbf{b}^T \mathbf{m}$ is the energy function of a binary RBM, and the joint distribution is thus given by: $p_{\text{FG}}(\mathbf{v}^F, \mathbf{m}) = \frac{1}{Z} \sum_{\mathbf{h}} \exp\{-E_{\text{mixed}}(\mathbf{v}^F, \mathbf{m}, \mathbf{h}; \Theta)\}$.

Inference: Although exact inference is intractable, an efficient Gibbs sampling scheme exists, as detailed in [4]. Let \mathbf{h}^F and \mathbf{h}^B denote the hidden units of the foreground and the background model respectively, then the following properties admit a Gibbs sampling scheme in which the three sets of variables $(\mathbf{h}^F, \mathbf{h}^B)$, \mathbf{m} , and $(\mathbf{v}^F, \mathbf{v}^B)$ are sampled in turn:

1. given \mathbf{v}^F and \mathbf{v}^B the hidden units \mathbf{h}^F and \mathbf{h}^B are conditionally independent, i.e. $p(\mathbf{h}^F | \mathbf{v}^F) = \prod_j p(h_j^F | \mathbf{v}^F)$ and $p(\mathbf{h}^B | \mathbf{v}^B) = \prod_j p(h_j^B | \mathbf{v}^B)$,
2. given the hidden variables \mathbf{h}^F , \mathbf{h}^B and the image \mathbf{x} , the variables of the mask, foreground image, and background image are pixel-wise conditionally independent, i.e. $p(\mathbf{v}^F, \mathbf{v}^B, \mathbf{m} | \mathbf{h}^F, \mathbf{h}^B, \mathbf{x}) = \prod_i p(v_i^F, v_i^B, m_i | \mathbf{h}^F, \mathbf{h}^B, \mathbf{x})$,
3. $p(v_i^F, v_i^B, m_i | \mathbf{h}^F, \mathbf{h}^B, \mathbf{x})$ can be decomposed as follows

$$p(v_i^F, v_i^B, m_i | \mathbf{h}^F, \mathbf{h}^B, \mathbf{x}) = p(v_i^F, v_i^B | m_i, \mathbf{h}^F, \mathbf{h}^B, \mathbf{x}) p(m_i | \mathbf{h}^F, \mathbf{h}^B, \mathbf{x}) \quad (3)$$

$$p(m_i = 1 | \mathbf{h}^F, \mathbf{h}^B, \mathbf{x}) = \frac{p_{\text{FG}}(v_i^F = x_i, m_i = 1 | \mathbf{h}^F)}{p_{\text{BG}}(v_i^B = x_i | \mathbf{h}^B) p_{\text{FG}}(m_i = 0 | \mathbf{h}^F) + p_{\text{FG}}(v_i^F = x_i, m_i = 1 | \mathbf{h}^F)} \quad (4)$$

$$p(v_i^F, v_i^B | m_i, \mathbf{h}^F, \mathbf{h}^B, \mathbf{x}) = \begin{cases} \delta[v_i^F = x_i] p_{\text{BG}}(v_i^B | \mathbf{h}^B) & \text{if } m_i = 1 \\ \delta[v_i^B = x_i] p_{\text{FG}}(v_i^F | \mathbf{h}^F) & \text{otherwise.} \end{cases} \quad (5)$$

Learning: During learning with unlabeled data only \mathbf{x} is observed. We use an EM-like approach in which inference of \mathbf{v}^F , \mathbf{v}^B , and \mathbf{m} alternates with updates of the model parameters. Once these variables have been inferred, they can be treated as “observed” data for one of the usual learning schemes for RBMs such as contrastive divergence (CD, [1]) or stochastic maximum likelihood (SML, also referred to as “persistent CD” [9]), which we use in the experiments below. SML relies on persistent chains of samples to represent the model distribution which are updated by one step of Gibbs sampling per iteration. Note that, due to the directed nature of the mixture in (1), the persistent Markov chains representing the model distributions of the two RBMs for foreground and background do not

interact, i.e. we run independent chains as if we were training both RBMs separately. Fully unsupervised learning is possible in principle but likely to be very difficult for all but very simple datasets. We therefore consider a “weakly supervised” scenario related to [11]: we assume that we have some general knowledge of the statistical regularities to be expected in the *background*. This approximate model of the background can be obtained by training on general natural images and it allows us to bootstrap learning of the foreground model from unsegmented training data. Foreground objects stand out from the background, i.e. they appear as “outliers” to the background model which forces them to be explained by the foreground model. Although this detection is initially unreliable, the foreground model can then learn about the consistencies of these outliers across the training images which eventually leads to a good model of the foreground objects without any additional information being provided or prior knowledge (e.g. coherence or convexity) being used.

3 Experiments

Datasets & evaluation: We evaluate the model and the learning scheme on two datasets: a toy dataset and a more challenging, modified version of the “Labeled faces in the wild-A” (LFW-A)-dataset [5,12]. The toy data consist of 16×16 pixel image patches that contain two classes of foreground “objects” against backgrounds that are randomly cropped patches from the VOC 2009 dataset. The two classes differ in appearance and shape (i.e. shape and appearance are *dependent*) and objects can appear at various positions in the patch (see Fig. 1). For the LFW-A dataset, the original images of size 250×250 were cropped to 210×210 pixels and down-scaled to 32×32 pixels. We used the first 13000 images in the dataset for training and the remaining 233 images for test purposes. Examples are shown in Fig. 2a. We evaluate the quality of learned models for both datasets by sampling¹ and by performing FG-BG segmentation on unseen test images². We also investigate whether the FG-BG model’s ability to model the object of interest independently of the background provides additional robustness of the latent representation to background clutter in a recognition task.

Models & learning: For our weakly supervised learning scheme, we first learned models of the background by training Beta RBMs on large sets of natural image patches using SML. We next trained the foreground RBMs in the context of the full model (eq. 1 as described in section 2). For each training data point, we store the state of the latent variables \mathbf{v}^F , \mathbf{v}^B , and \mathbf{m} from one epoch to the next. We update them by one step of Gibbs sampling before using the state of the latent variables for the current batch and the particles in the persistent chains for the foreground model to compute the gradient step for the parameters of the foreground RBM in the usual manner [9]. Fig. 2e illustrates how the mask \mathbf{m}

¹ Markov chains are initialized with random noise; we use the conditional means of the visible units given the hidden units in the final step for visualization.

² We provide a comparison with the performance of a simple conditional random field (CRF) in the supplemental material [13].

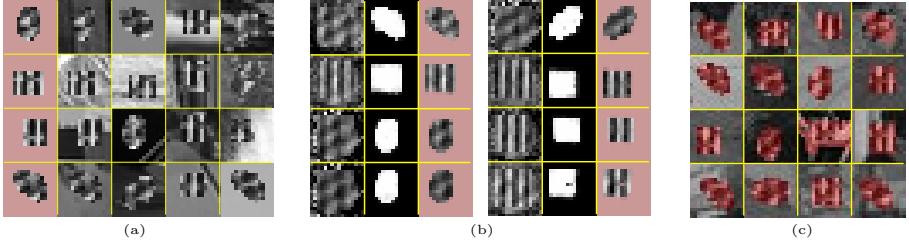


Fig. 1. (a) Training toy data: The two classes of “objects” are (1) rectangles of 9 different sizes and (2) four kinds of round shapes. Objects can appear at different locations in the image and are filled with two different types of textures. The texture varies from training image to training image but rectangles are always filled with one type of texture and round objects with the other. The first columns shows “ground truth”, i.e. objects in isolation. The remaining columns show actual training data, i.e. objects embedded in natural image backgrounds. (b) Samples from the model: In each block the left column shows v^F (appearance), the middle column shows m (shape) and the right column shows the joint sample where shape and appearance have been combined (red indicates the invisible part of the sample). (c) Test images with inferred masks m superimposed in semi-transparent red. The model largely identifies the foreground objects correctly, but struggles sometimes, especially if the background is poorly explained under the background model.

converges for a particular *training* image *during learning* over 1000 epochs. For the model to converge well on the face dataset we further initialized the weights for the foreground appearance by training a Beta RBM directly on the full face images (i.e. including background; this pre-training does not teach the model to distinguish between foreground and background in the training images). For highly structured images the background models were sometimes not sufficiently powerful so that part of the background was assigned to the foreground even after consolidation of the foreground model. This does not completely prohibit learning of the foreground model but leads to a noisy final model. We addressed this issue by introducing an outlier component into the background model³, i.e. each background pixel was modeled either by the background RBM or by a uniform distribution ($p = 0.3$), which can be incorporated into the Gibbs sampling scheme by modifying equations (3-5). Additional details can be found in the supplemental material available on the first author’s homepage [13].

4 Results

Toy Data: For the toy data the model successfully learned about the shapes and appearances of the foreground objects: after learning, samples closely matched

³ Using an “outlier-indicator” $o_i \in \{0, 1\}$ ($P(o_i = 1) = p$) the constraints in eq. (1) are replaced by $\left(\prod_i \delta[v_i^F = x_i]^{m_i} \delta[v_i^B = x_i]^{(1-m_i)(1-o_i)} [U(x_i)]^{(1-m_i)o_i} \right)$. For the toy dataset we initially trained only with the basic model and introduced the outlier component only for fine-tuning once the model had largely converged. For the faces we trained with the outlier component from the beginning.

the foreground objects in the training set (cf. Fig. 1a) and inference on test patches led to largely correct segmentations of these patches into foreground and background: it labeled 96% of the pixels in our set of 5000 test images correctly⁴ (see Fig. 1c for examples). To investigate to what extent the ability to ignore the background may help in recognition tasks we trained a simple RBM on the same data (same total number of hidden units as for the FG-BG model) and performed inference for a set of test patches in both models⁵. We trained a simple logistic classifier (with L2-regularization) on the inferred activation of the hidden units (only \mathbf{h}^F for the FG-BG model) in order to classify patches whether they contain rectangles or round shapes. This is an easy classification task if enough training data is available (e.g. 100 training patches per class), but as the number of training data points is reduced the classification performance drops strongly for the simple RBM but to a far lesser extent for the FG-BG model (classification performance 66% vs. 88% at 10 data points per class), suggesting that being able to ignore the background can help to improve recognition performance.

Faces: The model also learned a good representation of faces. Fig. 2b shows samples from the trained model. Although the samples do not exhibit as much detail as the faces in the training data (this is to be expected given the relatively small number of hidden units used) they exhibit important features of the training data, for instance, there are male and female faces, and the model has learned about different head positions and hair styles. Figure 2c shows segmentation results for a subset of the test images. In most cases the model has largely correctly identified the pixels belonging to the face. Test images for which the model tends to make mistakes typically show heads in extreme poses. Fig. 2d demonstrates that the model does not simply choose the same region in all images: randomly re-assigning the inferred masks to test images leads to considerably worse results. To investigate the effect of the mask in a very simple recognition task we manually segmented a small subset of the images in the dataset ($N = 65$). For each segmented person we created two test images by pasting the person into two different natural image background patches, thus obtaining two sets of images containing the same 65 different faces but against different backgrounds (two example pairs are shown in Fig. 2f). We inferred the segmentation and subsequently the hidden activation of the latent units of the foreground model. For each image from the first set we determined the most similar image in the second set (in terms of the RMS difference between the hidden unit activations). For 52 (80%) of the images in the first set the corresponding image with the same face (different background) in the second set was the closest match. This compares to 15 out of 65 (23%) for a simple Beta RBM (same number of hidden units as the foreground model) trained on the same dataset, suggesting that here, too, “ignoring” the background can lead to a representation in the hidden units that is less affected by the background than for a normal RBM.

⁴ Chance is 50%; a CRF using a histogram of the background and contrast dependent smoothness term achieves 79%; see supplemental material [13] for further details.

⁵ Inference in the simple RBM involves computing the activation of the hidden units given the test image; in the FG-BG model it involves inferring \mathbf{m} , \mathbf{v}^F , and \mathbf{v}^B .



Fig. 2. (a) Examples of the training data. (b): Samples from the learned model. For the first three columns the format is similar to Fig. 1b, they demonstrate how shape (\mathbf{m} , left) and appearance (\mathbf{v}^F , middle) combine to the joint sample (right). The remaining columns show further samples from the model. For the joint samples the red area is not part of the object. (c): Inferred masks \mathbf{m} (foreground-background segmentations) for a subset of the test images. Masks are superimposed on the test images in red. In most cases the model has largely correctly identified the pixels belonging to the face. Test images for which the model tends to make mistakes typically show the head in extreme poses. Labeling of the neck and the shoulders is somewhat inconsistent, which is expected given that there is considerable variability in the training images and that the model has not been trained to either include or exclude such areas. The same applies if parts of a face are occluded, e.g. by a hat. (d) Test images with random masks superimposed. If masks are randomly assigned to test images the alignment of mask and image is considerably worse. Additional training images and segmentation results and a comparison with results for a conventional conditional random field can be found in the supplemental material [13]. (e): Convergence of the segmentation *during learning* (inferred mask \mathbf{m} superimposed on training image before joint training (left most) and after 10, 20, 100 and 1000 epochs of joint training). At the beginning the segmentation is driven primarily by the background model and thus very noisy. (f) Two examples of pairs of images used for the recognition task.

5 Discussion

We have demonstrated how RBMs and layered representations can be combined to obtain a model that is able to represent the joint shape and appearance of foreground objects independently of the background and we have shown how to learn the model of the foreground directly from cluttered training images using only very weak supervision. The architecture is very flexible: it can be applied to images of different types (e.g. binary), the background model can be re-used for different foreground models, and it is possible to replace the background model independently of the foreground (e.g. if the statistics of the background change). Also, DBNs or DBMs could be used instead of RBMs. One interesting extension would be to include a third layer that is *in front* of the object layer. This would allow modeling occlusion of the foreground object (such occlusions, although rare in the face dataset, may explain part of the uncertainty in the learned shape model). Using *semi*-supervised schemes (e.g. with a few pre-segmented images) to learn more challenging object categories, is another exciting direction.

Acknowledgments. NH is supported by a EPSRC/MRC scholarship from the Neuroinformatics DTC at the University of Edinburgh. NLR is supported by a grant from the European Research Council (SIERRA-ERC-239993).

References

1. Hinton, G.E.: Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation* 14(8), 1771–1800 (2002)
2. Hinton, G.E., Osindero, S., Teh, Y.: A fast learning algorithm for deep belief nets. *Neural Computation* 18, 1527–1554 (2006)
3. Lee, H., Gross, R., Ranganath, R., Ng, A.Y.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: ICML (2009)
4. Le Roux, N., Heess, N., Shotton, J., Winn, J.: Learning a Generative Model of Images by Factoring Appearance and Shape. *Neural Computation* 23(3), 593–650 (2011)
5. Huang, G.B., Rames, M., Berg, T., Learned-Miller, E.: Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. TR 07-49; Univ. of Mass., Amherst (2007)
6. Ranzato, M., Hinton, G.: Modeling Pixel Means and Covariances Using Factorized Third-Order Boltzmann Machines. In: CVPR (2010)
7. Salakhutdinov, R., Hinton, G.E.: Deep Boltzmann Machines. In: AISTATS (2009)
8. Tang, Y.: Gated Boltzmann Machine for Recognition under Occlusion. In: NIPS Workshop on Transfer Learning by Learning Rich Generative Models (2010)
9. Tieleman, T.: Training Restricted Boltzmann Machines using Approximations to the Likelihood Gradient. In: ICML (2008)
10. Wang, J.Y.A., Adelson, E.H.: Representing moving images with layers. *IEEE Transactions on Image Processing* 3(5), 625 (1994)
11. Williams, C.K.I., Titsias, M.K.: Greedy Learning of Multiple Objects in Images using Robust Statistics and Factorial Learning. *Neural Comp.* 16(5), 1039–1062 (2004)
12. Wolf, L., Hassner, T., Taigman, Y.: Similarity scores based on background samples. In: Zha, H., Taniguchi, R.-i., Maybank, S. (eds.) ACCV 2009. LNCS, vol. 5995, pp. 88–97. Springer, Heidelberg (2010)
13. Suppl. Material, <http://homepages.inf.ed.ac.uk/s0677090/papers/icannSuppl.pdf>

Recursive Multi-Way PLS for Adaptive Calibration of Brain Computer Interface System

Andrey Elishev^{1,2}, Alim-Louis Benabid², and Tatiana Aksanova^{1,2}

¹ Foundation Nanosciences, Grenoble, France

² Clinatec/LETI/CEA, Grenoble, France

{andriy.yelisyeyev, alim-louis.benabid, tetiana.aksanova}@cea.fr

Abstract. In the present article a Recursive Multi-Way PLS algorithm for adaptive calibration of a BCI system is proposed. It combines the NPLS tensors decomposition with a scheme of recursive calculation. This Recursive algorithm allows treating data arrays of huge dimension. In addition, adaptive calibration provides a fast adjustment of the BCI system to mild changes of the signal. The proposed algorithm was validated on artificial and real data sets. In comparison to generic Multi-Way PLS, the recursive algorithm demonstrates good performance and robustness.

Keywords: Partial least square, recursive estimation, multi-way analysis tensor factorization, brain-computer interface (BCI), adaptive control.

1 Introduction

Movement related Brain Computer Interfaces (BCI) aim to provide an alternative non-muscular communication pathway for individuals with severe motor disability (such as post-traumatic quadriplegia) to send commands to effectors of the external world originating from measures of the brain neuronal activity. During the last decades, several approaches were developed to face the problem of neuronal signal decoding. In particular, multi-way analysis was reported recently as an effective tool for neuronal signal processing. It allows simultaneous treatment of data in several domains. For instance, multi-way analysis was applied recently at the self-paced BCI in natural environment in freely moving animals ([1], [2]). Signals of neuronal activity were mapped by continuous wavelet transformation to the temporal-frequency-spatial space. Then the Iterative N-way PLS (INPLS) was applied to extract the predictors (neural electrical features preceding an intention to act). One of the major problems of BCI studies is the variability of neuronal signals, due in particular to the brain plasticity. These changes in neuronal activity require recalibration of BCI systems. The full system recalibration is a time and labor consuming procedure. Adaptive calibration aims to provide a fast adjustment of the BCI system to mild changes of the signal. Although the INPLS allows treating data arrays of huge dimension, this method cannot be applied for adaptive learning. In this paper a Recursive NPLS (RNPLS) algorithm is proposed. It allows online processing of multi-modal data. Moreover, weighted RNPLS can be applied for adaptive learning to treat time-dependent recordings. This algorithm can be efficiently used for numerous applications beyond BCI.

2 Methods

The RNPLS algorithm combines the NPLS tensors (multi way arrays) decomposition and modeling [3] with the scheme of recursive calculation [4].

2.1 Generic PLS

Partial Least Squares (PLS) is a statistical method for data analyses, particularly suited to the case of high dimensions of observations [5]. PLS regression is an approach for modeling the linear relationship between the vector of dependent (output) variables \mathbf{y} and the vector of independent (input) variables \mathbf{x} on the basis of matrices of observations \mathbf{X} and \mathbf{Y} : $\mathbf{Y} = \mathbf{X}\mathbf{C} + \mathbf{V}$, where \mathbf{V} and \mathbf{C} are noise and coefficient matrices. To build the model, the observations are projected into low dimensional spaces in such a way that the maximum variances of \mathbf{X} and \mathbf{Y} are explained simultaneously. PLS represents the iterative procedure. At the first step, the matrices \mathbf{X} and \mathbf{Y} are presented as bilinear terms:

$$\mathbf{X} = \mathbf{t}_1 \mathbf{p}_1^T + \mathbf{E}_1, \quad \mathbf{Y} = \mathbf{u}_1 \mathbf{q}_1^T + \mathbf{F}_1,$$

where \mathbf{t}_1 and \mathbf{u}_1 are the latent variables (score vectors), whereas \mathbf{p}_1 and \mathbf{q}_1 are the loading vectors. \mathbf{E}_1 and \mathbf{F}_1 are the matrices of residuals. The score vectors are linear transforms of the matrices of observation in a way to maximize the covariance of \mathbf{t}_1 and \mathbf{u}_1 [5]. The score vectors are related by a linear model minimizing the norm of the residuals \mathbf{r}_1 : $\mathbf{u}_1 = b_1 \mathbf{t}_1 + \mathbf{r}_1$. The same procedure is applied iteratively to the residual matrices. It is repeated F times.

Let us note that latent variables could be constructed as orthonormal [4]:

$$\mathbf{T}^T \mathbf{T} = \mathbf{I}_F, \quad (1)$$

where $\mathbf{T} = [\mathbf{t}_1, \dots, \mathbf{t}_F]$, \mathbf{I}_F is identity matrix.

2.2 Recursive PLS

The recursive PLS algorithms were invented to take into account time-dependent changes of data as well as to be able to handle large data sets. Qin [4] and by Dayal and MacGregor [6] introduced the most known approaches. The Dayal and MacGregor algorithm exhibits a better performance but it stores in the memory the covariance matrix $\mathbf{X}^T \mathbf{X}$, the size of which is equal to the square of the dimension of variable \mathbf{x} . Thus the method is not suited to the case of high dimension of the variable \mathbf{x} . In our BCI program, we are particularly interested in multimodal data analyses. In this case, the data dimensionality is extremely large (in the order of hundreds of thousands). That is why we chose Qin's algorithm [4] as a basic approach. This article is devoted to the extension of this method to multimodal data.

According to the algorithm of Qin [4], matrices \mathbf{X} and \mathbf{Y} are decomposed by the batch-wise PLS algorithm with orthonormal latent variables (1):

$$\mathbf{X} = \mathbf{TP}^T + \mathbf{E}_F, \quad \mathbf{Y} = \mathbf{UQ}^T + \mathbf{F}_F = \mathbf{TBQ}^T + \mathbf{F}_F,$$

here $\mathbf{P} = [\mathbf{p}_1, \dots, \mathbf{p}_F]$, $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_F]$, $\mathbf{Q} = [\mathbf{q}_1, \dots, \mathbf{q}_F]$, $\mathbf{B} = \text{diag}\{b_1, \dots, b_F\}$, and $\mathbf{T}^T \mathbf{T} = \mathbf{I}_F$. Additionally by construction it holds

$$\mathbf{T}^T \mathbf{E}_F = \mathbf{0}, \quad (2)$$

$$\mathbf{T}^T \mathbf{F}_F = \mathbf{0}. \quad (3)$$

It is shown [4] that if F is large enough to give $\mathbf{E}_F^T \mathbf{E}_F = 0$ then $\mathbf{X}^T \mathbf{X} = \mathbf{P} \mathbf{P}^T$ and $\mathbf{X}^T \mathbf{Y} = \mathbf{P} \mathbf{B} \mathbf{Q}^T$. This yields that, for the new data pair, $\{\mathbf{X}_1, \mathbf{Y}_1\}$ regressions on the next data sets will be equivalent:

$$\left\{ \begin{bmatrix} \mathbf{X} \\ \mathbf{X}_1 \end{bmatrix}, \begin{bmatrix} \mathbf{Y} \\ \mathbf{Y}_1 \end{bmatrix} \right\} \Leftrightarrow \left\{ \begin{bmatrix} \mathbf{P}^T \\ \mathbf{X}_1 \end{bmatrix}, \begin{bmatrix} \mathbf{B} \mathbf{Q}^T \\ \mathbf{Y}_1 \end{bmatrix} \right\}.$$

Thus, the old data \mathbf{X} and \mathbf{Y} are captured by the loading matrices \mathbf{P} , \mathbf{Q} , and by the coefficient matrix \mathbf{B} , and the new data are added to them. As a result, the algorithm always keeps the size of the matrices. In addition, it presents an effective tool for the adaptive learning introducing the weights for the old and new data [4].

2.3 NPLS

The multi-way PLS [3] is introduced as a generalization of PLS to multi-way data sets (tensors). Tensors (multi-way arrays) are a higher-order generalization of vectors and matrices. Elements of the tensor $\underline{\mathbf{X}} \in R^{I_1 \times I_2 \times \dots \times I_N}$ are denoted as x_{i_1, i_2, \dots, i_N} , here N is the order of the tensor, i.e., the number of dimensions (ways or modes). The number of the variables I_i in the mode i shows the dimensionality of this mode [7]. Without loss of generality, let us consider the case of the fourth-order tensor of observations $\underline{\mathbf{X}} \in R^{n \times I_1 \times I_2 \times I_3}$ which contains n samples $\mathbf{x}_i \in R^{I_1 \times I_2 \times I_3}$, $i = 1, \dots, n$, and the vector of n scalar observations $\mathbf{y} \in R^n$. Similar to the ordinary PLS approach, $\underline{\mathbf{X}}$ is represented as:

$$\underline{\mathbf{X}} = \mathbf{t}_1 \circ \mathbf{w}_1^1 \circ \mathbf{w}_1^2 \circ \mathbf{w}_1^3 + \underline{\mathbf{E}}_1, \quad (4)$$

here operation “ \circ ” is called the outer product (see [7]). The latent variable $\mathbf{t}_1 \in R^n$ is extracted from the first mode of the tensor $\underline{\mathbf{X}}$ providing maximum of covariance between \mathbf{t}_1 and \mathbf{y} . In parallel, the algorithm forms the factor, i.e. the set of vectors $\{\mathbf{w}_1^1 \in R^{I_1}, \mathbf{w}_1^2 \in R^{I_2}, \mathbf{w}_1^3 \in R^{I_3}\}$, $\|\mathbf{w}_1^i\| = 1$, $i = 1, 2, 3$ related to the second, the third, and the fourth modes of $\underline{\mathbf{X}}$, respectively, in such a way that projection of the tensor $\underline{\mathbf{X}}$ on these vectors results in \mathbf{t}_1 . The coefficient b_1 of regression $\mathbf{y} = b_1 \mathbf{t}_1 + \mathbf{f}_1$ is calculated with Minimal Least Squares (MLS). Next factors are calculated in the same way by decomposing the residuals. Let us note that the NPLS latent variables $\{\mathbf{t}_f\}_{f=1}^F$ are not orthogonal.

Similar to the generic PLS regression, the NPLS projects data onto the feature space of low dimension. The tensor decomposition uses the set of projectors, which corresponds to each modality.

2.4 RNPLS

In this article we propose the method of recursive tensor-data processing (RNPLS), which is derived from the NPLS and the RPLS. To apply the recursive approach to the NPLS algorithm described above, the orthonormality of the latent variables as well as the orthogonality of the latent variables to the residuals should be provided (see conditions (1-3)).

Let us consider the fourth-order tensor $\underline{\mathbf{X}} \in R^{n \times I_1 \times I_2 \times I_3}$ of observations and the vector of observations $\mathbf{y} \in R^n$. At the first step, the tensor $\underline{\mathbf{X}}$ is represented by the set of factors $\{\mathbf{w}_i^1\}_{i=1}^3$ constructed by the NPLS algorithm (4). For simplicity of the notation, let us unfold the tensor $\underline{\mathbf{X}}$ along the first mode into the matrix $\mathbf{X} \in R^{n \times (I_1 I_2 I_3)}$ [7]. At the same time let us denote $\mathbf{w}_1 \in R^{I_1 I_2 I_3}$ the vectorization of the tensor $\underline{\mathbf{w}}_1 = \mathbf{w}_1^1 \circ \mathbf{w}_1^2 \circ \mathbf{w}_1^3$. The score vector \mathbf{t}_1 is a projection of the observation matrix \mathbf{X} on \mathbf{w}_1 : $\mathbf{t}_1 = \mathbf{X} \mathbf{w}_1$. In this matrix notation, equation (4) can be rewritten in the form $\mathbf{X} = \mathbf{t}_1 \mathbf{w}_1^T + \mathbf{E}_1$. On the iteration f : $\mathbf{X} = \mathbf{T}_f \mathbf{W}_f^T + \mathbf{E}_f$.

As it was mentioned above, the NPLS latent variables are not orthogonal. Let us apply orthonormalization: $\mathbf{T}_f^\perp = \mathbf{T}_f \mathbf{A}_f$ $(\mathbf{T}^\perp)^T \mathbf{T}^\perp = \mathbf{I}_F$. Here \mathbf{A}_f is the matrix of orthonormalization. For the new latent variables $\mathbf{X} = \mathbf{T}_f \mathbf{W}_f^T + \mathbf{E}_f = \mathbf{T}_f^\perp \mathbf{P}_f + \mathbf{E}_f$, where $\mathbf{P}_f^T = \mathbf{A}_f^{-1} \mathbf{W}_f^T$. The OLS coefficients of regression of \mathbf{y}_f on the orthonormal latent variable $\mathbf{y}_f = b_f^\perp \mathbf{t}_f^\perp + \mathbf{f}_f$ equal to $b_f^\perp = \mathbf{y}_f^T \mathbf{t}_f^\perp$. After F iterations:

$$\mathbf{X} = \mathbf{T}^\perp \mathbf{P}^T + \mathbf{E}_F, \quad \mathbf{y} = \mathbf{T}^\perp \mathbf{b}^\perp + \mathbf{f}_F.$$

At this step conditions (1) and (3) are satisfied: $(\mathbf{T}^\perp)^T \mathbf{T}^\perp = \mathbf{I}_F$ and $(\mathbf{T}^\perp)^T \mathbf{f}_F = \mathbf{0}$. Then to provide orthogonality of the matrix \mathbf{T}^\perp and the residual matrix \mathbf{E}_F (2), let us subtract from the residual matrix its projection on all latent variables $\{\mathbf{t}_f^\perp\}_{f=1}^F$: $\tilde{\mathbf{E}}_F = \mathbf{E}_F - \sum_{f=1}^F \mathbf{t}_f^\perp (\mathbf{t}_f^\perp)^T \mathbf{E}_F$. Thus $\mathbf{X} = \mathbf{T}^\perp \tilde{\mathbf{P}}^T + \tilde{\mathbf{E}}_F$, with the new matrix of loadings $\tilde{\mathbf{P}} = [\tilde{\mathbf{p}}_1, \dots, \tilde{\mathbf{p}}_F]$, $\tilde{\mathbf{p}}_f = \mathbf{p}_f + \mathbf{E}_F^T \mathbf{t}_f^\perp$. For the new residuals $\tilde{\mathbf{E}}_F$ it holds $(\mathbf{T}^\perp)^T \tilde{\mathbf{E}}_F = \mathbf{0}$. Since conditions (1), (2) and (3) are satisfied, we get $\mathbf{X}^T \mathbf{X} = \tilde{\mathbf{P}} \tilde{\mathbf{P}}^T$ and $\mathbf{X}^T \mathbf{y} = \tilde{\mathbf{P}} \mathbf{b}^\perp$. Similar to the RPLS for the new data pair $\{\underline{\mathbf{X}}_1, \mathbf{y}_1\}$ regressions on the next data sets will be equivalent:

$$\left\{ \begin{bmatrix} \underline{\mathbf{X}} \\ \underline{\mathbf{X}}_1 \end{bmatrix}, \begin{bmatrix} \mathbf{y} \\ \mathbf{y}_1 \end{bmatrix} \right\} \Leftrightarrow \left\{ \begin{bmatrix} \tilde{\mathbf{P}}^T \\ \underline{\mathbf{X}}_1 \end{bmatrix}, \begin{bmatrix} \mathbf{b}^\perp \\ \mathbf{y}_1 \end{bmatrix} \right\}.$$

The tensor $\tilde{\mathbf{P}}$ is obtained from the matrix $\tilde{\mathbf{P}}$, with F as dimensionality of the first mode. Dimensions of the other modes equal to the ones of the tensor $\underline{\mathbf{X}}$. Thus, the

RNPLS algorithm inherits the tensor representation of data from the NPLS and allows effective adaptive learning, which is the proper of the recursive PLS.

3 Results

We tested the proposed RNPLS algorithm on artificial and real data sets. To study the performance and the prediction accuracy of the RNPLS, we compared it with the traditional NPLS algorithm on simulated data sets with different level of noise and for different amount of factors. Taking into account specificity of the binary-BCI task (described below), the tests were performed for tensor input and binary output variables. An artificial data set $\{\underline{\mathbf{x}}_k \in \mathbb{R}^{100 \times 200}, y_k \in \{0, 1\}\}_{k=1}^{1600}$ was created in the following way. Binary y_k were randomly generated with equal probabilities. Tensors $\underline{\mathbf{x}}_k$ were calculated according to $\underline{\mathbf{x}}_k = \underline{\mathbf{c}}(y_k) + \lambda \underline{\mathbf{\epsilon}}_k$, where $\underline{\mathbf{c}}(y_k) \in \mathbb{R}^{100 \times 200}$ is set as $c_{ij} = \cos(2.5\pi(i - 0.4j) + 2)$ if $y_k = 0$, and $c_{ij} = \sin(2.5\pi(i + 0.4j) + 1)$ if $y_k = 1$ (Fig. 1). The random noise $\underline{\mathbf{\epsilon}}_k \in \mathbb{R}^{100 \times 200}$ was drawn from a multivariate normal distribution $N(\underline{\mathbf{0}}, \underline{\Sigma})$, which is a generalization of Gaussian distribution for tensors [8]. It was added to the templates with parameter λ introducing signal-to-noise ratio: $\lambda = \{0.5; 1; 5; 10\}$. The noise has the same amplitude as the signal $\underline{\mathbf{c}}(y_k)$ in the case of $\lambda = 1$. The entire data set was split into training and test data sets of equal size.

The NPLS algorithm processed the whole training set. For the recursive calculation with RNPLS, the training set was split into 20 disjoint windows, each one containing 40 points. For all conditions (level of the noise and amount of factors), the experiment was repeated 10 times with new realizations of noise. Then the predictions of the output variable were averaged over these 10 experiments. The resulting percentage of prediction errors is shown in (Fig. 2). In all tests, the RNPLS demonstrated significantly smaller amount of factors, which are necessary for efficient prediction. Moreover, it showed a better robustness. Variation of prediction error was essentially smaller for the recursive algorithm for small amount of factors for the level of noise up to 500%. Advantages of the RNPLS can be explained by overfitting effect suppression.

At the second stage, the proposed RNPLS algorithm was tested with real data sets which were collected during BCI experiments with freely moving rat, carried out by the team of Clinatec/LETI/CEA. The animal can push a pedal mounted in a wall of the cage. Every pushing event activates the food dispenser. Animal's electrocorticogram (ECoG) signal and information about pedal's state were recorded simultaneously. The experiment lasted 10 minutes, the rat made 69 pushing. More detailed description of the experiment see in [1] and [2]. The purpose of the BCI is to predict the rat's intention to press the pedal from animal's neuronal activity. The purpose of the BCI calibration stage is the predictive model identification. The NPLS and the RNPLS algorithms were applied to solve the problem. For multimodal analyses, the recorded signal was mapped onto the spatial-temporal-frequency space by continuous wavelet transformation (Fig. 3).

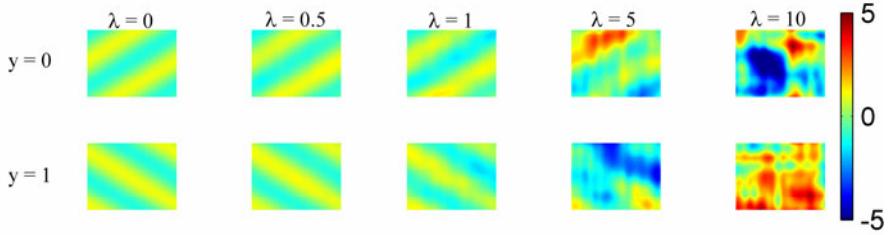


Fig. 1. Example of points $\{\underline{x}_k, y_k\}$ from the artificial dataset, with different levels of noise

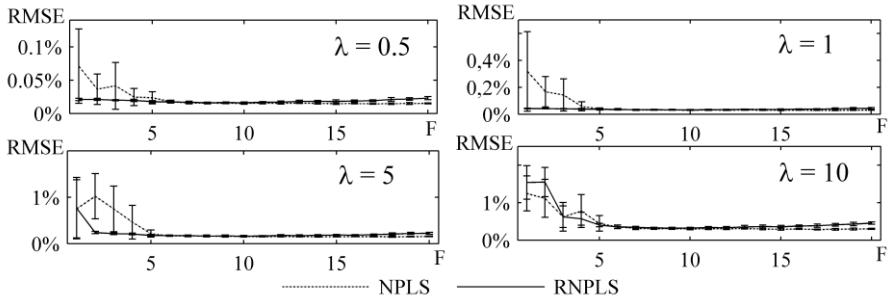


Fig. 2. Comparison of prediction errors (root mean squared error divided by the range of y , RMSE) for the NPLS and the RNPLS algorithms on the test sets for different levels of noise λ and different amount of used factors F

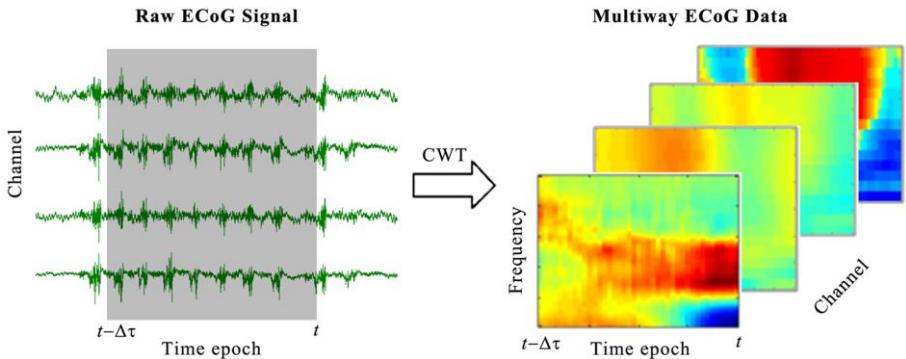


Fig. 3. Time epochs of multi-channel ECoG recording are mapped by continuous wavelet transform to the spatial-temporal-frequency feature space

1000 randomly selected points (700 correspond to “non-events” and 300 to “events”) formed the training set, whereas 400 points (300+100) were used as the test set. The NPLS algorithm was trained on the whole training set. For the RNPLS the training set was split into disjoint subsets with 10 and 100 points. Then projectors and predictive models were identified (Fig. 4). Fig. 5 shows the percentage of resulted

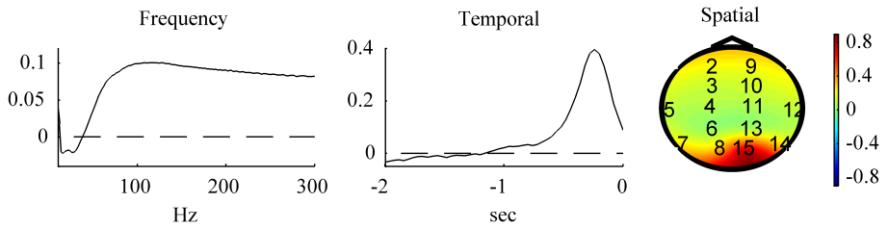


Fig. 4. Frequency, temporal, and spatial projectors of the first factor identified by the RNPLS (100)

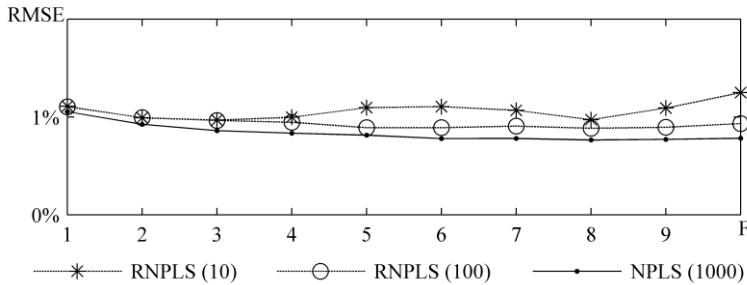


Fig. 5. Comparison of the test data prediction error (RMSE). RNPLS(10) – the training set is split into 10-points disjoint subsets; RNPLS(100) – the training set is split into 100-points disjoint subsets, NPLS(1000) – generic NPLS using whole training set.

prediction errors. With respect to the NPLS, the RNPLS algorithm demonstrates minimal deterioration in prediction quality: for the RNPLS (100) it is about 0.1%, whereas for the RNPLS (10) it is about 0.2%.

The proposed recursive algorithm demonstrated an excellent performance in comparison to the NPLS in terms of accuracy and convergence rate. The requirements for computation resources (memory) are low and do not depend on the size of processed data. Therefore, the method could be efficiently applied to the BCI systems.

Acknowledgments. The authors are grateful to C. Moro, N. Torres, T. Costecalde, S. Gharbi, G. Charvet, F. Sauter, J. Porcherot, A. Serpollet and C. Mestais, as well as to all members of Clinatec/LETI/CEA team, who organize and realize the experiments and provide the recordings used in these studies. The work was partially supported by project CE ICoBI, Nanosciences Foundation, RTRA.

References

1. Moro, C., Aksanova, T., Torres, N., Elisseyev, A., Costecalde, T., Charvet, G., Sauter, F., Gharbi, S., Porcherot, J., Mestais, C., Benabid, A.L.: First successful self-paced non-supervised ECoG based on-line BCI in freely moving rats performing a binary behavioral task during long term experiments. Submitted to J. Neurosciences (2011)

2. Eliseyev, A., Moro, C., Costecalde, T., Torres, N., Gharbi, S., Mestais, C., Benabid, A.L., Aksanova, T.: Iterative N-way PLS for self-paced BCI in freely moving animals. Submitted to *Journal of Neural Engineering* (2010)
3. Bro, R.: Multiway calibration. multilinear pls. *J. Chemom.* 10, 47–61 (1996)
4. Qin, S.J.: Recursive PLS algorithms for adaptive data modeling. *Computers Chem. Engng* 22, 503–514 (1998)
5. Geladi, P., Kowalski, B.R.: Partial least-squares regression: a tutorial. *Anal. Chim. Acta* 185, 1–17 (1986)
6. Dayal, B.S., MacGregor, J.F.: Improved PLS algorithm. *J. Chemometrics* 11, 73–85 (1997)
7. Kolda, T.G., Bader, B.W.: Tensor Decompositions and applications. Sandia report, SAND2007-6702 (2007)
8. Hamedani, G.G., Tata, M.N.: On the determination of the bivariate normal distribution from distributions of linear combinations of the variables. *The American Mathematical Monthly* 82, 913–915 (1975)

Transformation of Edge Weights in a Graph Bipartitioning Problem

I.M. Karandashev and B.V. Kryzhanovsky

Center of Optical and Neural Technologies of SRISA RAS

119333 Moscow, Vavilova St., 44 b. 2, Russia

Yakov.Karandashev@phystech.edu, Kryzhanov@mail.ru

Abstract. In this paper we consider the problem of partitioning a graph into two parts of equal sizes with minimal sum of edge weights between them. It is known that this problem is NP-complete and can be reduced to the minimization of a quadratic binary functional with constraints. In previous work it was shown that raising the matrix of couplings to some power leads to a significant increase of the basin of attraction of the deepest functional minima. This means that such transformation possesses great optimizing abilities. In this paper we show that in spite of the constraints present in the graph bipartitioning problem, the proposed matrix transformation approach works very well with this problem.

1 Introduction

Let $G = (V, E)$ be a graph with N vertices $V = (v_1, v_2, \dots, v_N)$ and weights of edges E , described by matrix of coupling T of size $N \times N$. Without loss of generality, we suppose that T is symmetrical and has zero diagonal.

In this paper we consider the problem of minimal cost partitioning of this graph into two parts of equal size, i.e., to divide the graph vertices V into two disjoint parts V_1 and V_2 such that the sum of the weights of the edges connecting the two parts is minimal:

$$\min_{V_1, V_2} \sum_{i \in V_1} \sum_{j \in V_2} T_{ij} \quad (1)$$

subject to

$$|V_1| = |V_2| = N/2 \quad (2)$$

Let's introduce new variables $s_i = \pm 1$ such that $s_i = 1$ if $i \in V_1$ and $s_i = -1$ if $i \in V_2$, then it is easy to show that problem (1)–(2) reduces to the minimization of the quadratic binary functional

$$E(S) = - \sum_{i=1}^N \sum_{j=1}^N T_{ij} s_i s_j, \quad s_i = \pm 1 \quad (3)$$

subject to

$$\sum_{i=1}^N s_i = 0. \quad (4)$$

To solve the graph bipartitioning problem various heuristic methods have been proposed, e.g.: Kernighan-Lin algorithm [1], Hopfield neural network dynamics [2], simulated annealing [3], spectral methods, various multilevel schemes [4], genetic algorithms [5], etc..

It is known that the problem of the minimization of (3) is of great interest because it presents a wide class of NP-complete problems.

Minimization of quadratic functional (3) in a configuration space is quite different from minimization in a continuous-variable space where it can be solved with the help of conventional methods. The problem of binary minimization is NP-complete in the general case, i.e., it does not have a polynomial-time solution algorithm. For this reason heuristic methods are used to solve it.

The functional (3) can be regarded as energy of the state of a neural network with N neurons (see Hopfield neural network [2]). Similarly to the alternating-variable (gradient) descent method, which can be applied to functional minimization in a continuous space, the asynchronous dynamics of the Hopfield neural net [2] is used in a discrete space. In works [6], [7] we evolve this theory and suggested a new kind of minimization algorithm, which minimize the new functional constructed on matrix $W = T^k$:

$$E_k(S) = -\sum_{i=1}^N \sum_{j=1, i \neq j}^N W_{ij} s_i s_j, \quad s_i = \pm 1$$

The solutions obtained due to minimization of this functional are used to find deep minima of the initial functional (3).

Analogously to papers [6-7] we propose a new double step algorithm for graph bipartitioning problem.

2 The Double-Step Algorithm

In most of cases, heuristic algorithms solve the problem (1)–(2) as is. In this paper we show how to change the matrix T so that the effectiveness of the already known algorithms increases.

To test our principle we chose the simplest algorithm of graph bipartitioning, which consists of the following:

Firstly, we generate at random an initial bipartition.

Then, we look for a pair of vertices from different parts such that their interchange decreases the value of the energy, (3). If such vertices exist we interchange them and look for the next pair. If there are no such vertices the algorithm terminates.

Because there are many local minima the procedure is repeated several times until the partition that satisfies the necessary value of cost function (3) is found or until the uptime ends.

This algorithm of local optimization is shown in Fig. 1. The computational complexity of the algorithm is $O(N^2)$.

Initial partition $S = (s_1, s_2, \dots, s_N)$, $s_i = \pm 1$ such that $\sum_{i=1}^N s_i = 0$

For each i from 1 to N

$$h_i = \sum_{j=1, j \neq i}^N T_{ij} s_j$$

stop = 1

While (*stop* > 0)

stop = 0

For each i from V_1

For each j from V_2

If ($h_i s_i + h_j s_j - 2T_{ij} s_i s_j < 0$)

$s_i = -s_i$

$s_j = -s_j$

For each k from 1 to N

$h_k = h_k + 2T_{ki} s_i + 2T_{kj} s_j$

Remove i from V_1 and add it to V_2

Remove j from V_2 and add it to V_1

stop = 1

endIf

endFor

endFor

endWhile

$$E = -\sum_{i=1}^N h_i s_i$$

Fig. 1. The local optimization algorithm

Hereafter we will consider only matrices of coupling that are random and may have both positive and negative elements T_{ij} . In this case, [6]–[7] read as follows:

1. In general case functional (3) has a great many of local minima in binary space of variables $s_i = \pm 1$. Different minima have different basins of attraction and, hence, different probabilities of being detected during a random search.
2. The radius of the basin of attraction of a minimum is proportional to the energy of this minimum and, therefore, the probability of finding a minimum exponentially increases with its increasing depth. I.e., during a random search it is the deepest minima (the best solutions) which are found with the highest frequency.
3. The energy landscape of functional (3) can be transformed so as to increase the basin of attraction of good solutions. by raising T to a power $k = 2, 3, \dots$.

4. Upon exponentiating of the matrix, minima not only change their depth, but also shift. However, the deepest minima of the functional with matrix $W = T^k$ are located close to the deepest minima of the initial functional (3).

5. As the dimensionality N of the problem increases, exponentiation of the matrix leads to more and more relative growth of the basin of attraction of deep minima.

All these statements were obtained for the problem of unrestricted binary functional minimization (3). However, we suppose the same rules are also true for problems with constraints, for instance, for graph bipartitioning problem. We know that during random search good partitions are found more often than bad solutions. And we think that raising the matrix to a power can significantly increase this chance.

Thus, our method consists of the following two steps:

The 1st step. We create a graph G_k with matrix of coupling T^k with zero diagonal. Generate at random an initial bipartition and, using the algorithm of local optimization described in Fig. 1, find an optimum bipartition S_k of graph G_k .

The 2nd step. Use the solution S_k obtained in step 1 as the *initial partition* of graph G . Again we apply the local optimization algorithm and find an optimal bipartition S of graph G .

If the obtained solution does not meet our wishes, we have to repeat this two-step procedure several times, i.e., use a random search with different starting points.

3 Obtained Results

To estimate the efficiency of the proposed method we performed several computer experiments. Graphs of two types were chosen:

The first type is a graph of the 2D Ising model [8], also known as Edwards-Anderson spin glass [9]. Graphs of this kind correspond to a two dimensional quadratic lattice, where couplings exist only between neighbouring nodes. The value of a coupling (weight of an edge) is a random normal quantity with zero mean. Such a graph is very sparse, i.e., the number of edges $E \ll N^2$.

Graphs of the second type have full matrices of couplings ($E \approx N^2$). Weights are quantities uniformly distributed within a symmetric range with zero mean.

The dimensionalities of the problems considered are $N = 100, 200$. We chose relatively small dimensionalities in order to be able to find the global minimum and thus estimate the efficiency of the method. As the ‘global’ minimum, we got the deepest minimum found. Because we run the program many times and the best output was found with high probability, so we can expect that it is the global minimum. So hereinafter we do not distinguish the found ‘global’ minimum and the true global one.

At first, let’s estimate the efficiency of the standard algorithm described in Fig. 1, without matrix exponentiation.

During experiments we performed 10^5 starts for 50 graphs of each type. One of the good parameters that characterize the efficiency of an optimization algorithm is:

$$\delta E = \frac{E_{gl} - E_{mean}}{E_{gl}} \quad (5)$$

where E_{gl} is the energy of the global minimum and E_{mean} is the mean energy of the local minima found. The relative difference δE is rather a changeless quantity, which is self averaging with increasing dimensionality N of the problem. For example, for graphs with uniform matrices the standard algorithm gives the following results:

$$\begin{aligned} \delta E &= 8.0\% \pm 1.0\% & \text{at} & \quad N = 100 \\ \delta E &= 8.5\% \pm 0.8\% & \text{at} & \quad N = 200. \end{aligned}$$

And for Ising graphs:

$$\begin{aligned} \delta E &= 10.4\% \pm 1.1\% & \text{at} & \quad N = 100 \\ \delta E &= 10.5\% \pm 0.9\% & \text{at} & \quad N = 200. \end{aligned}$$

Another parameter that characterizes the efficiency of an optimization algorithm is:

$$P = \Pr\{E \in [E_{gl}, 0.99E_{gl}]\},$$

the probability of finding very deep minima of energy $E \in [E_{gl}, 0.99E_{gl}]$, which is very close to the energy of the global minimum.

In further experiments we created a graph with a matrix T^k for $k = 2, 3, \dots, 5$. The partition of this graph was used as a preliminary partition to find the partition of the initial graph. The results obtained are shown in Table 1 and in Fig. 2.

Table 1. Results for method with matrix exponentiation. Results for matrices with uniform distribution are on the left side of the table, results for the Ising matrices are on the right. The dimensionality $N=100$ is shown on the top, and $N=200$ is on the bottom. For the description of the parameters see the text.

Uniform matrix, N=100		Ising matrix, N=100	
δE	$\Pr\{E \in [E_{gl}, 0.99E_{gl}]\}$	δE	$\Pr\{E \in [E_{gl}, 0.99E_{gl}]\}$
T	$8.0\% \pm 1.0\%$	0.05	$10.4\% \pm 1.1\%$
T^2	$5.0\% \pm 1.3\%$	0.16	$8.0\% \pm 1.3\%$
T^3	$2.8\% \pm 1.0\%$	0.31	$5.4\% \pm 0.9\%$
T^4	$4.3\% \pm 2.0\%$	0.23	$7.6\% \pm 1.3\%$
T^5	$2.4\% \pm 1.3\%$	0.32	$4.5\% \pm 0.9\%$
Uniform matrix, N=200		Ising matrix, N=200	
δE	$\Pr\{E \in [E_{gl}, 0.99E_{gl}]\}$	δE	$\Pr\{E \in [E_{gl}, 0.99E_{gl}]\}$
T	$8.5\% \pm 0.8\%$	0.01	$10.5\% \pm 0.9\%$
T^2	$5.0\% \pm 0.9\%$	0.08	$8.4\% \pm 1.0\%$
T^3	$2.8\% \pm 0.7\%$	0.19	$5.1\% \pm 0.7\%$
T^4	$4.5\% \pm 1.5\%$	0.12	$8.0\% \pm 1.0\%$
T^5	$2.5\% \pm 0.9\%$	0.21	$4.2\% \pm 0.7\%$

As we see from the table and from the diagram the average distance between the energy of the global minimum and the local minima decreases with the growth of the exponent. As a consequence, the probability of finding deep minima increases by orders of magnitude.

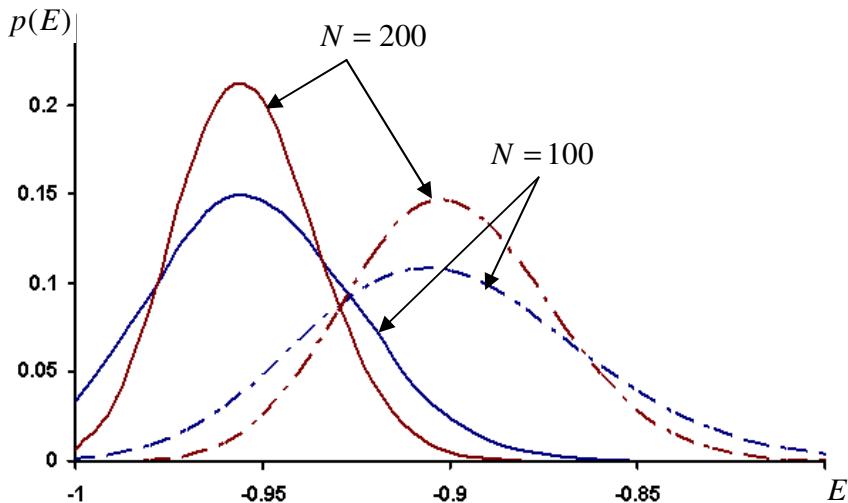


Fig. 2. Distribution of probability density for standard algorithm (dash-dot lines) and for new algorithm with cube of the matrix (solid lines), $N=100$ and $N=200$. Energy $E = -1$ corresponds to energy of the global minimum. The diagram represents experimental results for Ising type graph

4 Conclusions

In the paper we proposed a new method for graph bipartitioning. The method consists of the following two steps. Firstly, we suggest partitioning a graph with matrix of coupling T^k , where T is the matrix of coupling of the original graph we want to partition, and k is some exponent (for example, $k = 2, 3, \dots, 5$). Further, the solution obtained in the first step is used as an initial bipartition in the second step, which minimizes the cuts needed to partition the original graph.

We did not expand the theory of the double step algorithm with matrix transformation obtained in [6-7] for the case of “balanced” graph partitioning problem considered here.

But experiments showed that this two-step approach significantly raises the productivity of the simple algorithm for graph bipartitioning. Namely, the proposed transformation of edge weights decreases the mean energy of the local minima found. As a result, the mean energy becomes comparable with the energy of the global minimum and the probability of finding good solutions that are close to the optimum is increased by orders of magnitude (of course, if it was enough small).

In particular, for uniform matrices, the use of the method allows the following improvements: the difference between the average energy E_{mean} and the energy of the global minimum decreases by three times; and the probability that $E_m \in [E_0, 0.99E_0]$ (the energy interval around the global minimum) grows by more than a factor of 6 in case of $N = 100$ and by more than a factor 20 in case of $N = 200$.

As for Ising matrices, the method allows decreasing the difference between the average energy E_{mean} and the energy of the global minimum by half; and increasing the probability of finding very good solutions (such that $E_m \in [E_0, 0.99E_0]$) by more than a factor of 30 for $N = 100$ and by more than 2 orders of magnitude for $N = 200$.

All these results remain valid no matter what algorithm of local optimization for graph bipartitioning you use: the simplest greedy algorithm (that we used in our tests), Kernighan–Lin algorithm, simulated annealing, etc. Moreover, we suppose that adding Kernighan–Lin’s graph-partitioning algorithm instead of the simplest one will improve the algorithm considerably.

In this paper we considered only graphs in which the weights of edges may be both positive and negative. In practice there is often the need to partition graphs with only nonnegative edge weights. At the moment we don’t really know how our double-step approach works on such graphs. But we suppose that if in the 1st step of our approach matrix $(T - \bar{T})^k$ (where \bar{T} is the average value of matrix elements) is used instead of T^k then the new algorithm will again be effective.

Acknowledgements. This work was supported by RFBR grant # 09-07-00159a.

References

1. Kernighan, B.W., Lin, S.: An Efficient Heuristic Procedure for Partitioning Graphs. *Bell System Tech. Journal* 49, 291–307 (1970)
2. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. *Proc. Nat. Acad. Sci. USA* 79, P2554–P2558 (1982)
3. Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P.: Optimization by Simulated Annealing. *Science* 220(4598), 671–680 (1983)
4. Karypis, G., Kumar, V.: A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs. *SIAM J. Sci. Comput.* 20(1), 359–392 (1997)
5. Houdayer, J., Martin, O.C.: Hierarchical approach for computing spin glass ground states. *Phys. Rev. E* 64, 56704 (2001)
6. Karandashev, Y.M., Kryzhanovsky, B.V.: Transformation of Energy Landscape in the Problem of Binary Minimization. *Doklady Mathematics* 80(3), 927–931 (2009)
7. Karandashev, Y.M., Kryzhanovsky, B.V.: Binary Optimization: Efficient Increasing of Global Minimum Basin of Attraction. *Opt. Memory & Neural Net. (Information Optics)* 19(2), 110–125 (2010)
8. Hartmann, A.K., Rieger, H. (eds.): *Optimization Algorithms in Physics*. Wiley-VCH, Berlin (2001)
9. Hartmann, A.K., Rieger, H. (eds.): *New Optimization Algorithms in Physics*. Wiley-VCH, Berlin (2004)

A Distributed Behavioral Model Using Neural Fields

Mohamed Oubbati, Josef Frick, and Günther Palm

Institute of Neural Information Processing, University of Ulm.

Albert-Einstein-Allee 11. 89081 Ulm, Germany

{mohamed.oubbati, josef.frick, guenther.palm}@uni-ulm.de

Abstract. We investigate the use of neural fields for building a distributed behavioral model enabling several agents to move in a flock. No leader is required, and each agent is implemented as an independent element that follows its own behavioral model which is composed of four steering behaviors: *separation*, *cohesion*, *alignment* and *obstacle avoidance*. The synchronized motion of the flock emerges from combination of those behaviors. The control design will be discussed in theoretical terms, supported by simulation results.

Keywords: Neural fields, autonomous agent, group behavior, navigation.

1 Introduction

The creation and evolution of a group of animals, such as flock of birds or a bank of fishes, has received attention in several research fields such as artificial life [1], virtual reality [2], data mining [3] and robotics [4]. The aim is to understand how they stay close together, never collide with each other or with obstacles, but rather create a perfectly synchronized motion. In his pioneering work, Reynolds [5] proposed a distributed behavioural model for simulating the motion of a flock of birds. Each element (*boid*) of the flock is implemented as an independent actor that follows its own behavioral model composed of few rules, such as keep distance from other members, fly toward a goal and avoid collisions. Flake [6] added a fourth rule, “*view*”, that indicates that a boid should move away from any boid that blocks its view.

In this paper, we investigate the use of neural fields [7] to model some steering behaviors proposed by Reynolds in [8]. Neural fields are equivalent to continuous recurrent neural networks, in which neurons are laterally coupled through an interaction kernel and receive external inputs (stimuli). The correct choice of the parameters of the field enables the existence of one solution, called *single-peak* or *mono-modal* solution. In this solution, when an input of a stimulus is very large compared with the within-field cooperative interaction, a single-peak will be stabilized by interaction, and it remains there even if the stimulus is removed. The key idea of using neural fields in navigation is to provide sensory information as stimulus, and the position of the peak will decode the correspondant behavior [9,10,11]. In a previous work we investigated how neural fields can offer a solution for the problem of moving multiple agents as a team in formation [12]. The objective was to *acquire a target*, *avoid obstacles*, and *keep a geometric configuration* at the same time. The strategy was to design a leader which guides followers. In this work, no leader is available, and the synchronized motion of the “flock”

results from combination of every agent's steering behaviors. For an agent to participate in a flock, it adjusts its movements in accordance with the movements of its neighbors, i.e. stay close to its neighbors (*cohesion*), avoid collisions with them (*separation*) and move in their average direction (*alignment*) (Fig. 1). The global stimulus is designed by defining the relevance of each behavior relatively to the actual situation. We also consider *obstacle avoidance* with the highest priority among other stimuli. The control design will be discussed in theoretical terms, supported by simulation results.

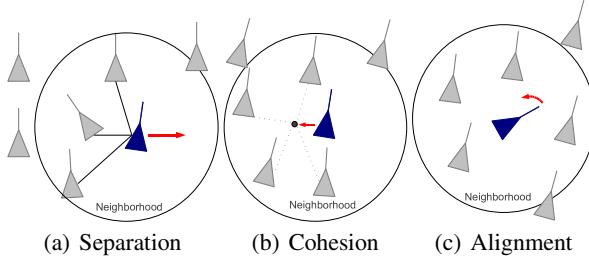


Fig. 1. Steering Behaviors

2 Behavioral Model

The equation of a one-dimensional dynamic neural field (DNF) is given by

$$\tau \dot{u}(\varphi, t) = -u(\varphi, t) + S(\varphi, t) + h + \int_{-\infty}^{+\infty} w(\varphi, \dot{\varphi}) f(u(\dot{\varphi}, t)) d\dot{\varphi} \quad (1)$$

The parameter $\tau > 0$ defines the time scale of the field, $u(\varphi, t)$ is the field excitation at the position (neuron) φ and at time t . We used the symbol φ for neurons, because each neuron will encode a possible heading direction for the agent. The constant h defines the resting activity level of the neurons, and $f(u)$ is the local activation function, usually chosen as a sigmoid function. The stimulus $S(\varphi, t) \in \mathbb{R}$ represents the input of the field at position φ and at time t . A nonlinear interaction between the excitation $u(\varphi, t)$ at the position φ and its neighboring positions $\dot{\varphi}$ is achieved by the convolution of an interaction kernel $w(\varphi, \dot{\varphi})$. That is, the activity change of a neuron φ depends on its actual activity level, the weighted input from other neurons, and the external input (stimulus) at its position. Depending on the parameter h , and the functions S , f , and w , equation (1) can produce a so called *single-peak* solution [7]. In this solution, when an input of a stimulus is very large compared with the within-field cooperative interaction, a single-peak will be stabilized by interaction, and it remains there even if the stimulus is removed. This interaction has also the effect to “push” the peak towards positions where the stimulus has maximum values. In our application the position of the peak decodes the movement direction, which is defined through distances and angles to the neighbors. The stimulus $S(\varphi, t)$ contains information about desired behaviors *separation*, *cohesion* and *alignment*.

2.1 Stimulus Design

We consider N_a autonomous agents moving in the plane, and each one updates its position and heading according to its neighbors. The neighbors of an agent i ($1 \leq i \leq N_a$) at time t are those which lie within a circle of radius r centered at the current position of the agent i . We denote by $N_i(t)$ the number of neighbors of the agent i at time t , such that

$$N_i(t) = \{\forall j / \|x_i^2(t) - x_j^2(t)\| + \|y_i^2(t) - y_j^2(t)\| < r\} \quad (2)$$

where $(x_i(t), y_i(t))$ is the position of the agent i at time t . The distance of an agent i to each neighbor $j \in N_i(t)$ at time t is

$$d_{ij}(t) = \sqrt{(x_i(t) - x_j(t))^2 + (y_i(t) - y_j(t))^2} \quad (3)$$

A behavior becomes active, if it holds the highest stimulus. We define λ_{min} as a threshold, such that if $d_{ij} < \lambda_{min}$, *Separation* is active. In this situation, each agent should move into the direction, which is opposite to the angle of the nearest neighbor. The behavior *Cohesion* aims to approach the agent to its neighbors as close as possible. This behavior becomes active when $d_{ij} > \lambda_{max}$, where λ_{max} is a predefined threshold distance. Finally, *Alignment* attempt to adjust the agent's direction according to the average direction of its neighbor's angle. The global stimulus is designed by defining the relevance of each behavior through the amplitudes $A_{separation}$, $A_{cohesion}$ and $A_{alignment}$ (Fig. 2). Obviously, *Separation* has the highest priority when $d_{ij} < \lambda_{min}$. On the other hand, when the agents are “too” far from each other, i.e. $d_{ij} > \lambda_{max}$, *Cohesion* behavior must take place, in order to keep the shape of the group. Finally, *Alignment* behavior occurs in the region when $\lambda_{min} < d_{ij} < \lambda_{max}$.

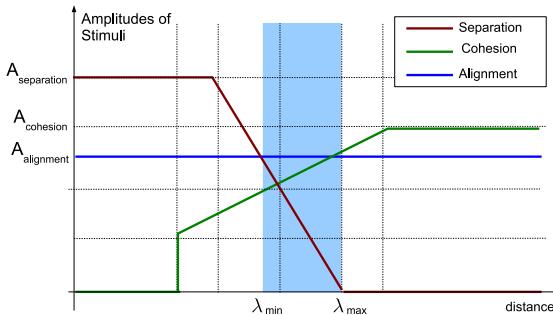


Fig. 2. Relevance of behaviors vs. distance between neighbors

3 Implementation and Results

We consider 4 agents, and each one has its own DNF. We choose the agents' headings φ_k ($k \in \{1, 2, 3, 4\}$) relative to a world-fixed reference direction as behavioral

variables. By means of a codebook we use N discrete directions from $-\pi$ to $+\pi$. We discretize (1) as

$$\tau \dot{u}_k(\varphi_i, t) = -u_k(i, t) + \sum_{j=1}^N w(\varphi_i, \varphi_j) f[u_k(\varphi_j, t)] + S_k(\varphi_i, t) + h \quad (4)$$

and we chose $N = 60$ neurons, which means that each direction N decodes a step of 6° , and the interaction kernel as:

$$w(\varphi_i, \varphi_j) = 2.8 \cdot \left(1 - \left(\frac{d(\varphi_i, \varphi_j)}{6} \right)^2 \right) \cdot \exp \left(- \left(\frac{d(\varphi_i, \varphi_j)}{2 \cdot 6} \right)^2 \right) + h \quad (5)$$

where $d(\varphi_i, \varphi_j)$ defines the distance between neurons φ_i and φ_j . The global inhibition $h = -1.3$ allows only one localized peak on the field. Based on the stimulus design described earlier, we implement the behaviors as (Fig. 3)

Separation:

$$s_{separation}(\varphi_i) = \max_k \left(A_{separation} \left(1 - \frac{2 \cdot d(\varphi_i, \phi_{sep})}{30} \right) \right) \quad (6)$$

where φ_{sep} is the position on the field decoding the opposite direction to the nearest neighbor.

Cohesion:

$$s_{cohesion}(\varphi_i) = A_{cohesion} \left(1 - \frac{2 \cdot d(\varphi_i, \phi_{middle})}{30} \right) \quad (7)$$

where φ_{middle} is the position on the field decoding the direction to the middle of the neighbors.

Alignment:

$$s_{alignment}(\varphi_i) = A_{alignment} \cdot \left(1 - \frac{2 \cdot d(\varphi_i, \phi_{flock})}{30} \right) \quad (8)$$

where φ_{flock} is the position on the field that decodes the flock direction.

At each situation the *flock* stimulus is chosen as

$$S_{flock} = \max(S_{separation}, S_{cohesion}, S_{alignment}) \quad (9)$$

The behavior *obstacle avoidance* is chosen as a Gaussian function centered at the direction of an obstacle φ_O

$$S_{obstacle}(\varphi_i) = C_O e^{-\sigma_O(\varphi_i - \varphi_O)^2} \quad (10)$$

where C_O and σ_O are positive constants, and σ_O defines the range of inhibition of an obstacle. The global stimulus of each agent will be then

$$S_{global} = S_{flock} - S_{obstacle} \quad (11)$$

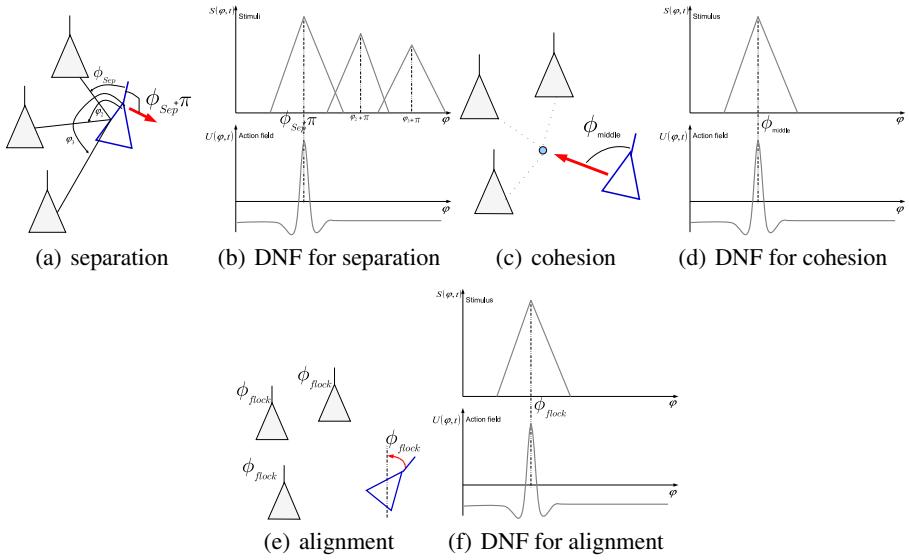
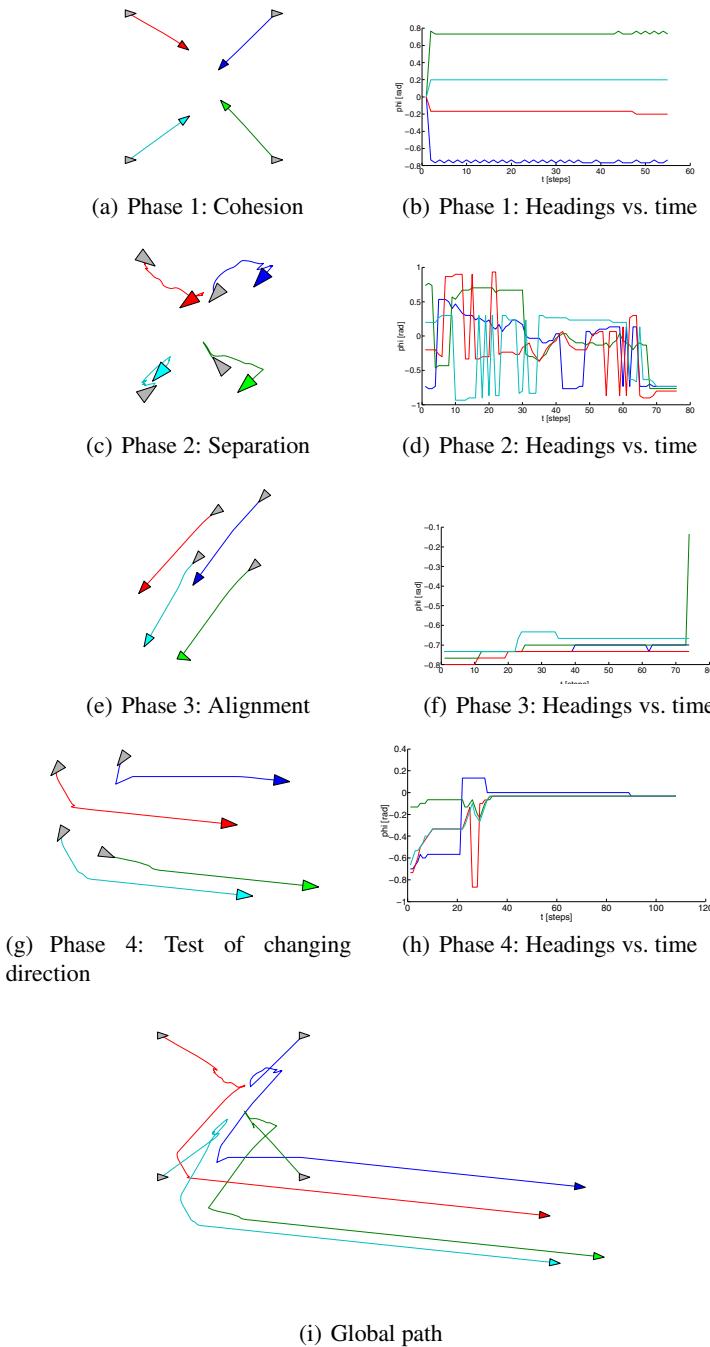


Fig. 3. Behaviors and their representation on the DNF.

After the stabilization of the field, the most activated neuron decodes the movement direction to be executed:

$$\varphi_{peak} = \arg \max \{u_k(\varphi_i) | i \in [1, N]\} \quad (12)$$

Two experiments are designed to test the validity of the behavioral model with DNFs. In each experiment the initial position and orientation are assigned beforehand. The first experiment (Fig. 4) consists in observing the agents' behavior in an obstacle-free environment. At the beginning the agents started with the *cohesion* behavior, in order to reach their desired positions in the flock (Phase 1). As they were too close to each other, their global stimuli were dominated by the *separation* stimulus, which brought each agent away from its neighbors (Phase 2). Once the flock was stabilized, each agent tried through *alignment* to follow the global direction of the group (Phase 3). To test robustness of the model we forced one agent to change its direction (Phase 4). The model could recover to this “perturbation”, and the agents adjust their positions and orientation according to this new situation, and maintain the flock motion. The global path is illustrated in Fig. 4. (i). The second experiment (Fig. 5) shows the case of flock motion with obstacle avoidance. There is an important distinction between *obstacle avoidance* and *separation* behavior. *Separation* is used to steer away from the predicted future position caused by the *cohesion*, whereas *obstacle avoidance* takes action only when a nearby obstacle is detected by the agent's sensors. Fig. 5. (c) shows how the contribution of all stimuli provide the appropriate heading directions, which permits to avoid the obstacle. After passing the obstacle the stimulus of obstacle avoidance is removed, and the group continues its movement by combining the three flock behaviors. The global path is illustrated in Fig. 5. (g).

**Fig. 4.** Flock Motion

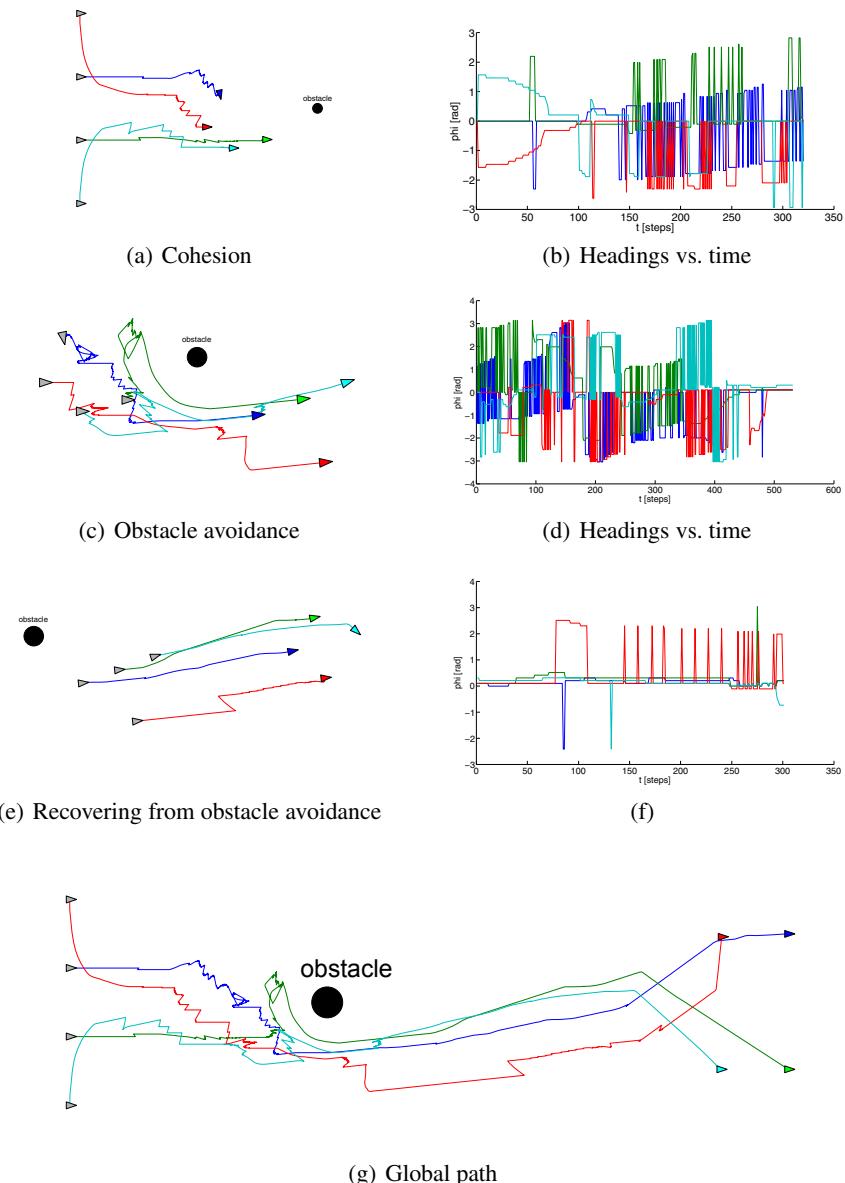


Fig. 5. Flock Motion with Obstacle Avoidance

4 Conclusion

We designed a behavioral model for flock motion using DNFs. No leader was available, and the synchronized motion emerged from combination of every agent's steering behaviors *separation*, *alignment*, *cohesion* and *obstacle avoidance*. These behaviors were first transformed to separate stimuli entries, and then combined in a global stimulus by assigning a situation-based priority to each behavior. The simulation tests demonstrate the feasibility of the approach, but a real-world implementation is still needed to confirm the effectiveness of the results achieved. Actually, we are working on implementing this design on 3 small real robots to test its performance in uncertain environments with real-time constraint.

References

1. Terzopoulos, D., Tu, X., Grzeszczuk, R.: Artificial fishes: Autonomous locomotion, perception, behavior, and learning in a simulated physical world. *Artificial Life* 1, 327–351 (1994)
2. Ulicny, B., Thalmann, D.: Towards interactive real-time crowd behavior simulation. *Computer Graphics Forum* 21(4), 767–775 (2002)
3. Abraham, A., Grosan, C., Ramos, V.: *Swarm Intelligence in Data Mining*. SCI, vol. 34. Springer, Heidelberg (2006)
4. Beni, G.: From swarm intelligence to swarm robotics. In: Şahin, E., Spears, W.M. (eds.) *Swarm Robotics 2004*. LNCS, vol. 3342, pp. 1–9. Springer, Heidelberg (2005)
5. Reynolds, C.W.: Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.* 21, 25–34 (1987)
6. Flake, G.W.: *The Computational Beauty of Nature: Computer Explorations of Fractals, Chaos, Complex Systems, and Adaptation*. MIT Press, Cambridge (2000)
7. Amari, S.: Dynamics of pattern formation in lateral-inhibition type neural fields. *Biol. Cybern.* 27, 77–87 (1977)
8. Reynolds, C.W.: Steering behaviors for autonomous characters. In: *Game Developers Conference*, pp. 763–782 (1999)
9. Schöner, G., Dineva, E.: Dynamic instabilities as mechanisms for emergence. *Developmental Science* 10(1), 69–74 (2007)
10. Oubbat, M., Palm, G.: A neural framework for adaptive robot control. *Journal of Neural Computing and Applications* 19(1), 103–114 (2010)
11. Sandamirskaya, Y., Schöner, G.: An embodied account of serial order: How instabilities drive sequence generation. *Neural Network* 23, 1164–1179 (2010)
12. Oubbat, M., Palm, G.: Neural fields for controlling formation of multiple robots. In: *International Symposium on Computational Intelligence in Robotics and Automation*, pp. 90–94. IEEE Computer Society Press, Los Alamitos (2007)

A Hypothetical Free Synaptic Energy Function and Related States of Synchrony

Karim El-Laithy and Martin Bogdan

Faculty of Mathematics and Computer Science

Dept. of Computer Engineering

Universität Leipzig, Germany

{kellaithy, bogdan}@informatik.uni-leipzig.de

Abstract. A simple hypothetical energy function is proposed for a dynamic synaptic model. It is an approach based on the theoretical thermodynamic principles that are conceptually similar to the Hopfield ones. We show that using this approach a synapse exposes stable operating points in terms of its excitatory postsynaptic potential (EPSP) as a function of its synaptic strength. We postulate that synapses in a network operating at these stable points can drive this network to an internal state of synchronous firing. The presented analysis is related to the widely investigated temporal coherent activities (cell assemblies) over a certain range of time scales (binding-by-synchrony). The results illustrate that a synaptic dynamical model has more than one stable operating point regarding the postsynaptic energy transfer. This proposes a novel explanation of the observed synchronous activities within networks regarding the synaptic (coupling) functionality.

Keywords: Dynamic synapse, synaptic energy, states of synchrony.

1 Introduction

Based on the lattice Ising model, Hopfield introduced his network as a recurrent neural network having symmetric synaptic connection pattern [1]. Within a Hopfield network there is an underlying monotonically decreasing energy function controlling the network dynamics [1]. Started in any initial state, the state of the system evolves to a final state that is a (local) minimum of the energy function. Each energy minimum acts as a point attractor in the space field of the system. It is an ongoing task in neuroscience to quantify and define those operating points, even with less regard to the underlying energy concepts. Many behavioural aspects were analyzed trying to relate these behavioural observations to known attractor dynamics, see e.g. [2,3]. It was illustrated that there is a class of features that can be described as dynamic attractors, referred as attractor networks. Levina et al. tackled the self-organizing behaviour regarding critical-stability in a fashion similar to the analysis of Avalanche dynamics [4]. It has been shown that a network interacting via timing-dependent synapses attains criticality in a certain region of the parameter space. That space is bounded by phase transitions that can be reached by tuning the synaptic strengths. The ensuing remarks from the above are: a) The physical system that may describe a biological neural system tries to find stable (or critically stable) operating points. These operating points act, in an abstract

way, as dynamic attractors. b) The search for such stable operating points is believed to be performed in a self-organized manner, i.e. either unsupervised or via reinforcement learning. Apart from those related to the Hopfield network, the idea of studying those operating points was, however, less studied from the energy point of view.

Some studies have investigated the concept of energy minimization and its relation to synchronized activity of networks acting as dynamic systems, see e.g. [5]. Torrealdea et al. studied extensively the energy and its related global balance in two bidirectionally coupled neurons [6]. It has been shown that in order to maintain the synchronized regime, there must be some kind of steady energy flow holding the coupling strength between the neurons. It has been shown that this flow of energy is also required for both the cooperative and collective behaviour in any neural ensemble. Furthermore, it was stated that this energy should be supplied (according to their analysis) through *the coupling mechanisms itself*, i.e. the synaptic connections [6]. This conceives the synapses to have a more vital role in the manifestation of synchrony in an artificial neural network than a passive dynamic coupling element.

Friston et al. have theoretically shown that for a biological system to hold its self-organizing behavior, seeking for stable operating points, should always exhibit a bounded interaction with *its environment* [7]. Under the condition that this bound quantity can be defined in terms of the internal states of the system. It was therefore proposed that the *energy* of the system that is *delivered* to this environment can represent this bound functionality, and that the system continuously tries to minimize it. It was referred as the Free-Energy principle. This bound (or exchange energy) is not the thermodynamical free energy of the system.

Therefore, here we inspect the energy-related intrinsic process that may be involved in the synaptic action. By this we extend the analysis made in [8] to determine the role of synaptic dynamics as whether it can be responsible for coherent activity, state of synchrony, and to what extent. We adopt the energy-based principle for biological neural systems from Friston et al. [7] and consider the synapse as a dynamical system. The analysis shows that a synaptic model with enough embedded dynamic features renders bistable energy levels as a function of its synaptic strengths. Thus, we postulate the argumentation that these stable energy levels are responsible for the observed stable synchronous discharge. The presented work aims at determining an analytical explanation of the role of the synaptic dynamic itself in achieving synchrony.

2 Model

The modified stochastic synaptic model (MSSM) introduced in [8,9] estimates the transmission probability of an arriving action potential, i.e. a spike, from a presynaptic neuron via a synapse to a postsynaptic neuron. Thus, $P(t_i)$ is the probability that the i th spike in a presynaptic spike train $\sum_i \delta(t - t_i)$ (input spikes) triggers the release of a vesicle at time t_i at the synapse. The involved probability-of-release $P(t) = 1 - \exp(-C(t) \cdot V(t))$ is governed by two counteracting mechanisms: facilitation and depression. Facilitation reflects the calcium concentration in the presynaptic neuron, $C(t)$, while depression represents the effect of the concentration of ready-to-release vesicles in the presynaptic terminal, $V(t)$. The model reads [8,10]:

$$\dot{C} = \frac{(C_o - C)}{\tau_C} + \alpha \cdot \sum_i \delta(t - t_i), \quad (1)$$

$$\dot{V} = \frac{(V_o - V)}{\tau_V} - P(t) \cdot \sum_i \delta(t - t_i), \quad (2)$$

$$\dot{N}_t = \max(0, -\frac{dV}{dt}) + \left(\frac{N_{to} - N_t}{\tau_{N_t}} \right), \quad (3)$$

$$\tau_{\text{epsp}} \dot{\text{EPSP}} = -\text{EPSP} + k_{\text{epsp}} \cdot N_t, \quad (4)$$

In Eq. 1, the intracellular calcium concentration starts at $C_o = 0.45$ ¹. It is raised incrementally by each stimulus impulse, approximated herein by a Dirac Delta function $\delta(t - t_i)$. The impact of each stimulus impulse to the intracellular calcium concentration is equal to the product of calcium gain (calcium current), α , caused by action potential and set to 0.095. Once the stimulus sequence ends, $C(t)$ decays with time constant $\tau_C = 3.34$ msec toward C_o . $C(t)$ represents, in an abstract way, the intrinsic synaptic processes of synaptic facilitation [9]. In Eq. 2, $V(t)$ is the expected number of vesicles of neurotransmitter molecules in the ready-for-release pool at time instant t . $V_o = 3.4$ is the maximum number of vesicles that can be stored in the pool. In a similar fashion to $C(t)$, $V(t)$ follows first-order nonlinear differential equations with a time constant $\tau_V = 9.95$ msec.

N_t determines the concentration of the released neurotransmitter in the synaptic cleft, Eq. 3. This concentration can be estimated by tracing the amount of vesicles of neurotransmitter that remains in the presynaptic neuron, $V(t)$, over time. It is worth mentioning that there is a formal distinction between release site and synapse. Alternatively, each quantum of neurotransmitter is stored in one synaptic vesicle. The concentration of neurotransmitter in the synaptic cleft is meant to be the corresponding concentration of quanta of neurotransmitter. Hence, in Eq. 3 we use here a first-order nonlinear differential equation similar in nature to Eq. 1. The incremental raise in this case is then the decrease in the concentration of vesicles (first term). The drift term (second term) allows the value of N_t to decay, in case of no input, to a minimum accepted concentration N_{to} with a decay time constant τ_{N_t} set to 0 and 0.05 sec respectively. This decay reflects the biological cleaning action (or complete removal) of the neurotransmitter from the cleft. As the binding process of neurotransmitter in the postsynaptic membrane induces EPSP, it is calculated as in Eq. 4; where $\tau_{\text{epsp}} = 0.075$ sec is a decay time constant and k_{epsp} is a scaling factor set to 13.33.

3 The Ansatz

It is important to clarify that there is a conflict between two conceptual considerations of what is the change in the state of synchrony within a network. The state change can be regarded as a phase transition [4] or through the thermodynamics-based approach by Friston et al. as a state transition [7]. Friston et al. argued that a biological system utilizes the *bounded* functionality to avoid what is called the "surprise in interaction"

¹ All values are arbitrarily chosen based on preliminary analysis, see [9].

with its environment. This surprise is, for example, the phase transition. Following this argumentation, the system (here the network) in general can selectively decide either to go through a phase transition or not, with high preference not to undergo a phase transition. The system tries to minimize (bound) this interaction, which is the energy exchange. State transition, therefore, is achievable if the energy dynamics of the system can reach more than one minimum as a function of its internal states. Here, the approach entertaining the potential of state transition by [7] is adopted.

The Postulates. Here, it is proposed that the aforementioned free-energy principle can be adopted describing the dynamics of the synaptic behaviour, specifically regarding the synchronous activity. Considering each synapse as a physical dynamical system, $EPSP$ is the only output exchange quantity between the synapse and a postsynaptic neuron. The electrical energy held by (and transferred via) $EPSP$, consequently, represents, in an abstract way, what we call here "the free synaptic energy" (FSE). It reads

$$FSE \equiv E_{Syn} = EPSP^2, \quad (5)$$

with units of energy per unit resistance ($J \cdot \Omega^{-1}$). In other words, the energy represented in $EPSP$ is the bound function that the biological system (here the synapse) tries to minimize by optimizing its internal states. These states (in the synapse) are the activity-dependent concentrations of the chemical constituents holding (and responsible for) the synaptic transmission. E_{Syn} does not represent the thermodynamic free energy of the synapse. The notation "free" is adopted since the synapse as a biological system tries to minimize this exchange energy in a fashion similar to that known from the second law of thermodynamics. This is performed through the minimization of the free energy of a system parallel to the maximization of its entropy. Based on the considerations above, we postulate the following:

- (a) A synapse is in continuous trial to bound (regulate) the interaction between itself and its postsynaptic neuron. Assuming that the synapse is an open biological system (from a thermodynamic point of view), the bounding is accomplished, through its inherited dynamics, by keeping its transferred output energy to its postsynaptic neuron (here $E_{Syn} = EPSP^2$ as energy per unit resistance) as minimal as possible. It is implicitly assumed that the synapse is able to change those controlling parameters affecting these dynamics (via e.g. learning or an update rule) in a self-organizing manner.
- (b) The synaptic transferred energy is a direct function of the internal synaptic electrochemical activities. These activities can be combined together in a non-explicitly determined nonlinear function to give the observable synaptic strength, d_{Syn} . As introduced in [4], the synaptic strength is: $d_{Syn} = X_{fac} \cdot X_{dep}$, where $X_{fac,dep}$ are the state variables of the dynamical system that represent facilitation and depression respectively. Applying this to the synaptic model, it follows: $d_{Syn} = C \cdot N_t$, where C , N_t are the internal counteracting dynamics of facilitation and depression (indirectly via the dependence of N_t on V) in Eq. 2 and 3.
- (c) At each minimum (either global or local, if any exists) along the trajectory of the state space in the E_{Syn} - d_{Syn} plane, the synapse, thus, realizes a stable level of energy transfer that represents a certain state of stable activity.

In order to summarize the idea, let us consider the following premises for a *fully recurrent network* consisting of n neurons interconnected with m synapses: a) An input with stationary randomly distributed (e.g. Poisson distributed with mean λ) inter-spike intervals (ISI) and a suitable background noise intensity are maintained, b) some synapses s , where $s \subset m$, are allowed to reach for a stable minimum of energy transfer provided that there are l neurons, where $l \subset n$, are directly connected to the s synapses, and c) these l neurons have enough excitatory synaptic connections with the rest of the neurons ($n - l$). We define:

Proposition 1. *For a given network with n neurons, if s synapses operate at any of the local energy minima and sustain stable E_{Syn} to a number of l neurons, then the rest of the network (($n - l$) neurons) are forced to follow gradually the stable activity of the l neurons. If the coherent activity of these neurons is observed over a time window \mathcal{W} , after a suitable time T , where $T \gg \mathcal{W}$, a general new state of synchronous discharge from all n neurons should be observed.*

4 Analysis and Results

The General Case. In Eq. 1 and 2, the input is represented as the Dirac-Delta function $\sum_i \delta(t - t_i)$, where t_i is the arrival time of the spike. The input to the synapse is considered to be a train of spikes that has a Poisson distributed ISI with a mean value of λ . If Δ_{isi} is the observed ISI, then $\langle \Delta_{\text{isi}} \rangle = \lambda$. In the implementation of the equations, stationarity is assumed. Taking that the instantaneous firing rate is changing slowly and let the discretization step be big enough for the occurrence of more than one spike; then, the input train of spikes can be replaced with the mean of the Poisson process λ^{-1} [9]. Thus, the dynamics of the synapse are function of what we call here the controlling parameters. They are: a) λ , the expected value (statistical mean) of ISI of the random process used for generating the input, b) the synaptic resources that are represented in the initial conditions for the internal concentration of chemical constituents (state variables), and c) the decay timing parameters.

In order to investigate the central hypothesis of *whether the FSE (E_{Syn}), as a function in the synaptic strength, has any local minima*, the synaptic dynamical system is solved to get the trajectories of solution. These trajectories relate the synaptic energy, E_{Syn} , to d_{Syn} , the synaptic strength. The system is solved using MATLAB (ode45 solver). The trajectories are illustrated in Fig. 1. Time span for solution is 150 msec. λ is chosen to allow that the effective overall mean firing rate in the input to be 100-110 Hz. Fig. 1a shows that the physical system of MSSM exposes two minima (indicated by the two small arrows). According to the above proposition, these minima are operating points at which the synapse maintains stable postsynaptic energy transfer to a postsynaptic neuron. The effect of changing the contribution of different values of the controlling parameters on the dynamics of MSSM is investigated, see Fig. 1. This figure illustrates the effect of changing different parameters in case of holding other at standard values. The input to the synapse is a spike train with a Poisson-distributed ISIs. In Fig. 1b, the effect of changing the overall mean input firing rate is illustrated by changing the expected value of the Poisson process of the spike generator, λ . In Fig. 1c, the effect of decreasing the available synaptic resources is illustrated. The synaptic resources are

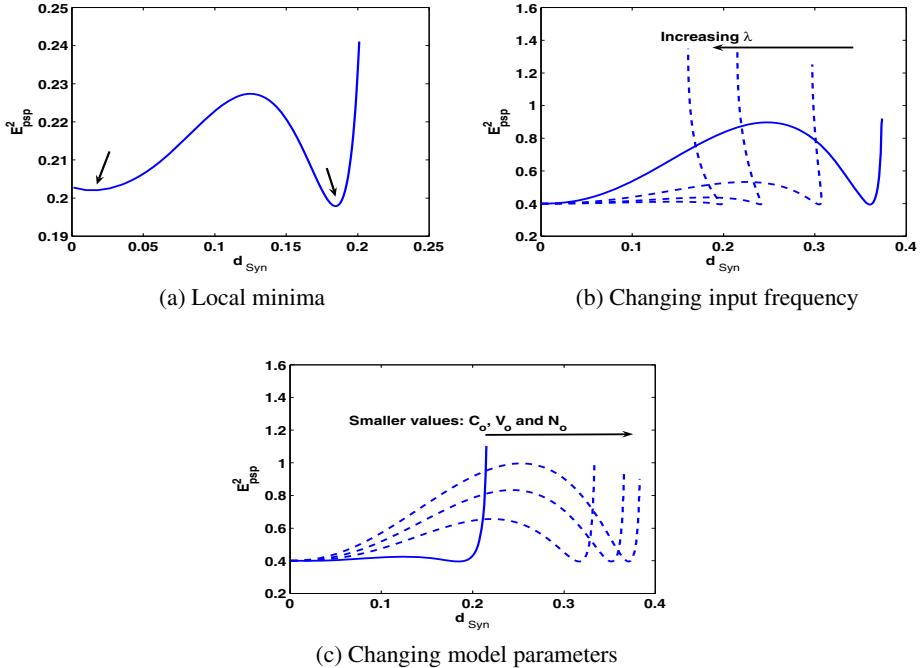


Fig. 1. Simulation of the trajectories describing the dynamics of MSSM in the $E_{\text{Syn}}-d_{\text{Syn}}$ plane. a) Trajectory relating the FSE vs. the synaptic strength for the used synaptic models with parameter values mentioned in section 2. The arrows indicate the minima of stable postsynaptic energy transfer. b) The effect of increasing the mean ISI of the Poisson distributed input spike train. c) The influence of decreasing some synaptic model parameters on the location of the energy minima.

the initial conditions for the differential equations describing the dynamics C , V and N_t for MSSM. These influences on the trajectories (that describe the regime of synaptic response) are similar to the expected effects of an update rule for the synaptic parameters. At certain initial conditions, the trajectory is totally shifted left showing a minimum in the negative range of d_{Syn} (Fig. 2a). The later point with negative synaptic strength reflects the idea that the synapse can achieve the stable behaviour with inhibitory activity as well. Fig. 2b illustrates the solution in case of limiting the time span for solving the system to be less than 50 msec.

5 Discussion

The results in Fig. 1 show that MSSM has indeed stable operating points in terms of its synaptic energy (E_{Syn}) as function in its own synaptic efficacy d_{Syn} . It exhibits a

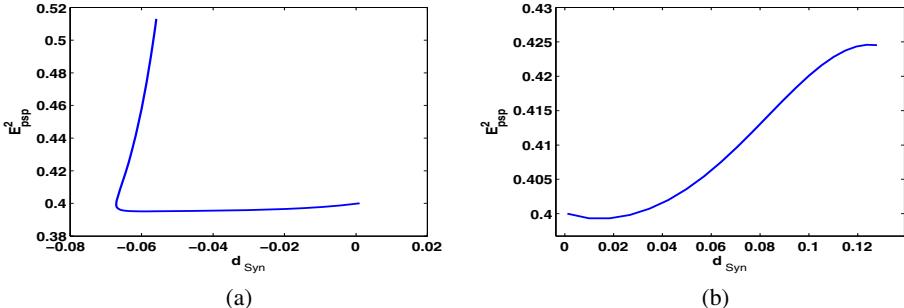


Fig. 2. Samples of the influence of the parameters controlling the synaptic dynamics on the trajectories describing MSSM. a) The negative synaptic strength indicates the inhibitory regime of the synaptic model. b) Effect of limiting the time span to 50 msec on the trajectory. It yields only one minimum as a stable operating point.

dichotomous capacity of operating points. Following the postulates in Sec. 3, a network that uses this synaptic model, on one hand, and when it features a learning update rule for its model parameters (see e.g. [11]), should be able to exhibit *at least* two levels of internal synchrony. A small condition should be added, these internal synchrony states can be observed if the window \mathcal{W} of observing this synchrony satisfies $\mathcal{W} > 100$ msec. Otherwise, the capacity of the network drops to one level of synchrony, as illustrated in Fig. 2b as the trajectory has only one global minimum. This time window constraint copes surprisingly with the statements from von der Malsburg [12,13] regarding minimum accepted window of synchrony for binding-by-synchrony and those recent argumentations by W. Singer [14]. The analysis here provides therefore a novel analytical explanation for the latter experimentally deduced time constraints.

These statements are also in agreement with many previous studies and findings. In [8,9], we have effectively shown that a small recurrent network of up to eight neurons with MSSM as synaptic connections depict two internal states (levels) of synchronous discharge among their neurons. That was achieved with one input neuron and the input was a train of spikes with Poisson distributed ISI. Updating the values of the synaptic parameters were done via basic Hebbian rules [9]. Hence, the introduced hypothetical free synaptic energy function and the shown stable operating point can be considered in general a novel analytical explanation to the experimentally and theoretically observed aspects of the relation between synaptic action and states of synchrony.

Regarding the capacity of states, specifying the expected total number of internal states of a network, adopting the introduced analysis, still represents an open task to be tackled in a succeeding study. Based on the minimum synaptic *dichotomous* capacity of minima, it can be shown that the maximum number of states is then $1 + S$; $S \geq 1$, where S can be a function of the network structure, update rule and input variability. The exact estimation of the maximum number of allowed states in the network is left for further study.

6 Conclusion

In this study, a hypothetical energy function is proposed to describe the energy flow via synaptic action to postsynaptic neurons. The trajectories relating this energy to the synaptic efficacy exposed stable operating points. These operating points are proposed to be responsible for the observed synchronous activity within neural networks. The analysis and results generally agree with the theoretical as well as the experimental discussions concerned with the generation of synchrony states, see e.g. [12,14]. The analysis explains and supports our previous results concerned with MSSM [8] via the novel treatment of synapses as dynamical systems with separate dynamic energy profile.

References

1. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America* 79(8), 2554–2558 (1982)
2. Eliasmith, C., Anderson, C.H.: *Neural Engineering (Computational Neuroscience Series): Computational, Representation, and Dynamics in Neurobiological Systems*. MIT Press, Cambridge (2002)
3. Stringer, S.M., Rolls, E.T., Trappenberg, T.P.: Self-organizing continuous attractor network models of hippocampal spatial view cells. *Neurobiology of Learning and Memory* 83(1), 79–92 (2005)
4. Levina, A., Herrmann, J.M., Geisel, T.: Phase transitions towards criticality in a neural system with adaptive interactions. *Physical Review Letters* 102(11), 860–875 (2009)
5. Sarasola, C., d'Anjou, A., Torrealdea, F.J., Graña, M.: Minimization of the energy flow in the synchronization of nonidentical chaotic systems. *Phys. Rev. E* 72(2), 26223 (2005)
6. Torrealdea, F.J., d'Anjou, A., Graña, M., Sarasola, C.: Energy aspects of the synchronization of model neurons. *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)* 74(1), 11905 (2006)
7. Friston, K.: The free-energy principle: a rough guide to the brain? *Trends in Cognitive Sciences* 13(7), 293–301 (2009)
8. El-Laithy, K., Bogdan, M.: Synchrony state generation in artificial neural networks with stochastic synapses. In: Alippi, C., Polycarpou, M., Panayiotou, C., Ellinas, G. (eds.) *ICANN 2009. LNCS*, vol. 5768, pp. 181–190. Springer, Heidelberg (2009)
9. El-Laithy, K., Bogdan, M.: synchrony state generation: An approach using stochastic synapses. *J. of Artificial Intelligence and Soft Computing Research* 1(1), 17–26 (2011)
10. El-Laithy, K., Bogdan, M.: Predicting spike-timing of a thalamic neuron using a stochastic synaptic model. In: *ESANN Proceedings*, pp. 357–362 (2010)
11. El-Laithy, K., Bogdan, M.: A hebbian-based reinforcement learning framework for spike-timing-dependent synapses. In: Diamantaras, K., Duch, W., Iliadis, L.S. (eds.) *ICANN 2010. LNCS*, vol. 6353, pp. 160–169. Springer, Heidelberg (2010)
12. von der Malsburg, C.: The what and why of binding: The modeler's perspective (September 1999)
13. Singer, W.: Neuronal synchrony: a versatile code for the definition of relations. *Neuron* 24, 49–65 (1999)
14. Singer, W.: Understanding the brain. *European Molecular Biology Org.* 8, 16–19 (2007)

Observational Learning Based on Models of Overlapping Pathways

Emmanouil Hourdakis and Panos Trahanias

Institute of Computer Science, Foundation for Research and Technology – Hellas (FORTH)
Science and Technology Park of Crete, GR70013, Crete, Greece
ehourdak, trahania@ics.forth.gr

Abstract. Brain imaging studies in macaque monkeys have recently shown that the observation and execution of specific types of grasp actions activate the same regions in the parietal, primary motor and somatosensory lobes. In the present paper we consider how learning via observation can be implemented in an artificial agent based on the above overlapping pathway of activations. We demonstrate that the circuitry developed for action execution can be activated during observation, if the agent is able to perform action association, i.e. relate its own actions with the ones of the demonstrator. In addition, by designing the model to activate the same neural codes during execution and observation, we show how the agent can accomplish observational learning of novel objects.

Keywords: Computational Brain Modeling, Observational Learning, Connectionist Model, Grasping Behaviors.

1 Introduction

Recent imaging experiments of the parieto-frontal cortex of Macaque monkeys performing grasping tasks have revealed the existence of an extended overlapping pathway of activations between action execution and action observation [1]. These areas include the forelimb representations of MI and SI cortices (both activated at 50% during observation), ventral (F5) area of the premotor cortex (100% activation during observation), as well as areas of the inferior (IPL) and superior (SPL) parietal lobes (each with 50% activation during observation).

The functional roles of the above mentioned regions during grasping tasks and their activation during observation, have led neuroscientists to believe that primates are using the circuitry developed for action execution in order to simulate an observed behavior and its anticipated consequences [1]. In the current paper, we consider the activation results reported in [1] in order to develop a computational model that accomplishes observational learning of novel objects, i.e. a model that is able to associate known actions to novel objects only by observation. Similarly to their biological counterparts, this is accomplished by activating the same extended circuitry of regions used for action execution during action observation. To enable learning during observation, the model encodes an observed behavior using the neural codes stored for its execution.

The importance of the overlapping activations between action execution and action observation in primates is well established in the computational modeling community. Previous recording studies in the cerebral cortex of macaque monkeys have revealed the existence of a specific group of neurons that discharge when the primate performs or observes a goal directed behavior [2]. These overlapping activations have provided computational modelers an important foundation for implementing imitation mechanisms in artificial agents. In this context, Oztop and Kawato developed a biologically inspired model of action observation based on the process of inferring the mental states of conspecifics [3]. A more biologically faithful model has been developed by Fagg and Arbib that focused on the parietal-premotor associations during primate grasping [4], which was also extended by Arbib in the Mirror Neuron System [5] to include regions of the primary and supplementary motor cortices. Finally, a model exploiting the overlapping activations in an extended network of brain areas in order to implement the reaching component of grasping acts has been developed in our previous work using genetic algorithms to tune model parameters [6,7].

2 Computational Modeling

In the current section we provide the design and implementation details of a computational model that replicates the results described in [1] in order to accomplish observational learning of novel objects. To activate the same neural codes during execution and observation we need to track how input/output information propagates within the model. For this reason we identify certain pathways within the model, i.e. sets of regions that participate in a certain transformation of information from one type to another. In the current model we define three different pathways: *(i)* object recognition, *(ii)* proprioceptive association and *(iii)* behavior learning. The following section provides details on the implementation of the input, neuron model and network topologies, as well as the design of the specified three pathways.

2.1 Input Encoding

Our simulated agent receives three types of input: *(i)* information regarding the objects present in the scene, *(ii)* proprioceptive input that indicates the joint positions of its fingers and *(iii)* information portraying the demonstrator's finger joint positions. All three aspects of input are encoded using population codes [8], i.e. each variable is encoded using a group of neurons, with each neuron being selective to a range of its values. For each object we encode a set of discriminative features, namely its XY axis ratio and number of corners. Behaviors (proprioception and demonstrator's motion) are encoded based on the joint value for each finger at each time step.

2.2 Neurons and Connections

The layout of the model is shown in Fig 1. Each area is replicated using a distinct neural network, comprising of interconnected neurons that are modelled based on the Leaky Integrate and Fire model suggested by [9].

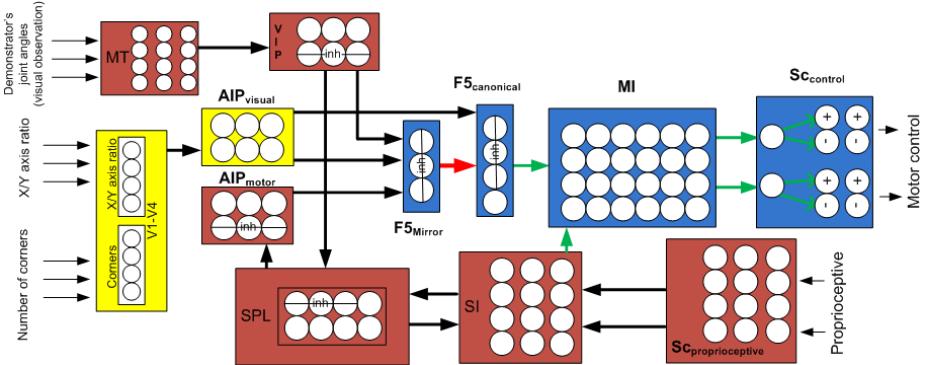


Fig. 1. Layout of the proposed model. The three pathways are marked with different colors; object recognition: yellow; proprioceptive association: red; behavior learning: blue. Different types of synapses are also marked with different colors: STDP: black, reinforcement: red, GA: green. The lines crossing the neurons in some networks (e.g. SPL) indicate the existence of lateral inhibitory connections in the respective networks

The inter and intra connectivity of each network is modelled using Spike Timing Dependent Synaptic Plasticity (STDP, [10]). Since STDP is an associative learning rule, the role of each network is to bind the information from its input to a common neural code. To form the associations between two neural networks, we use excitatory STDP synapses sparsely created from the neurons of one network towards the neurons of another. In addition, STDP is used to promote the competition between the neurons of the same networks using lateral-inhibitory connections. This is implemented as inhibitory STDP connections densely formed among the neurons of the same network (networks that employ lateral-inhibitory connections are marked with a line crossing their neurons in Fig. 1, e.g. SPL network). The lateral-inhibitory connections ensure that the dominant firing neurons of a network will suppress the stimulation of less active cells.

New behaviors are taught using a derivation of the BCM learning rule for the spiking neuron model [11]. This is implemented in the connections between the F5_{canonical}-F5_{mirror} neurons, and is described later in section 2.4.

2.3 Model Pathways

Following the design principle of pathways mentioned in the introduction, this section provides details about the implementation of the (i) object recognition, (ii) proprioceptive association and (iii) behavior learning pathways.

Object Recognition Pathway. The first entry point of information in the current model is through regions V1V4_{corners} and V1V4_{XYaxisRatio}. Those two networks are responsible for encoding the properties of the demonstrated object into population code. The output from those two networks is associated in region AIP_{visual} which during training forms neuronal clusters in response to its inputs. This is accomplished by connecting neurons that are close together with excitatory links, and neurons that are distant from each other with inhibitory synapses. The V1V4_{corners} and V1V4_{XYaxisRatio}

networks are densely connected to the AIP_{visual} region, so that when an object is viewed by the agent more than one cluster of neurons is activated. These compete during training (through their inhibitory connections), and the dominant cluster suppresses the activation of others. To ensure that diverse objects are clustered in different topological regions in the network space of AIP_{visual}, the weights of all synapses of a neuron in AIP_{visual} from the V1V4_{corners} and V1V4_{XYAxisRatio} networks are normalized in the [0..1] range. As a result, when a certain neuron in the input strengthens its connections with a specific cluster, it also suppresses the strength of the connections between that cluster and the remaining neurons in the input.

Proprioceptive Association Pathway. This pathway includes regions Sc_{proprioceptive}, SI, SPL, AIP_{motor}, VIP and MT, and is assigned two tasks: (i) to form the neural codes that represent the motion of the fingers of our cognitive agent and (ii) to build a correspondence between the agent's and the demonstrator's actions (through the VIP-SPL circuit).

The process that allows the formation of the *proprioceptive codes* of the pathway involves the Sc_{proprioceptive}, SI, SPL, AIP_{motor} and VIP regions. Sc_{proprioceptive} contains three sub-populations, each assigned to one of the index, middle and thumb fingers and projects to the SI network which also contains neuron sub-populations assigned to specific fingers. In turn each sub-population in the SI network projects to a different cluster of neurons in SPL. Finally, SPL projects to the AIP_{motor} network, which through its connections with F5_{mirror} provides information on the proprioceptive codes of the agent when generating a behavior. The neural codes formed in regions SI, SPL and AIP_{motor} become progressively sharper (i.e. less distributed and concentrating more on the peaks of their tuning curves) as they are projected from the one region to the other.

In addition to the formation of the proprioceptive codes, the proprioceptive association pathway is also responsible for the *action association* function. This is accomplished using the SI-SPL-VIP circuitry. SPL, apart from SI, also accepts connections from VIP, i.e. the region encoding a distributed representation of the demonstrator's active fingers. These synapses (VIP-SPL) undergo a competition process which aims at associating the neural representations of the agent's fingers with the neural representation formed for the demonstrator's fingers. This is accomplished through the MT network which projects directly to VIP, with excitatory STDP synapses, resulting in VIP encoding a distributed representation of the demonstrator's motion. This representation is associated with the actions of the agent, using the synapses between VIP and SPL based on a competition mechanism, that normalizes the weights of all synapses (in respect to the sum of their presynaptic weights) leading to the same neuron in SPL from VIP in the [0..1] range. Because of the above process, after learning a behavior, the agent is able to identify it during observation using the same neural codes used for its execution.

Behavior Learning Pathway. The behavior learning pathway exploits information from the previous two pathways in order to observe and execute a behavior using the same networks. The entry point for the behavior learning pathway is in the F5_{mirror} network which accepts connections from: (a) AIP_{motor} that provides information about the motor behavior that is being executed by the agent, (b) VIP network encoding the current demonstrated behavior, and (c) AIP_{visual} which provides information about the

viewed object. To make mirror neurons respond only to transitive actions, i.e. only when an object is present in the scene, we normalize the input current sent to the $F5_{\text{mirror}}$ neurons from the AIP_{motor} , AIP_{visual} and VIP regions, to appropriate ranges, and thus force the neurons in the $F5_{\text{mirror}}$ network to become active when the AIP_{visual} (object present) and at least one from the AIP_{motor} (executing) or VIP (observing) networks is active. More details on how the $F5_{\text{mirror}}$ neurons are programmed, as well as a discussion on related theories regarding the formation of mirror neurons are given in [12].

2.4 Motor Control of the Simulated Agent

The Sc_{control} neural network, which controls the fingers of our simulated agent, accepts neuron signals from the MI neural network which encodes in a distributed manner the input signals received from $F5_{\text{canonical}}$. The $F5_{\text{canonical}}$ network contains three neurons, each corresponding to a specific finger in the simulated body of the agent. The $F5_{\text{canonical}}\text{-}MI\text{-}Sc_{\text{control}}\text{-}SI\text{-}Sc_{\text{proprioceptive}}$ circuitry (motor control circuitry) is evolved using genetic algorithms so that when a neuron in the $F5_{\text{canonical}}$ network is active the finger assigned to that neuron will move. The complete layout of the motor control circuitry is described more thoroughly in [12].

New behaviours are taught through the same circuit for execution and observation, using the connections between the $F5_{\text{mirror}}\text{-}F5_{\text{canonical}}$ neural networks. The synapses between those networks are adjusted in a series of observation/execution cycles (execution phase), using reinforcement learning. To calculate the reward signal, we assign a binary variable the value of 0 or 1 depending on whether the demonstrator's and observer's fingers are moving. The reward signal is simply the subtraction of the two variables, rescaled to the $-0.5..0.5$ range. The synapses of the $F5_{\text{mirror}}\text{-}F5_{\text{canonical}}$ neurons are updated using a derivation of the BCM learning rule for spiking neurons [11].

3 Results

The experimental setup consists of two simulated robots. The first is assigned the role of the demonstrator and the second the role of the observer. The experiments consist of two phases, execution and observational learning. During the execution phase the demonstrator exhibits two behaviors to the observer, each associated with a different object. During the observational learning phase a novel object is shown to the observer and the demonstrator exhibits one of the behaviors taught during the execution phase. The goal for the observer is to identify the perceived behavior and associate it with the novel object on the scene.

Behavior Learning. After training, given a known object the agent is able to select and execute the correct behavior without any assistance from the demonstrator (i.e. the MT and VIP networks are inactive). During the execution phase the model activated regions SPL , IPL , MI and SI at a lower rate compared to the activation level during execution (Fig 2).

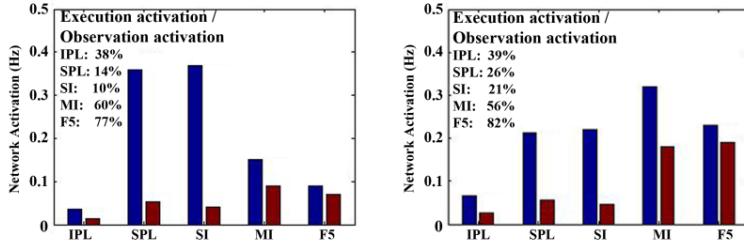


Fig. 2. The activations of the IPL, SPL, SI, MI and F5 networks, during the execution(blue bars) and sole observation cycle (red bars). The left activation plot shows the network activations during the first behavior (close middle and thumb), while the right plot shows the network activations during the second behavior (close index).

Computationally we can attribute the network activations during sole observation to the active visual input of the model, originating from regions $V1V4_{\text{corners}}$, $V1V4_{\text{XAxisRatio}}$ and MT. During observation, the agent is still shown the object and demonstrated with the associated behavior and consequently connections between the SPL-SI, SI-MI and SPL-AIP_{motor} networks activate the latter networks in each pair.

More importantly, a comparison of the active neurons during observation and execution indicates that the above mentioned networks activate the same neurons during the two phases (Fig 3).

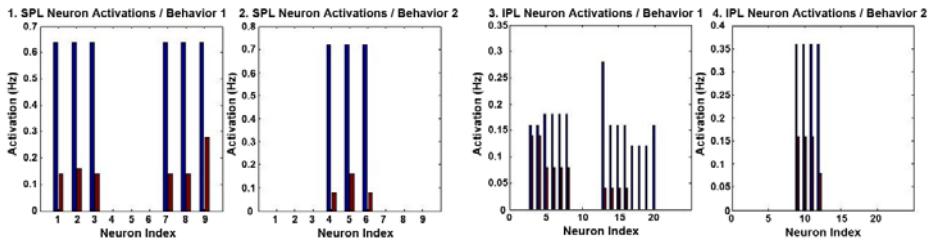


Fig. 3. Neuron activations for the execution (blue bars) and sole observation(red bars) phases for the SPL network during the first (plot 1, close middle and thumb) and second (plot 2, close index) behavior and IPL network during the first (plot 3, close middle and thumb) and second (plot3, close index) behavior

Observational Learning. During this phase the goal is to assess the ability of the agent to associate a novel object with the one of the behaviors taught during the execution phase and subsequently be able to execute this behavior whenever this object is presented.

Fig 4 illustrates the behavior executed by the agent when viewing a novel object before (left) and after (right) the observational learning phase of training.

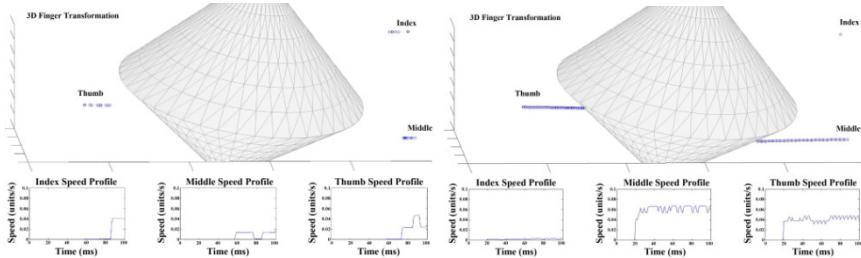


Fig. 4. Trajectories and speed profiles of the three fingers of the agent in response to a novel object before (left) and after (right) the observational learning phase. Both plots show the world coordinates of the three finger tips, along with a wireframe, transparent version of the object. (above), and the velocity profiles for the index, middle and thumb fingers during the 100ms cycle (below).

Finally, we note that after the observational learning phase, the agent is still able to execute the two behaviours learned during the execution phase. This is due to the fact that in the AIP_{visual} network, different objects activate different clusters of neurons. Thus when the novel object is presented during the observational learning stage, it activates a different cluster in the AIP_{visual} network. Consequently, learning of the new behaviour during the observational learning stage will employ a different set of synapses between the AIP_{visual} and $F5_{canonical}$ networks and will not interfere with the synapses used in previous behaviours.

4 Discussion

Recent neuroscientific experiments investigated the activation of regions in the brain cortex of primates when observing or executing a grasp behavior [1]. Results from these studies indicate that the same pathway of regions was activated during observation and execution of a behavior. This has led researchers to believe that during observation the primate is internally simulating the observed act, using its own cognitive circuits to comprehend it. The current paper, based on the neuroscientific results of [1], suggests how this extended overlapping pathway can be reproduced in a computational model of action observation/execution, and used for implementing learning during sole observation. Results from the model evaluation indicate that by shunting the motor output of an observing agent, there are several regions in the computational model that are activated at a lower firing rate compared to their activation during execution. Computationally, and maybe biologically, the lower activations that those networks exhibit are attributed to the fact that during sole observation the proprioceptive input of the agent is not available. As a result, during observation the number of active afferent projections towards each network is smaller, and therefore its activation is lower. The activations during observation are accomplished using an action correspondence circuitry, within the proprioceptive association pathway, that allows the agent to learn to associate its own joint angles with the joint angles of the demonstrator during the execution phase. The fact that this

circuitry activates the same neural codes during observation and execution facilitates learning during observation.

The current work constitutes an initial attempt towards the derivation of a detailed computational model of observational learning using the overlapping pathway of activations between execution and observation. We are currently exploring how the aforementioned activations in the somatosensory cortex and parietal lobe can be used in order to extend the observational learning capabilities of an agent to learning novel actions only by observation.

References

1. Savaki, H.E., Raos, V., Evangelou, M.N.: Observation of action: grasping with the mind's hand. *Neuroimage* 23(1), 193–201 (2004)
2. Rizzolatti, G., Craighero, L.: The mirror neuron system. *An. Rev. of Neuroscience* 27 (2004)
3. Oztop, E., Wolpert, D.M., Kawato, M.: Mental state inference using visual control parameters. *Cognitive Brain Research* 22, 129–151 (2005)
4. Fagg, A.H., Arbib, M.A.: Modeling parietal-premotor interactions in primate control of grasping. *Neural Networks* 11, 1277–1303 (1998)
5. Oztop, E., Arbib, M.A.: Schema design and implementation of the grasp-related mirror neuron system. *Biological Cybernetics* 87, 116–140 (2002)
6. Hourdakis, E., Maniadakis, M., Trahanias, P.: A biologically inspired approach for the control of the hand. *Congress on Evolutionary Computation* (2007)
7. Hourdakis, E., Trahanias, P.: A framework for automating the construction of computational models. *Congress on Evolutionary Computation* (2009)
8. Rieke, F., Warland, D., de Ruyter van Steveninck, R., Bialek, W.: *Spikes: Exploring the Neural Code*. The MIT Press, Cambridge (1999)
9. Stein, R.B.: Some models of neuronal variability. *Biophysical Journal* 7, 37–68 (1967)
10. Song, S., Miller, K.D., Abbott, L.F.: Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nature Neuroscience* 3, 919–926 (2000)
11. Baras, D., Meir, R.: Reinforcement learning, spike-time-dependent plasticity and the BCM rule. *Neural Computation* 19, 2245–2279 (2007)
12. Hourdakis, E., Savaki, H., Trahanias, P.: Computational modeling of cortical pathways involved in action execution and action observation. *Journal of Neurocomputing* 74, 1135–1155 (2011)

On the Capacity of Transient Internal States in Liquid-State Machines

Karim El-Laithy and Martin Bogdan

Faculty of Mathematics and Computer Science

Dept. of Computer Engineering

Universität Leipzig, Germany

{kellaithy, bogdan}@informatik.uni-leipzig.de

Abstract. Liquid-state machines (LSM) represent a class of neural networks that are able to introduce multitasking by implicit representation of input information over the entire network components. How exactly the input information can be represented and how the computations are accomplished, stay however unresolved. In order to tackle this issue, we demonstrate how LSM can process different input information as a varying set of transiently stable states of collective activity. This is performed by adopting a relatively complex dynamic synaptic model. Some light is shed on the relevance of the usage of the developed framework to mimic complex cortical functions, e.g. content-addressable memory.

Keywords: Dynamic synapses; liquid state machines; transient states.

1 Introduction

How can a neural system, either biological or artificial, with a real finite physical state space be related to the infinite cognitive state space? The simplest answer is that each existing physical representation in the network should be able of the perception and delivery of more than one behavioural or cognitive action, i.e. real-time multitasking [1,2]. This implies that the same physical system (here to be an artificial neural network) carries out implicitly multiple computational tasks within the same activity period that corresponds to certain input. I.e. different readout outputs can be processed out from the same machine status. Taking the feature "multitasking" as a key property, liquid-state-machines (LSM) proposed by Maass [3] represent a suitable materialization of the answer to the above question. LSM are an abstraction of the possible techniques used in the biological organism to *compute*. This class of state-machines accounts for time as a major factor in representing input and getting a response. The term *liquid* in this context refers to two main issues: First, the relation to biology, and second, that the collective (overall) physical state changes during the continuous presentation of input [4]. The general framework and implementation of LSM in [3] has shown that multitasking is achievable. A single recurrent network arranged in a column was used as a core engine that processes certain input spike trains. The activity of this column is then read from a number of side network (readout maps). Each readout map is trained for a specific task. All these readout maps get their inputs from the same pool (but not from the same neurons) of activity and transfer this activity to different responses [3,5]. This

implementation reveals many issues, mainly that mapping from an infinite state space to the finite physical state space is achievable provided that enough transformation side systems (readout maps) are available.

LSM allow to benefit from the wide range of already available computational representations for the building components of a neural network. It has been shown mathematically [4] that the implementation of LSM with various computational representation does not alter the main properties of the LSM (e.g. multiprocessing and multitasking abilities), but may affect the computational power of the system. AP, on the other hand, addresses the ability of the system via the readout maps to transform its internal activities into predefined outputs.

2 Liquid-State Machines and Internal States

The discussions concerned with LSM did not investigate adequately, however, an important issue that characterizes the biological counterparts: the incorporation of collective behaviour, known as brain states or neural states, in the computations. In other words, how are neural states represented in the system? How do they evolve in time? And can the state capacity affect and be related to the computational power? From a biological perspective the experiment and theoretical evidences presume that the collective behaviour (also called binding) and related brain states are major features that explain the ability of computations in neural systems [6,7,8,9,10].

Although the real mechanism underlying neuronal binding is not completely clarified, a couple of important issues have been proposed [11]: a) Any cortical area computes adaptively according to the immediate sensory input. b) This adaptive processing ability is inherited from the ability to realize more than one state of activity. c) These collection of states correspond to various computational algorithms that are in general confined by the behavioural context. Moreover, evidences are accumulating that cortical regions communicate in the temporal domain depending on the synchronization of activity [12]. This statement is widely supported through both experiment studies, see e.g. [13,14], and theoretical analysis, see e.g. [15].

Hence, the capacity of any neural system to compute is then related to its capacity of representing its sensory information on a various set of collective states of synchrony; these aforementioned states reflect, in an abstract way, the overall available algorithms for computations. Studies introducing LSM have investigated only one aspect of temporal features that is the consideration of time factor in general [5]. This does not help in probing the power of the LSM in terms of its capacity to internal representation of the information as *internal states*.

We have proposed a study that compared between performances of a neural network seeking to realize a state of synchronous activity depending on the utilized synaptic representation [16]. Specifically, how the synaptic representation may affect the exhibition of a state of synchronous activity with the neural network. Two synaptic representations have been used: the well established synaptic model from Markram et al. [17]¹ and our modified stochastic synaptic model (MSSM) [18]. It has been shown that the utilization

¹ This is the model used for the LSM implementation in [3,4,5].

of MSSM in the simulations provides the network with the ability to realize two different *transiently stable states* of synchronous activity while the network using Markram's model can show only one state of synchrony. At this point, there should be a formal distinction between the state of a network (or system) and its activity. On one side, the spiking activity across all neurons in the network at any time instant are called here the network activity. On the other hand, we use the term state to indicate that a group of neurons collectively and synchronously act through their spiking activity.

Here, we investigate the added values to the LSM, as a computational framework, by utilizing the MSSM in the implementation. The analysis is concerned only with the SP of the LSM. However, this is not meant to get a better performance in classification. We use the same measures used in [3] in indicating the performance of separation. We propose that these measures indicate the existence of stable transient states of collective activities. As the work presented here is a propositional study, we shed some light on the relevance of accomplishing such capacity of transient states. This capacity enables LSM to address more complex cortical functions, e.g. memory, and specifically content-addressable memory.

3 Simulation Results and Discussion

We apply a similar network structure to the one used in [3]. Thus, 135 neurons are randomly connected in a single column ($15 \times 3 \times 3$), see Fig. 1a, of which 20% are randomly chosen to be inhibitory. Regarding connectivity structure, the probability of a synaptic connection from neuron a to neuron b (as well as that of a synaptic connection from neuron b to neuron a) was defined as $A \cdot e^{-(D(a,b)/\lambda)^2}$, where λ is a parameter that controls both the average number of connections and the average distance between neurons that are synaptically connected. $D(a,b)$ is the Euclidean distance between neurons a and b . Depending on whether a and b are excitatory (E) or inhibitory (I), the value of A is 0.3 (EE), 0.2 (EI), 0.4 (IE) and 0.1 (II) [3]. The neurons are modelled as leaky Integrate-and-Fire units and the synapses are set to MSSM ones. The formulations of both the neuronal and synaptic models are listed in Appendix I. The input signals spike trains with Poisson distributed inter-spike intervals generated such that the have a certain overall firing frequency. The epoch of the input signals is 500 msec and the firing frequencies are 50-100 Hz with 10 Hz steps.

The generated input spike trains with different firing rates are injected in separate trials as input to the described network. The resulting neural activity is recorded for each of these inputs. An output, e.g. y_{50} represents the time-varying liquid state corresponds to the input x_{50} which is the input with firing rate of 50 Hz. The average distances² between the inputs over 200 randomly generated pairs $\|x_i(t) - x_j(t)\|$ are plotted in Fig. 1b, where $i, j = 60-100$ Hz with 10 Hz steps and $i \neq j$. $\|\cdot\|$ denotes the Euclidean norm. The average liquid state differences caused by the same spike train applied with

² In order to define the distance $d(m, n)$ between two spike trains m and n , each spike is replaced by an exponentially decaying function $\exp(-t/\tau)$ for $\tau = 5$ ms (m and n are convolved with the exponential kernel) and defined $d(m, n)$ as the distance of the resulting two continuous functions in the L_2 -norm (divided by the maximal lengths 0.5 sec of the spike trains m and n).

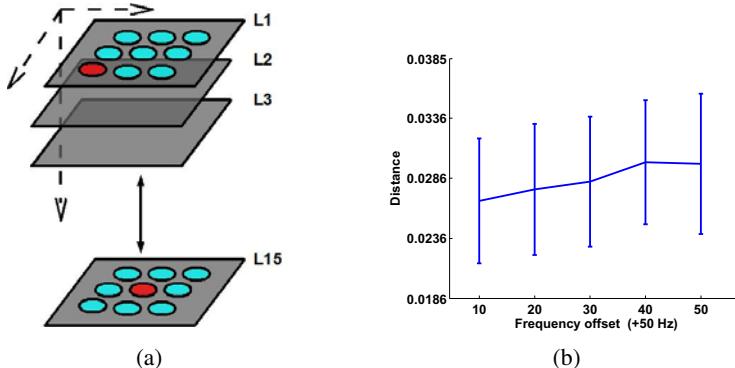


Fig. 1. a) Schematic of network setup. Gray circles represent excitatory neurons while dark ones indicate inhibitory neurons (20% of excitatory ones). The synaptic connections are omitted for clarity. Layers are indicated with the letter "L". b) The average distances between the inputs used for the simulation, it is $\|x_i(t) - x_j(t)\|$, where $i, j = 60\text{-}100$ Hz with 10 Hz steps and $i \neq j$.

two different randomly chosen initial conditions of the circuit is lower than 0.001 and is omitted from the plots for clarity.

In order to illustrate the generation of internal states within the LSM, the response y_{50} is taken as a reference one. The state distances of the outputs $\|y_i(t) - y_{50}(t)\|$ are calculated as $i = 60\text{-}100$ Hz with 10 Hz steps. The traces are given in Fig. 2a as a function of the time at $\lambda = 2$. The distances are calculated over all network activity. Therefore, the transient stable difference between one response to the reference one indicates a kind of internal synchrony level in this response. I.e. transient stable distance represent transient internal state of synchrony. Within this state single spikes and the degree of coherence are irrelevant [6].

Basically, the network processes the different inputs in different manners. This can be clearly seen from the distances exhibited between the responses corresponding to different inputs. The distances reported for the responses vary between 0 and 0.6 (see Fig. 2a) while the distances between the input signals are found below 0.03, as shown in Fig. 1b.

Different from results reported by Maass et al. [3], the traces of distances in Fig. 2a demonstrate that the difference in liquid states after the first 30 msec is not proportional to the distance between the corresponding input spike trains; see the zoomed-in captures of the first 150 msec from this plot given in Fig. 2b. The state differences are transiently stable states distributed in a nonuniform fashion. Recalling the randomness of both the network connectivity and the input signals, only the relative location of these transient states to each other is relevant rather than the absolute value of the distance to y_{50} . The main observation from this simulation is the rich repertoire of stable states achieved by the LSM when MSSM is used. By studying Fig. 2b, at least five levels (stable parts of the lines with tiny rapid fluctuations) of activity can be counted. These levels of activities represent certain neural states that encode input features. Moreover, each line visits different levels of activity and certain levels are commonly visited by more than once over time; e.g. the state distance between 0.3 and 0.4 is common between three lines, see Fig. 2b.

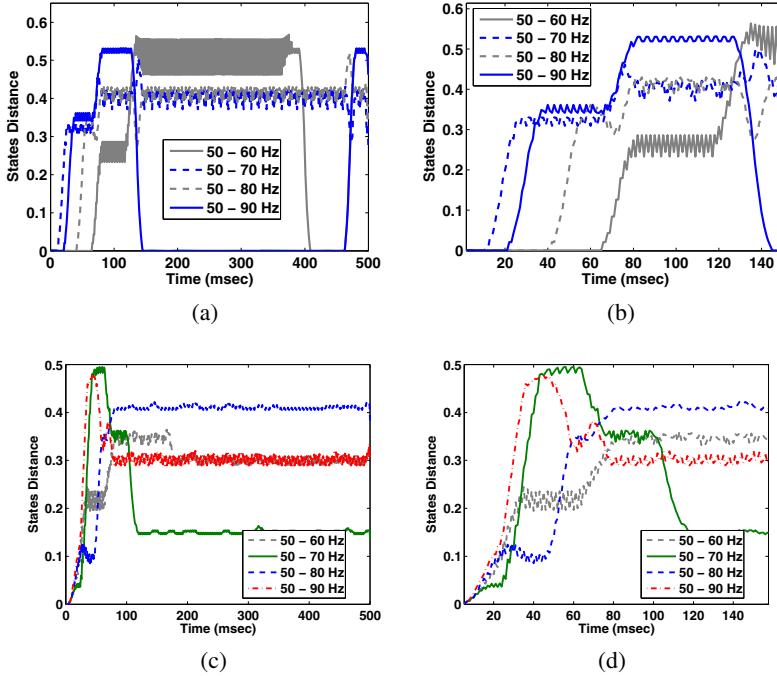


Fig. 2. Simulation results. a) The time evolution of state distances $\|y_i(t) - y_{50}(t)\|$, where $i = 60-90$ Hz at 10 Hz steps at $\lambda = 2$. b) Zoom in of (a) for the first 150 msec. c) The time evolution of state distances $\|y_i(t) - y_{50}(t)\|$, where $i = 60-90$ Hz at 10 Hz steps at $\lambda = 3$. d) Zoom in of (c) for the first 150 msec.

The change in the network structure affects the internal representation of the information within the network. This statement is investigated as well. The whole simulation is repeated with the interconnectivity factor $\lambda = 3$. The results for the latter case are shown in Fig. 2c and 2d. In this case, the variation and the span of the transient states change significantly. In Fig. 2d, eight different levels can be clearly counted.

LSM vs. Cortical columns. The novel transient stable states reflect an emergent ability to process and represent input features at internally well defined states of activities. To understand the relevance of achieving these transient stable states, let us consider the following features observed by the analysis made to the brain circuitry in a behavioural context [11]:

- The representation of sensory information is made on "as-needed basis" that is reflected through the switching between appearing/disappearing of cell assemblies (binding-by-synchrony) throughout the same cortical area.
- Sensory information are represented "at level of synapses rather than cells" [11]. Thus, the expressive power of a neural circuitry in terms of its capacity of internal transient states reveals, for a certain extent, its computational power. This capacity may not be probed merely as "how many" but with "how long" as well.

The results we report here realize both features. In particular the switching between states and the variability of their time span is accomplished for the simulated neural column. Adopting these features permits LSM to inherit novel and powerful computational aspects similar to the abilities of cortical mini columns; this is achievable only when the synaptic dynamics are complex enough to implicitly encode the required input information [11]. Apart from knowing the exact mechanism of content-addressable memory in cortical columns, these conceptual analysis may be adequate to entertain the assumption that LSM with MSSM can simulate the content-addressable memory features observed in cortical circuitry.

4 Conclusion

In this paper we introduce a new implementation of the LSM that incorporates the recently introduced synaptic model (MSSM) [18,16]. The main computational feature, the separation property, is preserved with additional ones; they are the ability to sustain certain levels of synchronous activity and switch between these levels in response to input sensory information. Such capabilities permit LSM to capture more complex functions and features from cortical mini/hyper columns. We leave further investigations to probe the realization of content-addressable memory in LSM for a future study. The implemented simulations and analysis here are regarded only a scratch on the surface of the possible findings from this framework.

References

1. van der Velde, F.: Is the brain an effective Turing machine or a finite-state machine?. *Journal Psychological Research* 55(1), 71–79 (1993)
2. van der Velde, F., de Kamps, M.: Neural blackboard architectures of combinatorial structures in cognition. *Behavioral and Brain Sciences* 29(01), 37–70 (2006)
3. Maass, W., Natschläger, T., Markram, H.: Real-time computing without stable states: a new framework for neural computation based on perturbations. *Neural Comput.* 14(11), 2531–2560 (2002)
4. Maass, W., Markram, H.: On the computational power of circuits of spiking neurons. *J. Comput. Syst. Sci.* 69(4), 593–616 (2004)
5. Maass, W.: Liquid computing. In: Cooper, S.B., Löwe, B., Sorbi, A. (eds.) *CiE 2007. LNCS*, vol. 4497, pp. 507–516. Springer, Heidelberg (2007)
6. von der Malsburg, C.: The what and why of binding: The modeler’s perspective (September 1999)
7. Singer, W.: Neuronal synchrony: a versatile code for the definition of relations. *Neuron* 24, 49–65 (1999)
8. Sejnowski, T.J., Paulsen, O.: Network Oscillations: Emerging Computational Principles. *J. Neurosci.* 26(6), 1673–1676 (2006)
9. Singer, W.: Understanding the brain. *European Molecular Biology Org.* 8, 16–19 (2007)
10. Tsodyks, M., Uziel, A., Markram, H.: Synchrony generation in recurrent networks with frequency-dependent synapses. *J. Neurosci.* 20, 50 (2000)
11. Gilbert, C.D., Sigman, M.: Brain states: top-down influences in sensory processing. *Neuron* 54(5), 677–696 (2007)

12. Ford, J.M., Krystal, J.H., Mathalon, D.H.: Neural Synchrony in Schizophrenia: From Networks to New Treatments. *Schizophr Bull.* 33(4), 848–852 (2007)
13. Bhattacharya, J., Petsche, H.: Shadows of artistry: cortical synchrony during perception and imagery of visual art. *Cognitive Brain Research* 13(2), 179–186 (2002)
14. Cheadle, S., Bauer, F., Parton, A., Müller, H., Bonneh, Y.S., Usher, M.: Spatial structure affects temporal judgments: Evidence for a synchrony binding code. *Journal of Vision* 8(7), 1–12 (2008)
15. Wennekers, T., Ay, N.: Spatial and temporal stochastic interaction in neuronal assemblies. *Journal Theory in Biosciences* 122(1), 5–18 (2003)
16. El-Laithy, K., Bogdan, M.: synchrony state generation: An approach using stochastic synapses. *J. of Artificial Intelligence and Soft Computing Research* 1(1), 17–26 (2011)
17. Markram, H., Wang, Y., Tsodyks, M.: Differential signaling via the same axon of neocortical pyramidal neurons. *Proc. of the Nat. Academy of Sciences of the USA* 95(9), 5323–5328 (1998)
18. El-Laithy, K., Bogdan, M.: Synchrony state generation in artificial neural networks with stochastic synapses. In: Alippi, C., Polycarpou, M., Panayiotou, C., Ellinas, G. (eds.) *ICANN 2009. LNCS*, vol. 5768, pp. 181–190. Springer, Heidelberg (2009)
19. Brunel, N., Rossum, M.v.: Quantitative investigations of electrical nerve excitation treated as polarization: Louis lapicque 1907; Translated *Biol. Cybern.* 97(5), 341–349 (2007)
20. El-Laithy, K., Bogdan, M.: Predicting spike-timing of a thalamic neuron using a stochastic synaptic model. In: *ESANN Proceedings*, pp. 357–362 (2010)

Appendix I: The Models

Neuronal representation. Neurons are modeled as leaky Integrate-and-Fire neurons. Each neuron is described by its voltage membrane potential h , as following [19]:

$$\tau_h \frac{dh}{dt} = h_{\text{rest}} - h + \text{EPSP}, \quad (1)$$

where τ_h is the membrane time constant set to 20 msec. EPSP is the total observed excitatory postsynaptic potential from all pre-synaptic terminals. When $h(t) \geq h_{\text{th}}$, a spike is generated and $h(t^+) := h_{\text{rest}}$ where $h_{\text{rest}} = -70\text{mV}$ and $h_{\text{th}} = -60\text{mV}$.

Synaptic representation. The modified stochastic synaptic model (MSSM) introduced in [18,16] estimates the transmission probability of an arriving action potential, i.e. a spike, from a presynaptic neuron via a synapse to a postsynaptic neuron. Thus, $P(t_i)$ is the probability that the i th spike in a presynaptic spike train $\sum_i \delta(t - t_i)$ (input spikes) triggers the release of a vesicle at time t_i at the synapse. The involved probability-of-release $P(t) = 1 - \exp(-C(t) \cdot V(t))$ is governed by two counteracting mechanisms: facilitation and depression. Facilitation reflects the calcium concentration in the presynaptic neuron, $C(t)$, while depression represents the effect of the concentration of ready-to-release vesicles in the presynaptic terminal, $V(t)$. The model reads [18,20]:

$$\dot{C} = \frac{(C_o - C)}{\tau_C} + \alpha \cdot \sum_i \delta(t - t_i), \quad (2)$$

$$\dot{V} = \frac{(V_o - V)}{\tau_V} - P(t) \cdot \sum_i \delta(t - t_i), \quad (3)$$

$$\dot{N}_t = \max(0, -\frac{dV}{dt}) + \left(\frac{N_{to} - N_t}{\tau_{N_t}} \right), \quad (4)$$

$$\tau_{epsp} EPSP = -EPSP + k_{epsp} \cdot N_t, \quad (5)$$

In Eq. 2, the intracellular calcium concentration starts at $C_o = 0.45^3$. It is raised incrementally by each stimulus impulse, approximated herein by a Dirac Delta function $\delta(t - t_i)$. The impact of each stimulus impulse to the intracellular calcium concentration is equal to the product of calcium gain (calcium current), α , caused by action potential and set to 0.095. Once the stimulus sequence ends, $C(t)$ decays with time constant $\tau_C = 3.34$ msec toward C_o . $C(t)$ represents, in an abstract way, the intrinsic synaptic processes of synaptic facilitation [16]. In Eq. 3, $V(t)$ is the expected number of vesicles of neurotransmitter molecules in the ready-for-release pool at time instant t . $V_o = 3.4$ is the maximum number of vesicles that can be stored in the pool. In a similar fashion to $C(t)$, $V(t)$ follows first-order nonlinear differential equations with a time constant $\tau_V = 9.95$ msec.

N_t determines the concentration of the released neurotransmitter in the synaptic cleft, Eq. 4. This concentration can be estimated by tracing the amount of vesicles of neurotransmitter that remains in the presynaptic neuron, $V(t)$, over time. It is worth mentioning that there is a formal distinction between release site and synapse. Alternatively, each quantum of neurotransmitter is stored in one synaptic vesicle. The concentration of neurotransmitter in the synaptic cleft is meant to be the corresponding concentration of quanta of neurotransmitter. Hence, in Eq. 4 we use here a first-order nonlinear differential equation similar in nature to Eq. 2. The incremental raise in this case is then the decrease in the concentration of vesicles (first term). The drift term (second term) allows the value of N_t to decay, in case of no input, to a minimum accepted concentration N_{to} with a decay time constant τ_{N_t} to 0 and 0.05 sec respectively. This decay reflects the biological cleaning action (or complete removal) of the neurotransmitter from the cleft. As the binding process of neurotransmitter in the postsynaptic membrane induces EPSP, it is calculated as in Eq. 5; where $\tau_{epsp} = 0.075$ sec is a decay time constant and k_{epsp} is a scaling factor set to 13.33.

³ All values are arbitrarily chosen based on preliminary analysis, see [16].

Hybrid Parallel Classifiers for Semantic Subspace Learning

Nandita Tripathi¹, Michael Oakes¹, and Stefan Wermter²

¹ Department of Computing, Engineering and Technology, University of Sunderland,
St Peters Way, Sunderland, SR6 0DD, United Kingdom

Nandita.Tripathi@hotmail.com, Michael.Oakes@sunderland.ac.uk

² Institute for Knowledge Technology, Department of Computer Science,
University of Hamburg, Vogt Koelln, Str. 30, 22527 Hamburg, Germany
wermter@informatik.uni-hamburg.de

Abstract. Subspace learning is very important in today's world of information overload. Distinguishing between categories within a subset of a large data repository such as the web and the ability to do so in real time is critical for a successful search technique. The characteristics of data belonging to different domains are also varying widely. This merits the need for an architecture which caters to the differing characteristics of different data domains. In this paper we present a novel hybrid parallel architecture using different types of classifiers trained on different subspaces. We further compare the performance of our hybrid architecture with a single classifier and show that it outperforms the single classifier system by a large margin when tested with a variety of hybrid combinations. Our results show that subspace classification accuracy is boosted and learning time reduced significantly with this new hybrid architecture.

Keywords: Parallel classifiers, hybrid classifiers, subspace learning, significance vectors, maximum significance.

1 Introduction

The *curse of dimensionality* [1] degrades the performance of many learning algorithms. Therefore, we need ways to discover clusters in different subspaces of datasets which are represented with a high number of dimensions [2]. Subspace analysis lends itself naturally to the idea of hybrid classifiers. Each subspace can be processed by a classifier best suited to the characteristics of that subspace. Instead of using the complete set of full space dimensions, classifier performance can be boosted by using only a subset of the dimensions. The method of choosing an appropriate reduced set of dimensions is an active research area [3]. The use of Random Projections [4] in dimensionality reduction has also been studied. In the Random Subspace Method (RSM) [5], classifiers were trained on randomly chosen subspaces of the original input space and the outputs of the models were then combined. However random selection of features does not guarantee that the selected inputs have necessary discriminating information. Several variations of RSM have been proposed such as Relevant random feature subspaces for co-training (Rel-RASCO) [6], the

Not-so-Random Subspace Method (NsRSM) [7] and the Local Random Subspace Method [8].

In the real world, documents can be divided into major semantic subspaces with each subspace having its own unique characteristics. The above research does not take into account this division of documents into semantic categories. We present here a novel hybrid parallel architecture (Fig. 1) which takes advantage of the different semantic subspaces existing in the data. We further show that this new hybrid parallel architecture improves subspace classification accuracy as well as significantly reduces training time. In this architecture, we tested various hybrid combinations of classifiers using the conditional significance vector representation [9] which is a variation of the semantic significance vector [10], [11] to incorporate semantic information in the document vectors. We compare the performance of this hybrid parallel classifier against that of single Multilayer Perceptron (MLP) classifiers using the significance vector as well as the Term Frequency – Inverse Document Frequency (tf-idf) vector representation. Our experiments were performed on the Reuters corpus (RCV1) [12] using the first two levels of the topic hierarchy.

2 Methodology Overview and Overall Architecture

Ten thousand Reuters headlines along with their topic codes were extracted from the Reuters Corpus. These headlines were chosen so that there was no overlap at the first level categorization. At the second level, since most headlines had multiple level 2 subtopic categorizations, the first subtopic was taken as the assigned subtopic. A total of fifty subtopics were included in these experiments. Headlines were then preprocessed to separate hyphenated words. Dictionaries with term frequencies were generated based on the TMG toolbox [13]. These were then used to generate the Full Significance Vector [9] and the Conditional Significance Vector [9]. The tf-idf [14] representation for each document was also generated by the TMG toolbox.

The WEKA machine learning workbench [15] was used to examine this architecture with various learning algorithms. Seven algorithms were compared for our representations to examine the different categories of classification algorithms. Classification Accuracy, which is a comparison of the predicted class to the actual class, and the Training Time were recorded for each experiment run.

3 Data Vector Sets Generation

Three different vector representations (Full Significance Vector, Category-wise separated Conditional Significance Vector and tf-idf) were generated for our data.

Full Significance Vector: For each Reuters Full Significance document vector the first four columns, representing the four main topics CCAT, ECAT, GCAT & MCAT, were ignored leaving a vector with 50 columns representing 50 subtopics. The order of the data vectors was then randomised and divided into two sets – training set of 9000 vectors and a test set of 1000 vectors.

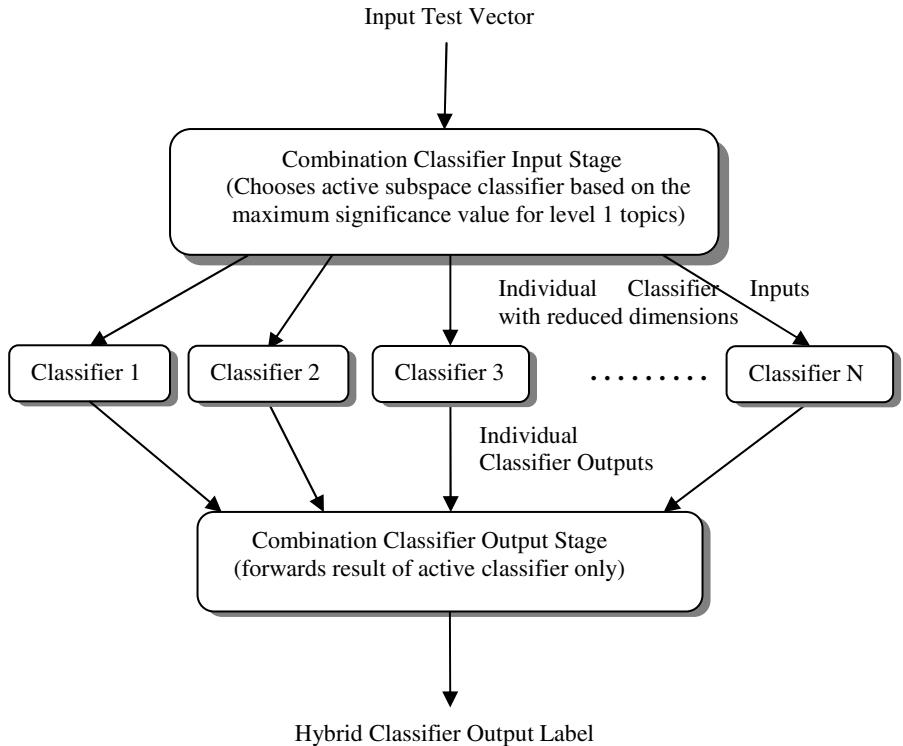


Fig. 1. Hybrid Parallel Classifier Architecture for Subspace Learning

Category-wise separated Conditional Significance Vector: The order of the Reuters Conditional Significance document vectors was randomised and divided into two sets – a training set of 9000 vectors and a test set of 1000 vectors. The training set was then divided into 4 sets according to the main topic labels. For each of these four sets, only the relevant subtopic vector entries (e.g. C11, C12, etc for CCAT; E11, E12, etc for ECAT; etc) for each main topic were retained. These 4 training sets were then used to train the 4 parallel classifiers of the Reuters hybrid classifier. The main category of a test data vector was determined by the maximum significance vector entry for the first four columns representing the four main categories. After this, the entries corresponding to the subtopics of this predicted main topic were extracted along with the *actual* subtopic label and given to the classifier trained for this predicted main category. The accuracy of choosing the correct main topic by selecting the maximum significance level 1 entry has been measured to be 96.80% for the 1000 test vectors i.e. 968 vectors were assigned the correct trained classifiers whereas 3.20% or 32 vectors were assigned to a wrong classifier – resulting in a wrong classification decision for all these 32 vectors. Hence the upper limit for classification accuracy is 96.80% for our hybrid parallel classifier.

TFIDF Vector: The TMG toolbox [13] was used to generate tf-idf vectors for the ten thousand Reuters headlines used in these experiments. The tf-idf vector dataset was then randomized and divided into a training set of 9000 vectors and a test set of 1000 vectors.

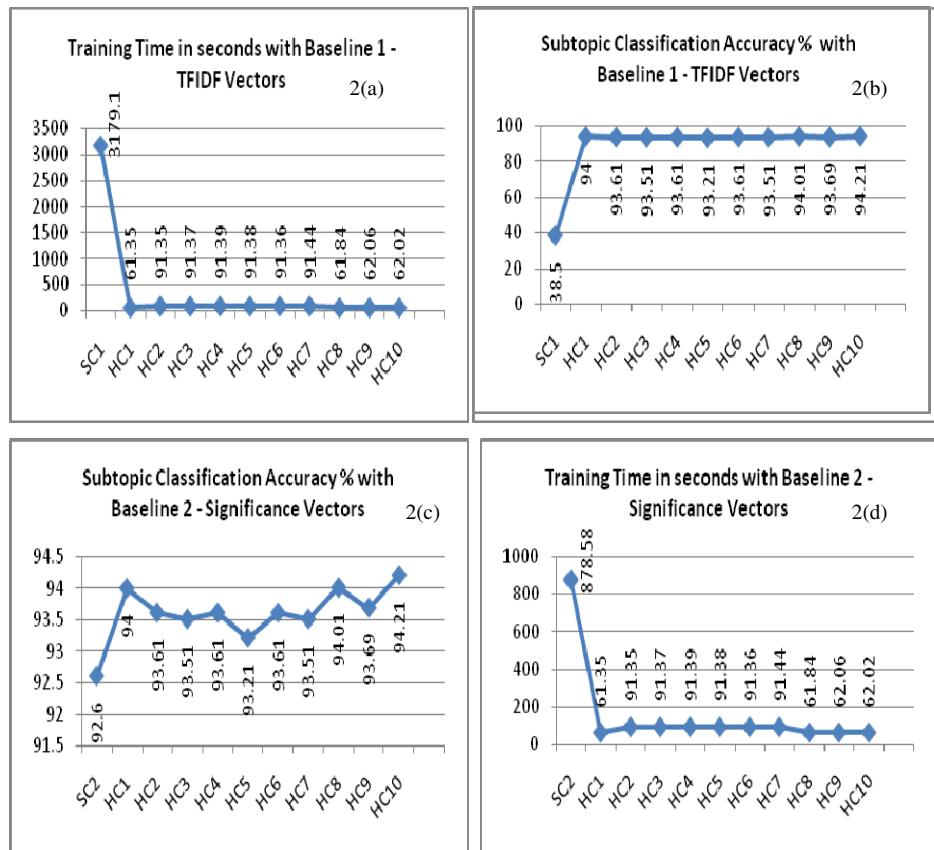
4 Classification Algorithms

Seven Classification algorithms were tested with our datasets namely Random Forest, C4.5, Multilayer Perceptron, Naïve Bayes, BayesNet, NNge and PART. Random Forests [16] are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently. C4.5 [17] is an inductive tree algorithm with two pruning methods: subtree replacement and subtree raising. Multilayer Perceptron [18] is a neural network which uses backpropagation for training. Naïve Bayes [19] is the simplest form of Bayesian network, in which all attributes are independent given the value of the class variable. BayesNet [20] implements Bayes Network learning using various search algorithms and quality measures. NNge [21] is a nearest neighbor-like algorithm using non-nested generalized exemplars. A PART [22] decision list uses C4.5 decision trees to generate rules.

5 Results and Their Analysis

Experiments were run using seven algorithms from Weka on the Reuters Corpus. In the first step, the algorithms were run using category-wise separated data from the training set to select the algorithm with the highest classification accuracy for each main category. In case of a tie between two algorithms, the one with the lower training time was chosen. Subsequently these selected algorithms were applied to the test data and the performance of the hybrid classifier was measured. The category-wise separated Conditional Significance Vectors were used here. Each of the algorithms was also run on the full (not category-wise separated) data set to provide a comparison for the hybrid classifier. Two vector representations were used for the comparison baseline – the Full Significance Vector and tf-idf. As the performances of many classifiers for each main category were quite close to each other, we also ran some experiments using a predefined set of classifiers. The combination of MLP with different types of classifiers (Bayesian, rule-based and tree-based classifiers) was evaluated and the best combination was identified. For a two-classifier combination, MLP and the other classifier were used alternately on the main category topics while for a four-classifier system four different classifiers were used on the four main topics.

The charts in Fig 2 show a comparison of the performance of hybrid classifiers with that of MLP. The baseline single MLP classifier experiment is run with two different vector representations - Significance Vector and tf-idf. The accuracies of all the hybrid parallel classifiers are better than that of the single MLP classifier. The Hybrid 4-classifier system (HC10) shows the best result which is quite similar to that of the hybrid classifier with category-wise classifiers chosen from the training set (HC1).



Classifier Index

SC1- Single MLP over full data using tf-idf Vectors; **SC2**- Single MLP over full data using Significance Vectors; **HC1**- Hybrid Parallel Classifier with category-wise classifiers chosen from training data performance

Hybrid 2-Classifier Systems :

HC2 (MLP/NB), HC3 (MLP/BNet)
 HC4 (MLP/NNge) , HC5 (MLP/PART)
 HC6 (MLP/J48), HC7 (MLP/RF)

Abbreviations: MLP - Multilayer Perceptron; NB - Naïve Bayes; BNet – BayesNet; RF – Random Forest

Hybrid 4-Classifier Systems :

HC8 (MLP/NB/ NNge/ J48)
 HC9 (MLP/BNet/PART/RF)
 HC10 (MLP/NNge/PART/NB)

Fig. 2. Hybrid Parallel Classifiers Performance Metrics

Overall, it was observed that there was an improvement in subtopic classification accuracy along with a significant reduction in training time. The classification accuracies of all the hybrid classifiers were quite close to each other but all of them were much better than the classification accuracy of the single classifier with tf-idf baseline. The difference with the significance vector baseline was smaller but even there the classification accuracies of the hybrid systems were better. The training times showed a very steep reduction compared to both baselines. The average of 10 runs was taken for each experiment. In the hybrid classifier, even though we are using more classifiers, the training time is reduced. This is because each classifier now trains on a reduced set of data with a reduced set of vector components. This two-fold reduction translates to a significance decrease in training time.

6 Conclusion

In this work, we attempt to leverage the differences in the characteristics of different subspaces to improve semantic subspace learning. The main objective here is to improve document classification in a vast document space by combining various learning methods. Our experiments show that hybrid parallel combinations of classifiers trained on different subspaces offer a significant performance improvement over single classifier learning on full data space. Individual classifiers also perform better when presented with less data in lower dimensions. Our experiments also show that learning based on the semantic separation of data space is more efficient than full data space learning. Combining different types of classifiers has the advantage of integrating characteristics of different subspaces and hence improves classification performance. This technique can work well in other domains like pattern / image recognition where different classifiers can work on different parts of the image to improve overall recognition. Computational Biology too can benefit from this method to improve recognition within sub-domains.

In our experiments, subspace detection is done by processing a single document vector. This method is independent of the total number of data samples and only compares the level 1 topic entries. The time complexity of the combining classifier is thus $O(k)$ where k is the number of level 1 topics. The novelty of our approach is in the use of a maximum significance based method of input vector projection for a hybrid parallel classifier along with the use of Conditional Significance Vectors (which increase the distinction between categories within a subspace) for improved subspace learning. Combining MLP in parallel with a basic classifier (Bayesian, tree based or rule based) improves the classification accuracy and significantly reduces the training time. The experiments also show that using maximum significance value is very effective in detecting the relevant subspace of a test vector.

References

1. Friedman, J.H.: On Bias, Variance, 0/1—Loss, and the Curse-of- Dimensionality. *Data Mining and Knowledge Discovery* 1(1), 55–77 (1997)
2. Parsons, L., Haque, E., Liu, H.: Subspace Clustering for High Dimensional Data: A Review. *ACM SIGKDD Explorations Newsletter* 6(1), 90–105 (2004)

3. Varshney, K.R., Willsky, A.S.: Learning dimensionality-reduced classifiers for information fusion. In: Proceedings of the 12th International Conference on Information Fusion, Seattle, Washington, pp. 1881–1888 (2009)
4. Fradkin, D., Madigan, D.: Experiments with Random Projections for Machine Learning. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 517–522 (2003)
5. Ho, Kam, T.: The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20(8), 832–844 (1998)
6. Yaslan, Y., Cataltepe, Z.: Co-training with relevant random subspaces. *Neurocomputing* 73, 1652–1661 (2010)
7. Garcia-Pedrajas, N., Ortiz-Boyer, D.: Boosting Random Subspace Method. *Neural Networks* 21, 1344–1362 (2008)
8. Kotsiantis, S.B.: Local Random Subspace Method for constructing multiple decision stumps. In: International Conference on Information and Financial Engineering, pp. 125–129 (2009)
9. Tripathi, N., Wermter, S., Hung, C., Oakes, M.: Semantic Subspace Learning with Conditional Significance Vectors. In: Proceedings of the IEEE International Joint Conference on Neural Networks, pp. 3670–3677, Barcelona (2010)
10. Wermter, S., Panchev, C., Arevian, G.: Hybrid Neural Plausibility Networks for News Agents. In: Proceedings of the Sixteenth National Conference on Artificial Intelligence, pp. 93–98 (1999)
11. Wermter, S.: Hybrid Connectionist Natural Language Processing. Chapman and Hall, Boca Raton (1995)
12. Rose, T., Stevenson, M., Whitehead, M.: The Reuters Corpus Volume 1 - from Yesterday's News to Tomorrow's Language Resources. In: Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002), pp. 827–833 (2002)
13. Zeimpekis, D., Gallopoulos, E.: Grouping Multidimensional Data: Recent Advances in Clustering. In: Kogan, J., Nicholas, C. (eds.) TMG: A MATLAB Toolbox for Generating Term Document Matrices from Text Collections. Springer, Heidelberg (2005)
14. Manning, C., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
15. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.: The WEKA Data Mining Software: An Update. *ACM SIGKDD Explorations Newsletter* 11(1), 10–18 (2009)
16. Breiman, L.: Random Forests. *Machine Learning* 45(1), 5–32 (2001)
17. Quinlan, J.R.: C4.5: Programs for Machine Learning. Machine Learning. Morgan Kaufmann Publishers, San Mateo (1993)
18. Verma, B.: Fast training of multilayer perceptrons. *IEEE Transactions on Neural Networks* 8(6), 1314–1320 (1997)
19. Zhang, H.: The optimality of Naïve Bayes. American Association for Artificial Intelligence (2004), <http://www.aaai.org>
20. Pernkopf, F.: Discriminative learning of Bayesian network classifiers. In: Proceedings of the 25th IASTED International Multi-Conference: Artificial Intelligence And Applications, pp. 422–427 (2007)
21. Martin, B.: Instance-Based learning: Nearest Neighbor With Generalization. Master Thesis, University of Waikato, Hamilton, New Zealand (1995)
22. Frank, E., Witten, I.H.: Generating Accurate Rule Sets Without Global Optimization. In: Shavlik, J. (ed.) Machine Learning: Proceedings of the Fifteenth International Conference. Morgan Kaufmann Publishers, San Francisco (1998)

Temperature Prediction in Electric Arc Furnace with Neural Network Tree

Mirosław Kordos¹, Marcin Blachnik², and Tadeusz Wieczorek²

¹ University of Bielsko-Biala, Department of Mathematics and Computer Science,
Bielsko-Biała, Willowa 2, Poland
mkordos@ath.bielsko.pl

² Silesian University of Technology, Department of Management and Informatics,
Katowice, Krasinskiego 8, Poland
marcin.blachnik@polsl.pl

Abstract. This paper presents a neural network tree regression system with dynamic optimization of input variable transformations and post-training optimization. The decision tree consists of MLP neural networks, which optimize the split points and at the leaf level predict final outputs. The system is designed for regression problems of big and complex datasets. It was applied to the problem of steel temperature prediction in the electric arc furnace in order to decrease the process duration at one of the steelworks.

1 Introduction

There are several concerns with regression tasks in data mining. There is usually no clear rule about which data point is an outlier or wrong point and thus should be excluded from the training set. Another issue is the determination of the error function; should be the mean squared error or a different measure. The next problem is how to adjust the system to the requirements of different sensitivity in different range of input and output variables, what frequently occurs in practical technological applications. Another problem is that frequently the systems do not provide us with the decision rules explaining; why a given output value was predicted. The comprehensibility of the system can be sometimes extremely important: if the person, who decides upon the process does not understand the system decision, he may not trust the system and may not want to use it. For that reason after the system has learned from the training data it needs to undergo a verification by an expert in a given domain.

Some of the requirements oppose the others. For example, one dimensional decision trees that split the nodes based on single attribute values produce clear rules (as long as the tree is not too big), which can be further optimized. On the other hand they generate decision borders perpendicular to particular attribute axes, which frequently does not describe well the properties of the real physical process. Oblique decision trees or neural networks for example, can map the data properties much more faithfully, but the logical rule extraction from trained neural networks is not so obvious as from decision trees. The requirements of different sensitivity of the model in different ranges of input and output data can be best obtained by creating several models, each for particular data range – that makes also the expert verification much easier. In previous years we

implemented an intelligent expert system for steel production optimization in one of the Steelworks in Poland, where we tried to consider the above factors [8]. Our current work adds new functionality to that system. The other purpose of the work is to create a more general system that can be used not only for prediction of temperature in the electric arc furnace but also in other regression problems with big and complex datasets. The solution is based on a merge of an MLP neural network with a regression tree with additional features added.

Several literature sources mention the idea of neural network tree. Hang and Chung [1] proposed a Fuzzy Neural Network Tree based on the normal fuzzy reasoning, where the output of the previous level network was the input to the next lever network. In their architecture there were several networks and the lower level and only one network at the top lever, which generated the final output. Schetinin [2] developed a neural network decision tree to solve the multi-class problems. His algorithm exploits a bottom up search of the features that provide the best classification accuracy of the linear tests. Golea and Marchand [3] explores the application of neural network with variable structure to the construction of decision trees from examples. Foresti and Christian Micheloni [4] presents an approach to the tree construction in classification tasks, where after each time the tree is increased by a new level, the whole tree is reevaluated. Fong et. Al [5] proposes a hybrid model of neural network and decision tree classifier . Chen et. Al [6] presents a model, which is generated initially as a flexible multi-layer feed-forward neural network and then evolved using an evolutionary procedure. In our works [7] we discussed parameterization of regression trees and in [8] hybrid systems with neural networks and data preprocessing for the prediction on additives in steelmaking process.

Our actual work is based on our experiences from [7,8,9]. However, we introduced quite new concepts and tested the solution for the prediction of temperature in steelmaking process using the electric arc furnace in a real production environment. The system is mentioned for regression an works well with big and complex datasets. In our case the dataset was gathered during two years of steel production and contained about 12,000 vectors with 104 attributes. Knowing actual temperature allows the EAF user to avoid overheating the steel past the optimum point which will save energy and reduce refractory wear however the largest saving comes from reducing the tap to tap time (duration of a single process) by identifying the optimum tapping temperature thus allowing the steelmaker to increase productivity.

Temperature measurement and control are critical factors in the operation of a modern electric arc furnace. Temperature is measured a few times during every melt. It is done via special lance with thermocouple that dip into the liquid steel. But every measurement takes time, approximately one minute. The arc has to be turn off and the process suspended. Modern DC EAFs for 150-200 tons have got “tap to tap “times of 30 minutes. Waste of time for two, three or even more measurements is thus quite significant.

Some papers on continuous temperature measurement appeared recently. But the problem is very complex. The temperatures are very high and the radiant heat from furnace creates major problems for cables and instrumentation. Another problem to overcome are the effects of electro-magnetic radiation on sensitive electrical systems within the pyrometer. Survey showed that conventional temperature measurement systems do

not lead to the required solutions. The available systems suffered from numerous application problems that affect availability of the measurement.

Kendall [10] presented the first stages in the development of a temperature measuring system, which shows the promise of being able to accurately measure the temperature of the steel in the furnace for the whole furnace cycle. However, it is still a long way to reasonable performance of the system.

Millman [11] presented a camera-based technology for monitoring the scrap melting process in an EAF. However, after about two to three minutes of arcing, generation of high dust density had a significantly deleterious effect on the resolution of the images. In addition to dust, the EAF process also generates flames which are composed of a variety of combustion gases that absorb light to varying degrees and at different wavelengths. Therefore, it is not possible to see through a flame, using a camera system that operates under white light.

Instead of measuring important parameters of the EAF process they can be calculated. Wendelstorf [12] basing on quantitative time-dependent measurements of energy and mass inputs modeled the meltdown process. The model allows the assessing of the actual mean temperature of liquid steel. Using the model and temperature measured 10 minutes before the tap the melt temperature is calculated and the end of the meltdown phase is predicted. However, we need to know the temperature much earlier, to be able to optimize the parameters during the whole process. Fernandes [13] proposed a system for online estimation of electric arc furnace tap temperature by using fuzzy neural networks.

2 Method

The purpose of the system is to minimize the number of temperature measurements in the EAF process. That will lead to shortening the process time and consequently the costs of steel production. For this purpose we used the following data preprocessing: first we decided to predict not the absolute temperature, but the difference between the current temperature and the last measured temperature. The temperatures are at the level of 1900K and the differences at the level of 30K. Thus, that allowed us to build a more precise model. Then we decided which attributes should be taken to the training set as they are and for which attributes the difference between the current value and the value at the time of last temperature measurement is preferred. The first group consists of values such as mass of the steel, chemical compounds, condition of the furnace, etc. while the second group consists of the values that are constantly changing during the process, such as electric energy, temperature at the base of the furnace, amount of gas and oxygen used, etc. All the values were obtained from PLC controllers with the frequency from one to several seconds and recorded in the real time into an SQL database. Then from the database we extracted useful information in a reasonable time intervals to prevent excessive growth of the training dataset.

At the next stage the training data set was standardized. Then we calculated the correlations between all 104 input variables and the temperature and rejected from the training set the values for which the absolute value of the correlation was below 0.03.

That left 50 attributes in the training set. We do not perform a further explicit feature selection, but it will be included in the future versions of the system.

Based on our previous experiences with implementing regression systems in the metallurgical industry, we knew that using the mean squared error (MSE) as the measure of the solution quality is not the best choice here [8]. The reason for that is, the MSE tends to penalize more the big errors and thus cause the neural network to fit better to the extreme values of the temperature change. The extreme values happen relatively infrequently and they suggest some abnormality of the process. So if the extreme value is predicted the process operator has to measure the temperature despite of the prediction, because he can't trust in such abnormal value. So this kind of prediction, no matter how accurate, really doesn't reduce the number of temperature measurements. It can alarm the operator of some abnormality. So it is useful, but does not have to be precise. Instead, we would rather prefer to have the typical values predicted more accurately. For that purpose during the system training we transfer as well all input attributes as the output by hyperbolic tangent function.

$$y = (1 - \exp(-x)) / (1 + \exp(-x)) \quad (1)$$

That changes the distribution of the data from Gaussian-like to a uniform-like and smoothly reduces the influence of the outliers on the predictive model. In this way we do not have to use any special outliers detection methods [14]. The outliers not necessarily are erroneous points and they still can carry some useful information. We reject only the values that are incorrect from the technological point of view. We use a three layer MLP network with hyperbolic tangent activation functions in all three layers. The beta parameters in the hidden layer are constant and in the input and output layer they are optimized along with the network weights during the training process. For that purpose we use a Variable Step Size MLP training algorithm, described in [15]. The algorithm allows for an easy incorporation of the sigmoid steepness as a training parameter and performs reasonably good for big datasets.

The whole system is built upon a decision tree, which contains MLP networks in the nodes. The training process is shown in Fig. 1:

1. After the initial data preprocessing, the whole training dataset is given to the MLP network input and it is observed for which vectors the network produced the error below the maxE.
2. The decision tree, which is limited to a maximum of four leaf nodes, tries to divide the data set into two parts; the vectors with the error $< \text{maxE}$ and with error $> \text{maxE}$.
3. After this division the vectors sent by the decision tree to the low error nodes are again given to the network input and the network is retrained.
4. The remaining vectors are split into two groups using the variance reduction criterion.
5. Then step 1 repeats with the two data subsets. However, if the MLP network does not produce low error for a significant number of vectors – step four occurs once respectively once again, as long as the networks work well or as the split criteria are reached. In the second case the final value is determined by the mean value of the vectors in the tree node.

6. In the test phase, each test vector goes down the decision tree as long as it meets the MLP network or a leaf node. The network predicts output for this vector or if the vector falls into a leaf node its value is given by the mean value of training set outputs in the tree node.

Logical rules are very easily extracted from the decision tree and they give an approximate accuracy. More accurate results are given by the neural networks. Rule extraction from the neural networks can also be performed, as described in [16]. The final rule in general consists of two parts: the general part generated by the decision tree and the detailed part generated by the MLP network. Depending on a given node, two or only one part of the rule may exist.

Every subset is standardized again from the original data (Fig. 1.) and then transferred by hyperbolic tangent in order to utilize the whole range of MLP input (-1;1). The stop criteria for creating the next tree nodes are: minimum number of vectors in the current node, minimum number of vectors in the child node and minimum variance in the current node. All three are used together. The output of the MLP at the leaf level is the predicted value. It must be then transformed by the area hyperbolic tangent function and destandardized. The split criteria on the decision tree is based on variance reduction maximization:

$$v = v_0 - p_L v_L - p_R v_R \quad (2)$$

where v is the variance reduction, v_0 is the variance of the output variable in the current node p_L is the ratio of the number of vectors in the left child node to the total number of vectors in the current node and v_L is the variance of the output value in the left child node, and p_R , v_R – respectively for the right child node. We use the formula (2), because we experimented with several different split criteria and the one above, turned to perform best, ensuring an equilibrium between the variance reduction of a single node and the symmetry of the decision tree (left and right children tend to have similar number of vectors) The split point is searched for between every two successive values of every input attribute. Details on the regression tree algorithm can be found in [8].

There are two ways to improve the results: to use a tree forest and to perform global optimization of the splitting points once the tree is created. There are several ways known from the literature of optimizing the tree structure [17]. The time of learning the model does not play any role in this application, once learned the model is used for a long time only with periodic re-learning as new data comes.

3 Experimental Results

Using this data we have built the system for one of the polish steelworks. For that reason we cannot release the data as it is. But we have made the data available in a form that is still useful for machine learning purposes, but which do not allow for discovering physical properties of the process. The dataset in its standardized form, with the original values subtracted from one another as needed (the differential form) is available from [18]. The software used in the experiments, which we created in Java and C# is also

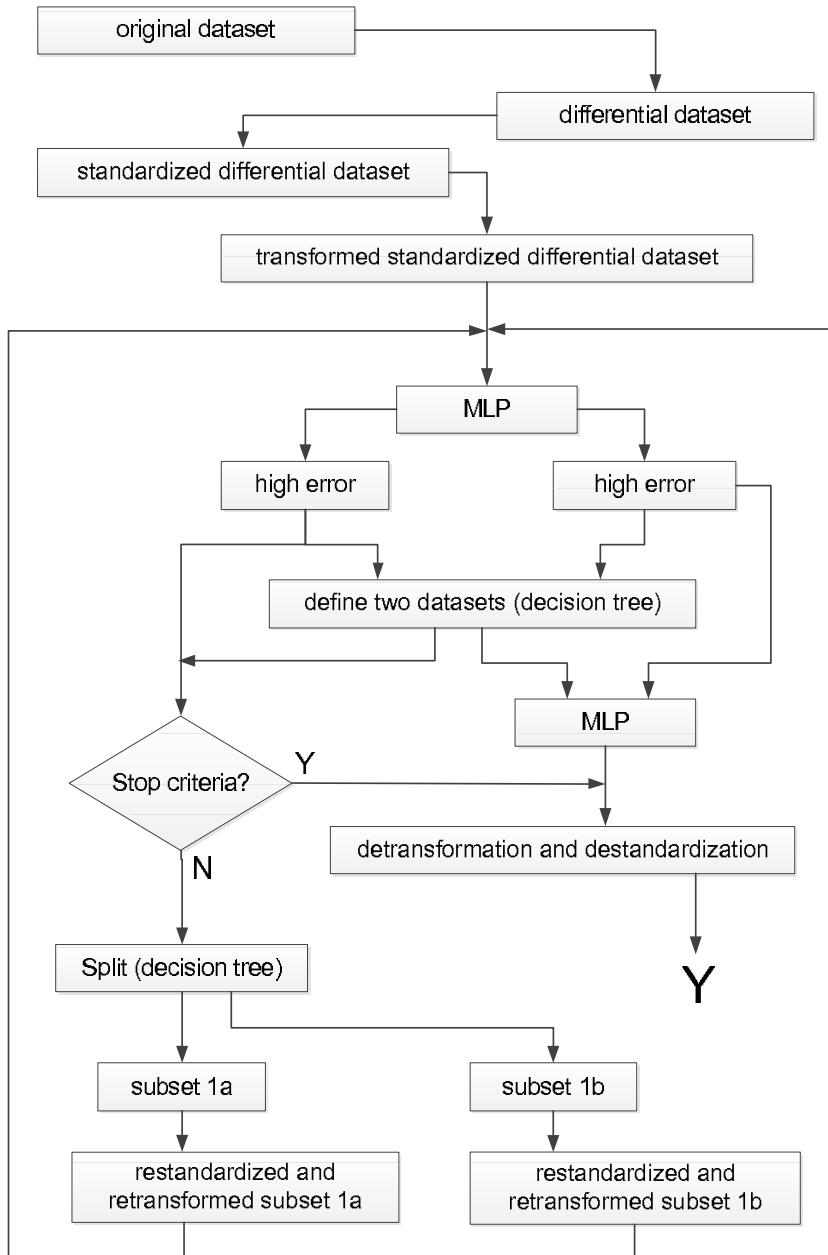


Fig. 1. The Neural Network Tree System

Table 1. Comparison of results obtained with various methods in 10-fold crossvalidation

method	MLP-OC	MLPTree	MLP	Tree	Tree Forest	SVR-L,SVR-G
MSE	0.46 ± 0.05	0.51 ± 0.07	0.60 ± 0.10	0.80 ± 0.08	0.70 ± 0.06	0.61 ± 0.06

available from [18]. Only the inputs with correlation to the output value above 0.03 were left in the dataset and the names of the inputs were changed to x_1, x_2, \dots, x_{50} . The output y is in the first column.

Although it is possible to build a complex system that can obtain lower MSE on that data than this solution, one must remember that the purpose of this system is not to minimize the MSE equally over all the variable range but to allow for shortening the LHF process as much as possible and the other purpose is to enable generation of comprehensive logical rules. For instance, we were able to achieve lower MSE on that data with a complex committee of neural networks with evolutionary parameter optimization (MLP-OC). However, the drawback of the MLP-OC method was that it did not allow for obtaining comprehensive rules. Although the practical usability of particular approaches for the task of LHF process optimization cannot be assessed with a simple single measure, such as MSE, we provide this measure for the comparison in Table 1, as a first step in the assessment of the model quality.

4 Conclusions

The paper presented a neural network tree system, which was applied for the prediction of the steel temperature in the electric arc furnace in one of the Steelworks in Poland. The dataset depicting the process and the result expected from such a system impose special requirements on the predictive system, such as effectively dealing with big and complex data and optimizing rather the gain the process operator can have from the system than the pure MSE function.

The system uses dynamic transformations of the input values, incorporated in the neural network training and uses a post-training tree optimization. The other advantage of the system, which is not fully implemented yet, is the possibility to generate logical rules. Logical rules are the embedded properties of decision trees, however, they require some modifications to the neural networks in the nodes to keep the rules simple and applying rule extraction algorithm from neural networks train for regression problems in the leaf nodes [15]. Thus, the system is intended to merge the advantages of decision trees, which is generating comprehensible logical rules with the advantages of neural networks, which is a more advanced possibility of partitioning the space and modeling the function that transposes inputs into outputs.

Because of the complexity of the model and the limited space for this publication we cannot provide here rigorous mathematical discussion of this approach. However, that discussion is provided in our other papers focused on several subsystems of this model as [7,8,9,15,16], which are also accessible online at our web page [18].

Acknowledgment. The work was sponsored by the Polish Ministry of Science and Higher Education, project No. 4866/B/T02/2010/38 (N508/486638).

References

1. Yan-Qing, Z., Fu-lai, C.: A Fuzzy Neural Network Tree with Heuristic Backpropagation Learning. *Neural Networks* (2002)
2. Schetinin, V.: A Neural Network Decision Tree for Learning Concepts from EEG Data. In: NIMIA-SC (2001)
3. Golea, M., Marchand, M.: A Growth Algorithm for Neural Network Decision Trees. *Euro-physics Lett.* (1990)
4. Foresti, G.L., Micheloni, C.: Generalized Neural Trees for Pattern Classification. *IEEE Transactions on Neural Networks* 13(6) (2002)
5. Fong, S., et al.: Applying a Hybrid Model of Neural Network and Decision Tree Classifier for Predicting University Admission. In: ICICS (2009)
6. Chen, Y., et al.: Time-series forecasting using flexible neural tree model. *Information Sciences* (2004)
7. Kordos, M., Blachnik, M., et al.: A Hybrid System with Regression Trees in Steelmaking Process. In: HAIS (2011)
8. Wieczorek, T., Kordos, M.: Neural Network-based Prediction of Additives in the Steel Refinement Process. *Computer Methods in Materials Science* 10(1) (2010)
9. Wieczorek, T., Blachnik, M., Maczka, K.: Building a model for time reduction of steel scrap meltdown in the electric arc furnace (EAF): General strategy with a comparison of feature selection methods. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2008. LNCS (LNAI), vol. 5097, pp. 1149–1159. Springer, Heidelberg (2008)
10. Kendall, M., et al.: A window into the electric arc furnace, a continuous temperature sensor measuring the complete furnace cycle. *Archives of Metallurgy and Materials* 53(2), 451–454 (2008)
11. Millman, M.S., et al.: Direct observation of the melting process in an EAF with a closed slag door. *Archives of Metallurgy and Materials* 53(2), 463–468 (2008)
12. Wendelstorf, J.: Analysis of the EAF operation by process modeling. *Archives of Metallurgy and Materials* 53(2), 385–390 (2008)
13. Fernandez, J., et al.: Online estimation of electric arc furnace tap temperature by using fuzzy neural networks. *Engineering Applications of Artificial Intelligence* 21 (2008)
14. Angiulli, F., Pizzuti, C.: Fast outlier detection in high dimensional spaces. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) PKDD 2002. LNCS (LNAI), vol. 2431, pp. 15–27. Springer, Heidelberg (2002)
15. Kordos, M., Duch, W.: Variable step search algorithm for feedforward networks. *Neurocomputing* 71(13–15), 2470–2480 (2008)
16. Setiono, R., Thong, J.: An approach to generate rules from neural networks for regression problems. *European Journal of Operational Research* 155(1) (2004)
17. Czajkowski, M., Krętowski, M.: Globally induced model trees: An evolutionary approach. In: Schaefer, R., Cotta, C., Kolodziej, J., Rudolph, G. (eds.) PPSN XI. LNCS, vol. 6238, pp. 324–333. Springer, Heidelberg (2010)
18. Dataset, www.kordos.com/his.html

Optimizing Linear Discriminant Error Correcting Output Codes Using Particle Swarm Optimization

Dimitrios Bouzas, Nikolaos Arvanitopoulos, and Anastasios Tefas

Department of Informatics, Aristotle University of Thessaloniki
Box 451, 54124 Thessaloniki, Greece
`{bouzas, niarvani}@ieee.org,`
`tefas@aiai.csd.auth.gr`

Abstract. Error Correcting Output Codes reveal an efficient strategy in dealing with multi-class classification problems. According to this technique, a multi-class problem is decomposed into several binary ones. On these created sub-problems we apply binary classifiers and then, by combining the acquired solutions, we are able to solve the initial multi-class problem. In this paper we consider the optimization of the Linear Discriminant Error Correcting Output Codes framework using Particle Swarm Optimization. In particular, we apply the Particle Swarm Optimization algorithm in order to optimally select the free parameters that control the split of the initial problem's classes into sub-classes. Moreover, by using the Support Vector Machine as classifier we can additionally apply the Particle Swarm Optimization algorithm to tune its free parameters. Our experimental results show that by applying Particle Swarm Optimization on the Sub-class Linear Discriminant Error Correcting Output Codes framework we get a significant improvement in the classification performance.

Keywords: Error Correcting Output Codes, Sub-classes, Particle Swarm Optimization, Support Vector Machines.

1 Introduction

Many real life classification problems are usually multi-class. However, many powerful classifiers like *Support Vector Machines (SVM)* [4] or *AdaBoost* [14] are capable of solving only binary classification problems. To deal with this problem the *Error Correcting Output Codes (ECOC)* emerged [1]. Due to its ability to correct the bias and variance errors of the base classifiers [5][11], the ECOC procedure has been applied to a wide range of applications, such as face recognition [16], face verification [10] and handwritten digit recognition [17].

On the ECOC framework Pujol et al. proposed that we can use a ternary problem dependent design of ECOC, the so called *Discriminant Error Correcting Output Codes (DECOC)* where, given a number of N_c classes, we can achieve a high classification performance by training only $N_c - 1$ binary classifiers [13].

Escalera et al. extended the DECOC algorithm with the use of *sub-classes* [7]. According to this technique we use a guided problem dependent procedure to group the classes and split them into subsets with respect to the improvement we obtain in the training performance. Recently, in [2] the use of the *Fisher's Linear Discriminant Ratio (FLDR)* was proposed as a problem decomposition criterion in the DECOC framework, resulting in a new algorithm, the so called *Linear Discriminant Error Correcting Output Codes (Linear DECOC)*. The new framework not only improves the classification performance, but it also reduces significantly the running time of the algorithm. This fact enables the promising DECOC with sub-classes approach to be applied efficiently on large scale datasets.

In the DECOC with sub-classes and consequently in the linear DECOC frameworks the split of the initial multi-class problem's classes into sub-classes is controlled by a triplet of thresholds. However, these thresholds are chosen in an arbitrary manner and thus, the risk of overtraining in the resulting classification is high. In this work, we propose the use of the *Particle Swarm Optimization (PSO)* algorithm to optimally select the values of these thresholds. Furthermore, by using as classifier the SVM we can simultaneously apply the PSO algorithm to select the values of the SVM's free parameters. Our experimental results show that by using PSO for tuning the values of the split control thresholds and of the free parameters of the SVM classifier the generalization capabilities of the linear DECOC classification algorithm are significantly increased.

2 Linear Discriminant Error Correcting Output Codes Framework

In this section we give a brief description of the previously mentioned Linear DECOC framework.

As already mentioned, the idea that lies behind the ECOC is to decompose the multi-class problem into several binary ones, and combine the resulting solutions in order to solve the initial problem. We assume that the training dataset contains N data samples $\{\mathbf{x}_i\}_{i=1}^N \in \mathbb{R}^d$ that belong to N_c different classes with labels $\{y_i\}_{i=1}^N \in \{1, \dots, N_c\}$. The ECOC procedure consists of the *coding* and the *decoding* step. In the coding step, the goal is to construct L binary classifiers according to which, each class attains a distinct codeword comprised of the outputs of these classifiers on the trained classes. A classifier contributes a zero value to the codeword of a specific class if this class is not considered in the training of the given classifier. Otherwise, a -1 or $+1$ value is contributed to the codeword according to the classifier's training partition this class belongs to. By arranging these codewords as rows of a matrix we obtain the so called *coding matrix* $\mathbf{M} \in \{-1, 0, +1\}^{N_c \times L}$. The number of classifiers L depends strongly on the coding strategy used, that is, the method which controls how to decompose the problem and create the binary sub-problems. In the decoding step, each test sample \mathbf{x} is feeded into the L previously mentioned classifiers and a codeword for this sample is obtained. The resulting codeword is then compared with all

the codewords in the coding matrix \mathbf{M} and the test sample is assigned to the class that corresponds to the closest codeword according to a decoding measure.

Additionally to the ECOC framework, Pujol et al. proposed that we can use a binary tree design of ECOC, the so called *discriminant ECOC (DECOC)* where, given a number of N_c classes, we can achieve a high classification performance by training only $N_c - 1$ binary classifiers [13]. On the resulting DECOC algorithm Escalera et al. proposed that from an initial set of classes \mathcal{C} of a given multi-class problem, we can define a new set of classes \mathcal{C}' , where the cardinality of \mathcal{C}' is greater than that of \mathcal{C} , that is $|\mathcal{C}'| > |\mathcal{C}|$ [7]. The new set of binary problems that will be created will improve the created classifiers' training performance.

In both the DECOC and DECOC with sub-classes procedures, the discriminant binary tree structure is built by choosing the partitioning that maximizes the *mutual information (MI)* between the data samples and their respective class labels. The structure as a whole describes the decomposition of the initial multi-class problem into an assembly of smaller binary sub-problems. Each node of the tree represents a pair that consists of a specific binary sub-problem with its respective classifier. The construction of the tree's nodes is achieved through the *Sequential Forward Floating Search (SFFS)* algorithm [12]. On each step of the DECOC with sub-classes tree creation procedure, we can split the bi-partitions that consist the current sub-problem. The split can be achieved using K-means or some other clustering method. After the split, two new problems are formed that can be examined separately. The assignment of tree nodes to these sub-problems, depends on the satisfaction of three user defined thresholds. If these thresholds are satisfied, two new tree nodes are added to the overall tree structure along with their respective binary classifiers. Otherwise, the split is rejected and a tree node with its respective classifier is created assigned to solve the previously unsplitted problem. The three thresholds are defined as follows:

1. θ_{perf} : Split the classes if the created classifier attains greater than $\theta_{perf}\%$ training error.
2. θ_{size} : Minimum cluster's size of an arbitrary created sub-class.
3. θ_{impr} : Improvement in the training error attained by classifiers for the newly created problems against previous classifier (i.e., before split).

The computation of the MI in each step of the SFFS algorithm has complexity proportional to $\mathcal{O}(d^2N^2)$ where d is the number of dataset's attributes. Hence, the DECOC algorithm can't be applied on large datasets in reasonable running time. In dealing with this problem The use of the FLDR as an alternative optimization criterion in the SFFS algorithm resulting in the so called linear DECOC algorithm has been proposed in [2]. The resulting algorithm not only reduces the computational complexity of the overall procedure but also displays an improvement in the classification performance.

3 Particle Swarm Optimization

The PSO algorithm is a population-based search algorithm whose initial intent was to simulate the unpredictable behavior of a bird flock [9]. In this context,

a simple and efficient optimization algorithm emerged. Individuals in a particle population called *swarm* emulate the success of neighboring individuals and their own successes. A PSO algorithm maintains a swarm of these individuals called *particles*, where each particle represents a potential solution to the optimization problem. The position of each particle is adjusted according to its own experience and that of its neighbors. Let $\mathbf{p}_i(t)$ be the position of particle i in the search space at time step t . The position of the particle is changed by adding a velocity $\mathbf{v}_i(t)$ to the current position. This update can be written as

$$\mathbf{p}_i(t+1) = \mathbf{p}_i(t) + \mathbf{v}_i(t+1) \quad (1)$$

with $\mathbf{p}_i(0) \sim U(\mathbf{p}_{min}, \mathbf{p}_{max})$, where U denotes the uniform distribution.

In our approach, we implement the *global best* PSO, or *gbest* PSO. The gbest PSO is summarized in Algorithm 1.

Algorithm 1. gbest PSO

```

1: Create and initialize an  $n_p$ -dimensional swarm;
2: repeat
3:   for each particle  $i = 1, \dots, n_s$  do
4:     {set the personal best position}
5:     if  $f(\mathbf{p}_i) < f(\mathbf{y}_i)$  then
6:        $\mathbf{y}_i = \mathbf{p}_i$ ;
7:     end if
8:     {set the global best position}
9:     if  $f(\mathbf{y}_i) < f(\hat{\mathbf{y}})$  then
10:       $\hat{\mathbf{y}} = \mathbf{y}_i$ ;
11:    end if
12:   end for
13:   for each particle  $i = 1, \dots, n_s$  do
14:     update the velocity using (2)
15:     update the position using (1)
16:   end for
17: until stopping criterion is true

```

In this algorithm, the neighborhood for each particle is the entire swarm. For gbest PSO, the velocity of particle i is calculated as

$$\mathbf{v}_i(t+1) = \mathbf{v}_i(t) + c_1 \mathbf{r}_1(t)[\mathbf{y}_i(t) - \mathbf{p}_i(t)] + c_2 \mathbf{r}_2(t)[\hat{\mathbf{y}}(t) - \mathbf{p}_i(t)] \quad (2)$$

where $\mathbf{v}_i(t)$ is the velocity of particle i at time step t , $\mathbf{p}_i(t)$ is the position of particle i at time step t , c_1 and c_2 are positive *acceleration coefficients*, and $\mathbf{r}_1(t), \mathbf{r}_2(t) \sim U(0, 1)$ are random vectors with elements in the range $[0, 1]$, sampled from a uniform distribution. These vectors introduce a stochastic element to the algorithm.

The personal best position \mathbf{y}_i associated with particle i is the best position the particle has visited since the first time step. Considering minimization problems, the personal best position at the next time step $t+1$ is calculated as

$$\mathbf{y}_i(t+1) = \begin{cases} \mathbf{y}_i(t) & \text{if } f(\mathbf{p}_i(t+1)) \geq f(\mathbf{y}_i(t)) \\ \mathbf{p}_i(t+1) & \text{if } f(\mathbf{p}_i(t+1)) < f(\mathbf{y}_i(t)) \end{cases} \quad (3)$$

where $f : \mathbb{R}^{n_p} \rightarrow \mathbb{R}$ is the fitness function. This function measures how close the corresponding solution is to the optimum.

The global best position $\hat{\mathbf{y}}(t)$ at time step t is defined as

$$\hat{\mathbf{y}}(t) \in \{\mathbf{y}_0(t), \dots, \mathbf{y}_{n_s}(t)\} | f(\hat{\mathbf{y}}(t)) = \min f(\mathbf{y}_0(t)), \dots, f(\mathbf{y}_{n_s}(t)) \quad (4)$$

where n_s is the total number of particles in the swarm.

Using the standard gbest PSO algorithm, we observe that the velocity of the particles quickly explodes to very large values and as a result the swarm diverges from the optimal solution. In order to control this phenomenon we use the so-called *Velocity clamping* in our approach [6]. That is, we used the classical gbest PSO algorithm that integrates an inertia weight w [15]. This weight w controls the momentum of the particle by weighing the contribution of the previous velocity. So, the equation of the gbest PSO becomes

$$\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1\mathbf{r}_1(t)[\mathbf{y}_i(t) - \mathbf{p}_i(t)] + c_2\mathbf{r}_2(t)[\hat{\mathbf{y}}(t) - \mathbf{p}_i(t)] \quad (5)$$

The value of w is extremely important to ensure convergent behavior of the algorithm. For $w \geq 1$, velocities increase over time, accelerating towards the maximum velocity and the swarm diverges. For $w < 1$, particles decelerate until their velocities become zero.

The previously mentioned, user defined thresholds of the linear DECOC with sub-classes approach are clearly problem-dependent. As a result, due to the fact that we have no a priori knowledge about the structure of the data, we are in no position to efficiently select their values. Therefore, we purpose the use of the PSO algorithm in order to tune the thresholds $\{\theta_{perf}, \theta_{size}, \theta_{impr}\}$. Additionally, if we choose to use SVM as classifier in the linear DECOC technique, we proposed the use of the PSO for selecting simultaneously optimal values for the SVM classifier's free parameters (i.e., the cost parameter C and, in the case of an RBF kernel SVM, the kernel's σ). In this case, the PSO algorithm searches in a 5-dimensional swarm in order to find the optimal solution.

4 Experimental Results

For our experiments we used eight datasets of the UCI Machine Learning Repository [8]. The features of each dataset were scaled to the interval $[-1, +1]$. The characteristics of each dataset can be seen in Table 1.

Table 1. UCI Machine Learning Repository Data Sets Characteristics

Database	Samples	Attributes	Classes
Iris	150	4	3
Ecoli	336	8	8
Wine	178	13	3
Glass	214	9	7
Thyroid	215	5	3
Vowel	990	10	11
Balance	625	4	3
Yeast	1484	8	10

In our experimental configuration we proceeded as follows: We split randomly the datasets into two sets, a training and a test set. The training set contained approximately 60% of the whole data samples, and the test set the remaining 40%. As an objective function f in our PSO algorithm we used the 10-fold cross validation error of our classifier in the training set.

The respective PSO parameters used were the following:

- Inertia weight: $w(0) = 0.9$, $w(n_t) = 0.4$
- Number of particles: 20
- Number of iterations: 100
- Additional stopping criterion:

$$\frac{1}{p} \sum_{i=1}^{n_s} \|\hat{\mathbf{y}}(t) - \mathbf{x}_i(t)\| < tol \quad (6)$$

where $tol = 10^{-3}$

- Acceleration coefficients: $c_{1,max} = c_{2,max} = 2.5$ and $c_{1,min} = c_{2,min} = 0.5$

After the termination of the optimization procedure, we obtained as an output three optimized thresholds $\{\theta_{perf}, \theta_{size}, \theta_{impr}\}$ and also the optimized parameters of the SVM, C and in the case of RBF SVM also the width σ of the kernel function. Finally, we evaluated the optimized classifier in the test set by computing the resulting test error in each dataset. It is worth noting that no information about the test data was used during the parameters optimization. That is, the 10-fold cross validation inside the training set provided all the information needed for optimizing the parameters. We note this because it is common practice for many researchers to report optimized parameters in the test set without proposing a procedure on how someone can find these optimal parameters.

The PSO resulting splitting parameters were compared with the set of default parameters $\theta = \{\theta_{perf}, \theta_{size}, \theta_{impr}\}$ which were fixed for each dataset to the values as proposed in [7]:

- $\theta_{perf} = 0\%$, i.e., split the classes if the classifier does not attain zero training error.
- $\theta_{size} = \frac{|J|}{50}$, minimum number of samples in each constructed cluster, where $|J|$ is the number of features in each dataset.
- $\theta_{impr} = 5\%$, the improvement of the newly constructed binary problems after splitting.

Furthermore, as a clustering method we used the K-means algorithm with the number of clusters $K = 2$. As stated by Escalera et al. [7], the K-means algorithm obtains similar results with other more sophisticated clustering algorithms, such as hierarchical and graph cut clustering, but with much less computational cost.

As a standard classifier for our experiments we used the LIBSVM [3] implementation of the Support Vector Machine with linear and RBF kernel. We compared our optimized classifier against the default classifier used in the LIBSVM package, that is a linear SVM with $C = 1$ and an RBF SVM with $C = 1$

and $\sigma = 1/attributes_{nr}$, where $attributes_{nr}$ is the number of features in each dataset.

The resulting classification accuracies obtained are shown in Table 2. we also give the (number of rows \times number of columns) of the encoding matrices formed as a measure of the split.

Table 2. UCI Repository Experiments for Linear and RBF SVM

Database	Linear SVM		RBF SVM	
	PSO	Default	PSO	Default
Iris	97% (3×4)	96.67% (3×4)	98.3% (3×4)	96.67 % (3×4)
Ecoli	84.14% (13×15)	54.48% (15×17)	84.83% (14×17)	22.76% (13×17)
Wine	95.71% (3×4)	94.29% (3×4)	98.57% (3×4)	97.14% (3×4)
Glass	55.81% (8×10)	52.53% (6×5)	61.63% (6×5)	54.65% (6×5)
Thyroid	95.24% (3×2)	92.06% (3×2)	93.65% (3×2)	84.13% (4×4)
Vowel	50.43% (27×31)	46.75% (37×46)	57.14% (11×10)	51.08% (15×15)
Balance	92.4% (21×26)	49.2% (59×67)	98.8% (3×2)	46.8% (3×2)
Yeast	53.20% (11×10)	38.38% (11×10)	57.07% (10×9)	34.68% (17×21)

From the results, it is obvious that the optimized linear DECOC using PSO always outperforms the default classifiers in all of the experiments conducted. The improvement is attributed to the fact that PSO finds the optimum values for the thresholds that control the resulting number of sub-classes. Furthermore, by finding via PSO optimum values for the SVM parameters (i.e., C and σ), the classification performance is further improved. In certain datasets the thresholds returned by PSO do not result in the creation of any sub-classes. In this case, PSO reveals that, in the specific dataset, it is highly probable that the use of sub-classes will lead to over-fitting. We can also see that in the RBF SVM the performance improvement is more significant than in the Linear SVM. This can be associated to the major role the σ parameter plays in the classification performance of the RBF SVM.

5 Conclusion

In this paper we used PSO in order to optimize the linear DECOC framework. As was shown, PSO can be a reliable method for choosing optimal values for the free parameters that control the split of the initial multi-class problem's classes into sub-classes resulting in improved classification performance.

References

1. Allwein, E.L., Schapire, R.E., Singer, Y.: Reducing Multi-class to Binary: A Unifying Approach for Margin Classifiers. *Journal of Machine Learning Research* 1, 113–141 (2002)
2. Arvanitopoulos, N., Bouzas, D., Tefas, A.: Subclass Error Correcting Output Codes Using Fisher’s Linear Discriminant Ratio. In: 20th International Conference on Pattern Recognition (ICPR 2010) , pp. 2953–2956 (2010)
3. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001), <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
4. Cortes, C., Vapnik, V.: Support-vector networks. *Machine Learning* 20, 273–297 (1995)
5. Dietterich, T.G., Bakiri, G.: Solving Multi-class Learning Problems via Error-Correcting Output Codes. *Journal of Machine Learning Research* 2, 263–282 (1995)
6. Eberhart, R.C., Shi, Y.: Computational Intelligence (2007)
7. Escalera, S., Tax, D.M.J., Pujol, O., Radeva, P., Duin, R.P.W.: Subclass Problem-Dependent Design for Error-Correcting Output Codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30(6), 1041–1054 (2008)
8. Frank, A., Asuncion, A.: UCI Machine Learning Repository (2010), <http://archive.ics.uci.edu/ml>
9. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference On Neural Networks, Perth, Australia, vol. 4, pp. 1942–1948 (1995)
10. Kittler, J., Ghaderi, R., Windeatt, T., Matas, J.: Face verification using error correcting output codes. In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2001, vol. 1 (2001)
11. Kong, E.B., Dietterich, T.G.: Error-Correcting Output Coding Corrects Bias and Variance. In: Proceedings of the 12th International Conference on Machine Learning, pp. 313–321 (1995)
12. Pudil, P., Ferri, F.J., Novovicova, J., Kittler, J.: Floating Search Methods for Feature Selection with Non-monotonic Criterion Functions. In: Proceedings of the International Conference on Pattern Recognition, vol. 3, pp. 279–283 (March 1994)
13. Pujol, O., Radeva, P., Vitria, J.: Discriminant ECOC: A Heuristic Method for Application Dependent Design of Error Correcting Output Codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6, 1001–1007 (2006)
14. Rudin, C., Daubechies, I., Schapire, R.E.: The Dynamics of Adaboost: Cyclic Behavior and Convergence of Margins. *J. Mach. Learn. Res.* 5, 1557–1595 (2004)
15. Shi, Y., Eberhart, R.C.: Empirical study of particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation, CEC 1999 (1999)
16. Windeatt, T., Ardesir, G.: Boosted ECOC ensembles for face recognition. In: International Conference on Visual Information Engineering, VIE 2003, pp. 165–168 (2003)
17. Zhou, J., Suen, C.Y.: Unconstrained Numeral Pair Recognition Using Enhanced Error Correcting Output Coding: A Holistic Approach. In: Document Analysis and Recognition, vol. 0, pp. 484–488 (2005)

SOS-HMM: Self-Organizing Structure of Hidden Markov Model

Rakia Jaziri, Mustapha Lebbah, Younès Bennani, and Jean-Hugues Chenot

LIPN-UMR 7030 - CNRS, Université Paris 13,

99, av. J-B Clément F-93430 Villetaneuse

Institut National de l'Audiovisuel,

4, av. de l'Europe 94 366 Cedex Bry-sur-Marne

{younes.bennani, mustapha.lebbah, rakia.jaziri}@lipn.univ-paris13.fr,

{rjaziri, jhchenot}@ina.fr

Abstract. We propose in this paper a novel approach which makes self-organizing maps (SOM) and the Hidden Markov Models (HMMs) cooperate. Our approach (SOS-HMM: Self Organizing Structure of HMM) allows to learn the Hidden Markov Models topology. The main contribution for the proposed approach is to automatically extract the structure of a hidden Markov model without any prior knowledge of the application domain. This model can be represented as a graph of macro-states, where each state represents a micro model. Experimental results illustrate the advantages of the proposed approach compared to a fixed structure approach.

1 Introduction

Hidden Markov Models (HMMs) [2] appears among the best approaches adapted to sequences analysis, because of both their ability to deal with sequences of variable lengths, and of their power to model the dynamic of a phenomenon described by sequences of events. This is why these models were widely used in many pattern recognition areas, including biological sequence modeling [6], speech recognition [11], optical character recognition [10] and information extraction [7], [8]. Clearly, there are many possible Markov structures that can be constructed according to the needs of particular applications. One of the major restrictions of HMM is their inability to automatically extract the architecture of models without any prior knowledge of the application domain. In most cases, the model topology is defined according to some prior domain's knowledge. Automatic techniques for extracting the HMM topology are interesting as the structures are sometimes hard to define a priori, or need to be tuned after some task adaptation. The work described here presents a novel approach towards this objective.

Self-Organizing Maps (SOM) proposed by Kohonen are interesting in unsupervised classification because of their topological structure [9]. Basically, their major interest is their capacity to summarize a set of multidimensional data in a simple way. They allow, on the one hand, compressing large quantities of data by

grouping together similar instances in classes, and on the other hand, to visualize projecting the classes obtained in a nonlinear way onto a map (as clusters). Thus they perform a reduction of dimension that unravels the structure of the data set in two dimensions, while preserving the topology of the data. Several methods were proposed in the literature for the adaptation of these models (SOM) to data structured in sequences. The principle of these methods consists of introducing the temporal dimension either into the coding of the input data, or in the learning process, as proposed in [14]. Other attempts [3,4] have been made for combining HMMs and SOM to form hybrid models that extract the topology. In many of these hybrid architectures, SOM models are used as front-end processors for vector quantization, and HMMs are then used in higher processing stages. In [12] the authors propose an original hybrid approach model where the topology of HMM can be defined from cluster's structure. In fact, each HMM represents a precise area of the map. It considers each cluster as an independent HMM. Note that this model does not include the link between clusters. The present contribution describes a novel approach to the structural extraction of HMMs. The general objective of SOS-HMM is to learn the topology of HMM without any prior knowledge of the application domain.

The organization of the paper is as follows: In section 2, the novel concept of the proposed approach which combines the strong points of HMM and SOM is presented. Section 3 reports the obtained results. Finally, conclusions and some future works are laid in section 4.

2 From Self-Organizing Map to SOS-HMM

To describe our approach, firstly, we will introduce the basis Self-Organizing Map that we have used for the preprocessing phase. As traditional self-organizing map, we assume that the lattice \mathcal{C} has a discrete topology defined by an indirect graph. Usually, this graph is a regular grid in one or two dimensions. We denote K the number of cells in the map \mathcal{C} . For each pair of cells (c, r) on the map, the distance $\delta(c, r)$ is defined as the length of the shortest chain allowing to

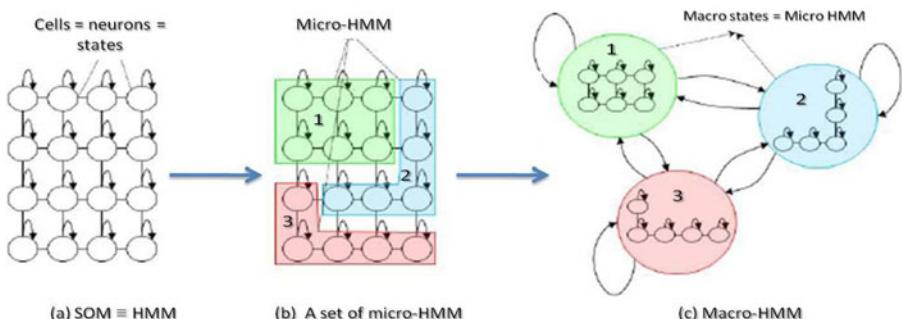


Fig. 1. General process used by SOS-HMM to built the macro-HMM structure

link cell r and c . If the cells are associated with probabilities, this leads to the concept of hidden Markov Models (HMM). In this lattice each unit represents an HMM state (Fig 1.a). In the following this model will be titled micro-HMM. Let $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n, \dots, \mathbf{x}_N\}$ be a sequence, of M -dimensional observations of size N . The states and their interactions are the core of the HMM. Formally the HMM is defined as triple $\lambda(\Pi, \mathbf{A}, \mathbf{B})$ where Π is the initial state probabilities, \mathbf{A} the state transition probabilities and \mathbf{B} is the emission probabilities.

The idea in the present work is that the micro-HMM is not trained but deducted from the SOM model. First we deduct a large HMM using SOM model and then partitioning it into clusters. Each cluster models macro-state. Thus all the probabilities characterizing a micro-HMM associated to a cluster will be estimated from the parameters of the topological map:

- Hidden states of micro-HMMs are denoted by $C = \{c_1, \dots, c_K\}$ where K is the number of states.
- $\Pi = \{\pi_1, \dots, \pi_K\}$ is a set of initial probability distribution. The initial probability distribution depends on the size of elements assigned to each cell.

$$\pi_i = \frac{\#(c_i)}{|X|}, 1 \leq i \leq K \quad (1)$$

Where $\#(c_i)$ is the number of element assigned to cell c_i and $|X|$ is the length of a sequence.

- The generation of the observed variable \mathbf{x}_i in state c_i at a given time is conditional on the neighborhood states at that same time. Thus, a high proximity implies a high probability to contribute to generation. This proximity is quantified using neighborhood function \mathcal{K} defined as follows:

$$\mathcal{K}(\delta(c_i, c_j)) = e^{\frac{-\delta(c_i, c_j)}{T}}, T_{min} \leq T \leq T_{max} \quad (2)$$

Where $\delta(c_i, c_j)$ is the Manhattan distance and the parameter T allows to control the size of influencing neighborhood of a given cell on the map. Thus, the transition probability from state c_i to state c_j is defined by:

$$a_{i,j} = P(c_i/c_j) = \frac{\mathcal{K}(\delta(c_i, c_j))}{\sum_{i=1}^K \mathcal{K}(\delta(c_i, c_j))}, 1 \leq i, j \leq K \quad (3)$$

- The emission probability in a state c_i is defined by a Gaussian distribution which depends on two parameters: "standard deviation" σ_i and the mean \mathbf{w}_{c_i} .

$$P(\mathbf{x}_i/c_j) = \frac{1}{\sqrt{2\pi}\sigma_{c_j}} \exp \frac{-||\mathbf{x}_i - \mathbf{w}_{c_j}||^2}{2\sigma_{c_j}^2}, 1 \leq i \leq N \text{ and } 1 \leq j \leq K \quad (4)$$

The global HMM is clustered into S clusters (Fig 1.b) by applying *Hierarchical Ascending Classification* [5] on the prototypes \mathbf{w}_{c_i} . Each cluster represents a

macro state S_k with a model called macro-HMM (Fig1.c). In other words, a macro-HMM is a generalization of an HMM where each state in the macro-HMM is itself a micro HMM (cluster in the map). Thus, a macro-HMM consists of a set of micro-HMM. In a macro-HMM, each macro state S is itself a standard HMM defined by the parameter set $\lambda^* = (A^*, B^*, \Pi^*, C^*)$:

- The initial macro state probability distribution $\Pi = \{\pi_1^*, \dots, \pi_S^*\}$, is determined by the sum of initial probability of the micro-HMM's states.

$$\pi_k^* = \sum_{j \in S_k} \pi_j \quad (5)$$

- To define transition probabilities of macro-HMM, each macro-state is associated to one micro-HMM's state c^* representing the cluster. Then, the transition probability is defined using the neighborhood function \mathcal{K} as follows:

$$a_{i,j}^* = P(c_i^* / c_j^*) = \frac{\mathcal{K}^T(\delta(c_i^*, c_j^*))}{\sum_{c_i} \mathcal{K}^T(\delta(c_i^*, c_j^*))}, 1 \leq i, j \leq S \quad (6)$$

- Each macro state S_k is a mixture models of $|S_k|$ components. The emission probability is defined as follows:

$$b_i^* = P(\mathbf{x} / S_k) = \sum_{c_j \in S_k} P(c_j / c_k^*) P(\mathbf{x} / c_j) \quad (7)$$

Each macro-state models the dynamics of only one cluster. They are connected to each other within on resulting in a macro-HMM. The connections associated with transitions probabilities are computed using the neighborhood function. The neighborhood function is used in both model (micro and macro)-HMM, in order to control the states organization. The best state sequence is obtained by using the Viterbi algorithm. Selecting only the best sequence corresponds to the winner principle of the SOM.

3 Experiments

The dataset here, were used for a PhD study on primitive extraction using HMM based models. The data consists of 2858 character samples, contained in the cell array 'mixout' [1]. The data were captured using a WACOM tablet. Three dimensions were kept - x , y , and pen tip force. Data was captured at 200Hz. The data were numerically differentiated and Gaussian smoothed. The data were normalized. Only characters with a single 'PEN-DOWN' segment were considered. Character segmentation was performed using a pen tip force cut-off point. The characters were also shifted so that their velocity profiles best match the mean of the set. Each character sample is a 3-dimensional pen tip velocity trajectory. The single best state sequence is obtained by using the Viterbi algorithm [13]. This algorithm is used in this work for choosing the best-matching model sequence.

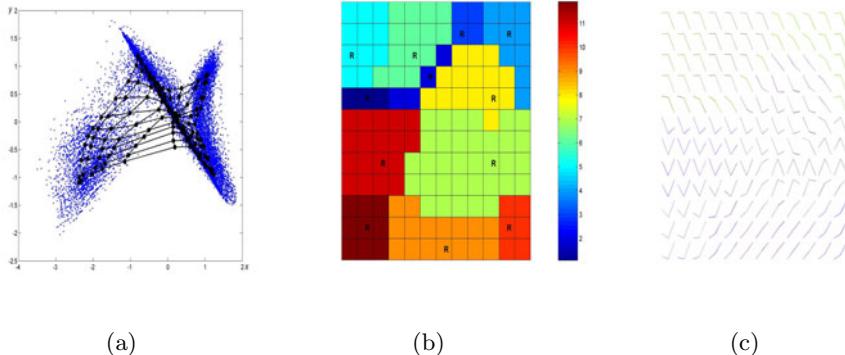


Fig. 2. *a*-data set training. (a) Macro-HMM map 12×12 . (b) Segmentation of the map, (c) Profiles of each prototype associated with map's units

Our first experiment was to run separately SOS-HMM model on each letter. For example, the figures 2.a shows the PCA projection and the map projection of the *a*-data set. The points on the left figure present the element of original sequence. It is necessary to establish macro-states to create a Markov model from map. One way to define these macro-states is with a clustering method, and as presented in the previous section, the Self-Organizing Map is a good tool. In this case, each cell in the SOM represents a micro-state of the Markov model. We can observe that the topological order of micro-states respect the topological order of data set: close samples are represented by close states. Our approach reveals an important implication to the neighborhood function to organize macro-HMM states. A clear organization of prototypes is observed in map figure 2.c. Various model selection techniques could be used to automate the selection of the appropriate features and states. We can detect the relevant features associated with each state or cluster. One of the characteristic features distinguishing macro-HMM from other HMM paradigms, is the topographic preservation using the neighborhood function. The figure 2.c shows the reference maps (profile) corresponding respectively to map figure 2.b. Each cell is a plot of the 3 variables : x velocity, y velocity and pressure differential. In the figure it is possible to identify some relationships between variables. The topographic maps in our macro-HMM model allow us to visualize the partition.

Usually, to build an HMM, we must first decide how many states the model should contain, and which transitions between states should be allowed. However, the SOS-HMM allows to automatically detect the structure of the models. Figure 3, shows illustration of Macro-HMM structure provided at the end of SOS-HMM process. As shown in figure each macro-state correspond to cluster. Low probabilities transition are thresholded out (≤ 0.001).

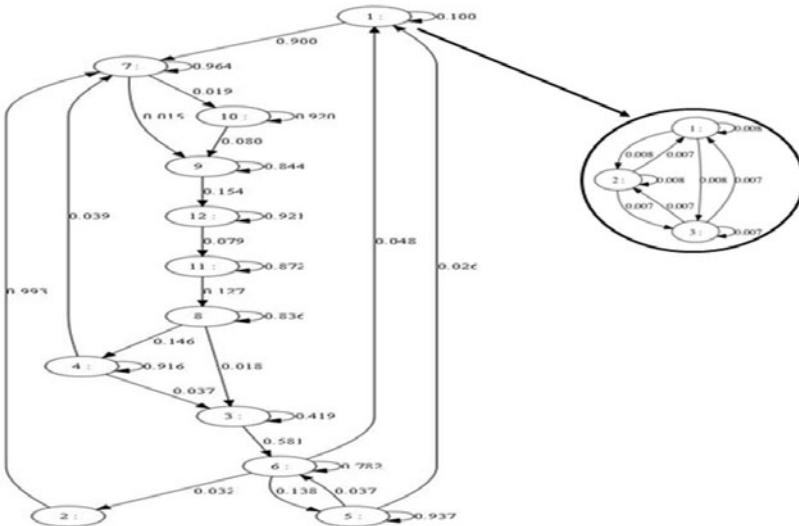


Fig. 3. Architecture of macro-HMM corresponding to the a -data set. The left fragment corresponds to macro-state.

Table 1. Cross validation with $\{a, b, h, m, n\}$ -data set. Classification accuracy.

Data set	HMM	Macro-HMM
a -data set	[96,38-99,53]	[98,48-100]
b -data set	[97,25-99,81]	[98,48-100]
h -data set	[80,77-89,46]	[96,04-99,40]
m -data set	[64,46-75,71]	[98,48-100]
n -data set	[57,71-69,54]	[98,48-100]

In order to show that learning phase is well done, we calculate for each example the Viterbi path. Figure 4 shows the original (in the left (1)) and the reconstruction sample (in the right (2)). We observe a clear topological organization of the global HMM states. These projections provide a topographical visualization of the sequential data set.

Our second experiment was to use the K -fold cross-validation technique, repeated n times for $n = 5$ and $k = 5$, to estimate the performance of SOS-HMM. In this case, we used $\{a, b, h, m, n\}$ -data set. For each run, the data set was split into five disjoint groups. We used four subsets for training and then tested the model on the remaining subsets. The labels generated were compared to the real labels of the test set for each run. Empirical results were obtained using SOS-HMM and HMM approaches. Table 1 shows that using self-organizing map improves the SOS-HMM performance and reduces the variability of the results. We point out here that SOS-HMM model provides more information for visualizations.

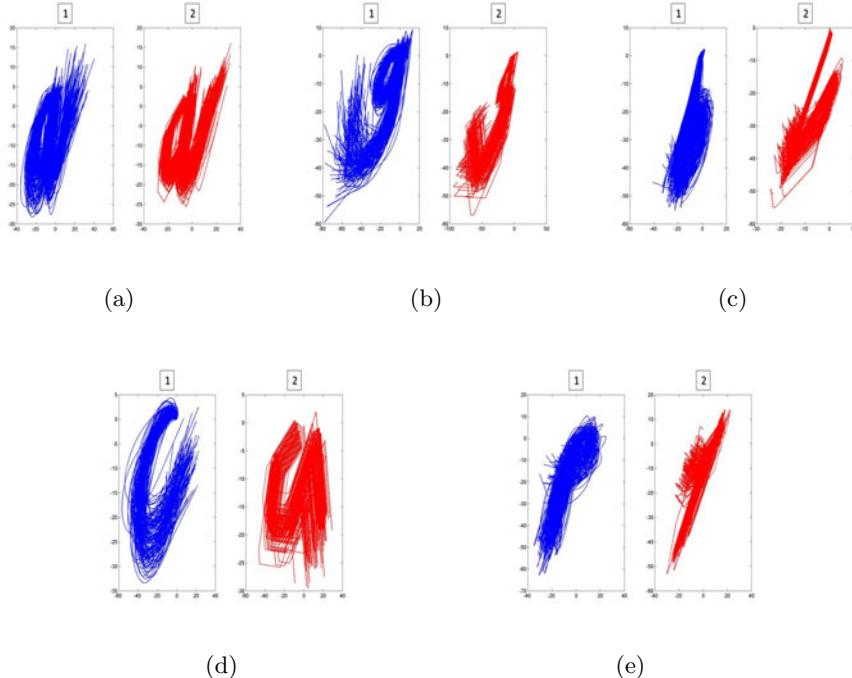


Fig. 4. Samples reconstructions using SOS-HMM map 12×12 . Left characters show the original samples and right characters show the reconstruction sample.

4 Conclusion

We propose in this paper a novel approach that enables to discover a structure of an hidden Markov model, without any prior knowledge of the application domain. It is an approach that determines an optimized and adapted structure of HMM, driven by sequential data. This topology is detected automatically using self-organizing map. The identified structure of HMM is adapted to the structure and the quality of data. The proposed approach SOS-HMM was evaluated on a sequences representing letters. The results obtained are encouraging and show the effectiveness of automatic structure learning. Our future work will consist of adapting the proposed approach to visual data processing.

References

1. Asuncion, A., Newman, D.: UCI machinelearning repository (2007), <http://www.ics.uci.edu/mlearn/MLRepository.html>
2. Baum, E.L., Petrie, T., Soules, G., Weiss, N.: A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The Annals of Mathematical Statistics* 41, 164–171 (1970)

3. Ferles, C., Stafylopatis, A.: Sequence clustering with the Self-Organizing Hidden Markov Model Map. In: 8th IEEE International Conference, pp. 1–7 (2008)
4. Ferles, C., Stafylopatis, A.: A Hybrid Self-Organizing Model for Sequence Analysis. In: 20th IEEE International Conference on Tools with Artificial Intelligence, pp. 105–112, Washington, DC, USA (2008)
5. Chavent, M.A.: Monothetic clustering method. *Pattern Recognition Letters* 19, 989–996 (1998)
6. Durbin, R., Eddy, S., Krogh, A., Mitchison, G.: *Biological Sequence Analysis*. Cambridge University Press, Cambridge (1998)
7. Freitag, D., McCallum, A.: Information extraction with HMMs and shrinkage. In: Proc. of the AAAI Workshop on Machine Learning for Information Extraction (1999)
8. Freitag, D., McCallum, A.: Information extraction with HMM structures learned by stochastic optimization. In: Proc of the Seventeenth National Conference on Artificial Intelligence, AAAI, pp. 584–589, (2000)
9. Kohonen, T.: *Self-Organizing Map*. Springer, Heidelberg (1995)
10. Levin, E., Pieraccini, R.: Planar Hidden Markov modeling: from speech to optical character recognition. In: Giles, C., Hanton, S., Cowan, J. (eds.) *Advances in Neural Information Processing Systems*, vol. 5, pp. 731–738. Morgan Kauffman, San Francisco (1993)
11. Rabiner, R.: A tutorial on hidden markov models and selected applications. In: speech recognition. *Proceedings of the IEEE* 77(2), 257–286 (1989)
12. Rogovschi, N., Lebbah, M., Bennani, Y.: Learning Self-Organizing Mixture Markov Models. *Journal of Nonlinear Systems and Applications, JNSA* (2010) ISSN. 1918-3704
13. Viterbi, A.J.: Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 260–267 (1967)
14. Zehraoui, F., Bennani, Y.: New self-organising maps for multivariate sequences processing. *International Journal of Computational Intelligence and Applications* 5(4), 439–456 (2005)

Image Receptive Fields Neural Networks for Object Recognition

Paméla Daum, Jean-Luc Buessler, and Jean-Philippe Urban

MIPS Laboratory

Université de Haute-Alsace, Mulhouse, France

{pamela.daum,jean-luc.buessler,jean-philippe.urban}@uha.fr

<http://www.trop.uha.fr>

Abstract. This paper extends a recent and very appealing approach of computational learning to the field of image analysis. Recent works have demonstrated that the implementation of Artificial Neural Networks (ANN) could be simplified by using a large amount of neurons with random weights. Only the output weights are adapted, with a single linear regression. Supervised learning is very fast and efficient. To adapt this approach to image analysis, the novelty is to initialize weights, not as independent random variables, but as Gaussian functions with only a few random parameters. This creates smooth random receptive fields in the image space. These *Image Receptive Fields - Neural Networks* (IRF-NN) show remarkable performances for recognition applications, with extremely fast learning, and can be applied directly to images without pre-processing.

Keywords: Artificial neural networks, image processing, classification, extreme learning machine, echo state networks, image receptive fields.

1 Introduction

The idea of learning from examples is compelling and well suited to image analysis. The efficiency of the ANN learning is confirmed by applications like character or handwritten numerals recognition [1] or face localization [2]. Research has been very active for the past thirty years and many neuronal architectures have been proposed and tested, from classical to more particular architectures [3]. These ANN remain however difficult to use and practical real-world solutions are still too scarce.

ANN are interesting tools, but new developments are necessary to apply them efficiently to the image analysis field. Recent works [4,5], in very different application fields, have shown that it was possible to considerably simplify the implementation of ANN. The general idea is to use a large number of neurons with randomly initialized weights that are kept constant. Only the output weights are adapted during the learning phase. The adaptation rules are therefore simple because the problem is linear.

This approach became first popular by succeeding in the identification of dynamical systems using recurrent networks, well-known for their difficult training. Various studies and applications have been introduced in terms of Reservoir Computing and Echo State Network (ESN) [4].

A similar approach, the Extreme Learning Machine (ELM) [5], has been devised for feed-forward networks. It performs very well for fitting applications or classification problems as long as the dimension of the input space is moderate. Therefore, the ELM cannot be used to process an image directly, but characteristic image features need to be extracted first.

This paper shows that it is possible to adapt the concept of randomly initialized constant weights to image processing, with the same efficiency and ease of use that made the success of the ESN and ELM. The proposed technique allows to use the image directly as a pixel array, without size reduction or prior feature extraction. No iterations or sliding window are introduced as a convolution filter would do. The technique applies to prior localized sub-images like in character recognition, or directly to entire medium sizes images for object classification.

The original contribution of this work is to constrain the weights of the neurons. They are not considered as independent random variables, but as elements forming a continuous and smooth receptive field in the image. Each neuron is endowed with limited degrees of freedom that determine the characteristics of its receptive field: position, shape, size, magnitude, etc. For this purpose, elliptic Gaussian functions are chosen. During the initialization of the network, the free parameters assigned to each neuron are set randomly; the Gaussian function determines the connection weight of each pixel to the neuron. The main characteristic of the approach is emphasized by naming it IRF-NN: Image Receptive Fields Neural Networks.

The paper is organized as follows. Section 2 introduces the proposed network and describes its characteristics. Section 3 analyzes its behavior on a set of geometrical shapes. Section 4 shows its performances on various sets of real-world images. The paper shows the benefits of the IRF network and its remarkable potential.

2 Image Receptive Fields Neural Network

2.1 Structure of the Network

The proposed neural network is based on a classical feedforward architecture with a single hidden layer (MultiLayer Perceptron - MLP) [6]. The input of the MLP is a copy of the image, presented as a pixel-array (fig. 1a). Each input element is connected to all neurons of the hidden layer and contributes to their activation proportionally to the connectivity weights. The neurons of the hidden layer perform thus a weighted sum of the input image and output a non-linear response, using typically a sigmoid function. The activation of neuron i of image k can be represented by equation

$$h_{ik} = \tanh(\alpha_i \sum_{x,y} (g_i(x,y) \cdot \phi_k(x,y)) + \beta_i), \quad (1)$$

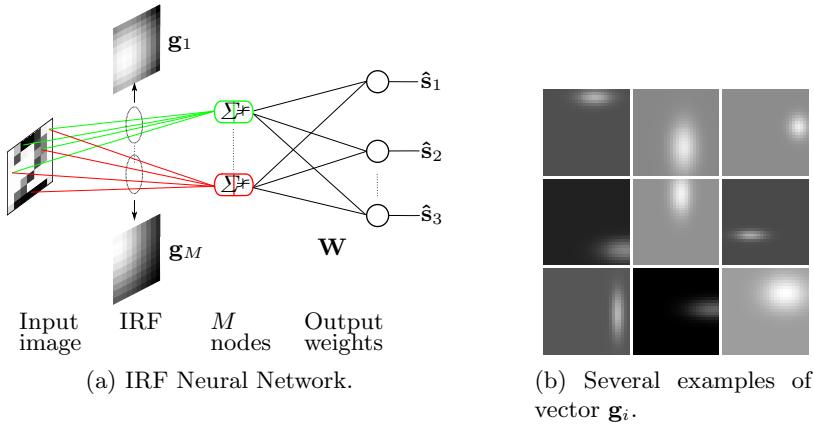


Fig. 1. Random Image Receptive Fields neural network and examples of Gaussian receptive fields. The random IRF of a neuron \mathbf{g}_i is represented as an image.

where α_i and β_i are respectively a multiplying factor and the neuron bias. The weight of the neuron, noted $g_i(x, y)$, is detailed in the next sub-section. Image k is represented by function $\phi_k(x, y)$, the gray-level of the pixel of coordinates (x, y) .

The output of a hidden layer of M neurons is vector \mathbf{h}_k . The output layer of the network consists of a single linear neuron for a scalar output, or a set of m linear neurons for multiple attributes learning. The output of the network is determined by weights \mathbf{w}_i ,

$$\hat{\mathbf{s}}_k = \sum_{i=1}^M \mathbf{w}_i h_{ik}, \quad (2)$$

where $\hat{\mathbf{s}}_k$ is a vector of the dimension m .

The output weights \mathbf{w}_i are determined by supervised learning based on a set of examples. In matrix notations, \mathbf{S} represents the desired outputs for the N images, and \mathbf{H} is the activation of the neurons for these images, with respective dimensions: $m \times M$ and $M \times N$. Learning determines weights \mathbf{W} , performing the linear regression

$$\mathbf{W} = \mathbf{S} \mathbf{H}^\dagger, \quad (3)$$

where \mathbf{H}^\dagger is the Moore-Penrose pseudoinverse of matrix \mathbf{H} .

2.2 Image and Receptive Fields Neurons

The original approach presented in this paper does not modify the basic structure of the network but rather extends the possibilities of using it. As for the ELM and ESN networks, the number of neurons M is taken relatively large, from several dozens to hundreds. The input weights \mathbf{g}_i are randomly initialized and

Table 1. *GEO* and *ALOI* configuration and recognition results

	M	σ	α	T (%)	$\varepsilon(x)$	$\varepsilon(y)$	$\varepsilon(h)$
<i>GEO-1</i>	200	[0.5 ; 2]	[-1 ; 1]/11	100	$1.6e^{-2}$	$1.6e^{-2}$	$4e^{-1}$
<i>GEO-2</i>	400	[0.5 ; 2]	[-1 ; 1]/32	99.99	$4e^{-3}$	$5e^{-3}$	$3e^{-1}$
<i>ALOI-V200</i>	1000	[0.01 ; 0.3]	[-1 ; 1]/15	99.5			
<i>ALOI-I1000</i>	1000	[0.01 ; 0.05]	[-1 ; 1]/0.7	99.5			

remain constant during the training phase. The training of the network for a set of examples adapts only the output weights \mathbf{w}_i .

To work effectively directly on images, without prior feature extraction, the input weights are organized as smooth functions in the image space. Non-linear parametrized functions provide some degrees of freedom that turn out to be sufficient to process images with precision. The sigmoid or Gaussian functions provide an interesting interpretation that will be developed in a future publication. In this paper, an elliptic Gaussian equation is chosen to determine \mathbf{g}_i :

$$g_i(x, y) = \gamma_i + \exp \left[-\frac{(x - \mu_{xi})^2}{(n_x \sigma_{xi})^2} - \frac{(y - \mu_{yi})^2}{(n_y \sigma_{yi})^2} \right], \quad (4)$$

where γ_i , σ_i and μ_i are randomly defined constants, n_x and n_y are the width and height of the images.

Figure 1b presents a few examples of weights under Gaussian constraint. With this initialization of the weights, each neuron responds in the input space to a localized and smooth region, named Receptive Fields in Neurosciences. We adopt the terminology *Image Receptive Fields* to characterize this approach.

3 Geometrical Figures in Binary Images

The IRF-NN presents interesting properties for supervised learning from a set of images:

- the network uses images without prior feature extraction;
- the adaptation phase is fast and requires no iteration;
- the parameters to be set are very few and straightforward to determine.

This section illustrates these property on geometrical objects in binary images. Results of photography classification are presented in section 4.

3.1 Presentation of the Experiments

The first set of experiments evaluates network learning and generalization when the size and location of the objects vary. The dataset *GEO-1* is constituted of 30×30 binary images representing simple geometrical figures. Each geometrical element takes successively all sizes from 10-28 pixels and all possible locations within the image boundary. In total, *GEO-1* is composed of 9,915 images for 3 geometrical object classes : square, disc and triangle (Fig. 2a).

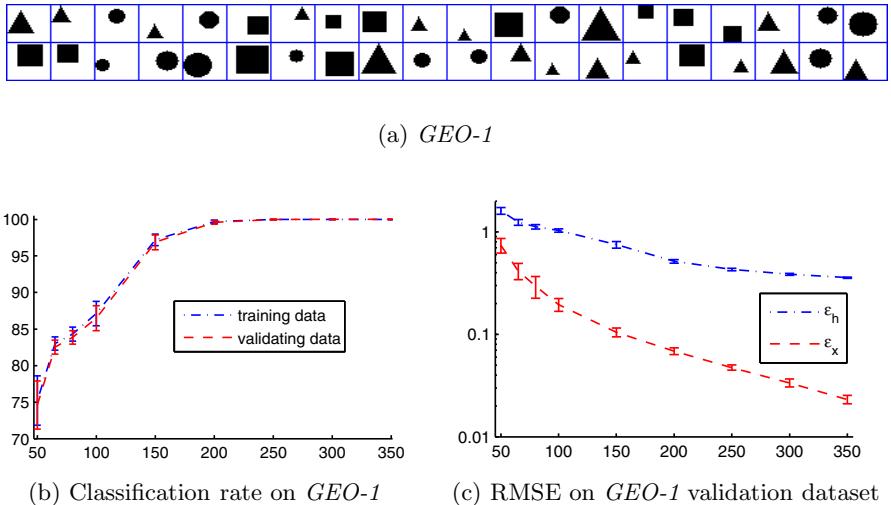


Fig. 2. *GEO-1* object samples (a) and recognition results (b, c) as a function of the number of neurons

Half of these images, picked randomly, form the training dataset. The IRF-NN is simultaneously trained to identify the classification model and numerical values with a data vector $\mathbf{S} = [\mathbf{s}_h, \mathbf{s}_x, \mathbf{s}_y, \mathbf{B}]^T$. Numerical values $s_h(k), s_x(k), s_y(k)$ are respectively height and coordinates of object in image k . Its class $s_c(k)$ is represented by using a 1-of- n_c code with the binary vector $\mathbf{b}_k \in \{-1, 1\}^{n_c}$ of dimension n_c , the number of classes. The index of the only element set to one indicates the class: $b_{s_c}(k) = 1$. The network classification response $\hat{s}_c(k)$ is the index of the largest output value in $\hat{\mathbf{b}}(k)$.

The validation dataset is formed with the images that were not presented during learning. The results are expressed as the root mean square error ε for the numerical values. Classification rate T is given as the percentage of true responses.

Table 1 summarizes the main configuration of the networks, setting the range of values for random initialization (with uniform distribution) of receptive fields attributes. The centers of the Gaussian functions μ_{xi} and μ_{yi} are set randomly in the image dimensions. Variables β_i and γ_i are equal to zero in all applications presented here.

3.2 Main Results

The experiments highlight and confirm several interesting characteristics of the IRF networks:

- its Gaussian receptive fields and sigmoid function transform the image in a remarkable internal representation which can approximate an arbitrary function of the image with good precision, with a simple linear regression;

- the adaptation of its output weights is precise and numerically stable;
- the generalization capabilities for new images is excellent.

The training dataset used here defines relatively complex functions, knowing that a given shape is represented by very different images with varying object location and size. A quite large number of neurons is therefore required, about 300. Figures 2b and 2c present classification rates, mean errors and standard deviation on the training and validation datasets, for series of 50 runs. For each run, all random variables and network weights are reinitialized, as well as the distribution between learning dataset and validation dataset. The graph shows that:

- after the adaptation of its output weights, the network is capable of determining the type of object without error, and to determine its position with a mean error $< 1.6e^{-2}$ pixel;
- the performances increase with the number of neurons, larger networks can guarantee a 100 % success rate for each run;
- the generalization works well even for large networks, the SVD-based Moore-Penrose technique ensuring a sufficient stability for the inversion of matrix \mathbf{H} (Eq. 3).

A second experimentation on binary figures illustrates the precision of the internal coding. A new class of objects is defined by modifying only a single pixel in the previous images: all squares of the first series have their upper left corner pixel removed (chamfered squares). The networks previously trained recognize, at 100%, these objects as squares, which complies with the desired generalization property. But it is also possible to train new networks to differentiate both classes, the complete squares and the chamfered squares. Therefore *GEO-2* is composed of 4 classes: square, disk, triangle, chamfered square. It is used in training and validation as with *GEO-1*. Although the two square classes differ of only one pixel, the classification is obtained at 99.99%. Similar results are obtained with triangles or disks: the modification of a single pixel is sufficient to discriminate the shapes.

The algorithmic cost of the IRF-NN is also interesting: the weight adaptation duration for 200 neurons on 4,959 examples (30×30 images) is 500 ms and the response time for one image is < 4 ms.¹

4 Photographic Object Recognition - the ALOI Database

The IRF network shows also good performances on larger images, in gray-level, and on real-world objects. The experiments presented in this section use the ALOI image database, a collection of images of a thousand photographic objects [7]. Hundred images of each object are recorded with systematical variation of viewing angle, illumination angle, and illumination color for each object. We

¹ The network is coded in Matlab version 7.9, runs on a Linux OpenSuse 11 (64bits) environment, and Intel Core 2 CPU 2.4GHz processor with 2Go of RAM.

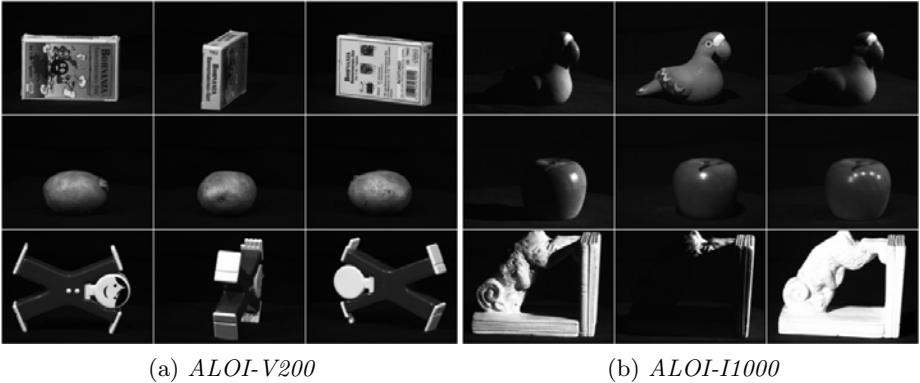


Fig. 3. Samples from the rotation (a) and illumination (b) databases

use several subsets of ALOI to train and test the network. All images are used directly without preprocessing or resizing.

ALOI-V200 is build up with the 200 first objects of the *ALOI-Viewpoint* database. For each object, 72 photographies are recorded by rotating the object about its vertical axis at 5° resolution. The gray-level images are of size 192×144 pixels. Figure 3a presents a few images that are used. The training dataset is formed of 3600 images, taking 1 view out of 4 of each object, i.e., at a 20° rotation step. The validation dataset comprises all the other images. The experiments are repeated by reinitializing the IRF-NN.

ALOI-I1000 takes the whole *ALOI-Illumination Direction* base, that is 1000 objects recorded with 24 different illumination conditions: 15 different illumination angles with single light source and 8 combinations of light sources (Fig. 3b). Half of the images for each object compose the training dataset.

The results (see Table 1) validate the direct use of the image by the network and show the capabilities of these random Receptive Fields. The object classification on the validation test is close to a 100% although the dataset comprises many difficulties: large number of objects, important appearance variations, a few very similar objects (e.g., two walnuts) that the network can still distinguish.

ALOI is a very challenging dataset and so far few results have been published [8,9] involving object rotation recognition. They show recognition rates of about 83% in [9] for the change in illumination direction, with training durations of several hours. A more thorough discussion of these comparisons is in preparation to take different methodologies and specific objectives into account.

The number of neurons of the IRF network, much larger than in the geometrical experiments, remains reasonable. With 1000 neurons for 192×144 pixels images, internal representation \mathbf{H} (Eq. 1) represents a strong image compression.

The learning time of 88 seconds represents a very good performance, compared to the volume of data: 3600 images, 200 objects, and 18 views at 20° increments. The learning time can be decomposed into: determination of representation \mathbf{H} (about 75 seconds) and pseudoinverse computation (13 seconds).

The computational cost of this last step is independent of the image size; it depends only on the number of neurons and the size of the training dataset.

5 Conclusions and Perspectives

This paper presents a tool adapted to learning with images. The IRF-NN introduces the concept of random Receptive Fields in ANN. It defines the weight values as a function of a few random variables at the time of initialization of the input layer of a feed-forward type network.

The network uses images directly for object recognition applications without prior feature extraction and presents excellent performances for object recognition as well as for learning functions of the image. The experiments show the good generalization capabilities for object shifting, resizing, or even for some re-shaping and illumination variations. Whenever the classification demands it, the generated representation is sufficiently precise to differentiate shape that differ by only a single pixel, even though the object is moving in the image.

The results described here are confirmed by numerous experiments. Several improvements are under way, for example the determination of optimal parameters, or the integration of iterative learning on subsets of examples.

References

1. Simard, P.Y., Steinkraus, D., Platt, J.C.: Best Practice for Convolutional Neural Networks Applied to Visual Document Analysis. In: International Conference on Document Analysis and Recognition. IEEE Computer Society, Los Alamitos (2003)
2. Yang, F., Paindavoine, M.: Implementation of an RBF Neural Network on Embedded Systems: Real-Time Face Tracking and Identity Verification. *IEEE Transactions on Neural Networks* 14(5), 1162–1175 (2003)
3. Egmont-Peterson, M., de Ridder, D., Handels, H.: Image Processing with Neural Networks - A Review. *Pattern Recognition* 35(10), 2279–2301 (2002)
4. Jaeger, H.: The Echo State Approach to Analysing and Training Recurrent Neural. Technical Report (GMD148), German National Research Center for Information Technology (2001)
5. Huang, G.B.: Extreme Learning Machine: Theory and Applications. *Neurocomputing* 70(1-3), 489–501 (2006)
6. Haykin, S.: Neural Networks: A Comprehensive Foundation. Macmillan College Publishing Compagny Inc., New York (1994)
7. Geusebroek, J.M., Burghouts, G.J., Smeulders, A.W.M.: The Amsterdam Library of Object Images. *International Journal of Computer Vision* 16(1), 103–112 (2005)
8. Song, X., Muselet, D., Tréneau, A.: Local color descriptor for object recognition across illumination changes. In: Blanc-Talon, J., Philips, W., Popescu, D., Scheunders, P. (eds.) *ACIVS 2009. LNCS*, vol. 5807, pp. 598–605. Springer, Heidelberg (2009)
9. Elazary, L., Itti, L.: A Bayesian model for efficient visual search and recognition. *Visual Search and Selective Attention* 50(14), 1338–1352 (2010)

A Comparison of the Electric Potential through the Membranes of Ganglion Neurons and Neuroblastoma Cells

Thiago M. Pinto, Roseli S. Wedemann, and Célia Cortez

Instituto de Matemática e Estatística, Universidade do Estado do Rio de Janeiro,
Rua São Francisco Xavier 524, 20550 – 900, Rio de Janeiro, Brazil
`{thiagomatos,roseli,ccortezs}@ime.uerj.br`

Abstract. We have modeled the electric potential profile, across the membranes of the ganglion neuron and neuroblastoma cells. We considered the resting and action potential states, and analyzed the influence of fixed charges of the membrane on the electric potential of the surface of the membranes of these cells, based on experimental values of membrane properties. The ganglion neuron portrays a healthy neuron, and the neuroblastoma cell, which is tumorous, represents a pathologic neuron. We numerically solved the non-linear Poisson-Boltzmann equation, by considering the densities of charges dissolved in an electrolytic solution and fixed on both glycocalyx and cytoplasmic proteins. We found important differences among the potential profiles of the two cells.

Keywords: Membrane model, electric potential, electrophoresis, neuroblastoma.

1 Introduction

We study the influence of surface electric charges on the stability of the neural cell membrane, by modeling the electric potential profile. This profile describes the behavior of the potential along the axis perpendicular to the cell membrane, from the outer bulk region to the inner one [1,2,3]. It has been shown that the electrophoretic behavior of neuroblastoma cells provides information about its surface charge, in different phases of the cellular cycle [4,5]. This evidence shows that membrane anionic groups are mainly responsible for the surface charges of murine neuroblastoma cells. These groups are distributed in a 0.2 e/nm^3 density, in a layer that covers the cell's outer surface, with a 10 nm thickness.

We compare the effects of fixed charges in the glycocalyx and those associated with cytoplasmic proteins, on the electric potential on the surfaces of the membranes of the lipid bilayer of the ganglion neuron and the neuroblastoma cells, considering both natural states of neuronal cells, *i.e.* the resting and the action potential (AP) states. The AP state refers to the state in which the neuron has been stimulated enough and is firing. We also calculated the potential profile across the membrane, including data from electrophoretic experiments

in our model. We have thus applied a model developed in [1,2,3] to the ganglion neuron, which is a healthy neuron, and to the neuroblastoma cell, which is a tumorous pathologic neuron. Although there are models for studying morphological and mechanical properties of cell membranes, we know of no other models for predicting the electric potential along an axis perpendicular to the membrane.

2 The Membrane Model

In the neuron membrane model we have adopted [2] shown in Fig. 1, four different regions are represented: extracellular, glycocalyx, bilayer and cytoplasm. The bilayer thickness is h and the width of the glycocalyx is h_g . Surface potentials are represented as $\phi_{-\infty e}$ for the potential in $-\infty$ in the electrolytic extracellular phase, $\phi_{S_{eg}}$ for the potential on the surface between the extracellular and glycocalyx regions, $\phi_{S_{gb}}$ is the potential on the surface between the glycocalyx and the bilayer, $\phi_{S_{bc}}$ is the potential on the surface between the bilayer and cytoplasm, and $\phi_{c+\infty}$ is the potential in $+\infty$, *i.e.* in the bulk cytoplasmic region.

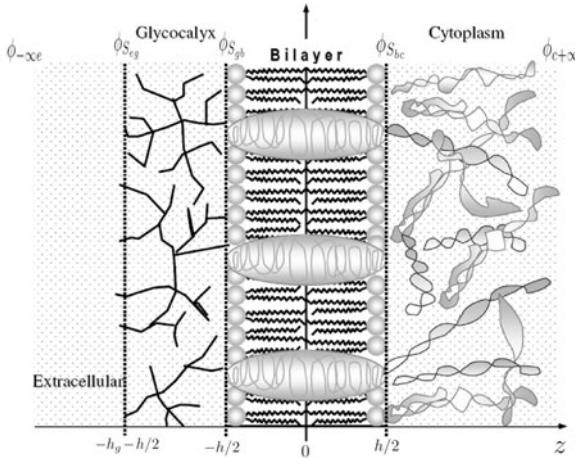


Fig. 1. Model for a neuron membrane. Different regions are represented, with the corresponding symbols for the potentials in the regions and on surfaces dividing regions. Symbols are explained in the text.

2.1 The Electric Potential in the Membrane Regions

In order to determine the potential profile across the membrane, we first considered as in [2] the Poisson equation, including the fixed charges on the surfaces

$$\nabla^2 \phi_i(x, y, z) = \frac{-4\pi(\rho_i + \rho_{fi})}{\epsilon_i} \text{ for } i = ext, g, b, c , \quad (1)$$

where $\phi_i(x, y, z)$ is the electric potential in any region i ; $i = ext$ for the outer electrolytic region; $i = g$ for the glycocalyx; $i = b$ for the bilayer; $i = c$ for the cytoplasm. The volumetric charge density due to the electrolytes in solution of area i is ρ_i , and ρ_{fi} is the density of charges fixed onto proteins of area i .

In a situation where the Boltzmann condition for equilibrium of the electrochemical potential for ionic solutes in a diluted solution holds, it is possible to use the Boltzmann distribution with the above Poisson equation Eq. (1) [2]. Considering a homogeneous charge distribution in directions x and y , Boltzmann equilibrium and Eq. (1), we obtain [2,6]

$$\frac{\partial}{\partial z} \phi_i(z) = \sqrt{\frac{Q_{m_i} \cosh \beta (\phi_i - \phi_{S_i})}{\beta} + \frac{Q_{d_i} \cosh^2 \beta (\phi_i - \phi_{S_i})}{\beta} + g_i \phi_i + W_i} , \quad (2)$$

where

$$Q_{m_i} = \left[\frac{8\pi e \eta_{1,S_i}}{\epsilon_i} \right] , \quad Q_{d_i} = \left[\frac{16\pi e \eta_{2,S_i}}{\epsilon_i} \right] , \quad \beta = \frac{e}{KT} , \quad \text{and} \quad g_i = -\frac{4\pi \rho_{fi}}{\epsilon_i} , \quad (3)$$

and ϕ_i is the electric potential at any point within region i ; ϕ_{S_i} is the limiting electric potential at surface S_i ; e is the electron charge; K is Boltzmann's constant; T is the temperature; ϵ_i is the dielectric constant in region i ; W_i is an integration constant for region i ; the monovalent ionic concentration is η_{1,S_i} and the divalent ionic concentration is η_{2,S_i} , both on surface S_i . Eq. (2) is the Poisson-Boltzmann equation for the electric potential in region i [2,6].

2.2 Surface Potentials

Considering the discontinuity of the displacement of the electric field vector on the surface S_{gb} and considering the solution of the Poisson-Boltzmann equation in the cytoplasmic and electrolytic regions, we have obtained [2,6]

$$\phi_{S_{bc}} = \phi_{S_{gb}} - \frac{4\pi Q_{S_{gb}} h}{\epsilon_b} + \frac{\epsilon_g h}{\epsilon_b} \sqrt{\alpha} , \quad (4)$$

where,

$$\alpha = 2 \frac{Q_{m_g} \sinh^2 \left(\frac{\beta}{2} (\phi_g - \phi_{S_{eg}}) \right)}{\beta} + \frac{Q_{d_g} \sinh^2 \left(\beta (\phi_g - \phi_{S_{eg}}) \right)}{\beta} + g_g (\phi_g - \phi_{S_{eg}}) + \left(\frac{4\pi Q_{S_{eg}} - \epsilon_{ext} \nabla \phi_{ext}|_{S_{eg}}}{\epsilon_g} \right)^2 , \quad (5)$$

and $Q_{S_{gb}}$ and $Q_{S_{eg}}$ stand for the charge density on the surfaces between the regions, glycocalyx and the bilayer, and electrolytic and glycocalyx, respectively. Applying the same procedure for the S_{bc} surface,

$$\phi_{S_{gb}} = \phi_{S_{bc}} - \frac{4\pi Q_{S_{bc}} h}{\epsilon_b} + \frac{\epsilon_c h}{\epsilon_b} \times \frac{\sqrt{2} \frac{Q_{mc} \sinh^2 \frac{\beta}{2} (\phi_c - \phi_{c+\infty})}{\beta} + \frac{Q_{dc} \sinh^2 \beta (\phi_c - \phi_{c+\infty})}{\beta} + g_c (\phi_c - \phi_{c+\infty})}}{(6)}$$

We have used data obtained from experimental results [5,7] for values of parameters, in order to solve the first order ordinary differential equations, obtained from the Poisson-Boltzmann Eq. (2), for the different regions of the membrane. Some experimental values were obtained from electrophoresis experiments. Since each kind of cell presents a specific electrophoretic mobility, the values of some parameters are different for the ganglion neuron and the neuroblastoma, in our calculations. Due to space limitations, we refer the reader to [6] for a table with all experimental values of the parameters used to solve the equations. We have thus examined the influence of parameters representing electric properties of the membrane, over resting and AP states, analyzing the differences between the healthy ganglion neuron and a neuroblastoma cell.

We implemented an algorithm for finding roots of functions, to calculate $\phi_{S_{gb}}$ and $\phi_{S_{bc}}$ from Eqs. (4) and (6), in C. The potential $\phi_{S_{eg}}$ was calculated from data obtained from electrophoretic experiments. We numerically calculated values of the potential profiles with Eq. (2), using the Runge-Kutta method, also in C.

3 Results

We examined the bilayer surface potentials as a function of ρ_{fc}/ρ_{fg} (ρ_{fc} and ρ_{fg} are the fixed charge densities in the cytoplasm and in the glycocalyx, respectively). Figs. 2 and 3 present the behavior of $\phi_{S_{gb}}$ and $\phi_{S_{bc}}$ with the variation of ρ_{fc}/ρ_{fg} , considering the same $Q_{S_{bc}}$ value, for both cells. During the resting potential state, results in Fig. 2 show that, while $\phi_{S_{gb}}$ remains constant while increasing ρ_{fc}/ρ_{fg} , by making the fixed charges in the cytoplasm more negative (decreasing negative values of ρ_{fc}), $\phi_{S_{bc}}$ decreases expressively, for both $Q_{S_{gb}} = 0$ (Fig. 2(a)) and $Q_{S_{gb}} \neq 0$ (Fig. 2(b)). However, comparing Figs. 2(a) and 2(b), we see that the increase of negativity of $Q_{S_{gb}}$ visibly decreases the value of $\phi_{S_{gb}}$, for the ganglion neuron. The same behavior is observed, when comparing Figs. 3(a) and 3(b). During the AP state (Fig. 3), the potential $\phi_{S_{bc}}$ of both cells shows a quick drop, when $\rho_{fc}/\rho_{fg} < 20$, becoming almost constant for $\rho_{fc}/\rho_{fg} > 20$. However, $\phi_{S_{gb}}$ remains constant for all values of ρ_{fc}/ρ_{fg} .

In Fig. 4, we compared the electric potential profile across the membranes of both cells. We verify that the gradual decrease of the potential along the z axis, up to the surface of the glycocalyx ($z < -h_g - h/2$) is higher for the ganglion neuron than for the neuroblastoma cell, and both curve shapes are similar. Through the glycocalyx ($-h_g - h/2 < z < -h/2$) we can see that the fall continues for the ganglion neuron, but is negligible for the cancerous cell. During the resting potential state, the value of the intracellular potential increases exponentially, from the bilayer surface to the bulk cytoplasmic region.

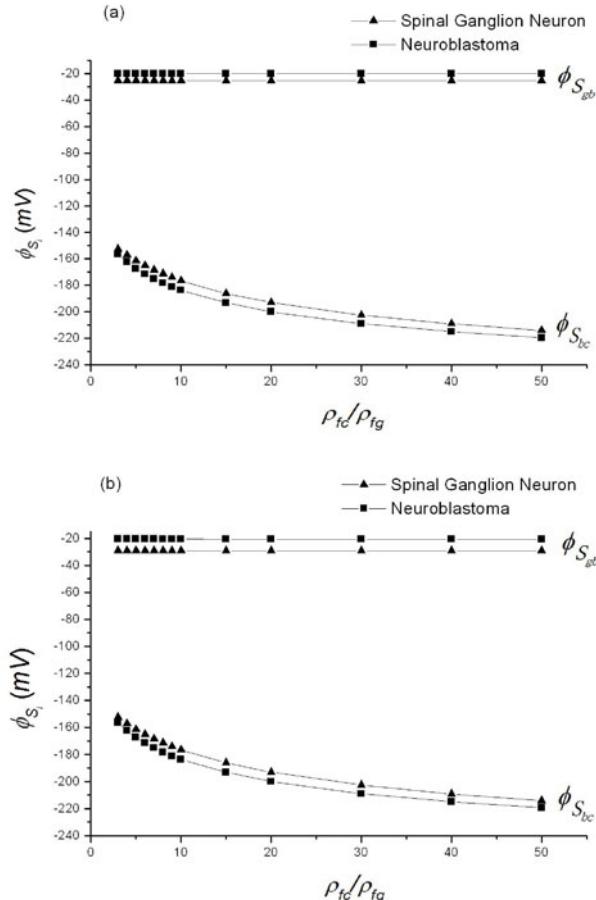


Fig. 2. $\phi_{S_{bc}}$ and $\phi_{S_{gb}}$ as a function of ρ_{fc}/ρ_{fg} , during resting state, on the ganglion neuron and neuroblastoma membranes. In both figures, $Q_{S_{bc}} = -5.4 \times 10^{-2} \text{ C/m}^2$. $Q_{S_{eg}} = -0.012 \text{ e/nm}^2$ and $\phi_R = -69 \text{ mV}$ for the ganglion neuron. $Q_{S_{eg}} = -0.02 \text{ e/nm}^2$ and $\phi_R = -64 \text{ mV}$ for the neuroblastoma. (a) $Q_{S_{gb}} = 0$. (b) $Q_{S_{gb}} = -3.20 \times 10^{-3} \text{ C/m}^2$, for the neuroblastoma; and $Q_{S_{gb}} = -1.92 \times 10^{-3} \text{ C/m}^2$, for the ganglion neuron. The resting transmembrane potential is ϕ_R .

4 Conclusions

Simulation experiments maintaining constant values of $Q_{S_{bc}}$ and $Q_{S_{eg}}$, resulted in no detectable changes in $\phi_{S_{gb}}$, but $\phi_{S_{bc}}$ of both neurons decreases gradually with the increase of ρ_{fc}/ρ_{fg} , by making the fixed charges in the cytoplasm more negative (decreasing negative values of ρ_{fc}), during the resting potential state (Fig. 2) and also during the AP state (Fig. 3). For the AP state, the drop in the values of $\phi_{S_{bc}}$ occurred mainly for the small values of ρ_{fc}/ρ_{fg} , tending to

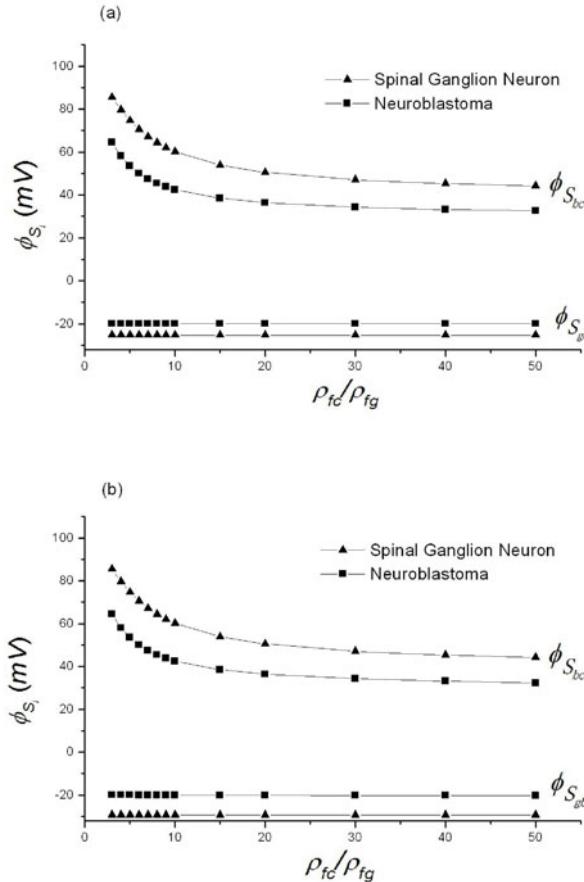


Fig. 3. $\phi_{S_{bc}}$ and $\phi_{S_{gb}}$ as a function of ρ_{fc}/ρ_{fg} , during action state, on ganglion neuron and neuroblastoma membranes. In all curves, $Q_{S_{bc}} = -5.4 \times 10^{-2} \text{ C/m}^2$. $Q_{S_{eg}} = -0.012 \text{ e/nm}^2$ and $\phi_R = -69 \text{ mV}$ for the ganglion neuron. $Q_{S_{gb}} = -0.02 \text{ e/nm}^2$ and $\phi_R = -64 \text{ mV}$ for the neuroblastoma. (a) $Q_{S_{gb}} = 0$. (b) $Q_{S_{gb}} = -3.20 \times 10^{-3} \text{ C/m}^2$, for the neuroblastoma; and $Q_{S_{gb}} = -1.92 \times 10^{-3} \text{ C/m}^2$, for the ganglion neuron.

become constant for higher values. Comparing Figs. 2(a) and 2(b), we verify, for $Q_{S_{gb}} \neq 0$, that $\phi_{S_{gb}}$ for the ganglion neuron is more negative than when $Q_{S_{gb}} = 0$, which was the only detectable alteration with this change in charge value. The results obtained for the ganglion neuron match those for the squid axon membrane found by Cortez et al. [2]. Using a model with similar equations as we used in this study, the authors observed variations of the surface potentials with a change in surface charge $Q_{S_{gb}}$ compatible with those observed here.

During the resting potential state, the net value of the protein charge in the cytoplasm is predominantly negative [2]. However, in our simulation experiments,

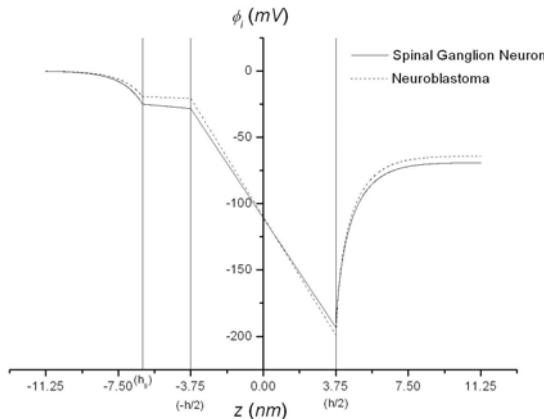


Fig. 4. Electric potential profile, during the resting state, across the ganglion neuron membrane, for $\phi_{S_{bc}} = -193.39\text{mV}$, $\phi_{S_{gb}} = -28.42\text{mV}$, $\phi_{S_{eg}} = -25.10\text{mV}$, $Q_{S_{gb}} = -1.92 \times 10^{-3}\text{C/m}^2$ and $\phi_R = -69\text{mV}$. Same profile for the neuroblastoma membrane, for $\phi_{S_{bc}} = -199.08\text{mV}$, $\phi_{S_{gb}} = -20.65\text{mV}$, $\phi_{S_{eg}} = -19.52\text{mV}$, $Q_{S_{gb}} = -3.20 \times 10^{-3}\text{C/m}^2$, $\phi_R = -64\text{mV}$. In both curves, $Q_{S_{bc}} = 30 \times Q_{S_{gb}}$ and $\rho_{fc} = 20\rho_{fg}$.

the contribution of these charges to the inner potential profile was smaller than the effect of the fixed charges in the inner surface of the bilayer, due to the curvature of the potential in this region, whereas the calculated value of $\phi_{S_{bc}}$ was smaller than the bulk region $\phi_{S_{bc}}$. It is known that the neuroblastoma cells, like all other cancerous cells, multiply themselves quickly. Alterations of the dynamics of cellular multiplication implicate changes in the synthesis, structure and degradation of the membrane components [8], which result in deformations in structure and composition of the plasma membrane surface [9]. These deformations implicate changes in the electric charge of the membrane.

Our results indicate that the alteration of the fixed electric charges of the membrane influences the behavior of its surface electric potential. Although we used the same model and equations for both types of cells, contrary to what was observed for the ganglion neuron, using the parameters of the neuroblastoma cells led to solutions of the equation for the electric potential, where a change of values of $Q_{S_{gb}}$ and ρ_{fc} charges practically didn't affect the surface potentials. It corroborates results of experimental observations, that the resting potential and the generation of action potentials in human neuroblastoma cells depend on the degree of the morphologic differentiation of the cell. Some of these cells are relatively non-excitable [10,11]. These properties should affect the transmission of signals through networks of these neurons and the functions of storage and communication of information.

The different values of the potential in the glycocalyx for the neuroblastoma and the spinal ganglion neuron must represent important alterations in the transport function of the membrane, due to the outer electric field, which is responsible

for the orientation of the charged particles which are closer to the membrane. Also, the potential at the outer surface of the membrane is determinant for many cell processes, such as the beginning of the process of triggering of the action potential, which depends on the opening of specific Na^+ channels.

References

1. Cortez, C., Bisch, P.: The effect of ionic strength and outer surface charge on the membrane electric potential profile: a simple model for the erythrocyte membrane. *Bioelectrochemistry and Bioenergetics* 32, 305–315 (1993)
2. Cortez, C., Cruz, F., Silva, D., Costa, L.: Influence of fixed electric charges on potential profile across the squid axon membrane. *Physica B* 403, 644–652 (2008)
3. Cruz, F., Vilhena, F., Cortez, C.: Solution of non-linear Poisson-Boltzmann equation for erythrocyte membrane. *Brazilian Journal of Physics* 3, 403–409 (2000)
4. Belan, P., Dolgaya, E., Mironov, S., Tepikin, A.: Relation between the surface potential of mouse neuroblastoma clone c1300 cells and the phase of the cell cycle. *Neurophysiology* 19(1), 130–133 (1987)
5. Dolgaya, E., Mironov, S., Pogorelaya, N.: Changes in surface charge of mouse neuroblastoma cells during growth and morphological differentiation of the cell population. *Neurophysiology* 17(2), 168–174 (1985)
6. Pinto, T.M.: Modelagem do Potencial Elétrico através da Membrana do Neurônio Ganglionar e Células de Neuroblastoma: Efeitos das Cargas Superficiais. Masters Dissertation, Universidade do Estado do Rio de Janeiro, Rio de Janeiro (2010)
7. Mironov, S., Dolgaya, E.: Surface charge of mammalian neurones as revealed by microelectrophoresis. *J. Membrane Biol.* 86, 197–202 (1985)
8. Dehlinger, P., Schimke, R.: Size distribution of membrane proteins of rat liver and their relative rates of degradation. *J. Biol. Chem.* 246(8), 2574–2583 (1971)
9. Schubert, D., Humphreys, S., Jacob, F.: Induced differentiation of a neuroblastoma. *Dev. Biol.* 25(4), 514–546 (1971)
10. Gérard, V., Rouzaire-Dubois, B., Dilda, P.: Alterations of ionic membrane permeabilities in multidrug-resistant neuroblastoma x glioma hybrid cells. *J. Exp. Biol.* 201, 21–31 (1998)
11. Kuramoto, T., Perez-Polo, J., Haber, B.: Membrane properties of a human neuroblastoma II: Effects of differentiation. *J. Neurosci. Res.* 6(4), 441–449 (1981)

SNPboost: Interaction Analysis and Risk Prediction on GWA Data

Ingrid Brænne^{1,2,3,*}, Jeanette Erdmann^{2,3},
and Amir Madany Mamlouk^{1,3}

¹ Institute for Neuro- and Bioinformatics

² Medizinische Klinik II

³ Graduate School for Computing in Medicine and Life Sciences
University of Lübeck,

Ratzeburger Allee 160, 23562 Lübeck, Germany
{braenne, madany}@inb.uni-luebeck.de
<http://www.inb.uni-luebeck.de>

Abstract. Genome-wide association (GWA) studies, which typically aim to identify single nucleotide polymorphisms (SNPs) associated with a disease, yield large amounts of high-dimensional data. GWA studies have been successful in identifying single SNPs associated with complex diseases. However, so far, most of the identified associations do only have a limited impact on risk prediction. Recent studies applying SVMs have been successful in improving the risk prediction for Type I and II diabetes, however, a drawback is the poor interpretability of the classifier. Training the SVM only on a subset of SNPs would imply a preselection, typically by the p-values. Especially for complex diseases, this might not be the optimal selection strategy. In this work, we propose an extension of Adaboost for GWA data, the so-called SNPboost. In order to improve classification, SNPboost successively selects a subset of SNPs. On real GWA data (German MI family study II), SNPboost outperformed linear SVM and further improved the performance of a non-linear SVM when used as a preselector. Finally, we motivate that the selected SNPs can be put into a biological context.

Keywords: Genome-wide association, risk prediction, SNP-SNP interaction.

1 Introduction

Genome-wide association (GWA) studies, which typically aim to identify single nucleotide polymorphisms (SNPs) associated with a disease, yield large amounts of high-dimensional data. During the last decade there has become a growing body of studies- mainly focusing on single-SNP statistics (p-values) - that have identified genetic loci (SNPs) associated to common complex diseases such as diabetes [14], myocardial infarction [11,2], and Crohn's disease [10]. However,

* Corresponding author.

so far these findings have only limited impact on risk assessment and clinical treatment [8,7].

Complex diseases are caused by a variety of genetic factors. These factors, e.g. SNPs, may interact positively or negatively to increase or reduce the effect of the individual factors; indeed, an appreciable disease effect may only come about through such an interaction. Studies that focus on single locus effects alone are thus not likely to reveal the more complex genetic mechanisms underlying multifactorial traits [14,15,9]. Since, so far, most of the identified genetic variants have only a limited effect on disease risk, it suggests itself to analyze several SNPs simultaneously.

A prominent algorithm for classification accounting multiple factors is the so-called Support Vector Machine (SVM) [5,6]. Recent studies using SVMs have been successful in improving the risk prediction for Type I and II diabetes [14,16,1]. However, the resulting classifier is using all SNPs, making it hard to interpret the resulting classifier in a biological context, especially for hundreds of thousands of SNPs. In order to allow a better interpretation of the results a feature selection approach is required. An intuitive way is to choose SNPs that are individually associated with the disease. But, clearly, SNPs might be missed that only show an effect in interaction with others.

Boosting algorithms like Adaboost might be an good alternative [4,3]. The main idea of Boosting is to combine several weak classifiers to one strong classifier. Weak classifiers, i.e. SNPs, are added one after another in order to gain a set of classifiers that together boost the classification. With this selection strategy, one can control the number of SNPs without having to preselect a subset of SNPs and possible SNP-SNP interactions might be found between the selected SNPs.

In this work, we propose a variation of the Adaboost algorithm for an application to GWA data. The algorithm is evaluated on the german MI family study II GWA data set [2].

2 Methods and Data

2.1 Support Vector Machine (SVM)

Support vector machines (SVM) aim to determine the hyperplane that separates two given classes with maximum margin [13]. It has been applied to a broad range of classification problems and is one of the standard benchmark methods. In this work, we train the SVM on an increasing number of SNPs, where the SNP subset is selected based on the single SNP p-values. In order to measure the classification performance for each subset, we train a SVM on the genotype data of the selected SNPs of the training set. For the linear and gaussian SVM the softness of the margin and the kernel width for the gaussian SVM is adjusted by a 10-fold cross-validation on the training set.

2.2 Adaboost

As SNPboost is derived from one of the most popular boosting algorithms, Adaboost [4,3], we will provide in the following a brief sketch of the algorithm.

In Adaboost, the classifiers are combined such that classification errors of the first single classifiers are compensated as good as possible by the second classifier etc.. Thus, the selection of the next classifier is biased in favor of the previously misclassified datapoints. This is done by applying weights to the datapoints, where the weights are updated after each step. More specific, the weights of the misclassified datapoints are increased and correspondingly decreased for the correctly classified datapoints. Consequently, step by step, the classifier gets stronger by subsequently adding weak classifiers that optimize the performance of the combined Adaboost classifier.

Let $X = (\mathbf{x}_1, \dots, \mathbf{x}_N)$, $\mathbf{x}_i \in \mathbb{R}^D$ be a set of given data samples and $Y = (y_1, \dots, y_N)$, $y_i \in \{1, -1\}$ the corresponding class information. Furthermore, we define a set of weights $\mathbf{w} = (\mathbf{w}_1, \dots, \mathbf{w}_N)$ with the two constraints $\mathbf{w} \geq 0$ and $\sum_{i=1}^N \mathbf{w}_i = 1$, and T be the number of weak classifiers combined to the strong classifier. Finally, let $H = (h_1, \dots, h_M)$ describe the set of weak classifiers to choose from and $L^{h_j} = (l_1^{h_j}, \dots, l_N^{h_j})$ be the classification of X by each classifier h_j . Then, Adaboost combines a desired number of T classifiers h_j to a strong new classifier according to Algorithm 1.

Algorithm 1. Basic Adaboost Algorithm

Input: Training data X , labels Y , and a set of naive classifier H

Output: List of T chosen classifier $\eta = (\eta_1, \dots, \eta_T)$ and their significance

$$\alpha = (\alpha_1, \dots, \alpha_T),$$

foreach epoch $t = 1, \dots, T$ **do**

 1. Find the classifier η_t that minimizes the training error ϵ

$$\eta_t = \arg \min_{h_j \in H} \epsilon_j \quad (1)$$

 with

$$\epsilon_j = \sum_{i=1}^N |l_i^{h_j} - y_i| \quad w_i(t) \quad (2)$$

 2. Choose α_t with

$$\alpha_t = 1/2 \ln\left(\frac{1 - \epsilon_t}{\epsilon_t}\right) \quad (3)$$

 3. Update the weights in \mathbf{w} with a normalization constant Z_t to

$$w_i(t+1) = \frac{w_i(t)}{Z_t} \cdot \begin{cases} e^{-\alpha_t} & \text{if } l_i^{\eta_t} = y_i \\ & \text{otherwise.} \end{cases} \quad (4)$$

end

Calculate final classification with $L^{ada}(x) = \text{sign}(\sum_{t=1}^T \alpha_t \cdot l_x^{\eta_t})$

2.3 SNPboost: Adaboost for GWA Data

To use Adaboost with GWA data, we have to define the set of weak classifiers H . In this work, we obtain H by deriving a set of 6 classifiers out of each SNP: Each of the given SNPs consists of three discrete states (**AA**, **AB**, **BB**), which denote the possible homozygot and heterozygot genotypes an individuum can express for this location. We use these three states to derive six naive classifiers that indicate a subject expressing one of the three states or its negation (a variant might just as well be protective in its effect). Hence, let p be the predicted genotype for all D SNPs with $p = p_1, \dots, p_M$ and $M = D \cdot 6$. Given the genotype $g_c(i)$ of the individual i we get the training error ϵ_c by:

$$\epsilon_c = \sum_{i=1}^N [1 - [p_c(i) \neq g_c(i)]] \quad w_t(i). \quad (5)$$

2.4 Data

We applied the SNPboost algorithm to the German MI family study II [2] in order to evaluate the performance of the algorithm. SNPs with a missing rate of ≤ 0.01 , minor allele frequency of ≤ 0.01 , and p-value $\leq 10^{-4}$ for deviation from Hardy-Weinberg equilibrium were excluded. After quality filtering we pruned the data for Linkage Disequilibrium (LD). The total number of SNPs left for analysis is $D = 127370$ SNPs for a total of 2032 individuals with 1222 controls and 810 cases. We filled up the remaining missings with imputed values.

For training and testing, the data was randomly divided in a training and test set with equal sample size for the cases and the controls with 405 individuals each.

3 Results and Discussion

We evaluated the performance of the SNPboost algorithm and the SVM by means of the receiver operator characteristic (ROC) obtained on the test set. We tested SNPboost against linear SVMs with p-values as a preselection method, then we evaluated the gain of using SNPboost as a feature selection method for a non-linear SVM, and finally we examined the selected SNPs for biological relevance.

3.1 Linear Classification

As shown in Figure (1), both algorithms yield a peak performance for small number of SNPs. Subsequently, the performance decrease with additional SNPs. This decrease is likely caused by overfitting of the algorithms: The more features used, the more likely the trained classifiers are adapted to the training set and hence no longer describe the test set appropriately. The maximum AUC (area under the (ROC) curve) for the SNPboost algorithm is 0.76 with a total of 5

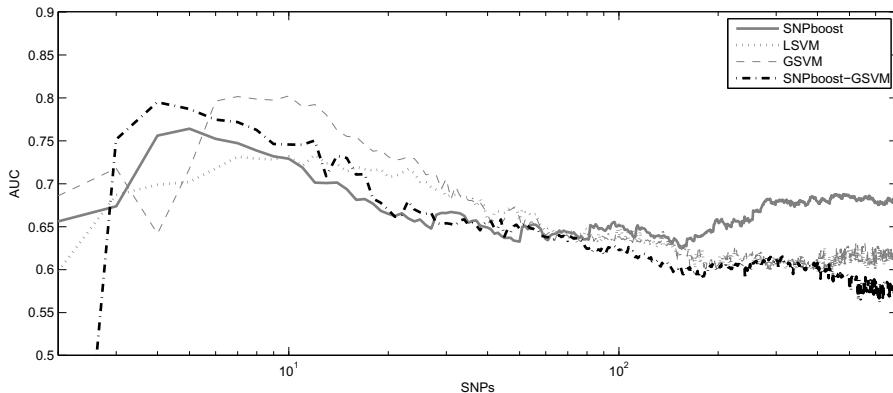


Fig. 1. Classification performance of linear and non-linear SVM with SNPboost selected SNPs vs p-value selected SNPs. A logarithmic (base 10) scale is used for the X-axis.

SNPs. For the LSVM the performance yield 0.73 for a total number of 11 SNPs. Whereas the performance of the LSVM remain weak for further increasing numbers of SNPs, the performance of the SNPboost algorithm recover. Hence, the SNP selection is no longer specific for the training set. The overall performance of the SNPboost algorithm clearly outranges the LSVM. Whereas the mean performance of the LSVM is 0.62 the mean performance of the SNPboost algorithm yield 0.67.

With SNPboost being a linear classifier we first evaluated the performance of the SNPboost algorithm and the linear kernel SVM (LSVM). The selection done by the SNPboost algorithm seems to be the better choice compared to the selection by the p-values. This might be due to the fact that the SNPboost algorithm selects SNPs one at a time in order to bit by bit increases the classification and might thus better fit together.

3.2 Non-linear Classification

The previous results indicated that the SNPs selected by the SNPboost algorithm might be more appropriate for combined classification than the SNPs selected due to the single significance values. Hence, combining the SNP selection strategy of the SNPboost algorithm with a more powerful classifier might improve the classification performance.

In order to test whether the classification performance can be further increased by applying a non-linear classifier, we trained a gaussian kernel SVM (GSVM) on the SNPs selected by the SNPboost algorithm and compared the results with the performance on the p-value selected SNPs.

The dash-dotted line in Figure 1 shows the performance of the gaussian classifiers with SNPboost as a preselector, while the dashed line depicts the performance of the gaussian kernal SVM with p-values as a feature selector.

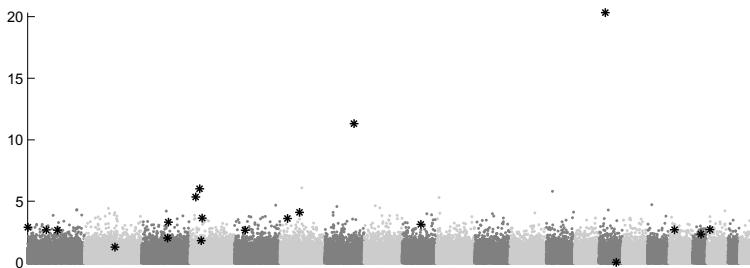


Fig. 2. Single p-values for all SNPs and the 20 first selected SNPs by the SNPboost algorithm. The SNPboost selected SNPs are shown as black dots.

Regardless of the selection strategies, the performance primarily increases by applying GSVM. For the p-value selection strategy, the GSVM yield better performance than the LSVM for small number of SNPs but the performances align for larger numbers of SNPs. Analogous to the p-values selection, for small number of SNPs, the GSVM on SNPboost selected SNPs improve the performance compared to the SNPboost algorithm. As for the performance of the SNPboost algorithm, also the performance of the GSVM decreases with a larger number of SNPs. However, in contrast to the SNPboost algorithm, for large number of SNPs, the performance does not recover but even drops below the performance of the LSVM and GSVM on p-value selected SNPs. This is probably due to the fact that the SNPboost algorithm selects a set of SNPs that fit well together. The GSVM further optimizes the classification with these matching SNPs and thus the classifier is too specific to the training data set.

The maximum performance of the GSVM is 0.80 for both the selection strategies. The peak performance is gained with 4 and 10 SNPs for the SNPboost selection and p-values selection respectively.

3.3 Interaction Analysis

The main advantage of SNPboost is the selection strategy: If a weak classifier positively interacts with a previous selected one, this weak classifier might be chosen since an interaction might improve the classification. Thus, possible interactions might be found within the selected SNPs. Hence, it might be of value to further analyze these SNPs.

Figure (2) shows the p-values of the 20 first selected SNPs by the SNPboost algorithm. Of the top ten p-value SNPs, three are selected by the SNPboost algorithm. While the first one corresponds to the strongest single classifier, which in this case is the SNP with the highest p-value, all upcoming SNPs are selected due to their interaction with the first classifiers, independent of their p-values.

First we assigned the p-value selected SNPs to their corresponding genes. 14 SNPs were found within genes. Two of the 14 genes can be linked through

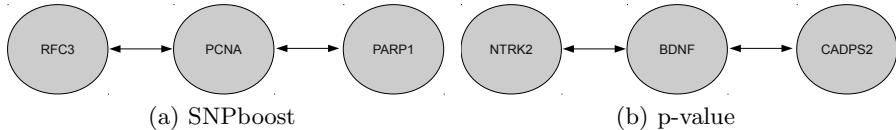


Fig. 3. Gene interaction, of the 20 first selected SNPs two genes can be linked through an additional gene for both selection strategies

an additional gene. As shown in Figure (3a) RFC3 interacts with PARP1 via PCNA [12]. Next we assigned the 20 SNPs, selected by the SNPboost algorithm, to their corresponding genes. 9 out of 20 lie within genes and two of these genes can be linked through an additional gene. As shown in Figure (3b) CADPS2 interacts with NTRK2 via BDNF [12]. Hence, through SNPboost a new possible interaction might have been identified. Whether this interaction increase the risk of a disease must however be further evaluated.

4 Conclusion

In this work, we propose a boosting algorithm for classification as well as for the identification of potential SNP-SNP interactions on GWA data. The SNPboost algorithm is a modified version of the well-known adaboost algorithm. Using each possible SNP genotype as a weak classifier, we build a strong combined classifier. We evaluated SNPboost on the German MI family study II data.

Initially, the classification performance of the SNPboost algorithm clearly outperforms the linear SVM (LSVM) hence leading to the assumption that the selection strategy of the SNPboost might be more appropriate in a multivariate context than the standard selection by p-values. Training a gaussian kernel SVM (GSVM) on these SNPs further improves the classification performance, however only for small number of SNPs. Since SNPboost selects the SNPs that combined improve classification, interacting SNPs are likely to be chosen by the algorithm. In this work, we extracted the first 20 selected SNPs and mapped these to the genes. Of the 20 SNPs, 9 were found within genes. Two of these SNPs can be linked through a third gene. However, before one can state any interaction or biological plausibility, these results need to be further evaluated.

SNPboost is a very fast and memory efficient algorithm and can thus be applied even on the largest datasets without any preselection step. We would thus propose this algorithm as an promising tool for feature selection, interaction analysis and classification on GWA data.

Acknowledgements. This work was supported by the Graduate School for Computing in Medicine and Life Sciences funded by Germanys Excellence Initiative [DFG GSC 235/1].

References

1. Ban, H.J., Heo, J.Y., Oh, K.S., Park, K.J.: Identification of type 2 diabetes-associated combination of snps using support vector machine. *BMC Genetics* 11(1), 26 (2010)
2. Erdmann, J., Großhennig, A., Braund, P.S., König, I.R., Hengstenberg, C., Hall, A.S., Linsel-Nitschke, P., et al.: New susceptibility locus for coronary artery disease on chromosome 3q22.3. *Nat. Genet.* 41(3), 280–282 (2009)
3. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. In: European Conference on Computational Learning Theory, pp. 23–37 (1995)
4. Freund, Y., Schapire, R.E.: A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 771–780 (1999)
5. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. *J. Mach. Learn. Res.* 3, 1157–1182 (2003)
6. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. *Machine Learning* 46, 389–422 (2002)
7. Ioannidis, J.P.: Prediction of cardiovascular disease outcomes and established cardiovascular risk factors by Genome-Wide association markers. *Circ. Cardiovasc. Genet.* 2(1), 7–15 (2009)
8. Manolio, T.A., Collins, F.S., Cox, N.J., Goldstein, D.B., Hindorff, L.A., Hunter, D.J., McCarthy, M.I., et al.: Finding the missing heritability of complex diseases. *Nature* 461(7265), 747–753 (2009)
9. Moore, J.H.: The ubiquitous nature of epistasis in determining susceptibility to common human diseases. *Human Heredity* 56(1-3), 73–82 (2003)
10. Raelson, J.V., Little, R.D., Ruether, A., Fournier, H., Paquin, B., Van Eerdewegh, P., Bradley, W.E.C., et al.: Genome-wide association study for crohn's disease in the quebec founder population identifies multiple validated disease loci. *Proceedings of the National Academy of Sciences* 104(37), 14747–14752 (2007)
11. Samani, N.J., Erdmann, J., Hall, A.S., Hengstenberg, C., Mangino, M., Mayer, B., Dixon, R.J., et al.: Genomewide Association Analysis of Coronary Artery Disease. *N. Engl. J. Med.* 357(5), 443–453 (2007)
12. Szklarczyk, D., Franceschini, A., Kuhn, M., Simonovic, M., Roth, A., Minguez, P., Doerks, T., Stark, M., Muller, J., Bork, P., Jensen, L.J., van Mering, C.: The STRING database in 2011: functional interaction networks of proteins, globally integrated and scored. *Nucleic Acids Research* 39(database), D561–D568 (2010)
13. Vapnik, V.N.: Statistical Learning Theory. Wiley, Chichester (1998)
14. Wei, Z., Wang, K., Qu, H.Q.Q., Zhang, H., Bradfield, J., Kim, C., Frackleton, E., et al.: From disease association to risk assessment: an optimistic view from genome-wide association studies on type 1 diabetes. *PLoS Genetics* 5(10), e1000678(2009)
15. Wray, N.R., Goddard, M.E., Visscher, P.M.: Prediction of individual genetic risk of complex disease. *Current Opinion in Genetics and Development* 18(73), 257–263 (2008)
16. Yoon, Y., Song, J., Hong, S.H., Kim, J.Q.: Analysis of multiple single nucleotide polymorphisms of candidate genes related to coronary heart disease susceptibility by using support vector machines. *Clinical Chemistry and Laboratory Medicine: CCLM / FESCC* 41(4), 529–534 (2003) PMID: 12747598

Binary Patterns Identification by Vector Neural Network with Measure of Proximity between Neuron States

Vladimir Kryzhanovskiy

Scientific Research Institute for System Analysis
of Russian Academy of Sciences, Moscow
Vladimir.Krizhanovsky@gmail.com

Abstract. I describe a new vector neural network, in which a priori information about the distribution of noise is easily and naturally embedded. Taking into account the noise distribution allows to essentially increase the system noise immunity. A measure of proximity between neuron states is embedded for the first time. It makes possible to use the prior information. On binary identification problem the one order increase of storage capacity is shown.

Keywords: Binary patterns identification, vector neural networks.

1 Introduction

Building effective systems of content-addressable memory requires the knowledge of conditions in which it will be operating. It is highly important to embed in the system the a priori information on:

- stored data – how many classes, to which class or subclass each reference pattern refers, what is the overlay of the reference patterns etc.;
- how the input vectors are distorted in input channel – autocorrelation function of noise and its distribution.

The content-addressable memory is often built on the basis of the neural network model developed for working in other conditions of use. E.g.: distribution of noise differs from distribution of the estimated noise when developing the used model. The ready model is hard and often impossible to modify. That is why the creation of the neural network models, easily adaptable to conditions of current tasks, is very important.

The present paper describes a model of the vector neural network (VNN), in which a priori information about the distribution of noise, laid on the components of recognized pattern in input channel, is easily and naturally embedded. (Hereinafter it will be implied that the noise is a white noise, i.e. input vector components are distorted independently.) The vector neural networks are designed for recognition of parametric vectors, each component of which adopts one of a number of discrete values. E.g.: $Q=256$ grey scale images may be used as reference vectors. In this case the number of

neurons in the VNN will be equal to the number of pixels and the number of neuron states will be determined by color grade Q . Value of the input vector component (pixel color) determines a state of the vector neuron assigned to such component at an initial instant. In the process of recognition, neurons, being influenced on each other, switch their states pixel after pixel, restoring the input vector.

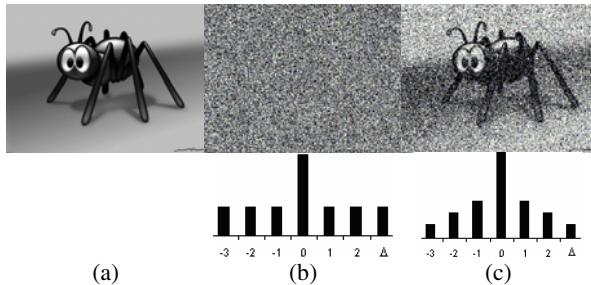


Fig. 1. Noise impact on the input vector: a) reference pattern; b) white noise with “uniform” distribution (with $b=0.98$ probability the pixel’s color is replaced with any other color); c) white noise with “normal” distribution with zero mean and mean-square deviation $\sigma=85$

Classic vector neural networks (e.g. Potts model) efficiently works with white noise [1], which has the “uniform” distribution (fig. 1.b) – with $1-b$ probability the pixel’s color is not distorted, while with b probability – the color is distorted with any other color. Fig. 1.b demonstrates that 98% distortion will result in the inability of a person to recognize patterns properly. On the other hand, the vector neural network can do that, since it is adjusted to the distortions of such type.

In practice, the “uniform” distribution of the noise is rare, it’s distribution is often closer to “normal”: with $1-b$ probability the pixel’s color is not distorted, and with $b \cdot f(\Delta)$ probability the pixel’s brightness is increased by Δ , where $f(\Delta)$ – is a density of normal distribution with zero mean and mean-square deviation σ . Picture 1.c shows the reference patterns, distorted by white noise with “normal” distribution. 98% of pixels here are also distorted, however sufficiently more information has been stored. It is seen that in this case a human eye can recognize the distorted patterns properly. It seems that nature has adjusted our vision system to such type of distortions as the most common one; otherwise the fig. 1.c would appear to us as non-informative as fig. 1.b. It should be noted that recognition of such vectors (fig. 1.c) of the classic VNN will be based solely on information with 2% of non-distorted pixels. Knowledge of the noise distribution enables to derive additional information with 98% of distorted pixels (a bit from each), which leads to the sufficient improvement of recognition reliability.

In order to enter information on the noise nature to the VNN it is suggested that the measure of proximity is entered between neuron states. It is accomplished by modification of synaptic weights of the classic VNN – multiplication of matrices, which describe synaptic coefficients, by matrix of proximity measures. It is suggested that the measure of proximity between two states is assigned in proportion to the probability of switching from one state to the other under the influence of distortions. It means that if a pixel replaces its color k with color r as a result of distortions with probability

of 30%, the measure of proximity between the states k and r will be assigned as equal to 0.3.

The proposed method has been tested with a problem of binary vectors recognition. The importance of such task is stipulated by the fact that well-known scalar models of neural networks which work with binary patterns have a memory of a very small capacity (where the memory capacity is rigidly connected to number of patterns dimensions) [2]. This circumstance does not allow using scalar models for development of practical applications. The paper [3,4] demonstrated how to adapt VNN for recognition of binary patterns. The adaptation consists in mapping of binary patterns (high dimensional black-and-white images) into Q -dimensional patterns (colored images of a smaller dimension). It was shown that this approach enables to reach a memory with exponential capacity per free parameter which is several orders greater than memory capacity of Hopfield model. However the distortions in the initial binary vector lead to a strong intensification of distortions in Q -dimensional vector, which has a negative influence on the memory capacity. This paper demonstrates that introduction of the proximity measure (i.e. accounting of noise distribution arising at the time of display) increases the memory capacity (by up to 30 times).

The paper has the following structure. The second section shows description of vector perceptron with a measure of proximity between neuron states. The third section gives the description of display algorithm of binary vectors in Q -dimensional, with specification of its advantages and disadvantages. It is shown that the display intensifies the distortions. The fourth section contains estimations of recognizing characteristics of the vector perceptron (memory capacity, probability of false identification). It is demonstrated that introduction of the proximity measure increases the memory capacity by up to 30 times.

2 Model Description

Let's consider the vector perceptron (VP), consisting of two layers of vector neurons, where each neuron of the input layer (N neurons) is connected with all the neurons of the output layer (n neurons). Neurons of input and output layers have Q and q of discrete states accordingly (in the general case $Q \neq q$). The state of the input layer neurons is described by unit vectors $\{\mathbf{e}_k\}^Q$ Q -dimensional space R^Q , and the state of the output layers – by unit vectors $\{\mathbf{v}_k\}^q$ q -dimensional space R^q .

Let each reference vector $\mathbf{X}_\mu = (\mathbf{x}_{\mu 1}, \mathbf{x}_{\mu 2}, \dots, \mathbf{x}_{\mu N})$ be put in a one-to-one correspondence to the response vector $\mathbf{Y}_\mu = (\mathbf{y}_{\mu 1}, \mathbf{y}_{\mu 2}, \dots, \mathbf{y}_{\mu n})$, where $\mathbf{x}_{\mu i} \in \{\mathbf{e}_k\}^Q$, $\mathbf{y}_{\mu i} \in \{\mathbf{v}_k\}^q$ and $\mu = 1, 2, \dots, M$. Then the synaptic connection between i -th and j -th neurons is assigned by $q \times Q$ -matrix according to the generalized Hebb's rule:

$$\mathbf{W}_{ij} = \sum_{\mu=1}^M \mathbf{y}_{\mu i} (\mathbf{x}_{\mu j} - \mathbf{e}_0)^T \mathbf{J}, \quad (1)$$

where

$$\mathbf{e}_0 = \frac{1}{Q} \sum_{k=1}^Q \mathbf{e}_k, \quad i = \overline{1, n} \text{ and } j = \overline{1, N}, \quad (2)$$

and \mathbf{J} – is a symmetric $Q \times Q$ -matrix of measures of proximities between the state of input layer's neurons, which $J_{kl} = J_{lk}$ elements – are the measure of proximities between the states k and l , $k, l = 1, 2, \dots, Q$. The measure of proximity between the states of the output layer neurons is not entered. If a unit matrix \mathbf{E} (i.e. $\mathbf{J}=\mathbf{E}$) is chosen as \mathbf{J} matrix, the expression (1) will describe weights of the classic Potts perceptron [1,5-8]. Therefore, in order to enter the measure of proximity to the Potts perceptron that has been trained already, it would be sufficient to modify interconnections, multiplying them by \mathbf{J} matrix on the right side.

\mathbf{J} matrix of the measures of proximity may be assigned either by problem's specifications, or based on the data analysis and nature of the noise. In order to enter information on noise distribution to the VNN it is suggested to assign proximity measure between the states of neurons as equal to probability of switching from one state to another under the influence of distortions:

$$J_{kl} = P_{kl}, \quad k, l = \overline{1, Q}, \quad (3)$$

where P_{kl} – is a probability of switching from state k to state l under the influence of distortions.

When presenting an unknown input vector $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N)$, where $\mathbf{x}_j \in \{\mathbf{e}_k\}^Q$, response \mathbf{Y} of the vector perceptron will be calculated as follows. The local field on i -output neuron will be calculated as follows:

$$\mathbf{H}_i = \sum_j^N \mathbf{W}_{ij} \mathbf{x}_j. \quad (4)$$

Then the i -th neuron, similar to the spin, locating in the magnetic field under the influence of the local field \mathbf{H}_i assumes the position which is the closest to the direction of this field (the neuron's state is discrete, that is why it can not be oriented exactly along vector \mathbf{H}_i). In other words the neuron is oriented along that unit vector \mathbf{v}_r , the local field's projection \mathbf{H}_i on which is maximal. Let it be, for instance, the projection on unit \mathbf{v}_3 . Then the i -th output neuron will be switching in state №3 described by a unit vector \mathbf{v}_3 :

$$\mathbf{y}_i = \mathbf{v}_3. \quad (5)$$

This procedure is carried out concurrently for all the output neurons ($i=1, 2, \dots, n$).

3 Binary Vector Pre-processing

The patterns identification is carried out in two stages. At the first stage the input binary signal is preprocessed: the binary L -dimensional vector \mathbf{Z} is displayed in Q -dimensional N -dimensional vector \mathbf{X} . At the second stage, the Q -dimensional pattern \mathbf{X} is identified (recognized) by vector perceptron, i.e. response vector \mathbf{Y} which carries all the required information on input binary vector \mathbf{Z} is generated.

The pre-processing – displaying a binary vector in Q -dimensional vector, involves the following. Let us have L -dimensional binary vector $\mathbf{Z} = (z_1, z_2, \dots, z_L)$, $z_i \in \{0, 1\}$. Let's imagine how we divide it by N fragments each containing per r elements ($N=L/r$). the i -th fragment of the binary vector may be considered as a certain integer number k_i

written as the binary code, $0 \leq k_i \leq Q-1$, $Q=2^r$, $i=1, \dots, N$. Let's put in correspondence with i -th binary fragment a unit vector, describing the k_i -th neuron state, i.e. $\mathbf{x}_i = \mathbf{e}_{k_i}$, where $\mathbf{e}_{k_i} = (0, \dots, 0, 1, 0, \dots, 0)$ – is a unit vector of Q -dimensional space, which k_i -th component is equal to 1. Therefore the whole binary vector \mathbf{Z} is put in a one-to-one correspondence with a set of Q -dimensional vectors, i.e. pattern $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N)$, $\mathbf{x}_i \in \{\mathbf{e}_k\}^Q$. Let's emphasize that \mathbf{X} is not a vector, but in the future will refer to it as a vector, keeping in mind that its components – are Q -dimensional unit vectors.

The presented display has both advantages and disadvantages. On the one hand the display procedure suppresses the existing correlations of reference binary patterns. E.g. let's take two binary vectors $\mathbf{Z}_1 = (10000001)$ and $\mathbf{Z}_2 = (10011001)$, 75% of which components coincide. If we would use the display procedure with parameter $r=2$ ($N=4$, $Q=4$) to them, i.e. dividing each vector by four fragments per two elements, we will get two Q -dimensional patterns $\mathbf{X}_1 = (\mathbf{e}_2, \mathbf{e}_0, \mathbf{e}_0, \mathbf{e}_1)$ and $\mathbf{X}_2 = (\mathbf{e}_2, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_1)$, which overlay is possibly only by 50% (only two components out of four coincide – the first one and the last one). Using the same procedure with display parameter $r=4$ ($N=2$, $Q=16$), we'll get two displays $\mathbf{X}_1 = (\mathbf{e}_8, \mathbf{e}_1)$ and $\mathbf{X}_2 = (\mathbf{e}_9, \mathbf{e}_9)$ which do not overlay at all. On the other hand, distortions in the initial binary vector will result in a strong intensification of distortions in Q -dimensional vector. The overlay of the reference pattern \mathbf{Z} and its distorted version $\tilde{\mathbf{Z}}$ after the display is at the average described by the following expression:

$$O_x = \frac{L}{r} (1 - \alpha)^r, \quad (6)$$

where α – is a probability that the binary vector \mathbf{Z} component will be inverted under the influence of the noise. It is seen that with the increase of r and α the overlay will rapidly tend toward zero. In the above example vector \mathbf{Z}_2 may be considered as vector \mathbf{Z}_1 distorted by 25%. The displaying with $r = 2$ parameter will accordingly result in 50% distortions and with $r = 4$ – in 100% distortions. This paper is aimed at elimination of such negative effect. The main idea is to use the vector perceptron at the second stage, with the measure of proximity, adjusted to distortions of such type. Probability that k -th state will be converted to l -th under the influence of distortions, shall be determined by the following expression:

$$P_{kl} = (1 - \alpha)^{r - \Delta_{kl}} \alpha^{\Delta_{kl}}, \quad (7)$$

where Δ_{kl} – is the Hamming distance between binary representation of k and l numbers. Then having assigned $J_{kl} = P_{kl}$ we'll sufficiently increase noise immunity of the system.

4 Efficiency of Identification

Deducing estimations of VNN efficiency with a measure of proximity is identical to the approach published in the papers [3,4]. Let's introduce only finite expressions for probability of a proper identification, memory capacity, computational complexity and capacity of main memory required for storage of synaptic coefficients.

The computational complexity of the identification process (4)-(5) with introduction of the measure of proximity has not been changed:

$$OP = nNq \quad [\text{operations}]. \quad (8)$$

The requirements for the capacity of main memory, necessary for storage of synaptic coefficients, have also remained unchanged (1):

$$RAM = 4nNqQ \quad [\text{bytes}]. \quad (9)$$

It is supposed that storage of a single element of \mathbf{W}_{ij} matrix requires to allocate 4 bytes.

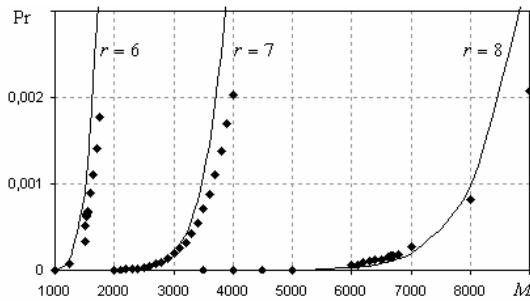


Fig. 2. Dependence on probability of error Pr of a number of M reference patterns written to the network for the fixed parameter of display r and level of distortions α . Thin lines – theory, markers – experiment (. Parameters: $L=1000$; $r=6,7,8$; $\alpha=20\%$).

The probability that vector perceptron would identify the input pattern \mathbf{X} improperly, i.e. restore the response vector \mathbf{Y} with errors, is estimated by the following expression:

$$Pr = \frac{n(q-1)}{\gamma \sqrt{2\pi}} \exp\left(-\frac{\gamma^2}{2}\right), \quad (10)$$

where γ – “signal/noise” relationship would appear as follows:

$$\gamma^2 = \frac{L2^{r-1}q}{rM} \left[[(1-\alpha)^2 + \alpha^2]^r - \frac{1}{2^r} \right]. \quad (11)$$

The obtained estimation of error probability has a fairly good agreement with an experiment (fig. 2). This enables to estimate memory capacity on the basis of such estimation.

Maximum amount of M reference patterns, which can be reliably identified with an error less than the preset value P_{max} , will be obtained by solving a transcendental equation $Pr(M)=P_{max}$. The memory capacity of the two-stage identification scheme, at the second stage of which a vector perceptron with measure of proximity is used, is given by:

$$M = \frac{L2^{r-2}q}{r\ln(nq/P_{\max})} \left[\left((1-\alpha)^2 + \alpha^2 \right)^r - \frac{1}{2^r} \right]. \quad (12)$$

For comparison let's show the memory capacity of the two-stage identification scheme, at the second stage of which a vector perceptron without measure of proximity is used [3,4]:

$$M_0 = \frac{L2^{r-2}q}{r\ln(nq/P_{\max})} \left[(1-\alpha)^{2r} - \frac{1}{2^r} \right]. \quad (13)$$

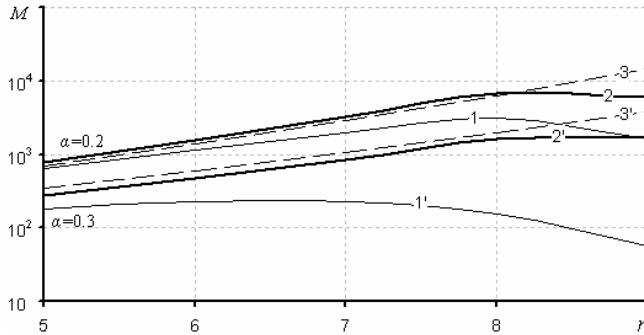


Fig. 3. Dependence of the memory capacity on a display parameter r : thin curves 1 and $1'$ – standard mapping, thick curves 2 and $2'$ – VNN with measure of proximity, dotted curves 3 and $3'$ – estimation (12). Curves 1, 2 and 3 are generated for $\alpha=0.2$ and $L=1000$, and the curves $1'$, $2'$ and $3'$ are generated for $\alpha=0.3$ and $L=1000$.

Fig. 3 demonstrates how the memory capacity of two-stage system without a measure of proximity (thin curves) and with measure of proximity (thick curves) depends on the value of the display parameter r , obtained as the result of experiment in identification of binary vectors with a dimension of $L=1000$. Memory capacity M was determined from the condition $\Pr(M)=M^{-1}$, i.e. as a maximum number of patterns, recognizable without an error. Upper curves $1', 2'$ and $3'$ correspond to the level of distortions $\alpha=0.2$, and the lower curves 1, 2 and 3 – $\alpha=0.3$. As we can see, with a rise of level of distortions α , the memory capacity of both models is reduced. However, a system with the measure of proximity is more resistant to distortions (thick curves 2 and $2'$ are located above the thin curves 1 and $1'$). Introduction of the measure of proximity increases the memory capacity by 2 times with $\alpha=0.2$ and by 7 times when $\alpha=0.3$. With increase of the display parameter r after a certain critical value, the memory capacity M starts to drop. This is related to the fact that distortions are heavily intensified at the time of display which results in a situation where at a given time all binary fragments contain at least one distorted bit, which leads to upsetting the recognition system without the measure of proximity (curve $1'$ in the field $r=8-9$). The system with a measure of proximity is also a subject to this effect, but to a substantially lesser degree: e.g. with $r=9$ and $\alpha=0.3$ the relationship $M/M_0=30$. Now let's

compare a theoretical evaluation of the memory capacity (12) with the experiment results (thick curves 2 and 2', and dotted curves 3 and 3' accordingly). It is seen that theory agrees well with the experiment except for the point $r=9$, where the memory capacity is not dropped due to the effect described above. Hopfield scalar model with noises $\alpha=0,2$ and $\alpha=0,3$ can store only 26 and 12 binary patterns accordingly, which is two orders lesser than the memory capacity of the two-stage system with the measure of proximity, able to store and reliably restore 6226 and 1680 reference vectors accordingly.

5 Conclusion

A model of neural network is developed, which enables to easily and naturally embed a priori information about the distribution of noise. The estimations of recognition characteristics of model have been obtained. The example with an identification of binary patterns shows agreement of the obtained estimations with computational experiment. Accounting of the noise nature enables to sufficiently increase memory capacity and system noise immunity. It has been demonstrated that the capacity of content-addressable memory, generated on the basis of vector neural network with the measure of proximity which by several orders is greater than the memory capacity of Hopfield's model.

High reliability parameters of the proposed algorithm enable to use it as a fast retrieval algorithm, operating where there are input data distortions.

Dedicated to the memory of academician Andrey Leonovich Mikaelyan. The work in part supported by RFBR #09-07-00159-a and DNT RAS 1.7.

References

1. Kryzhanovsky, B., Kryzhanovskiy, V., Litinskii, L.: Machine Learning in Vector Models of Neural Networks. In: Michalski. Koronacki, J., Ras, Z.W., Wierzchon, S.T., et al. (eds.) *Advances in Machine Learning II. SCI*, pp. 427–443. Springer, Heidelberg (2010)
2. Hopfield, J.J.: Proc. Nat. Acad. Sci. USA 79, 2554–2558 (1982)
3. Kryzhanovsky, V., Kryzhanovsky, B., Fonarev, A.: Application of potts-model perceptron for binary patterns identification. In: Kůrková, V., Neruda, R., Koutník, J. (eds.) *ICANN 2008*, Part I. LNCS, vol. 5163, pp. 553–561. Springer, Heidelberg (2008)
4. Kanter, I.: Potts-glass models of neural networks. *Physical Review A*, 37(7), 2739–2742 (1988)
5. Cook, J.: The mean-field theory of a Q-state neural network model. *Journal of Physics A* 22, 2000–2012 (1989)
6. Nadal, J., Rau, A.: Storage capacity of a Potts-perceptron. *J. Phys. I: France* 1, 1109–1121 (1991)
7. Bolle, D., Dupont, P., Huyghebaert, J.: Thermodynamics properties of the q-state Potts-glass neural network. *Phys. Rew. A* 45, 4194–4197 (1992)

Emerging Bayesian Priors in a Self-Organizing Recurrent Network

Andreea Lazar^{1,*}, Gordon Pipa², and Jochen Triesch³

¹Max-Planck Institute for Brain Research, Deutschordenstr. 46, 60528, Frankfurt am Main, Germany

²Institute of Cognitive Science, University of Osnabrueck, Albrechtstr. 28, 49069 Osnabrueck, Germany

³Frankfurt Institute for Advanced Studies, Ruth-Moufang-Str. 1, 60438, Frankfurt am Main, Germany
andreea.lazar@brain.mpg.de

Abstract. We explore the role of local plasticity rules in learning statistical priors in a self-organizing recurrent neural network (SORN). The network receives input sequences composed of different symbols and learns the structure embedded in these sequences via a simple spike-timing-dependent plasticity rule, while synaptic normalization and intrinsic plasticity maintain a low level of activity. After learning, the network exhibits spontaneous activity that matches the stimulus-evoked activity during training and thus can be interpreted as samples from the network's prior probability distribution over evoked activity states. Further, we show how learning the frequency and spatio-temporal characteristics of the input sequences influences network performance in several classification tasks. These results suggest a novel connection between low level learning mechanisms and high level concepts of statistical inference.

Keywords: Spontaneous activity, statistical priors, Bayesian inference, STDP, intrinsic plasticity, recurrent networks.

1 Introduction

Convincing evidence supports the idea that cortical processing is efficiently adapted to the statistical properties of the environment and that perception and action arise on the basis of Bayesian statistical calculations. However, little is known about the neuronal correlates of statistical learning and inference [1]. Specifically, it is not known how statistical priors are encoded by neurons in a dynamic setting and how they are combined with sensory information to generate an optimal percept.

Based on the finding that more frequently experienced stimuli gain stronger representations in sensory cortices [2], it has been hypothesized that Bayesian stimulus priors are stored via cortical plasticity. While spontaneous activity is highly structured both in space and time [3], the repetitive presentation of a

* Corresponding author.

given visual stimulus was shown to increase the similarity between spontaneous and sensory-evoked responses [4]. Berkes et al. [5] related stimulus-evoked and spontaneous activities in the visual cortex of ferrets to inferences and prior expectations in an internal model of the visual environment and predicted that they should match if the model is statistically optimal. The similarity between evoked responses to natural scenes and spontaneous activation increased with developmental age, suggesting an efficient rewiring in network connectivity via plasticity mechanisms during the critical period.

In previous work, we showed that cortical plasticity mechanisms can improve the performance of reservoir networks on memory and prediction tasks [6]. Reservoir Computing (RC), in the form of Echo State Networks [7] and Liquid State Machines [8], aims to circumvent the difficulties of training recurrent networks. The defining feature of RC is that the recurrent connections in the reservoir are fixed and random and only the projections to readout neurons are trained. In contrast, in self-organizing recurrent networks (SORNs) [6] the recurrent connections are trained via biologically plausible plasticity mechanisms while utilizing unlabelled data (unsupervised training).

In this study, we explore the role of local plasticity rules in incorporating priors that reflect the statistical properties of the stimuli. Following the repetitive presentation of a given set of stimuli, we omit the input and analyse the characteristics of spontaneous activity (Section 3). We find that the network revisits states similar to those evoked by sensory input and that it follows similar trajectories through its high-dimensional state space. Furthermore, the network has learned the statistical properties of the data: during spontaneous activity the network preferentially visits states that are similar to evoked activity patterns for inputs with a higher prior probability. In several classification tasks, we show that the network’s ‘perception’ of the current source of information is influenced by the learned prior probabilities (Section 4).

2 Self-Organizing Recurrent Network (SORN)

2.1 Network Architecture

The SORN model introduced in [6] consists of a population of excitatory cells ($N^E = 200$) and a smaller population of inhibitory cells ($N^I = 0.2 \times N^E = 40$). All possible connections between the excitatory and inhibitory neurons are present (W^{IE} and W^{EI}), while the excitatory-excitatory connections (W^{EE}) are sparse with a mean number $\lambda^W = 10$ of incoming and outgoing connections per neuron. The weight strengths are drawn from the interval $[0, 1]$ and subsequently normalized such that the incoming connections to a neuron sum up to a constant value: $\sum_j W_{ij}^{IE} = 1$, $\sum_j W_{ij}^{EI} = 1$ and $\sum_j W_{ij}^{EE} = 1$.

Inputs are time series of different symbols. Each input symbol is associated with a specific group of N^U excitatory input units, which all receive positive input drive ($\nu^U = 1$) when the associated symbol is active. The network state, at a discrete time t , is given by the binary vectors $x(t) \in \{0, 1\}^{N^E}$ and $y(t) \in$

$\{0, 1\}^{N^I}$ corresponding to the activity of the excitatory and inhibitory units:

$$x_i(t+1) = \Theta \left(\sum_{j=1}^{N^E} W_{ij}^{EE}(t) x_j(t) - \sum_{k=1}^{N^I} W_{ik}^{EI} y_k(t) + \nu_i^U(t) - T_i^E(t) \right) \quad (1)$$

$$y_i(t+1) = \Theta \left(\sum_{j=1}^{N^E} W_{ij}^{IE} x_j(t) - T_i^I \right) . \quad (2)$$

The threshold values for excitatory (T^E) and inhibitory units (T^I) are initially drawn from a uniform distribution in the interval $[0, T_{max}^E]$ and $[0, T_{max}^I]$, with $T_{max}^E = 0.5$ and $T_{max}^I = 0.9$. The Heaviside step function Θ constrains the network activation at time t to a binary representation: a neuron fires if the total drive it receives is greater than its threshold, otherwise it stays silent.

The initial values for excitatory weights (W^{EE}) and thresholds (T^E) stay constant for random reservoirs. These parameters have been tuned such that the dynamics of the reservoir is close to criticality and its performance in prediction tasks is high [6]. SORNs change their initial structure during stimulation in an unsupervised fashion following the rules described in Section 2.2.

2.2 Plasticity Mechanisms

Learning with spike-timing dependent plasticity (STDP) is constrained to the set of W^{EE} synapses. We use a simple model of STDP that strengthens (weakens) the synaptic weight W_{ij}^{EE} by a fixed amount $\eta_{STDP} = 0.001$ whenever unit i is active in the time step following (preceding) activation of unit j .

$$\Delta W_{ij}^{EE}(t) = \eta_{STDP} (x_i(t)x_j(t-1) - x_i(t-1)x_j(t)) ; \quad (3)$$

To keep the activity balanced during learning we make use of additional homeostatic mechanisms that are sensitive to the total level of synaptic efficacy and the post-synaptic firing rate. Synaptic normalization (SN) proportionally adjusts the values of incoming connections to a neuron so that they sum up to a constant value. At every time step:

$$W_{ij}^{EE}(t) \leftarrow W_{ij}^{EE}(t) / \sum_j W_{ij}^{EE}(t) . \quad (4)$$

An intrinsic plasticity (IP) rule spreads the activity evenly across units, such that on average each excitatory neuron will fire with the same target rate H_{IP} . To this end, a unit that has just been active increases its threshold while an inactive unit lowers its threshold by a small amount:

$$T_i^E(t+1) = T_i^E(t) + \eta_{IP} (x_i(t) - H_{IP}) , \quad (5)$$

where $\eta_{IP} = 0.001$ is a small learning rate.

The implementation of the model described above and the simulations presented in the following sections were performed in Matlab. Further details about the model dynamics and performance are available in [6] and a toolbox can be downloaded at www.fias.uni-frankfurt.de/~lazar/SORN.

3 Spontaneous Activity in SORNs Matches the Statistics of the Inputs

In the first analysis we show that following learning, SORN's internal representations match the input statistics, so that more common input sequences gain stronger influence on the network's dynamics. We compare periods of spontaneous activity before and after learning, and show that the latter faithfully reflects the structure and frequency of stimulation.

Our inputs are random alternations of two sequences of letters, ABCD and MNOP, where the second sequence is presented two times more often or three times more often than the first, during 50000 steps of network self-organization with plasticity. Evoked responses for SORNs are modulated by STDP, SN and IP. We condition synaptic learning to the periods containing input stimulation. In the absence of inputs, we analyse spontaneous activity in the presence of IP, but in the absence of synaptic learning. During spontaneous activity, the homeostatic nature of intrinsic plasticity restores stable activity rates within 2000 time steps.

In Fig 1a we compare the result of PCA on the network's internal representations during 100 steps of evoked activity at the end of 50000 steps of a learned sequence of letters (circles) with 100 steps of spontaneous activity (blue arrows). As the figure suggests, spontaneous activity follows transitions similar to those embedded during learning.

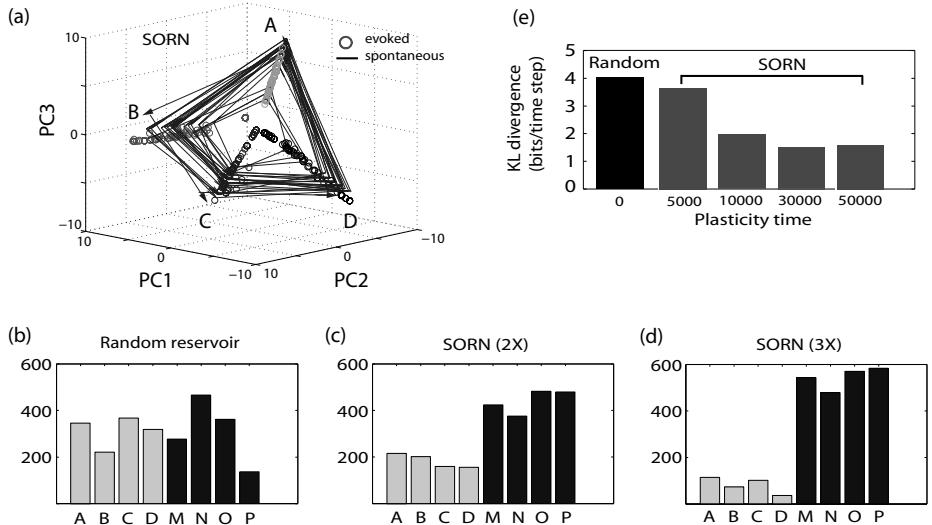


Fig. 1. (a) Result of PCA on SORN's internal representations; (b, c ,d) Distribution of spontaneous states for random networks in the absence of plasticity compared to SORNs when MNOP is presented two times more often or three times more often than ABCD during plastic self-organization. (e) Divergence between the distributions of evoked and spontaneous activity as a function of plastic self-organization.

We quantify the similarity between 5000 steps of evoked and 5000 steps of spontaneous activity. For each input letter, we store the unique patterns of evoked activity (vectors of zeroes and ones). We match each activation pattern during spontaneous activity to the closest evoked response in terms of Hamming distance. We find that during spontaneous activity the network preferentially visits states that are similar to evoked activity patterns and the mean number of visits matches the proportions in which the sequences were presented during training. Spontaneous activity is more similar to the letters MNOP, presented two times (Fig. 1c) or three times (Fig. 1d) more often during learning, while this effect cannot be noticed for random networks (Fig. 1b). These results indicate that the structural transformations induced by STDP, result in spatio-temporal representations that match the statistical structure of the stimuli.

Following the assumptions from [5], a statistical model that is optimally adapted to a set of stimuli exhibits priors that match the actual frequency with which features occur in that set. The fit of an internal model for processing stimuli can be quantified as the distance between the average evoked responses (corresponding to the posterior) and the spontaneous responses (corresponding to the prior distribution). We measured 5000 steps of evoked activity in response to our input sequences and 5000 steps of spontaneous activity in the absence of inputs at five stages of learning: for random networks without plasticity and after 5000, 10000, 30000 and 50000 steps of plasticity in SORNs (Fig. 1e). We find that SORNs have a decreasing Kullback-Leibler divergence between evoked and spontaneous distributions of patterns as a function of learning time, which indicates a gradual adaptation of internal models at the neural level to the statistics of the processed stimuli.

4 Influence of Learned Bayesian Priors on Classification

4.1 Bistable Stimuli

We next investigated how learning internal representations that match the stimulus statistics influences the performance of the network on a classification task. During 50000 steps of plastic self-organization we present two motion stimuli going from left to right (sequence 1234) or going from right to left (sequence 4321). Either the two sequences are presented equally often (Fig. 2 a,b), or right-motion is presented three times more often than left-motion (Fig. 2 c,d). We train a classifier, based on the last 5000 steps of network activity, to map network states to the corresponding directions. We use the Moore-Penrose pseudoinverse method which minimizes the squared difference between the output of the readout neurons and the target output value. During test we present the network with bistable stimuli in which we alternate the simultaneous presentation of symbols 1 and 3 with the simultaneous presentation of symbols 2 and 4 (5000 steps).

The SORN projects a similar amount of output drive towards the left and right-motion readouts, when the two motion directions have been presented equally often during self-organization, and more drive towards the right-motion

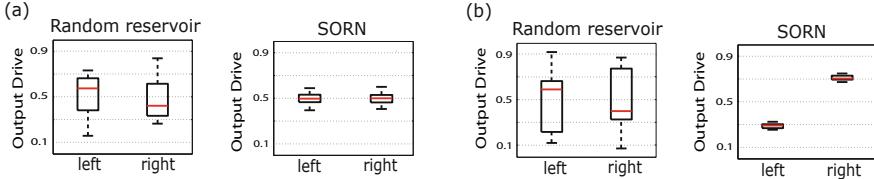


Fig. 2. Two motion stimuli are presented equally often (a), or right motion is presented three times more than left motion (b). For SORNs the output drive generated by bistable stimuli is biased according to the input priors and the output variance is significantly reduced compared to random networks.

readout, when right-motion was presented three times more often (average over 10 runs). In comparison, the random networks on the left column show no stimulus specificity and display a high level of output variance.

4.2 Multiple Sources of Information

We then investigated the ability of a secondary source of information to disambiguate the bistable percept from the previous task. During self-organization (50000 steps) we present two streams of inputs, both describing the same left-right motion: $v_1v_2v_3v_4$, and $a_1a_2a_3a_4$ for right motion and $v_4v_3v_2v_1$, and $a_4a_3a_2a_1$ for left motion. For convenience we will refer to these two sources of information as ‘visual’ and ‘auditory’, however we do not imply any particularities related to visual and auditory processing. The visual and auditory sequences are presented separately (50% of trials) or simultaneously (50% of trials). First, a readout is trained to map the network activity to the direction of motion, based on all trials containing auditory, visual input and both (Fig. 3, left). Secondly, a readout is trained to map the direction of motion based on trials containing visual information but not auditory (Fig. 3, right). During test, the visual input is bistable (v_1 and v_3 are presented simultaneously, followed by v_2 and v_4), while the auditory input is informative and alternates randomly between left and right motion. The performance of SORN to classify the correct direction of motion is higher compared to the performance of a random network. Interestingly, even when the read-out is trained entirely on visual information (Figure 3, right) the performance of the SORN is higher than chance level, suggesting an established

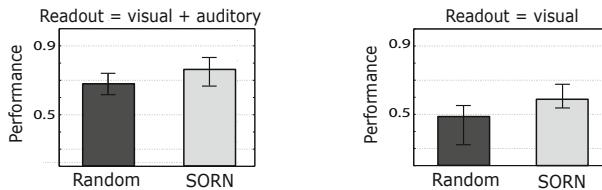


Fig. 3. Classification performance for the direction of motion of a readout trained on visual and auditory information (left) or only visual information (right). During test, the visual input is bistable and the auditory input is informative.

association between auditory and visual information at the level of neuronal dynamics.

4.3 Prediction Cues

In the last experiment, we examined the ability of SORNs to learn prediction cues and we analyzed the integration between the current source of information (a mixture of cues) and the pre-learned prior probabilities.

Inputs are two randomly alternating sequences of symbols: $AXXXXXM$ and $BXXXXXN$. Symbols A and B cue the network for the later appearance of M and N , respectively, following a 6 time steps delay. Networks are shaped by plasticity, while presented with these two cue associations for 50000 steps. Sequence $AXXXXXM$ is presented three times more often during learning. Based on 5000 steps of activity, a linear readout is trained to map the network's recurrent activity to the incoming input M or N . During the test phase the network is presented with mixtures of the two cues (e.g 80% cue A , 20% cue B). These mixtures correspond to the number of input neurons which receive external drive for each of the two input populations (e.g. 8 input neurons out of 10 for A , versus 2 input neurons out of 10 for B).

In Fig. 4, we contrast the behavior of random reservoirs to that of SORNs. For SORNs the output drive projected towards the predicted symbols M and N is strongly biased by the prior information. As a result, the network's prediction relies more strongly on cue A , which was presented more often during training. The output drive matches the optimal Bayesian solution (gray lines).

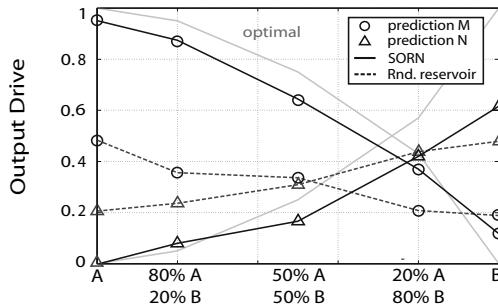


Fig. 4. Network output drive for a prediction task (avg. over 10 runs) when the network is presented with mixtures of two learned cues. For SORNs, the output relies more strongly on cue A , which was presented three times more often during self-organization

5 Conclusions

We have shown that a recurrent network model endowed with local neuronal plasticity mechanisms can learn the statistical structure of its input stimuli, by developing an internal model of its environment.

The resulting neuronal representations reflect Bayesian priors that match the frequency and the spatio-temporal characteristics of the stimuli and influence the network's performance in classification tasks. Interestingly, with learning, the network's output responses decrease in variability (Fig. 3). While our work was focused on the unsupervised emergence of Bayesian priors, rather than on the integration between priors and likelihoods, in the last example, the sensory-evoked activity (an ambiguous prediction cue) was combined with the prior probability which resulted in a nearly optimal statistical integration.

Our results suggest important connections between low level principles of learning via neuronal plasticity and high level concepts of statistical inference.

Acknowledgments. This work was supported by the NeFF Project of the Hessian LOEWE initiative, GABA Project EU-04330 and the ICT EU grant PHOCUS, Project 240763 GP.

References

1. Fiser, J., Berkes, P., Orban, G., Lengyel, M.: Statistically optimal perception and learning: from behavior to neural representations. *Trends Cogn. Sci.* 14(3), 119–130 (2010)
2. Kver, H., Bao, S.: Cortical plasticity as a mechanism for storing bayesian priors in sensory perception. *PLoS One* 5(5), e10497 (2010)
3. Ringach, D.L.: Spontaneous and driven cortical activity: implications for computation. *Curr. Opin. Neurobiol.* 19(4), 439–444 (2009)
4. Han, F., Caporale, N., Dan, Y.: Reverberation of recent visual experience in spontaneous cortical waves. *Neuron* 60(2), 321–327 (2008)
5. Berkes, P., Orban, G.: Spontaneous cortical activity reveals hallmarks of an optimal internal model of the environment. *Science* 331(6013), 83–87 (2011)
6. Lazar, A., Pipa, G., Triesch, J.: SORN: a self-organizing recurrent neural network. *Front Comput. Neurosci.* 3, 23 (2009)
7. Jaeger, H.: The "echo state" approach to analysing and training recurrent neural networks. GMD Report 148, GMD - German National Research Institute for Computer Science (2001)
8. Maass, W., Natschläger, T., Markram, H.: Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation* 14, 2531–2560 (2002)

Momentum Acceleration of Least–Squares Support Vector Machines

Jorge López, Álvaro Barbero, José R. Dorronsoro*

Dpto. de Ingeniería Informática and Instituto de Ingeniería del Conocimiento
Universidad Autónoma de Madrid
`{j.lopez,alvaro.barbero,jose.dorronsoro}@uam.es`

Abstract. Least–Squares Support Vector Machines (LS–SVMs) have been a successful alternative model for classification and regression Support Vector Machines (SVMs), and used in a wide range of applications. In spite of this, only a limited effort has been realized to design efficient algorithms for the training of this class of models, in clear contrast to the vast amount of contributions of this kind in the field of classic SVMs. In this work we propose to combine the popular Sequential Minimal Optimization (SMO) method with a momentum strategy that manages to reduce the number of iterations required for convergence, while requiring little additional computational effort per iteration, especially in those situations where the standard SMO algorithm for LS–SVMs fails to obtain fast solutions.

1 Introduction

LS–SVMs were introduced in [13] as a simplification of SVMs with several advantages: 1) the dual formulations for classification and regression are identical, 2) this common formulation can be rewritten as a linear system of Karush–Kuhn–Tucker (KKT) equations, 3) the subsequent models show a similar performance to the SVM ones in real-world problems.

Let a sample of N patterns $\{X, y\}$, where X is the $N \times d$ matrix with pattern X_i as its i -th row, and the target vector $y \in \mathbb{R}^N$, with $y_i \in \{+1, -1\}$ for classification and $y_i \in \mathbb{R}$ for regression. An LS–SVM finds a hyperplane, with normal w and bias b , by solving the primal problem

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2} w^T w + \frac{C}{2} \xi^T \xi \\ \text{s.t.} \quad & \Phi(X) w + b e = y - \xi \end{aligned} \tag{1}$$

* With partial support of Spain’s TIN 2010–21575–C02–01 project and Cátedra UAM–IIC en Modelado y Predicción. The first author is kindly supported by FPU–MEC grant AP2007–00142. The second author was partly supported by the FPU–MEC grant reference AP2006–02285 during the development of this work.

where T denotes transpose, e is a vector with ones in all its components, $\Phi(X)$ is a new pattern matrix induced from X , for instance, by a kernelization process, and ξ is a slack vector with components ξ_i that measure the deviations from the hyperplane $w^T\Phi(X_i) + b = y_i$. Introducing the equalities in the Lagrangian function with a vector of coefficients α , differentiating w.r.t. the primal variables and equalling to 0, we get the dual problem

$$\begin{aligned} \min_{\alpha} f(\alpha) &= \frac{1}{2} \alpha^T Q \alpha - y^T \alpha \\ \text{s.t.} \quad \alpha^T e &= 0, \end{aligned} \quad (2)$$

where Q is the matrix with elements $Q_{ij} = k(X_i, X_j) + \delta_{ij}/C$. Here, k is a Mercer kernel function giving $k(X_i, X_j) = \Phi(X_i)^T \Phi(X_j)$, whereas δ_{ij} stands for Kronecker's delta, that is, $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ if $i \neq j$.

In order to solve these formulations, there exist two different approaches. The first one is to combine (1) and (2) to obtain a system of linear equations, and apply linear algebra algorithms. The second one consists of solving iteratively the dual (2), as it is done for SVMs, with some decomposition algorithm. The main advantage of this second approach when compared to the first one is that it does not require to invert any matrix, being thus more suitable for large problems.

We will follow the second approach; more precisely, we propose to combine the Sequential Minimal Optimization (SMO) method with a momentum strategy that reduces the number of iterations required for convergence, while requiring little additional computational effort. The methods proposed here are a significant extension to the ideas presented in [2] (see [1] for similar ideas for standard SVMs).

The rest of the paper is organized as follows: in Section 2 we explain a bit more in detail these two different approaches. Section 3 describes our method. Its performance is studied in the experiments of Section 4. Finally, Section 5 gives conclusions and some pointers to further work.

2 Algorithms for LS-SVM Training

As was explained above, the first approach that was devised by the creators of LS-SVMs is based on writing the problem as a system of equations. This can be done because the primal (1) only has equality constraints. Combining the constraints of (1) and (2), and making use of the fact that w can be written as $w = \sum_i^N \alpha_i \Phi(X_i)$, yields the system

$$\left[\begin{array}{c|c} 0 & e^T \\ \hline e & Q \end{array} \right] \left[\begin{array}{c} b \\ \alpha \end{array} \right] = \left[\begin{array}{c} 0 \\ y \end{array} \right] \quad (3)$$

Specifically, the method they proposed to deal with this system is a Hestenes-Stiefel conjugate gradient method after transforming (3) to an equivalent system [12]. Although this algorithm is carefully designed to avoid any inversion of matrices, its convergence becomes very slow if the system matrix gets ill-conditioned.

In fact, in the last version of the Matlab package for LS-SVMs by these authors, the system of equations is simply solved by Matlab's backslash operator [6].

The second approach, coming from the literature of SVMs, deals with the dual (2). The idea is to solve iteratively the problem with a decomposition algorithm. The simplest algorithm of this kind is SMO [10], because it only chooses two coefficients α_L, α_U for updating (observe that this is the minimum number we can pick, since the constraint $e^T \alpha = 0$ must hold). SMO was initially adapted to the LS-SVM scenario in [8]. Subsequently, several enhancements to the SMO algorithm were suggested for the SVM case, where the most remarkable was the so-called Second Order pair selection in [7]. The convenience of such selection for LS-SVMs was pointed out very recently in [9]. We will term this variant as LS-SMO henceforth.

The improvement observed is explained by the fact that Second Order information is partially used to choose better updating directions. The version of SMO in [8] only used First Order information, resulting in slower convergence. A purely Second Order method such as Newton's one is not applicable in practice because again it implies expensively inverting a matrix, in this case the Hessian. An alternative to Newton or quasi-Newton methods is Conjugate Gradient, which was explored for this case in [4]. It was observed that it converges faster than SMO when the values of parameter C are large.

Now let us briefly describe LS-SMO, which will be extended in the next section to include a momentum term. First, we need an optimality criterion for deciding when to stop. The constraints in (1) and the optimality condition $\alpha_i = C\xi_i$ mean that $w^T \Phi(X_i) - y_i = \sum_j \alpha_j k(X_i, X_j) - y_i = -b \forall i$. Hence, we can use the approximate optimality condition

$$\max_i \{w^T \Phi(X_i) - y_i\} - \min_i \{w^T \Phi(X_i) - y_i\} \leq \epsilon , \quad (4)$$

for some $\epsilon > 0$ (if $\epsilon = 0$ we force an exact solution) [9]. If we keep track of these quantities $w^T \Phi(X_i) - y_i = \nabla f(\alpha)_i$ by storing the gradient of (2) in memory, Second Order LS-SMO chooses the coefficients α_L, α_U by

$$L = \arg \min_i \{\nabla f(\alpha)_i\} , \quad U = \arg \max_{i \neq L} \left\{ \frac{(\nabla f(\alpha)_i - \nabla f(\alpha)_L)^2}{2Z} \right\} , \quad (5)$$

where $Z = \|\Phi(X_L) - \Phi(X_i)\|^2$. The idea after such a choice is to minimize the dual as much as possible, since it can be shown that the fraction maximized while choosing U is exactly the change in the dual function f [9]. Second Order information is only partially used, since using it fully would imply not fixing L in advance, but finding the pair L, U which maximizes that fraction, which is too costly in practice.

From now on, we will denote $\Delta = \nabla f(\alpha)_U - \nabla f(\alpha)_L$, whereas we can summarize the stopping criterion (4) with $\Delta_{stop} = \nabla f(\alpha)_{max} - \nabla f(\alpha)_{min} \leq \epsilon$, where $\nabla f(\alpha)_{max}$ stands for the maximal component of the gradient and $\nabla f(\alpha)_{min}$ for the minimal one.

Algorithm 1. LS-SMO

```

while  $\Delta_{stop} > \epsilon$  do
  Find working set  $(L, U)$  and  $Z$  with (5).
  Compute  $\delta$  using (6).
   $\alpha = \alpha + \delta s$ .
end while

```

Once we have chosen L and U , the updating direction is $s = e_U - e_L$, with e_i an all-zeroed vector except for a value of 1 at the i -th entry. Using (2) for calculating $f(\alpha + \delta s)$, where δ expresses the updating stepsize, differentiating and equalling to 0 yields the value

$$\delta = -\frac{\Delta}{Z} , \quad (6)$$

where Z denotes the product $s^T Q s$. LS-SMO can thus be summarized in Algorithm 1.

3 Momentum Acceleration

In order to further speed-up LS-SMO, we propose here to include a momentum term in the updating directions chosen by SMO, so that convergence to the optimum is attained in fewer iterations. Thus, the updating direction follows now the expression

$$d = (1 - \lambda)s + \lambda m = \lambda(m - s) + s , \quad (7)$$

where λ stands for an angle parameter of the algorithm which balances the influence between the standard LS-SMO updating s direction and the new momentum term m . Although classic momentum is defined as $m_t = \alpha_t - \alpha_{t-1}$, here we use instead an approximate, memory-limited momentum defined as

$$m_t = \sum_{r=t-\tau}^{t-1} (1 - \lambda_r) \delta_r s_r , \quad (8)$$

in which only the s terms from the most recent τ updates are taken into account. This allows to control sparsity in the resultant direction d , which is mandatory to maintain low computational costs per iteration as shown later on. Note the larger τ the more information is gathered in d . This form of momentum also allows to obtain optimal values for both δ and λ in closed form, as after some algebra it can be shown that the choice

$$\lambda = \frac{-Z \nabla_m + R \nabla_s}{-H \nabla_s + (R - Z)(\nabla_m - \nabla_s)} , \quad \delta = -\frac{(\nabla_m - \nabla_s)(Z - R) + H \nabla_s}{-(R + Z)^2 + HZ} \quad (9)$$

guarantees maximum decrease in the objective function for the given choice of (s, m) . The new expressions in these formulas correspond to $M = m^T Q m$,

Algorithm 2. MLS-SMO

```

 $m = 0.$ 
while  $\Delta_{stop} > \epsilon$  do
  Find working set  $(L, U)$  and  $Z$  as in SMO with (5).
  Compute  $\nabla_m, \nabla_s, M, R, H$  using (10).
  Compute  $(\delta, \lambda)$  using (9).
   $\alpha = \alpha + \delta((1 - \lambda)s + \lambda m).$ 
  Update  $m.$ 
end while

```

$R = m^T Qs, H = (m - s)^T Q(m - s) = M + Z - 2R, \nabla_s = \nabla f(\alpha)^T s$ and $\nabla_m = \nabla f(\alpha)^T m$. Note that all the terms needed in these computations can be easily obtained by defining $u = Qm$, for we have

$$\nabla_m = \sum_{m_i \neq 0} m_i [\nabla f(\alpha)]_i, \nabla_s = [\nabla f(\alpha)]_U - [\nabla f(\alpha)]_L, \quad (10)$$

$$M = m^T u = \sum_{m_i \neq 0} m_i u_i, R = u_U - u_L, H = M + Z - 2R,$$

where we have made explicit use of the sparse structure of m . A simplified pseudocode of the whole algorithm, which we will call MLS-SMO (Momentum LS-SMO), is presented in Algorithm 2.

Addressing now the computational costs of the method, suppose that the gradient $\nabla f(\alpha)$ and u were already available. Then the cost would be $O(2N)$ for the (L, U) coefficients selection (as in LS-SMO), $O(4\tau)$ for the computation of M and ∇_m thanks to the sparsity of m (see equations 10) and $O(N)$ for the α update (the rest of computations have constant cost). Now, the gradient as well as the u updates can be performed efficiently by observing that

$$\nabla f(\alpha + \delta d) = \nabla f(\alpha) + \delta Qd = \nabla f(\alpha) + \delta(1 - \lambda)v + \delta\lambda u, \quad (11)$$

$$u' = Qm' = u + (1 - \lambda)\delta v - (1 - \hat{\lambda})\hat{\delta}v,$$

where $v = Qs$ and $(\hat{\delta}, \hat{\lambda}, \hat{v})$ stand for the corresponding quantities of the oldest term contributing to m (update at time $t - \tau$), which is to be removed due to the limited memory of the momentum, unless m still does not contain τ terms. If that is the case, no removal is needed, and thus $u' = u + \delta(1 - \lambda)v$. By storing in memory the quantities (δ, λ, v) for each term building up m , the MLS-SMO update requires no additional kernel evaluations with respect to the standard SMO update. Summing up costs, the total burden per iteration of MLS-SMO would be $O(4N) + O(4\tau) + \mathcal{K} \cdot O(2N) \simeq O(5N) + \mathcal{K} \cdot O(2N)$, as to ensure sparsity $\tau \ll N$. This should be compared against the LS-SMO cost of $O(3N) + \mathcal{K} \cdot O(2N)$.

Table 1. Average performances (% of misclassification or MSE), iterations and training times for the datasets used, along with the C and σ values used in training

Set	Par.		Perf.	$\tau = 0$		$\tau = 10$		$\tau = 20$	
	C	σ^2		Iters	Time	Iters	Time	Iters	Time
Ba	0.8368	0.8273	10.4	651.8	0.032	627.1	0.034	634.5	0.034
Bc	0.1462	20.128	26.0	332.2	0.010	333.2	0.011	334.6	0.012
Fl	0.5278	34.997	34.0	1277.2	0.126	1278.7	0.137	1272.5	0.136
Ge	0.5915	22.390	23.3	1154.8	0.138	1132.2	0.145	1138.9	0.147
He	0.6748	149.21	15.7	260.4	0.007	260.0	0.008	259.0	0.008
Im	103.46	9.6379	2.7	13211.0	2.868	7118.6	1.644	7128.2	1.651
Pi	256.06	183.93	23.9	5583.9	0.355	1985.6	0.139	2114.4	0.148
Sp	299.70	47.572	10.5	2010.7	0.524	1747.8	0.478	1767.4	0.483
Th	1850.0	0.4430	3.9	1905.3	0.034	1200.2	0.025	1328.3	0.028
Ti	6.3255	1.4087	22.4	382.3	0.006	355.0	0.007	341.2	0.007
Ab	10.797	10.402	80.2	2736.7	0.376	2295.1	0.344	2402.3	0.361
Bo	5810.7	15786.0	0.004	135.6	0.002	109.7	0.002	110.2	0.002
Ho	71.913	18.897	549.6	2257.3	0.069	1126.5	0.038	1112.3	0.038
Mg	1.2928	1.4330	0.056	891.0	0.071	808.2	0.070	817.8	0.071
Mp	12.122	16.4931	538.9	680.1	0.018	546.7	0.017	568.7	0.018

4 Experiments

In order to assess the performance of MLS-SMO, we compared it to LS-SMO on 10 classification datasets (*Banana*, *Breast Cancer*, *Flare*, *German*, *Heart*, *Image*, *Pima*, *Splice*, *Thyroid* and *Titanic*) and 5 regression datasets (*Abalone*, *Bodyfat*, *Housing*, *Mg* and *Mpg*). Classification ones were taken from [11], whereas regression ones were taken from [5]. The patterns from each dataset were randomly split in 100 different training and testing sets in order to average the results.

We used the RBF kernel $k(X_i, X_j) = \exp(-\|X_i - X_j\|/\sigma^2)$ and $\epsilon = 10^{-3}$ for all experiments. To obtain good parameter values for C and σ we applied the routine *tunelssvm* of the toolbox [6], using as cost function a 10-fold cross-validation over a whole dataset. The convenience of these parameter values is assessed by % of misclassification or MSE, depending on the kind of dataset. We tried with momentum sizes of $\tau = 0$ (i.e., LS-SMO), $\tau = 10$ and $\tau = 20$. Table 1 shows the results obtained.

As it can be seen, there are datasets for which little improvement is achieved, but in other datasets (such as *image*, *pima*, *thyroid* and *housing*) remarkable speedups are observed. In the former, MLS-SMO can be considered equal to LS-SMO in the sense that it does not make things worse, but in the latter MLS-SMO is clearly faster. We can also see that it is better to choose $\tau = 10$ than $\tau = 20$, indicating that a too large momentum term does not compensate for its overhead, but it does compensate when its size is moderate.

Noticing the values of the parameters, the datasets with speedups are in general those for which the value of C is large. When this value is very small,

MLS-SMO behaves very similarly to LS-SMO. The influence of σ is less noticeable, but it is also present. For instance, in *bodyfat* C is large, but σ is also large, and the improvement is not very remarkable. In order to understand what is their joint influence, we plot in Fig. 1 the cases of *thyroid* and *bodyfat* for $\tau = 10$.

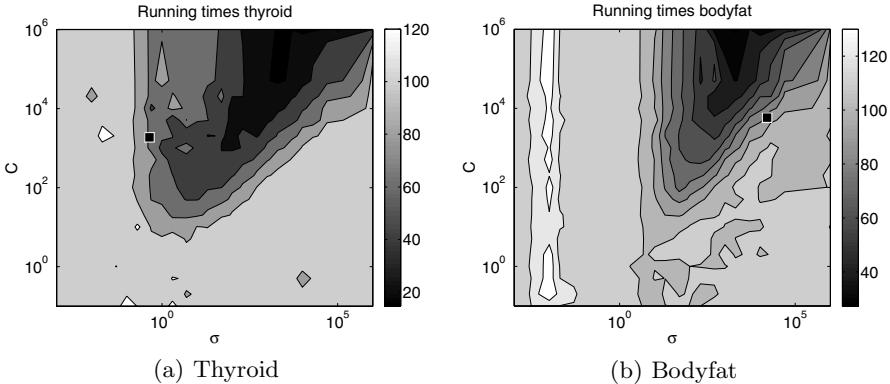


Fig. 1. Influence of parameters C and σ on % of training time of MLS-SMO w.r.t. LS-SMO, with $\tau = 10$. Parameter values of Table 1 are marked with a square.

The figure shows that the region where the improvement is greater is a cone, with one side parallel to the C axis, and the other one parallel to the diagonal $C = \sigma$ (a similar pattern is observed for the rest of datasets). This is why we have speedups if C is large. It is also worth observing that for the rest of the parameter space, MLS-SMO can be slightly slower than LS-SMO, but the performance of the models obtained is in general quite poor, and thus not of interest. Usually, it is inside the cone where the test error is low, so MLS-SMO accelerates the training phase for the most convenient values, whereas it does not imply a big overhead for other values.

5 Discussion and Further Work

In this work we have adapted a SMO accelerating strategy based on momentum to the case of the LS-SVM model, and shown experimentally that faster running times can be achieved through its use in some settings. This is especially true in those situations where a large C penalty value is to be used, since performance of LS-SMO greatly decays in these settings, as already shown in the past [4], where a Conjugate Gradient method produced significantly better results. Nevertheless, this approach seems to scale worse than SMO as the training set size grows. These observations, together with the results of this work, hint that in some cases of LS-SVM an optimization algorithm can largely benefit from the use of methods employing richer updating directions, as an opposite to SMO's

extremely simple directions. However, one must keep the cost per iteration at bay to ensure scalability; our limited momentum approach guarantees this while producing effective speedups.

As further work we intend to analyze the applicability in the context of the LS-SVM of recent optimization methods that make use of cheap yet theoretically strong updating directions, such as the FISTA [3] algorithm. Also, and realizing how our momentum approach presents interesting results for LS-SVM as well as for the SVM classification model, we intend to extend this method to other SVM models as well, such as ν -SVMs.

References

1. Barbero, Á., Dorronsoro, J.: Momentum Sequential Minimal Optimization: an Accelerated Method for Support Vector Machine Training. Submitted to IJCNN
2. Barbero, Á., López, J., Dorronsoro, J.R.: Cycle-breaking Acceleration of SVM Training. *Neurocomputing* 72(7-9), 1398–1406 (2009)
3. Beck, A., Teboulle, M.: A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM J. Imaging Sciences* 2(1), 183–202 (2009)
4. Chu, W., Ong, C., Keerthi, S.: An Improved Conjugate Gradient Scheme to the Solution of Least Squares SVM. *Neural Networks* 16(2), 498–501 (2005)
5. Chang, C.C., Lin, C.J.: LIBSVM Regression Datasets Repository, <http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/regression.html>, datasets available at <http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/regression.html>
6. De Brabanter, K., Karsmakers, P., Ojeda, F., Alzate, C., Brabanter, J.D., Pelckmans, K., Moor, B.D., Vandewalle, J., Suykens, J.: LS-SVMlab Toolbox User's Guide version 1.7. Tech. Rep. 10-146, Katholieke Universiteit Leuven (2010), software available at <http://www.esat.kuleuven.be/sista/lssvm>
7. Fan, R.-E., Chen, P.-H., Lin, C.-J.: Working Set Selection using Second Order Information for Training Support Vector Machines. *Journal of Machine Learning Research* 6, 1889–1918 (2005)
8. Keerthi, S., Shevade, S.: SMO Algorithm for Least-Squares SVM Formulations. *Neural Computation* 15(2), 487–507 (2003)
9. López, J., Suykens, J.: First and Second Order SMO Algorithms for LS-SVM classifiers. *Neural Processing Letters* 33(1), 33–44 (2011)
10. Platt, J.: Fast Training of Support Vector Machines using Sequential Minimal Optimization. In: *Advances in Kernel Methods: Support Vector Learning*, pp. 185–208. MIT Press, Cambridge (1999)
11. Rätsch, G.: Benchmark Repository (2000), datasets available at <http://ftp.tuebingen.mpg.de/pub/fml/raetsch-lab/benchmarks/>
12. Suykens, J., Lukas, L., Van Dooren, P., De Moor, B., Vandewalle, J.: Least Squares Support Vector Machine Classifiers: a Large Scale Algorithm. In: *ECCTD 1999: Proceeding of the European Conference on Circuit Theory and Design*, Stresa, Italy, pp. 839–842 (1999)
13. Suykens, J., Vandewalle, J.: Least Squares Support Vector Machine Classifiers. *Neural Processing Letters* 9(3), 293–300 (1999)

Fast Support Vector Training by Newton's Method

Shigeo Abe

Kobe University
Rokkodai, Nada, Kobe, Japan
`abe@kobe-u.ac.jp`
<http://www2.kobe-u.ac.jp/~abe>

Abstract. We discuss a fast training method of support vector machines using Newton's method combined with fixed-size chunking. To speed up training, we limit the number of upper or lower bounded variables in the working set to two so that the corrections of the variables do not violate the bounding conditions. If similar working sets occur alternately, we merge these two working sets into one, and if similar working sets occur consecutively, we use incremental Cholesky factorization in calculating corrections. By computer experiments, we show that the proposed method is comparable to or faster than SMO (Sequential minimum optimization) using the second order information.

1 Introduction

Support vector machines (SVMs), formulated in a quadratic programming program, are usually trained in dual form with the number of variables being the number of training data. Thus, to speed up training for a large data set, many training methods have been developed [1]. The widely used training methods use decomposition techniques with variable-size chunking or fixed-size chunking. The most well known method is sequential minimum optimization (SMO) [2], which uses the fixed-size chunking with the working set size of two. To further improve convergence, quadratic information is used [3]. In [4], if a sequence of selected violating variable pairs forms a loop, corrections are made by the combined descent direction. But the problem of SMO is that convergence becomes slow when the margin parameter value becomes large. To improve convergence more than two variables are optimized at a time [1].

In this paper we discuss a fast training method using Newton's method with fixed-size chunking. By Newton's method, the solutions may not be updated if some of the variables in the working set are at the upper or lower bound and the corrections violate the constraints. To make this does not happen frequently, we limit the number of upper or lower bounded variables in the working set to two. And we check the working set sequence in the iteration steps and if similar working sets occur alternately, we combine these working set into one. If similar working sets occur consecutively, we use incremental Cholesky factorization between consecutive steps to reduce factorization time. To limit candidates

for working set, we use incremental training, in which the working set variables are selected from the newly added data and the support vectors. Using several multiclass benchmark data sets, we evaluate the effectiveness of our proposed method.

In Section 2, we briefly summarize SVMs, and in Section 3 we discuss the proposed training method. And in Section 4, we compare the proposed method with SMO.

2 Support Vector Machines

A support vector machine for two-class problems is trained by solving the following optimization problem:

$$\text{maximize} \quad Q(\alpha) = \sum_{i=1}^M \alpha_i - \frac{1}{2} \sum_{i,j=1}^M \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \quad (1)$$

$$\text{subject to} \quad \sum_{i=1}^M y_i \alpha_i = 0, \quad 0 \leq \alpha_i \leq C \quad \text{for } i = 1, \dots, M, \quad (2)$$

where (\mathbf{x}_i, y_i) ($i = 1, \dots, M$) are M training input-output pairs, with $y_i = 1$ if \mathbf{x}_i belongs to Class 1, and $y_i = -1$ if Class 2, C is the margin parameter that determines the tradeoff between the maximization of the margin and minimization of the classification errors, α_i are Lagrange multipliers associated with \mathbf{x}_i , and $K(\mathbf{x}, \mathbf{x}')$ is a kernel function. We use polynomial kernels with degree d : $K(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T \mathbf{x}' + 1)^d$ and radial basis function (RBF) kernels: $K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2)$, where d is a positive integer and γ is a positive parameter for slope control.

The KKT complementarity conditions are given by

$$\alpha_i \left(\sum_{j=1}^M y_i y_j \alpha_j K(\mathbf{x}_i, \mathbf{x}_j) + y_i b - 1 + \xi_i \right) = 0 \quad \text{for } i = 1, \dots, M, \quad (3)$$

$$(C - \alpha_i) \xi_i = 0 \quad \alpha_i \geq 0, \quad \xi_i \geq 0 \quad \text{for } i = 1, \dots, M, \quad (4)$$

where b is the bias term in the decision function and ξ_i are non-negative slack variables associated with α_i . For the solution of (1) and (2), if $\alpha_i > 0$, \mathbf{x}_i are called support vectors; especially if $\alpha_i = C$, bounded support vectors and if $0 < \alpha_i < C$, unbounded support vectors.

To avoid using the bias term we use exact KKT conditions proposed in [5]. We rewrite (3) as follows:

$$\alpha_i (y_i b - y_i F_i + \xi_i) = 0, \quad (5)$$

where

$$F_i = y_i - \sum_{j=1}^M y_j \alpha_j K(\mathbf{x}_i, \mathbf{x}_j). \quad (6)$$

Then the KKT conditions are simplified as follows:

$$\bar{F}_i \geq b \geq \tilde{F}_i \quad \text{for } i = 1, \dots, M, \quad (7)$$

where

$$\tilde{F}_i = F_i \quad \text{if } (y_i = 1, \alpha_i = 0), \quad 0 < \alpha_i < C \quad \text{or} \quad (y_i = -1, \alpha_i = C), \quad (8)$$

$$\tilde{F}_i = F_i \quad \text{if } (y_i = -1, \alpha_i = 0), \quad 0 < \alpha_i < C \quad \text{or} \quad (y_i = 1, \alpha_i = C). \quad (9)$$

3 Training Methods

High-speed training can be achieved if violating variables do not increase very much as training proceeds. To realize this, we use a nested double loop; in the inner loop training is done incrementally and in the outer loop, training is done using all the training data. Let the index set $\{1, \dots, M\}$ be divided into k non-overlapping subsets V^i ($i = 1, \dots, k$). Then, at the i th stage of the inner loop, we calculate the violating index set W^i for $\{\alpha_j \mid j \in V^i \cup S\}$, where S is the set of unbounded support vector indices. We train the SVM until W^i is empty deleting the indices of zero variables that satisfy the KKT conditions from $V^i \cup S$. When training of the SVM using V^k is finished we move to the outer loop and train the SVM using all the training data. In the outer loop, we do not delete indices of zero variables that satisfy the KKT conditions.

For $V \cup S$, the violating index set W is calculated by

$$W = \{i \mid b_{\text{up}} < \tilde{F}_i - \tau \quad \text{or} \quad b_{\text{low}} > \tilde{F}_i + \tau \quad \text{for } i \in V \cup S\}, \quad (10)$$

where V is either V_i or $\{1, \dots, M\}$, τ is a tolerance of convergence, $b_{\text{low}} = \max_i \tilde{F}_i$, and $b_{\text{up}} = \min_i \tilde{F}_i$.

To check convergence at each step we need to calculate F_i . If F_i was calculated at the previous iteration step, it can be calculated either from scratch or incrementally. We calculate incrementally if the current working set size is smaller than the number of support vectors.

3.1 Calculating Corrections by Newton's Method

We optimize the variables α_i ($i \in W$) fixing α_i ($i \in N$), where $W \cup N = \{1, \dots, M\}$, and $W \cap N = \emptyset$, by

$$\begin{aligned} \text{maximize} \quad Q(\boldsymbol{\alpha}_W) = & \sum_{i \in W} \alpha_i - \frac{1}{2} \sum_{i, j \in W} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \\ & - \sum_{\substack{i \in W, \\ j \in N}} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \end{aligned} \quad (11)$$

$$\text{subject to} \quad \sum_{i \in W} y_i \alpha_i = - \sum_{i \in N} y_i \alpha_i, \quad 0 \leq \alpha_i \leq C \quad \text{for } i \in W, \quad (12)$$

where $\boldsymbol{\alpha}_W = (\dots, \alpha_i, \dots)^\top$, $i \in W$.

Solving the equality in (12) for α_s ($s \in W$), we obtain

$$\alpha_s = - \sum_{i \neq s, i=1}^M y_s y_i \alpha_i. \quad (13)$$

Substituting (13) into (11), we eliminate the equality constraint. Let $\boldsymbol{\alpha}_{W'} = (\dots, \alpha_i, \dots)^\top$ ($i \neq s, i \in W$). Then, neglecting the bounds, $\Delta Q(\boldsymbol{\alpha}_{W'})$ has the maximum at

$$\Delta \boldsymbol{\alpha}_{W'} = A^{-1} \mathbf{b}, \quad (14)$$

where

$$A = -\frac{\partial^2 Q(\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}_{W'}^2}, \quad \mathbf{b} = \frac{\partial Q(\boldsymbol{\alpha}_{W'})}{\partial \boldsymbol{\alpha}_{W'}}, \quad \frac{\partial Q(\boldsymbol{\alpha}_{W'})}{\partial \alpha_i} = y_i (F_i - F_s), \quad (15)$$

$$\frac{\partial^2 Q(\boldsymbol{\alpha}_{W'})}{\partial \alpha_i \partial \alpha_j} = y_i y_j (-K(\mathbf{x}_i, \mathbf{x}_j) + K(\mathbf{x}_i, \mathbf{x}_s) + K(\mathbf{x}_s, \mathbf{x}_j) - K(\mathbf{x}_s, \mathbf{x}_s)). \quad (16)$$

Here, we assume that A is positive definite.

Then from (12) and (14), we obtain the correction of α_s :

$$\Delta \alpha_s = - \sum_{i \in W'} y_s y_i \Delta \alpha_i. \quad (17)$$

For α_i ($i \in W$), if

$$\alpha_i = 0, \quad \Delta \alpha_i < 0 \quad \text{or} \quad \alpha_i = C, \quad \Delta \alpha_i > 0, \quad (18)$$

we delete these variables from the working set and repeat the procedure for the reduced working set. If (18) does not hold for any variable in the working set, we modify the variables by

$$\boldsymbol{\alpha}_W^{\text{new}} = \boldsymbol{\alpha}_W^{\text{old}} + r \Delta \boldsymbol{\alpha}_W, \quad (19)$$

where r ($0 < r \leq 1$) is determined so that the variables do not violate the upper or lower bound.

3.2 Working Set Selection

For the SMO, if the variable pair associated with the minimum \bar{F}_i and maximum \tilde{F}_i are selected, the convergence to the optimal solution is guaranteed. Thus, for more than two variables, it is natural to use the working set W given by (10) and if $|W|$ exceeds the working set size, we delete variables from the least violating variables in order. By this working set selection, if all the variables associated with the indices in W are not bounded, i.e., $C > \alpha_i > 0$, we can obtain the corrections that can improve the solution. But if bounded variables are included, there may be cases where we cannot correct the solution. Let $W = W_{\max} \cup W_{\min}$, where W_{\max} and W_{\min} include indices associated with \tilde{F}_i and \bar{F}_i , respectively.

If W_{\max} or W_{\min} includes more than one index whose associated variables are bounded, there is a possibility that the solution cannot be corrected. We call the index associated with the bounded variable bounded index. We order the elements of W_{\max} and W_{\min} in the descending and ascending orders of violation, respectively. Then, for each W_{\max} and W_{\min} , we limit the number of bounded indices to one. Namely, if W_{\max} or W_{\min} includes more than one bounded index, we delete the bounded indices in the lower orders, leaving the violating index in the highest order.

3.3 Calculating Corrections by Cholesky Factorization

We use the Cholesky factorization in calculating (14). We merge W_{\max} and W_{\min} into W , first $\arg \max \tilde{F}_i$ then $\arg \min \tilde{F}_i$, and then if there are bounded indices, we put these at the end of W . We set $s = \arg \max \tilde{F}_i$. Because A is positive semi-definite, A may be singular. In such a case, during factorization of A , if a diagonal element of the lower triangular matrix is smaller than the prescribed value, we discard the associated column and row and proceeds to the next element.

If (18) is satisfied they are either the first or second element, or the last element or the second last element of W . If the first or the second element is not corrected, we correct variables associated with the first and the second elements. This is the same as SMO. If the last or the second last element is not corrected, we delete this and recalculate (14). Because the factorized matrix for the reduced working set is the sub-matrix of the originally factorized matrix, this process is sped up by storing the original factorized matrix.

3.4 Detection of Two-Cycle and One-Cycle Loops

According to [4], cyclic loops, which consist of the same sequences of violating pairs, occur by SMO. To accelerate convergence in such a situation, we consider detecting two-cycle and one-cycle loops. Here, we consider a two-cycle loop occurs if the first two elements of W appear every two iteration steps, and a one-cycle loop every iteration step. If a two-cycle loop occurs, we merge the associated working sets into one, setting the first two elements of the first working set in the loop to the first two elements in the merged working set, and limiting the bounded index for each \tilde{F}_i and \bar{F}_i to one. If a one-cycle loop occurs, and if more than first three elements of the consecutive working sets are the same, we restart Cholesky factorization from the element that is different from the previous one. For this, we save the factorized matrix between iteration steps.

4 Performance Evaluation

We evaluated performance of the proposed training method using the benchmark data sets: thyroid, blood-cell, hiragana-50, hiragana-13, hiragana-105, satimage, and USPS data sets used in [6]. We used fuzzy pairwise SVMs [1] and measured the training time using a personal computer (3GHz, 2GB memory, Windows XP operating system). We prepared a cache memory with the size equal to the kernel matrix.

Table 1. Training time (s) for different working set sizes

Data	C	W_2	W_{10}	W_{50}	W_{100}	W_{200}
	1	0.812 (2)	1.64 (4.6)	2.13 (5.4)	2.44 (5.8)	2.89 (5.9)
Blood cell	10^3	1.14 (2)	0.531 (6.4)	0.593 (8.3)	0.640 (8.5)	0.593 (8.7)
	10^6	204 (2)	19.5 (6.8)	0.562 (8.7)	0.625 (9.1)	0.546 (9.6)
USPS	1	6.67 (2)	8.67 (6.7)	26.0 (26)	35.0 (31)	40.8 (34)
	10^3	5.22 (2)	5.41 (9.1)	5.58 (23)	5.94 (30)	6.89 (38)

Table 2. Training time (s) for addition of training data

Data	C	All	V_{10}	V_{50}	V_{100}	V_{200}
	1	2.89 (5.9)	1.00 (2.5)	1.02 (3.0)	1.19 (3.7)	1.61 (4.7)
Blood cell	10^3	0.593 (8.7)	0.296 (6.7)	0.281 (7.4)	0.281 (7.8)	0.343 (8.0)
	10^6	0.546 (9.6)	0.265 (7.6)	0.296 (8.4)	0.281 (8.9)	0.312 (9.0)
USPS	1	40.8 (34)	7.80 (9.5)	8.36 (13)	8.84 (16)	9.53 (20)
	10^3	6.89 (38)	6.84 (44)	6.84 (44)	6.30 (50)	6.47 (52)

Table 3. Effect of working set optimization

Data	C	Opt	Without
	1	1.0 (2.5)	8.86 (2.8)
Blood cell	10^3	0.296 (6.7)	1.34 (5.9)
	10^6	0.265 (7.6)	1.08 (6.6)
USPS	1	7.80 (9.5)	77.4 (6.5)
	10^3	6.8 (44)	7.16 (40)

To determine the working set size and the number of added data we used the blood cell and USPS data sets. Table 1 lists the training time for different working set sizes and the C values selected from 1, 10^3 , and 10^6 . We trained the SVM in a batch mode. The number in parentheses is the average working set size. The subscript of W is the working set size. We do not include the result of USPS for $C = 10^6$, because it is the same with that of 10^3 . The shortest time in the row is shown in boldface. For the blood cell data with $C = 10^6$, training time decreases as the working set size is increased. But the reverse is true for the USPS data set with $C = 1$. In the following study, unless otherwise stated, we fixed the working set size to 200.

Fixing the working set size to 200, we evaluated incremental training. Table 2 shows the results, where “All” means batch training and the subscript of V is the number of data added and the number in parentheses is the average working set size. From the table, incremental training is faster than batch training and addition of 10 is relatively good for all cases. Thus, in the following we fix the data addition size to 10.

We compared the effect of working set size optimization. Table 3 shows the results, where “Opt” and “Without” denote with and without working set optimization, respectively. The number in parentheses shows the average working set size. By optimization, usually the average working set size increases and training time is reduced. This is especially evident for the USPS data set with $C = 1$.

Table 4. Effect of two- and one-loop optimizations

Data	C	Opt	Two-Cycle Loop		One-Cycle Loop		
			Time (s)	Time (s)	Loops	Time (s)	Restart Size
Blood cell	1	0.890	0.906	4	0.921	180	4.6
	10^3	0.265	0.281	109	0.281	524	5.2
	10^6	9.23	45.3	31,699	9.27	510	5.5
USPS	1	6.28	6.30	123	6.28	729	4.7
	10^3	13.9	15.2	14,365	13.9	529	5.6

Table 5. Training time comparison (s)

Data	C	RBF			Polynomial		
		W_{10}	W_{200}	LIBSVM	W_{10}	W_{200}	LIBSVM
Blood cell	1	0.906	<i>1.00</i>	0.625	0.265	0.312	<i>0.359</i>
	10^3	0.250	0.296	<i>0.312</i>	<i>2.30</i>	0.265	0.234
	10^6	9.28	0.265	2.52	<i>13.0</i>	0.265	5.09
	10^9	34.4	0.281	27.8	13.0	0.265	62.3
Thyroid	1	1.09	6.78	0.375	2.91	<i>6.80</i>	0.312
	10^3	<i>3.70</i>	1.13	0.375	<i>1.38</i>	1.11	0.312
	10^6	4.63	0.312	2.73	1.08	0.296	<i>4.16</i>
	10^9	2.75	0.296	<i>13.5</i>	1.06	0.296	12.8
H50	1	2.70	<i>4.33</i>	2.53	1.17	0.921	1.69
	10^3	1.11	0.984	1.11	<i>1.17</i>	0.921	0.968
H13	1	6.11	<i>6.61</i>	3.78	0.781	0.765	<i>1.94</i>
	10^3	<i>0.921</i>	0.812	0.890	0.796	0.781	<i>0.812</i>
	10^6	<i>0.921</i>	0.828	0.906	—	—	—
H105	1	6.23	6.98	<i>7.80</i>	12.9	<i>13.3</i>	5.17
	10^3	<i>2.86</i>	2.28	2.81	<i>2.97</i>	2.42	2.55
	10^6	2.75	2.28	<i>2.86</i>	3.19	2.41	2.73
Satimage	1	1.02	<i>1.16</i>	0.859	10.5	4.31	0.703
	10^3	<i>11.1</i>	5.06	1.02	49.2	3.25	1.16
	10^6	<i>290</i>	3.53	10.8	49.2	3.25	14.7
USPS	1	6.25 (5.70)	7.80 (13.2)	<i>8.63</i>	7.95 (6.48)	<i>16.5</i> (16.3)	5.94
	10^3	13.9 (5.64)	6.84 (9.97)	5.50	18.6 (6.48)	<i>30.9</i> (15.1)	5.14
Ranking		3:8:10	10:6:5	8:8:5	1:10:9	11:5:4	8:5:7

Table 4 shows the results of 2-cycle and 1-cycle optimization for W_{10} (There was not much improvement for W_{200}). In the table, “Opt,” “Two-Cycle Loop,” and “One-Cycle Loop” denote that both optimization methods are used, two-cycle loop optimization is not used, and one-loop optimization is not used, respectively. The columns “Loops,” “Restarts,” “Size” denote the number of two-cycle loops detected, the number of restarts of Cholesky factorization, and the average size of the common sequences, respectively. Especially for the blood cell data with $C = 10^6$, the effect of two-cycle loop optimization is evident, but the effect of the one-cycle loop is not so prominent especially for the USPS data set.

Table 5 shows the training time of the proposed method with working set sizes of 10 and 200 and LIBSVM [7], which uses the second order information. We used RBF kernels with $\gamma = 1$ and polynomial kernels with degree 3. We set the same τ value of 0.001 for both methods. The shortest and longest training times are shown in bold and italic, respectively. For the USPS data set we show the training time without incremental training in parentheses. Ranking shows the numbers of shortest, middle, and longest training times among 21 cases. The proposed method with the working set size of 200 performed best and the

LIBSVM the second best. From the table, the proposed method works well if we use a small working set size for a small C , and a large size for a large C . For the USPS data set, incremental training was slower than without incremental training except for RBF kernels and W_{200} .

5 Conclusions

We discussed some methods for accelerating the convergence of SVM training by Newton's method; optimization of working set selection by deleting bounded variables, merging of consecutive working sets for two-cycle loops, and incremental Cholesky factorization for one-cycle loops. The computer experiment showed performance improvement by the proposed method and the proposed method showed comparable or faster training than LIBSVM.

References

1. Abe, S.: Support Vector Machines for Pattern Classification. Springer, Heidelberg (2010)
2. Platt, J.C.: Fast training of support vector machines using sequential minimal Optimization. In: Advances in Kernel Methods: Support Vector Learning, pp. 185–208. MIT Press, Cambridge (1999)
3. Fan, R.-E., Chen, P.-H., Lin, C.-J.: Working set selection using second order information for training support vector machines. *Journal of Machine Learning Research* 6, 1889–1918 (2005)
4. Barbero, Á., Dorronsoro, J.R.: Faster directions for second order SMO. In: Diamantaras, K., Duch, W., Iliadis, L.S. (eds.) ICANN 2010. LNCS, vol. 6353, pp. 30–39. Springer, Heidelberg (2010)
5. Keerthi, S.S., Gilbert, E.G.: Convergence of a generalized SMO algorithm for SVM classifier Design. *Machine Learning* 46(1-3), 351–360 (2002)
6. Abe, S.: Is primal better than dual. In: Alippi, C., Polycarpou, M., Panayiotou, C., Ellinas, G. (eds.) ICANN 2009. LNCS, vol. 5768, pp. 854–863. Springer, Heidelberg (2009)
7. Chang, C.-C., Lin, C.-J.: LIBSVM–A library for support vector machines, <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>

Linear Operators and Stochastic Partial Differential Equations in Gaussian Process Regression

Simo Särkkä

Department of Biomedical Engineering and Computational Science
Aalto University, Finland
`simo.sarkka@tkk.fi`

Abstract. In this paper we shall discuss an extension to Gaussian process (GP) regression models, where the measurements are modeled as linear functionals of the underlying GP and the estimation objective is a general linear operator of the process. We shall show how this framework can be used for modeling physical processes involved in measurement of the GP and for encoding physical prior information into regression models in form of stochastic partial differential equations (SPDE). We shall also illustrate the practical applicability of the theory in a simulated application.

Keywords: Gaussian process regression, linear operator, stochastic partial differential equation, inverse problem.

1 Introduction

This paper is concerned with Gaussian process (GP) regression [1,2], which refers to a Bayesian machine learning approach, where the regression functions are modeled non-parametrically as Gaussian processes. In the approach, we first postulate a Gaussian process prior for the unknown function $f(\mathbf{x})$ and then compute the posterior distribution of the function by conditioning to observed measurements $D = \{(\mathbf{x}_i, \mathbf{y}_i) : i = 1, \dots, n\}$. The predicted function values at arbitrary points \mathbf{x}^* are then estimated by computing the posterior predictive distributions of the unknown function values.

Linear operators and functionals, which act on functions in Hilbert spaces [3] are commonly used models in engineering and physics applications. In telecommunications [4], statistical signal processing [5] and image processing [6] the linear operators are typically integral operators in the form of convolution kernels, which model the effect of a linear system to the input or measurement signal. In physics [7,8] the linear operators are typically differential operators, which appear as parts of partial differential equation (PDE) models of the Nature. When PDE models include stochastic terms, they become stochastic partial differential equations (SPDE) [9], which can be used for modeling spatial physical processes with unknown sub-phenomena.

In this paper we shall show how linear operators can be naturally included into GP regression and used for encoding physical and other background information into the models. The information can be encoded either by including a linear operator to the measurement model or by forming the prior covariance function as a solution to a stochastic partial differential equation (SPDE). Similar ideas have been previously used, for example, in Kriging [10,11,12,13,14], image processing [15,16,6], Kalman filtering [17,18], and physical inverse problems [19,20,21]. In the machine learning context, the inclusion of linear operators into GP regression models has also been previously proposed. For example, convolved multi-output Gaussian processes [22,23], latent force models [24], noisy operator equations [25,26], as well as measurement and estimation of derivatives and integrals [27,28] fall into this class of models. In this paper we shall formulate a general model, which includes all of these models and present its solution.

2 Linear Operators and Functionals of GPs

In this article we shall denote the application of linear operator \mathcal{L}_x to the function $\mathbf{f}(\mathbf{x})$, which results in another function $\mathbf{g}(\mathbf{x})$ as $\mathbf{g}(\mathbf{x}) = \mathcal{L}_x \mathbf{f}(\mathbf{x})$. We shall assume that the operator produces functions with the same input domain as its input function has – if the input function is $\mathbf{f} : \mathbb{R}^d \mapsto \mathbb{R}^{m_f}$ then the output function is $\mathbf{g} : \mathbb{R}^d \mapsto \mathbb{R}^{m_g}$. That is, the output dimensionality might change, but input not. We shall explicitly allow vector and matrix valued operators, because scalar models are too restricted for most modeling purposes. Examples of allowed linear operators are, for example integration $\mathcal{L}_x \mathbf{f}(\mathbf{x}) = \int_{-\infty}^{x_i} \mathbf{f}(\mathbf{x}) dx_i$, and computation of the Jacobian matrix $[\mathcal{L}_x \mathbf{f}(\mathbf{x})]_{ij} = \partial f_i(\mathbf{x}) / \partial x_j$.

We shall also need linear functionals \mathcal{H}_x , which are just like linear operators, but produce vectors or matrices instead of functions. Functionals can be considered as operators combined with evaluation of the result function at certain point. For example the derivative $\mathcal{H}_x \mathbf{f}(\mathbf{x}) = \partial \mathbf{f}(\mathbf{x}) / \partial x_i$ evaluated at fixed point $\hat{\mathbf{x}}$ is a functional as well as integral of $\mathbf{f}(\mathbf{x})$ over a fixed area. The mathematical treatment of functionals is similar to operators, because functionals can be considered as a restricted class of operators.

In this article we are interested in the case where $\mathbf{f}(\mathbf{x})$ is modeled as a zero mean Gaussian process with covariance function $E[\mathbf{f}(\mathbf{x}) \mathbf{f}^T(\mathbf{x}')] = \mathbf{K}_{ff}(\mathbf{x}, \mathbf{x}')$, which is denoted as

$$\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_{ff}(\mathbf{x}, \mathbf{x}')). \quad (1)$$

By applying the well known rules for linear transformations of Gaussian processes [29,2], we get that if $\mathbf{g}(\mathbf{x}) = \mathcal{L}_x \mathbf{f}(\mathbf{x})$, then we have

$$\begin{aligned} \mathbf{K}_{gf}(\mathbf{x}, \mathbf{x}') &= \mathcal{L}_x \mathbf{K}_{ff}(\mathbf{x}, \mathbf{x}') \\ \mathbf{K}_{fg}(\mathbf{x}, \mathbf{x}') &= \mathbf{K}_{ff}(\mathbf{x}, \mathbf{x}') \mathcal{L}_{x'}^T \\ \mathbf{K}_{gg}(\mathbf{x}, \mathbf{x}') &= \mathcal{L}_x \mathbf{K}_{ff}(\mathbf{x}, \mathbf{x}') \mathcal{L}_{x'}^T. \end{aligned} \quad (2)$$

Here one has to be careful with the notation – the transpose is just a normal matrix transpose, but by operator application from the right we mean an analogous operation for kernels as right multiplication by matrix in linear algebra

is. The multiplication from right here means operating to the second argument of the kernel $\mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}')$ and this is here emphasized by writing $\mathcal{L}_{x'}$ instead of \mathcal{L}_x . This convention is required to be able to consistently cope with vector and matrix operators.

To illustrate the right multiplication, we may consider the operator $\mathcal{L}_x = (1, \partial/\partial x)$ acting on scalar Gaussian process $f(x)$ with scalar input and covariance function $k_{\text{ff}}(x, x')$. The cross covariance $\mathbf{K}_{\text{fg}}(x, x')$ is then given as

$$\mathbf{K}_{\text{fg}}(x, x') = k_{\text{ff}}(\mathbf{x}, \mathbf{x}') (1 \ \partial/\partial x') = (k_{\text{ff}}(\mathbf{x}, \mathbf{x}') \ \partial k_{\text{ff}}(\mathbf{x}, \mathbf{x}')/\partial x') . \quad (3)$$

The rules for computation with functionals are analogous to the rules for operators, but of course, the result of applying functional to the function, say, $\mathbf{h} = \mathcal{H}_x \mathbf{f}(\mathbf{x})$ is a random variable having the covariance matrix

$$\mathbf{K}_{\mathbf{h}\mathbf{h}} = \mathcal{H}_x \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}') \mathcal{H}_{x'}^T, \quad (4)$$

not a covariance function.

3 GP Regression Model with Linear Operators

If we model the unknown regression function $\mathbf{f} : \mathbb{R}^d \mapsto \mathbb{R}^m$ as a zero mean Gaussian process with covariance function $\mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}')$, then a basic multi-input multi-output Gaussian process regression model can be stated as

$$\begin{aligned} \mathbf{f}(\mathbf{x}) &\sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}')) \\ \mathbf{y}_i &= \mathbf{f}(\mathbf{x}_i) + \mathbf{e}_i, \end{aligned} \quad (5)$$

where $\mathbf{y}_i, \mathbf{e}_i \in \mathbb{R}^m$. The joint covariance of errors $\mathbf{e} = (\mathbf{e}_1, \dots, \mathbf{e}_n)$ is assumed to be given by the matrix Σ . The equations for the posterior mean and covariance given the concatenated vector of measurements $\mathbf{y} = (\mathbf{y}_1, \dots, \mathbf{y}_n)$ are then given by the well-known GP regression equations (see, e.g., [1,2]):

$$\begin{aligned} \mathbb{E}[\mathbf{f}(\mathbf{x}) | \mathbf{y}] &= \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}_{1:n}) [\mathbf{K}_{\text{ff}}(\mathbf{x}_{1:n}, \mathbf{x}_{1:n}) + \Sigma]^{-1} \mathbf{y} \\ \text{Cov}[\mathbf{f}(\mathbf{x}) | \mathbf{y}] &= \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}) - \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}_{1:n}) [\mathbf{K}_{\text{ff}}(\mathbf{x}_{1:n}, \mathbf{x}_{1:n}) + \Sigma]^{-1} \mathbf{K}_{\text{ff}}^T(\mathbf{x}, \mathbf{x}_{1:n}), \end{aligned} \quad (6)$$

where $\mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}_{1:n})$ is a block matrix formed of blocks $\mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}_j)$ where $j = 1, \dots, n$ and $\mathbf{K}_{\text{ff}}(\mathbf{x}_{1:n}, \mathbf{x}_{1:n})$ is a block matrix with blocks $\mathbf{K}_{\text{ff}}(\mathbf{x}_i, \mathbf{x}_j)$ where $i = 1, \dots, n$ and $j = 1, \dots, n$.

In image processing [6] and physical inverse problems [19,20] the Gaussian process based models are often stated in more general form than Equation (5). In order to model linear processes such as motion blur or physical phenomena occurring in the measurement process, the measurements are assumed to be linear functionals of the underlying Gaussian process, not direct (noisy) values of the process. Furthermore, the estimation objective often is not the underlying Gaussian process, but some linear operator transformation of it, such as its

derivative or integral. With this motivation, a suitably generalized Gaussian process regression model can be stated as

$$\begin{aligned}\mathbf{f}(\mathbf{x}) &\sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}')) \\ \mathbf{y}_i &= \mathcal{H}_{x,i} \mathbf{f}(\mathbf{x}) + \mathbf{e}_i,\end{aligned}\tag{7}$$

where $\mathcal{H}_{x,i}$ is a deterministic linear functional and the objective is to estimate a linear operator transformation of the signal $\mathbf{d}(\mathbf{x}) = \mathcal{L}_x \mathbf{f}(\mathbf{x})$, where \mathcal{L}_x is a linear operator. As in the basic model in Equation (5), $\{\mathbf{y}_i : i = 1, \dots, n\}$ are the m -dimensional measurements and the errors $\mathbf{e} = (\mathbf{e}_1, \dots, \mathbf{e}_n)$ have zero mean and joint covariance matrix Σ .

With these assumptions the posterior process $\mathbf{d}(\mathbf{x})$, after conditioning to measurements, will be a Gaussian process and the mean and covariance can be derived as follows. The joint distribution of $\mathbf{y} = (\mathbf{y}_1, \dots, \mathbf{y}_n)$ and $\mathbf{d}(\mathbf{x})$ is multi-dimensional Gaussian with zero mean and covariance

$$\text{Cov} \left[\begin{pmatrix} \mathbf{y} \\ \mathbf{d}(\mathbf{x}) \end{pmatrix} \right] = \begin{pmatrix} \mathcal{H}_x \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}') \mathcal{H}_{x'}^T + \Sigma & \mathcal{H}_x \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}') \mathcal{L}_{x'}^T \\ \mathcal{L}_x \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}') \mathcal{H}_{x'}^T & \mathcal{L}_x \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}') \mathcal{L}_{x'}^T \end{pmatrix} \Big|_{\mathbf{x}'=\mathbf{x}}, \tag{8}$$

where $\mathcal{H}_x = (\mathcal{H}_{x,1}, \dots, \mathcal{H}_{x,n})$. The substitution $\mathbf{x}' = \mathbf{x}$ after performing all the operations is needed just to get the notation consistent, that is, to make it clear that to which variables do the operators operate to. By the elementary rules for Gaussian random variables we can compute the conditional mean and covariance of $\mathbf{d}(\mathbf{x})$:

$$\begin{aligned}\mathbb{E}[\mathbf{d}(\mathbf{x}) | \mathbf{y}] &= \mathcal{L}_x \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}') \mathcal{H}_{x'}^T [\mathcal{H}_x \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}') \mathcal{H}_{x'}^T + \Sigma]^{-1} \mathbf{y} \\ \text{Cov}[\mathbf{d}(\mathbf{x}) | \mathbf{y}] &= \mathcal{L}_x \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}') \mathcal{L}_{x'}^T \\ &\quad - \mathcal{L}_x \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}') \mathcal{H}_{x'}^T [\mathcal{H}_x \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}') \mathcal{H}_{x'}^T + \Sigma]^{-1} \mathcal{H}_x \mathbf{K}_{\text{ff}}(\mathbf{x}, \mathbf{x}') \mathcal{L}_{x'}^T \Big|_{\mathbf{x}'=\mathbf{x}},\end{aligned}\tag{9}$$

where the substitution $\mathbf{x}' = \mathbf{x}$ is understood to be done for all instances of \mathbf{x}' after all the operations in the expression have been performed.

4 Stochastic Partial Differential Equations

Linear stochastic partial differential equation (SPDE) is an operator equation of the form

$$\mathcal{D}_x \mathbf{g}(\mathbf{x}) = \mathbf{n}(\mathbf{x}), \tag{10}$$

where \mathcal{D}_x is a linear differential operator and $\mathbf{n}(\mathbf{x})$ is a Gaussian process with zero mean and covariance function $\mathbf{K}_{\text{nn}}(\mathbf{x}, \mathbf{x}')$. Note that often SPDE refers to the special case where $\mathbf{n}(\mathbf{x})$ is a Gaussian white noise process [9], that is, a process with covariance function $\mathbf{K}_{\text{nn}}(\mathbf{x}, \mathbf{x}') = \mathbf{Q} \delta(\mathbf{x} - \mathbf{x}')$, but here we shall define SPDE in the above generalized sense.

Assume that we now want to solve the corresponding inverse problem, that is, estimate $\mathbf{g}(\mathbf{x})$ in the Equation (10), based on the measurements of $\mathbf{g}(\mathbf{x})$. If we convert this into Gaussian process regression model, denote the unknown function as $\mathbf{f}(\mathbf{x})$, and include the measurement functional, the general model can be written in form

$$\begin{aligned}\mathbf{f}(\mathbf{x}) &\sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_{ff}(\mathbf{x}, \mathbf{x}')) \\ \mathcal{D}_x \mathbf{g}(\mathbf{x}) &= \mathbf{f}(\mathbf{x}) \\ \mathbf{y}_i &= \mathcal{H}_{x,i} \mathbf{g}(\mathbf{x}) + \mathbf{e}_i.\end{aligned}\tag{11}$$

Unfortunately this model is incompatible with the GP model in Equation (7).

Fortunately, we can convert the model into compatible form by introducing the Green's function $\mathbf{G}(\mathbf{x}, \mathbf{x}')$ of the operator, which can be interpreted as inverse of the operator \mathcal{D}_x . Thus we get the formal solution for $\mathbf{g}(\mathbf{x}) = \int_{\mathcal{X}} \mathbf{G}(\mathbf{x}, \mathbf{x}') \mathbf{f}(\mathbf{x}') d\mathbf{x}'$ and the model can be rewritten as

$$\begin{aligned}\mathbf{f}(\mathbf{x}) &\sim \mathcal{GP}(\mathbf{0}, \mathbf{K}_{ff}(\mathbf{x}, \mathbf{x}')) \\ \mathbf{y}_i &= \mathcal{H}_{x,i} \int_{\mathcal{X}} \mathbf{G}(\mathbf{x}, \mathbf{x}') \mathbf{f}(\mathbf{x}') d\mathbf{x}' + \mathbf{e}_i,\end{aligned}\tag{12}$$

which is compatible with the model (7).

Physical laws such as the laws of electromagnetism are often linear PDEs. When linear PDE is driven by Gaussian process, we get a SPDE, whose solution is a Gaussian process. By suitable formulation of the Gaussian process regression model and the covariance functions is it also possible to encode physical laws into the model.

5 Simulation: Electrostatic Inverse Problem

Assume that we are measuring electrostatic potential $\phi(\mathbf{x})$ of a stationary charge density at some predefined points \mathbf{x}_i . If we denote the unknown charge density as $f(\mathbf{x})$, then from Maxwell's equations (see, e.g., [8]) we know that the potential is the solution to the Poisson equation $\nabla^2 \phi(\mathbf{x}) = -f(\mathbf{x})/\epsilon_0$. If we model the unknown charge density as a Gaussian process, then by using the procedure presented in Section 4 the model can be converted into the form

$$\begin{aligned}f(\mathbf{x}) &\sim \mathcal{GP}(0, k_{ff}(\mathbf{x}, \mathbf{x}')) \\ y_i &= \frac{1}{4\pi\epsilon_0} \int_{\mathbb{R}^3} \frac{f(\mathbf{x}')}{\|\mathbf{x}_i - \mathbf{x}'\|} d\mathbf{x}' + e_i,\end{aligned}\tag{13}$$

which indeed has the form of the extended GP regression model (7) with the measurement model functional defined as

$$\mathcal{H}_{x,i} f(\mathbf{x}) = \frac{1}{4\pi\epsilon_0} \int_{\mathbb{R}^3} \frac{f(\mathbf{x}')}{\|\mathbf{x}_i - \mathbf{x}'\|} d\mathbf{x}'.\tag{14}$$

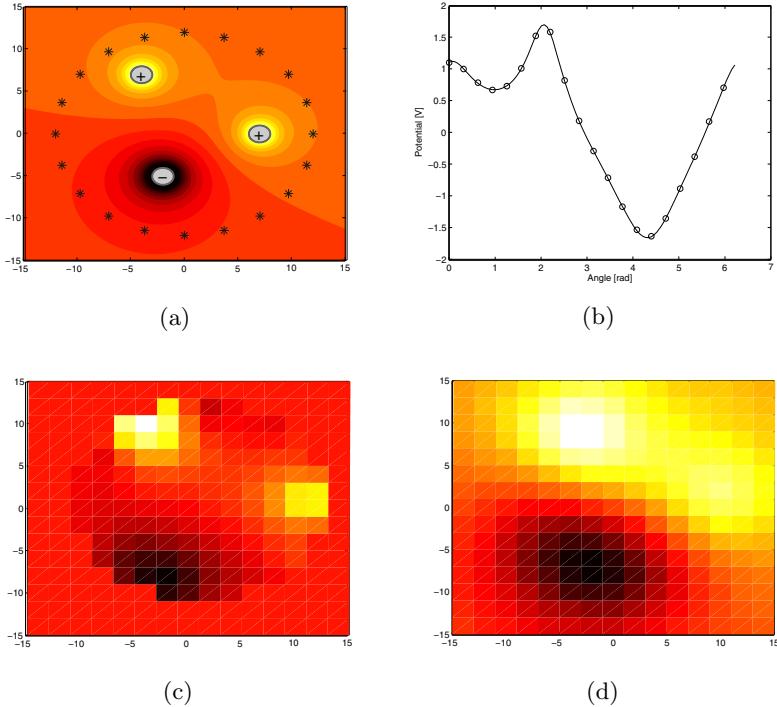


Fig. 1. (a) Charges, potential and measurements points. The measurement points are located at radius $r = 12$ around the origin. (b) Measured point potentials. (c) Estimated charge density. (d) Estimated potential.

In this case we assume that due to the physical setup we know that the charge density is confined into the area of radius $r_0 = 12$ centered at origin and outside that area the charge density is identically zero. If we assume that the prior covariance function of $f(\mathbf{x})$ is a squared exponential inside the specified area, the suitable non-stationary covariance function is

$$k_{ff}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\frac{1}{2l^2} \|\mathbf{x} - \mathbf{x}'\|^2\right) \mathcal{I}(\|\mathbf{x}\| < r_0) \mathcal{I}(\|\mathbf{x}'\| < r_0), \quad (15)$$

where the \mathcal{I} denotes the indicator function, which takes the value 1 when the argument is true, zero otherwise.

In the simulation, three charged objects were placed on the xy plane and electrostatic potential measurements were obtained on circle of radius $r = 12$ around the origin in xy -plane, as shown in Figure 1 (a). The measurements were corrupted by Gaussian noise with standard deviation 0.01. Note that the selection of the finite support covariance function also ensures that the singularities in the integrands disappear, because the measurements are strictly on the area with no charge. The electrostatic potential measurements obtained from the simulated measurement points are shown in Figure 1 (b).

The computation was performed by simple Euler discretization of the integral operator in Equation (14), but more efficient quadrature or cubature integration methods could be applied instead. With this approximation the mean and covariance Equations (9) reduced to matrix expressions for the grid points. The estimated charge density with prior parameter values $\sigma^2 = 10^{-12}$, $l = 2$ is shown in Figure 1 (c) and the corresponding predicted potential is shown in Figure 1 (d). As can be seen in the figures, both the estimated charge density and potential indeed visually resemble the true quantities, and the estimates are quite reasonable given that we have only observed the 20 measurements in Figure 1 (b). However, due to the ambiguities in the physical setup, the prior covariance defines the size and shape of the charge areas and with the current selection the charge areas and the corresponding potentials are always smoother and flatter than the true ones.

6 Conclusion

In this article we have discussed an extended Gaussian process regression model for machine learning problems, which allows modeling of linear functionals in measurement process and estimation of the result of a linear operator applied to the Gaussian process. We have also demonstrated how this extended model can be used for formulating Gaussian process based solutions to physical inverse problems. The practical applicability of the theory was illustrated in a simulation of a classical electrostatic inversion problem.

Acknowledgments. The author is grateful to the Centre of Excellence in Computational Complex Systems Research of Academy of Finland for the financial support and also likes to thank Aki Vehtari and Jouko Lampinen for helpful comments on the manuscript.

References

1. O'Hagan, A.: Curve fitting and optimal design for prediction (with discussion). *Journal of the Royal Statistical Society B* 40(1), 1–42 (1978)
2. Rasmussen, C.E., Williams, C.K.I.: *Gaussian Processes for Machine Learning*. MIT Press, Cambridge (2006)
3. Akhiezer, N.I., Glazman, I.M.: *Theory of Linear Operators in Hilbert Space*. Dover (1993)
4. Carlson, A.B.: *Communication Systems: An Introduction to Signals and Noise in Electrical Communication*, 3rd edn. McGraw-Hill, New York (1986)
5. Hayes, M.H.: *Statistical Digital Signal Processing and Modeling*. John Wiley & Sons, Chichester (1996)
6. Gonzalez, R.C., Woods, R.E.: *Digital Image Processing*. Prentice-Hall, Englewood Cliffs (2002)
7. Guenther, R.B., Lee, J.W.: *Partial Differential Equations of Mathematical Physics and Integral Equations*. Dover, New York (1988)
8. Griffiths, D.J.: *Introduction to Electrodynamics*, 3rd edn. Pearson, London (2008)

9. Holden, H., Øksendal, B., Ubøe, J., Zhang, T. (eds.): *Stochastic Partial Differential Equations: A Modeling, White Noise Functional Approach*. Birkhäuser, Basel (1996)
10. Whittle, P.: On stationary processes in the plane. *Biometrika* 41(3/4), 434–449 (1954)
11. Matérn, B.: Spatial variation. Technical report, Meddelanden från Statens Skogsforskningsinstitut, Band 49 - Nr 5 (1960)
12. Christakos, G.: *Random Field Models in Earth Sciences*. Academic Press, London (1992)
13. Cressie, N.A.C.: *Statistics for Spatial Data*. Wiley, Chichester (1993)
14. Gelfand, A.E., Diggle, P.J., Fuentes, M., Guttorp, P.: *Handbook of Spatial Statistics*. Chapman & Hall/CRC (2010)
15. Jain, A.K.: Partial differential equations and finite-difference methods in image processing, part 1: Image representation. *Journal of Optimization Theory and Applications* 23(1) (1977)
16. Jain, A.K., Jain, J.R.: Partial differential equations and finite difference methods in image processing – part II: Image restoration. *IEEE Transactions on Automatic Control* 23(5) (1978)
17. Curtain, R.: A survey of infinite-dimensional filtering. *SIAM Review* 17(3), 395–411 (1975)
18. Ray, W.H., Lainiotis, D.G.: *Distributed Parameter Systems*. Marcel Dekker, New York (1978)
19. Tarantola, A.: *Inverse Problem Theory and Methods for Model Parameter Estimation*. SIAM, Philadelphia (2004)
20. Kaipio, J., Somersalo, E.: *Statistical and Computational Inverse Problems*. Applied mathematical Sciences, vol. 160. Springer, Heidelberg (2005)
21. Hiltunen, P., Särkkä, S., Nissila, I., Lajunen, A., Lampinen, J.: State space regularization in the nonstationary inverse problem for diffuse optical tomography. *Inverse Problems* 27(2) (2011)
22. Alvarez, M., Lawrence, N.D.: Sparse convolved Gaussian processes for multi-output regression. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) *Advances in Neural Information Processing Systems*, vol. 21, pp. 57–64. The MIT Press, Cambridge (2009)
23. Alvarez, M., Luengo, D., Titsias, M.K., Lawrence, N.D.: Efficient multioutput Gaussian processes through variational inducing kernels. In: Teh, Y.W., Titterington, M. (eds.) *Proceedings of the 13th International Workshop on Artificial Intelligence and Statistics*, pp. 25–32 (2010)
24. Alvarez, M., Lawrence, N.D.: Latent force models. In: van Dyk, D., Welling, M. (eds.) *Proceedings of the 12th International Workshop on Artificial Intelligence and Statistics*, pp. 9–16 (2009)
25. Vapnik, V.: *Statistical Learning Theory*. Wiley, Chichester (1998)
26. Graepel, T.: Solving noisy linear operator equations by Gaussian processes: Application to ordinary and partial differential equations. In: *Proceedings of 20th International Conference on Machine Learning* (2003)
27. Solak, E., Murray-Smith, R., Leithead, W.E., Leith, D., Rasmussen, C.E.: Derivative observations in Gaussian process models of dynamic systems. In: *Advances in Neural Information Processing Systems*, vol. 15, pp. 1033–1040. MIT Press, Cambridge (2003)
28. O'Hagan, A.: Bayes-Hermite quadrature. *Journal of Statistical Planning and Inference* 29, 245–260 (1991)
29. Papoulis, A.: *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill, New York (1984)

Learning from Multiple Annotators with Gaussian Processes

Perry Groot¹, Adriana Birlutiu², and Tom Heskes²

¹ Technical University Eindhoven - Dept. of Electrical Engineering - Control Systems Potentiaal 4.28, 5600 MB Eindhoven - The Netherlands

² Radboud University Nijmegen - Intelligent Systems
Heyendaalseweg 135, 6525 AJ Nijmegen - The Netherlands
pcgroot@gmail.com, {A.Birlutiu,t.heskes}@science.ru.nl

Abstract. In many supervised learning tasks it can be costly or infeasible to obtain objective, reliable labels. We may, however, be able to obtain a large number of subjective, possibly noisy, labels from multiple annotators. Typically, annotators have different levels of expertise (i.e., novice, expert) and there is considerable disagreement among annotators. We present a Gaussian process (GP) approach to regression with multiple labels but no absolute gold standard. The GP framework provides a principled non-parametric framework that can automatically estimate the reliability of individual annotators from data without the need of prior knowledge. Experimental results show that the proposed GP multi-annotator model outperforms models that either average the training data or weigh individually learned single-annotator models.

1 Introduction

In most learning settings a function is learned from inputs to outputs and it is assumed that outputs are available for training. In contrast to this, there are many real-world scenarios in which the true values of the outputs used for training are unknown or very expensive to obtain. Instead, multiple annotators are available to provide subjective, noisy estimates of these outputs. For example, annotators can be radiologists [1,2] who provide a subjective (possibly noisy) opinion about a suspicious region on a medical image as being malignant or benign. The actual gold standard (whether it is cancer or not) can be obtained from biopsies, which is an expensive and invasive procedure. Text and image classification are other learning scenarios where multiple human annotators subjectively assign inputs to some categories [3,4,5,6,7,8]. The amount of noise in the annotators' estimates of the true output can range from very few (annotators which are domain experts) to very much (annotators which are only novice).

Learning with multiple annotators is a special case of supervised learning in which a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ is learned given a training data set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$. It can be considered in various learning settings, i.e., regression, binary and multi-class classification, and ranking. Previous work on learning with multiple annotators has focused on learning f using parametric models [1,2,9]. In

this paper we present a non-parametric approach based on Gaussian processes (GPs) [10]. Gaussian processes provide a rich, principled, and well-established alternative to parametric models. We provide details on predictive equations and hyperparameter optimization for regression with multiple annotators.

The rest of this paper is structured as follows. Section 2 gives background on Gaussian process regression. Section 3 describes regression with multiple annotators. Section 4 presents an experimental evaluation of our approach. Section 5 concludes and discusses some directions for future work.

2 Gaussian Process Regression

We denote vectors \mathbf{x} and matrices \mathbf{K} with bold-face type and their components with regular type, i.e., x_i , K_{ij} . With \mathbf{x}^T we denote the transpose of the vector \mathbf{x} . Let $\mathbf{x} \in \mathbb{R}^D$ be an input, $y \in \mathbb{R}$ an output, and let $\mathcal{D} = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N)\}$ a set of N observed input-output pairs. Denote with $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ and $\mathbf{Y} = \{y_1, \dots, y_N\}$ the inputs and outputs occurring in \mathcal{D} . We assume that \mathcal{D} is generated by an unknown function $f : \mathbb{R}^D \rightarrow \mathbb{R}$ where the observations are possibly corrupted with Gaussian noise, i.e., $y_i = f(\mathbf{x}_i) + \epsilon_i$ with $\epsilon_i \sim \mathcal{N}(0, \sigma_i^2)$. Usually, the noise is taken uniform $\sigma_i^2 = \sigma^2$ for every input \mathbf{x}_i , but this notation is consistent with the multi-annotator model described in Section 3 in which the noise model will be input dependent and the annotator identities involved.

A Gaussian process (GP) is a collection of random variables, here $\{f(\mathbf{x}_i)\}_{i \in I}$ for some index set I , any finite number of which have a joint Gaussian distribution [10]. A GP $\mathbf{f} \sim \mathcal{GP}(\mathbf{m}, \mathbf{K})$ specifies a prior distribution on functions and is completely specified by its mean function and kernel function. The kernel function k (also called (co)variance function) is a function mapping two arguments into \mathbb{R} that is symmetric, i.e., $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$, and positive definite, i.e., $\mathbf{z}^T \mathbf{K} \mathbf{z} > 0$ for all nonzero vectors \mathbf{z} with real entries ($z_d \in \mathbb{R}$) and \mathbf{K} as defined below. The kernel function can be used to construct a covariance matrix \mathbf{K} with respect to a set $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ by defining $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. An often used kernel is the Gaussian kernel:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp\left(-\frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{A}^{-1}(\mathbf{x}_i - \mathbf{x}_j)\right), \quad (1)$$

where $\mathbf{A} = \text{diag}(\ell_1^2, \dots, \ell_D^2)$ with hyperparameters specifying the signal variance (σ_f^2) and specifying the length-scales, i.e., how much the function can vary for each input dimension (ℓ_d for $d = 1, \dots, D$). A kernel function k leads to a multivariate Gaussian prior distribution on any finite subset $\mathbf{f} \subseteq \{f(\mathbf{x}_i)\}_{i \in I}$ of function values

$$p(\mathbf{f}) = \frac{1}{(2\pi)^{\frac{N}{2}} |\mathbf{K}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2} \mathbf{f}^T \mathbf{K}^{-1} \mathbf{f}\right). \quad (2)$$

Given a GP prior over functions and a set of observations \mathcal{D} , a posterior distribution $p(\mathbf{f}|\mathcal{D})$ can be computed that can be used to make predictions at new

test points \mathbf{x}, \mathbf{x}' . Let $\boldsymbol{\Sigma} = \text{diag}(\sigma_1^2, \dots, \sigma_N^2)$. The standard predictive equations for regression with a zero-mean GP are given by [10]

$$\begin{aligned}\bar{f}_{\mathcal{D}}(\mathbf{x}) &= k(\mathbf{x}, \mathbf{X})(\mathbf{K} + \boldsymbol{\Sigma})^{-1}\mathbf{Y}, \\ \text{cov}_{\mathcal{D}}(f(\mathbf{x}), f(\mathbf{x}')) &= k(\mathbf{x}, \mathbf{x}') - k(\mathbf{x}, \mathbf{X})(\mathbf{K} + \boldsymbol{\Sigma})^{-1}k(\mathbf{X}, \mathbf{x}').\end{aligned}\quad (3)$$

Maximum likelihood estimates for the hyperparameters can be obtained by minimizing the negative log marginal likelihood (e.g., using gradient descent), which can be evaluated exactly in the case of GP regression and is given by [10]

$$-\log p(\mathbf{Y}|\mathbf{X}) = \frac{1}{2}\mathbf{Y}^T(\mathbf{K} + \boldsymbol{\Sigma})^{-1}\mathbf{Y} + \frac{1}{2}\log|\mathbf{K} + \boldsymbol{\Sigma}| + \frac{N}{2}\log(2\pi). \quad (4)$$

3 Multi-annotator Regression

Let $\mathcal{D}_m = \{(\mathbf{x}_i^m, y_i^m)\}_{i=1}^{N_m}$ be the data set of the m th annotator. We assume there are M annotators, each annotator m providing noisy labels that follow a Gaussian distribution $\mathcal{N}(0, \sigma_m^2)$ with unknown noise-level σ_m . Let $N = \sum_m N_m$ be the total number of annotations. Let $\mathbf{X}_m, \mathbf{Y}_m$ be the inputs and outputs occurring in \mathcal{D}_m . Define $\mathbf{X} = \cup_{m=1}^M \mathbf{X}_m$, $\mathbf{Y} = \{\mathbf{Y}_1, \dots, \mathbf{Y}_M\}$. Define I to be the number of (unique) inputs in \mathbf{X} . Furthermore, we define for $i = 1, \dots, I$

$$\hat{\sigma}_i^2 = \sum_{m \sim i} \frac{1}{\sigma_m^2}, \quad \hat{y}_i = \hat{\sigma}_i^2 \sum_{m \sim i} \frac{y_i^m}{\sigma_m^2}, \quad \hat{\boldsymbol{\Sigma}} = \text{diag}(\hat{\sigma}_1^2, \dots, \hat{\sigma}_I^2), \quad (5)$$

with $m \sim i$ denoting the sum over annotators m that annotated sample \mathbf{x}_i .

We assume that annotators provide labels independently of each other. Therefore, the multi-annotator likelihood factorizes over cases in the training set and can, up to a constant, be rewritten in terms of the single-annotator model

$$\begin{aligned}p(\mathbf{Y}|\mathbf{f}) &= \prod_m \prod_{i \sim m} \mathcal{N}(y_i^m | f_i, \sigma_m^2) \propto \exp\left(-\frac{1}{2} \sum_i \sum_{m \sim i} \frac{(y_i^m - f_i)^2}{\sigma_m^2}\right) \\ &= \exp\left(-\frac{1}{2} \sum_i \frac{(\hat{y}_i - f_i)^2}{\hat{\sigma}_i^2} - \frac{1}{2} \sum_i \sum_{m \sim i} \frac{(y_i^m)^2}{\sigma_m^2} + \frac{1}{2} \sum_i \frac{\hat{y}_i^2}{\hat{\sigma}_i^2}\right) \\ &= \exp\left(-\frac{1}{2} \sum_i \frac{(\hat{y}_i - f_i)^2}{\hat{\sigma}_i^2} + c\right),\end{aligned}\quad (6)$$

with c independent of \mathbf{f} . The posterior process

$$p(\mathbf{f}|\mathbf{Y}) \propto p(\mathbf{f})p(\mathbf{Y}|\mathbf{f}) \propto \exp\left(-\frac{1}{2}\mathbf{f}^T \mathbf{K}^{-1} \mathbf{f} - \frac{1}{2}(\hat{\mathbf{Y}} - \mathbf{f})^T \hat{\boldsymbol{\Sigma}}^{-1}(\hat{\mathbf{Y}} - \mathbf{f})\right), \quad (7)$$

is thus Gaussian $\mathcal{N}(\mathbf{m}, \mathbf{V}) \propto \exp(-\frac{1}{2}\mathbf{f}^T \mathbf{V}^{-1} \mathbf{f} + \mathbf{f}^T \mathbf{V}^{-1} \mathbf{m})$ with mean and covariance given by

$$\begin{aligned}\mathbf{m} &= (\mathbf{K}^{-1} + \hat{\boldsymbol{\Sigma}}^{-1})^{-1} \hat{\boldsymbol{\Sigma}}^{-1} \hat{\mathbf{Y}}, \\ \mathbf{V} &= (\mathbf{K}^{-1} + \hat{\boldsymbol{\Sigma}}^{-1})^{-1}.\end{aligned}\quad (8)$$

From the posterior distribution we can derive the predictive equations for the multi-annotator model. The equations closely follow the predictive equations of the single-annotator model, but now with weighted output $\hat{\mathbf{Y}}$ and covariance $\hat{\Sigma}$ which is no longer homogeneous as it depends on the data sample:

$$\begin{aligned}\bar{f}_{\mathcal{D}}(\mathbf{x}) &= k(\mathbf{x}, \mathbf{X})(\mathbf{K} + \hat{\Sigma})^{-1}\hat{\mathbf{Y}}, \\ \text{cov}_{\mathcal{D}}(f(\mathbf{x}), f(\mathbf{x}')) &= k(\mathbf{x}, \mathbf{x}') - k(\mathbf{x}, \mathbf{X})(\mathbf{K} + \hat{\Sigma})^{-1}k(\mathbf{X}, \mathbf{x}'),\end{aligned}\quad (9)$$

where we used the fact that for a GP $\mathbb{E}[f_*|\mathbf{Y}, \mathbf{X}, \mathbf{x}_*] = k(\mathbf{x}_*, \mathbf{X})\mathbf{K}^{-1}\mathbb{E}[\mathbf{f}|\mathbf{X}, \mathbf{Y}]$, with $\mathbb{E}[\mathbf{f}|\mathbf{X}, \mathbf{Y}]$ denoting the posterior mean of \mathbf{f} given \mathbf{X} and \mathbf{Y} [10].

From Eq. (6) follows that the evidence for the multiple-annotator regression model with observations \mathbf{Y} is very similar to that of a single-annotator model with observations $\hat{\mathbf{Y}}$. Some bookkeeping to account for the constant c gives

$$\begin{aligned}-\log p(\mathbf{Y}) &= \frac{1}{2} \log |\mathbf{K} + \hat{\Sigma}| + \frac{1}{2} \hat{\mathbf{Y}}^T (\mathbf{K} + \hat{\Sigma})^{-1} \hat{\mathbf{Y}} + \frac{N}{2} \log(2\pi) + \\ &\quad - \frac{1}{2} \log |\hat{\Sigma}| - \sum_i \sum_{m \sim i} \log \frac{1}{\sigma_m} + \frac{1}{2} \sum_i \sum_{m \sim i} \frac{(y_i^m)^2}{\sigma_m^2} - \frac{1}{2} \sum_i \frac{\hat{\sigma}_i^2}{\sigma_i^2}.\end{aligned}\quad (10)$$

It can be checked that with one annotator the last four terms indeed cancel out because $\hat{\Sigma} = \sigma^2 \mathbf{I}$, $\hat{\mathbf{Y}} = \mathbf{Y}$, and $\hat{\sigma}_i^2 = \sigma^2$.

4 Experiments

We tested the GP multi-annotator model on the ‘housing’ benchmark dataset from the UCI machine learning repository. The target labels in the dataset were taken as ground truth from which we generated annotations for each annotator by adding Gaussian noise to the target label using their individual noise level. Inspired by [2,9], we choose the following set-up. First, we randomly splitted the dataset into a training dataset (70%) and test dataset (30%), which were normalized using the training data. Second, we generated annotations for each annotator. We used three annotators with a variance of 0.25, 0.5, and 0.75. We did not use all the training data but only annotated a portion of it. For each annotator we selected $A\%$ of the training data at random for annotation, with $A \in \{10, 20, \dots, 100\}$. Third, we report the root mean squared error ($RMSE(x, y) = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - y_i)^2}$) on the test data, averaged over 50 runs, for both the prediction of the targets/outputs and the hyperparameter prediction of the annotator noise-levels.

We show in Fig. 1 the RMSE results of six different models, the GP multi-annotator model, a GP fitted to the averaged training data provided by the annotators, three GP models fitted to each annotator individually, and a model that weighs the individual GP models. The latter model takes the mean prediction of each individual GP model and weighs it with the inverse predicted variance of that model. Note that the GP fitted to the average training data treats each annotator equally and does not learn individual noise levels. Each

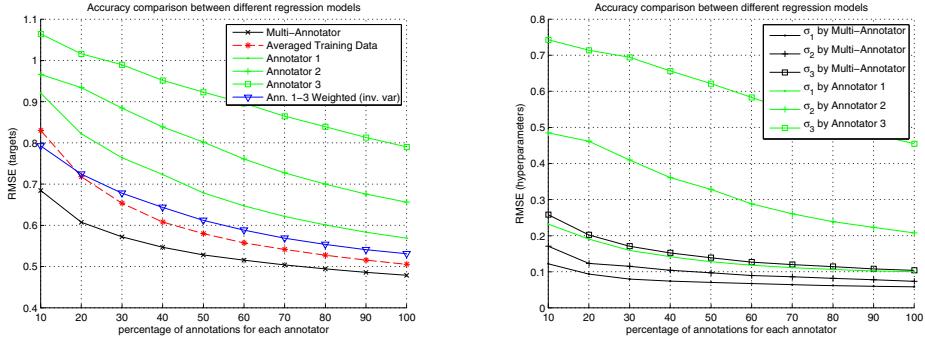


Fig. 1. The RMSE of the GP multi-annotator model, the GP fitted to the average response, and GPs fitted to each individual annotator on the ‘housing’ dataset (506 instances, 13 features). Left: RMSE for predicted targets. Right: RMSE for predicted noise-level hyperparameters.

model used a Gaussian kernel (cf. Eq. (1)), a zero mean function, and Gaussian likelihood function.

In Fig 1, left panel, we plot the RMSE results with respect to the target labels. In Fig 1, right panel, we plot the RMSE results with respect to the noise-levels of the annotators used for data generation. Clearly, the GP multi-annotator model outperforms the other models, on both target prediction and hyperparameter estimation, but the precise amount depends on the dataset, the number of annotations, and the number of annotators and their noise levels.

The GP multi-annotator model improves upon earlier reported results [2,9]; (1) It provides a non-parametric framework, (2) it is not necessary to annotate all samples by all annotators, and (3) the tuning of hyperparameters is fully automatic. In addition, since all data is properly combined using a Bayesian framework, ad hoc methods such as adding additional constraints to control annotator influence [11] or pruning low-quality annotators [12] are unnecessary.

5 Conclusions and Future Work

In this paper we presented a GP framework for regression with multiple noisy annotators. GPs provide a flexible, non-parametric framework, which naturally deals with missing annotations, and allows automatic tuning of hyperparameters using the evidence. The individual annotator noise levels can therefore be learned from data allowing for a better weighting of annotations leading to superior performance compared to a model fitted to an average response or a weighting of individually trained models.

For future work the GP multi-annotator model can be extended to be robust against outliers and relax the assumption of homogeneous Gaussian noise for each annotator. Furthermore, the model can be extended to other supervised

learning tasks like binary classification. GPs can be extended to binary classification by using the GP as a latent function whose sign determines the class label. Exact evaluations are, however, intractable, because of the non-Gaussian likelihood function and approximations are needed.

Acknowledgement. This work has been carried out as part of the OCTOPUS project with Océ Technologies B.V. under the responsibility of the Embedded Systems Institute. This project is partially supported by the Netherlands Ministry of Economic Affairs under the Bsik program.

References

1. Raykar, V.C., Yu, S., Zhao, L.H., Jerebko, A., Florin, C.: Supervised learning from multiple experts: whom to trust when everyone lies a bit. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 889–896 (2009)
2. Raykar, V.C., Zhao, L.H., Valadez, G.H., Florin, C., Bogoni, L., Moy, L.: Learning from crowds. *JMLR* 11, 1297–1322 (2010)
3. Smyth, P., Fayyad, U., Burl, M., Perona, P., Baldi, P.: Inferring ground truth from subjective labelling of venus images. In: Advances in Neural Information Processing Systems, vol. 7, pp. 1085–1092 (1995)
4. Sheng, V.S., Provost, F., Ipeirotis, P.G.: Get another label? Improving data quality and data mining using multiple, noisy labelers. In: Proceeding of the 14th ACM SIGKDD International Conference On Knowledge Discovery and Data Mining, pp. 614–622 (2008)
5. Snow, R., O'Connor, B., Jafar, D., Ng, A.: Cheap and fast—but is it good? Evaluating non-expert annotations for natural language tasks. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, pp. 254–263 (2008)
6. Sorokin, A., Forsyth, D.: Utility data annotation with Amazon Mechanical Turk. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 1–8 (2008)
7. Warfield, S.K., Zou, K.H., Wells, W.M.: Simultaneous truth and performance level estimation (STAPLE). An algorithm for the validation of image segmentation. *IEEE Trans. Med. Imag.* 23(7), 903–921 (2004)
8. Cholleti, S.R., Goldman, S.A., Blum, A., Politte, D.G., Don, S.: Veritas: combining expert opinions without labeled data. In: 20th IEEE International Conference on Tools with Artificial Intelligence (2008)
9. Ristovski, K., Das, D., Ouzienko, V., Guo, Y., Obradovic, Z.: Regression Learning with Multiple Noisy Oracles. In: 19th European Conference on Artificial Intelligence, pp. 445–450 (2010)
10. Rasmussen, C.E., Williams, C.K.I.: Gaussian processes for machine learning. MIT Press, Cambridge (2006)
11. Dekel, O., Shamir, O.: Good Learners for Evil Teachers In: Proceedings of the 26th International Conference on Machine Learning, pp. 233–240 (2009)
12. Dekel, O., Shamir, O.: Vox populi: Collecting high-quality labels from a crowd. In: Proceedings of the 22nd Annual Conference on Learning Theory, pp. 377–386 (2009)

Estimation of the Number of Clusters Using Heterogeneous Multiple Classifier System

Omar Ayad, Moamar Sayed-Mouchaweh, and Patrice Billaudel

Université de Reims Champagne-Ardenne, Centre de recherche en STIC (URCA-CReSTIC)

Moulin de la Housse, BP 1039, 51687 Reims Cedex, France

{omar.ayad, moamar.sayed-mouchaweh,
patrice.billaudel}@univ-reims.fr

Abstract. Assessing the number of clusters of statistical populations is a challenging problem in unsupervised learning. In this paper, we propose to overcome this problem by estimating the number of clusters using a novel clustering ensemble scheme. This one combines clustering and classification methods in order to increase the clustering performances. In the first time, the proposed approach divides the patterns into stable and ambiguous sets. The stable set gathers the patterns belonging to one cluster while the ambiguous set corresponds to ambiguous patterns located between different clusters. To detect the appropriate number of clusters, the proposed approach ignores ambiguous patterns and preserves the stable set as good “prototypes”. Then, the different partitions obtained from the stable set are evaluated by several cluster validation criteria. Finally, the patterns of the unstable set are assigned to the obtained clusters by supervised classifier.

Keywords: Clustering, ensemble classifier, support vector machines, k-means, unsupervised fuzzy pattern matching.

1 Introduction

Encouraged by the success of ensemble methods for supervised learning [1] ensemble clustering was recently proposed as a powerful method to improve the robustness and quality of unsupervised classification solutions. In supervised classification, an ensemble is a combination of classifiers with the goal to improve classification performances. Based on the same philosophy, the goal of combining an ensemble of clustering solutions is to find a consensus partition that optimally summarizes the ensemble and to obtain a clustering solution with improved accuracy and stability compared to the individual members of the ensemble. However, finding a consensus from the output partitions of various clustering algorithms is a difficult problem. The major difficulty is in the fact that patterns are unlabeled and there is no explicit correspondence between the labels delivered by different clusterings because the desired number of clusters is often not known in advance. Many recent studies have investigated the problem of constructing an accurate and robust ensemble committee of clustering algorithms [2] [3] [4]. Some of them execute a collection of different

clustering algorithms (*heterogeneous combination*), while others execute single clustering algorithm with different initializations or different clustering criteria (*homogenous combination*).

In this paper, we propose an ensemble clustering approach that combines clustering and classification algorithms. We used the terms *ambiguous ensemble* to indicate such patterns with a high probability of belonging to a different cluster. Ambiguity can be associated to different factors. For example, a high proximity of two or more underlying classes may produce a certain overlap between clusters, specially at patterns close to the class' boundaries. Contrary to the *ambiguous ensemble*, we used the terms *stable ensemble* to specify patterns assigned to only one cluster by the majority of clusterers in the ensemble. In the following sections we explain how to divide the original data set into a stable or ambiguous category. Following this classification of patterns, we propose a strategy to improve the clustering performance. In the first time, we propose to ignore patterns affected to the ambiguous ensemble, focusing on the patterns of stable ensemble as good prototypes to estimate the optimal number of clusters. In the second time, we use a supervised classifier to re-cluster the unstable patterns.

The rest of the paper is organized as follows. In section 2, the functioning of the proposed approach is detailed. Section 3 reviews significant experimental results obtained by applying the proposed approach. Finally, section 4 concludes the paper and presents the future work.

2 Heterogeneous Ensemble Classifier

In this section, we describe the proposed heterogeneous ensemble classifier approach (Fig. 1). This one is designed to address the general goal of clustering, which is to group data elements such that the intra-group similarities are high and the inter-group similarities are low. The functioning of the proposed scheme is based on four steps: 1) Clustering ensemble generation 2) Detection of ambiguous patterns 3) Estimation of the number of clusters 4) Re-clustering of ambiguous patterns. In the next we give details of each step.

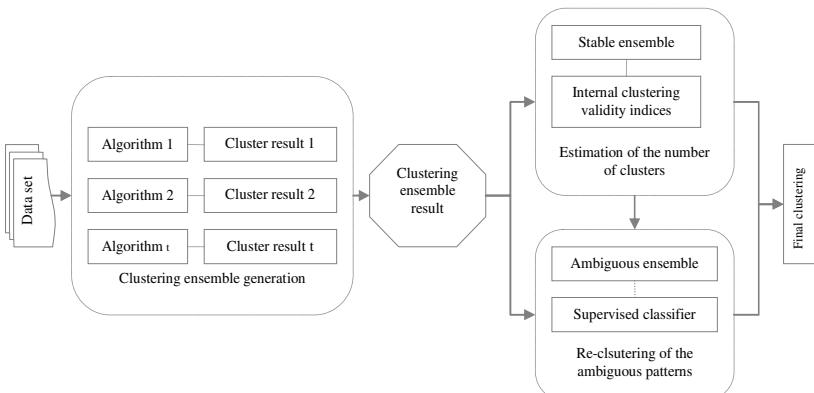


Fig. 1. Architecture of the proposed approach

2.1 Clustering Ensemble Generation

Many different strategies have been used to generate the initial partitions for a cluster ensemble. Given a set of objects $X = \{x_1, x_2, \dots, x_n\}$. The partitions of X can be obtained using different clustering algorithms or the same algorithm with different initialization values for its parameters. In this work we use the last method. The number of clusters in each partition P_i does not have to be necessarily the same.

2.2 Detection of the Ambiguous Patterns

In this subsection, we describe the approach used to detect stable and ambiguous patterns in the dataset.

Local Detection. Local detection of ambiguous patterns consists to look for the proper class of each pattern x according to each expert E_f , $f = 1 \dots l$. Two cases are possible. The first case corresponds when x belong only to one class. In other terms, x has a membership value $\pi_i(x) \neq 0$ to only one class. In the second case, the pattern x has a membership values $\pi_i(x), \pi_j(x) \neq 0$ to several classes. Where $\pi_i(x)$ and $\pi_j(x)$ are respectively the membership values of x to i and j . In this situation we defined a decision threshold ε . Then the final decision about this pattern is achieved as follows:

$$\pi_i(x) > \pi_j(x), \pi_j(x) \neq 0 \Rightarrow R_{ij}(x) = \frac{\pi_i(x)}{\pi_j(x)} \geq \varepsilon, \quad (1)$$

If $R_{ij}(x) \geq \varepsilon$, then x will be regarded as locally ambiguous pattern by the E_f clustering algorithm. Otherwise, the pattern will be classified as locally stable pattern.

Global Management of the Local Detection. After building the local decision of all clusterers in the ensemble, a pattern x can be globally stable if it is locally stable according to the majority of used experts. In this paper we use the majority voting rule. This one is a very popular combination scheme both because of its simplicity and its performance. In the proposed approach, the final decision is taken according to the number of votes given by the clusterers to each ensemble (stable/ambiguous), thus assigning x to the ensemble (stable/ambiguous) that has obtained a majority votes.

2.3 Estimation of the Number of Clusters

The simplest alternative to deal with uncertainty is to treat the ambiguous ensemble as noise patterns and ignore them, retaining the rest of data as good prototypes to detect the number of clusters. The stable ensemble is supposed contains the most informative samples for which classes' dose not completely overlapped. In this work, to attain the optimal representation of classes contained in the data set we use four popular, internal clustering validity indices, Davies-Bouldin, Calinski-Harabasz, the Silhouette and Kranowski-Lai indices. The choice of the CVIs was dictated by the fact that they are well-known measures, frequently used for the task of choosing the best clustering solution.

Davies-Bouldin Index. The Davies-Bouldin index aims at identifying sets of clusters that are compact and well separated. The Davies-Bouldin index validation index is defined as:

$$S_{DB} = \frac{1}{k} \sum_{i=1}^k \max_{j=1, \dots, k; i \neq j} \left(\frac{d_i + d_j}{d(c_i, c_j)} \right) \quad (2)$$

where k denotes the number of clusters. If i, j are cluster labels, then d_i and d_j are average distances of all patterns in clusters i and j to their respective cluster centroids, and $d(c_i, c_j)$ is the distance between these centroids. Hence the ratio is small if the clusters are compact and far from each other. Consequently, Davies-Bouldin index will have a small value for a good clustering.

Calinski-Harabasz Index. This index for n data points and K clusters is computed as

$$S_{CH} = \frac{\text{trace}(S_B)}{\text{trace}(S_W)} \cdot \frac{n-1}{n-k} \quad (3)$$

Where S_B is the between-cluster scatter matrix, S_W the within-cluster scatter matrix, n the number of clustered points, and k the number of clusters. When using the CH index to compare clustering solutions, maximal value of S_{CH} identifies the preferred candidate.

Silhouette Index. The Silhouette statistic is another well known way of estimating the number of groups in a data set. The Silhouette index (SI) computes for each point a width depending on its membership in any cluster. This silhouette width is then an average over all observations. This leads to Equation 4:

$$SI_k = \frac{1}{n} \sum_{i=1}^n \frac{(b_i - a_i)}{\max(a_i, b_i)} \quad (4)$$

Where n is the total number of points, a_i is the average distance between point x_i and all other points in its own cluster and b_i is the minimum of the average dissimilarities between x_i and points in other clusters. Finally, the partition with highest SI is taken to be optimal.

Krzanowski and Lai Index. Krzanowski and Lai index is defined as:

$$KL(k) = \left| \frac{Diff_k}{Diff_{k+1}} \right| \quad (5)$$

$$Diff_k = (k-1)^{2/m} W_{k-1} - k^{2/m} W_k \quad (6)$$

The parameter m represents the feature dimensionality of the input objects (number of attribute), and W_k is calculated as the within-group dispersion matrix of the clustered data

$$W_k = \sum_{i=1}^k \sum_{x_j \in C_i} (x_j - c_j)(x_j - c_j)^T \quad (7)$$

In this case, x_j represents an object assigned to the j^{th} cluster, and c_i denotes the centroid or mediod of the i^{th} cluster. The optimum k correspond to the maximum of $KL(k)$.

2.4 Re-clustering of Ambiguous Patterns

Now, we consider the clustering solution of the stable ensemble to re-cluster the unstable patterns. Since this solution is achieved in absence of unstable patterns, we assume a more robust representation of the surrogate classes is attained in the output clusters. Thereby, a training set is implicitly generated. The main goal now is to re-cluster the patterns assigned previously to the ambiguous ensemble. To achieve this task, we use supervised classifier algorithms.

3 Experimental Evaluation

In this section, we present experimental results of our model to demonstrate that it is both accurate and effective. We test it on a classical Iris flower data set [5]. The main goal of this evaluation is to show the significance of the different steps in the proposed approach. In the proposed heterogeneous ensemble classifiers two clustering algorithms assisted by a Support Vector Machines (SVMs) are employed. This latter is a method of supervised learning used for classification and regression [7]. In addition to SVMs, we have utilized Unsupervised Fuzzy Pattern Matching (NSFPM) [6] and k-means clustering methods. USFPM is chosen because it is capability to achieve clustering without the need to any prior information about clusters (shape, dimension) or their number. SVM is considered to be one of the best methods for the problem of supervised classification and the popular K-means method because it has the advantage of giving good modeling results in many cases.

In the first step different solutions were obtained by applying USFPM to the same data with different initialization. In this step the USFPM has access to all data and the difference between clustering solutions come from different initialization. Therefore, the clustering solutions obtained in this step are estimated to be in high diversity.

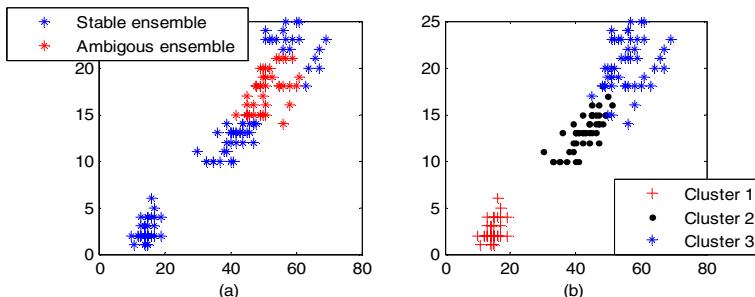


Fig. 2. (a) Detection of the stable and ambiguous patters, (b) Iris dataset

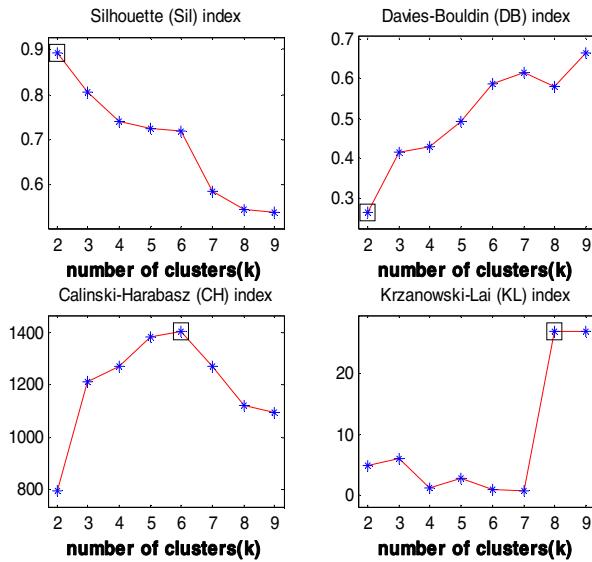


Fig. 3. Number of the clusters estimated from the original data set

When the clustering's solution are generated, we evaluate the qualities of each one of them by comparing its possibility membership and the threshold of decision ε . As result of this local evaluation, the initial data set will be divided into two ensembles. The first one contains the stable pattern or in other terms the pattern affected in one cluster. The second ensemble includes the pattern identified in multiple clusters. When the local decision for each used algorithms is established, the next step combine these opinion in order to achieve the global decision. This one is obtained using majority voting rule; a pattern is globally stable if it is known as stable by the majority of the clusterers in the ensemble. In figure 2(a) we illustrate the stable and ambiguous ensemble obtained by the proposed approach after combination of the individual decision.

In the second step, the patterns of the stable set are divides into clusters using the clustering method K-means. Then, the number of clusters is estimated using four internal clustering validity indices, Davies-Bouldin, Calinski-Harabasz, the Silhouette and Krzanowski-Lai indices for $k=2\dots 10$. In figure 3 is pictured the results obtained from the original data. Figure 4 show the result obtained by the proposed approach. Each plot shows one internal clustering validity index. We can notice that the proposed approach give the best number of clusters $k=3$. Finally we use the obtained clustering solution to train SVMs. This one is used to re-cluster the unstable pattern.

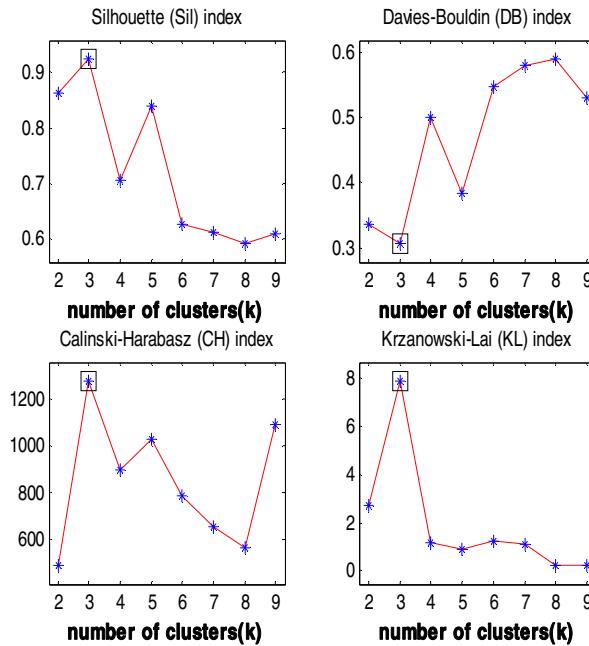


Fig. 4. Number of clusters obtained by the proposed approach

The following table represents the element composition of Iris data set in the true clusters and the best clusters using the proposed approach. 95,33 % of patterns are properly assigned in the best cluster.

Table 1. Final clustering result

Iris data set (3 clusters)	Element composition		
	True clusters	Best clusters using the proposed approach	
Clust er1	Group 1	50	50
	Group 2	0	0
	Group 3	0	0
	Group 1	0	0
Clust er2	Group 2	50	50
	Group 3	0	0
	Group 1	0	0
	Group 2	0	7
Clust er3	Group 3	50	43

4 Conclusion

In this work, we have presented new clustering ensemble scheme, based on the concept of clustering ensembles and on the combination of unsupervised and supervised classification results. We proposed to exploit the properties of ensemble clustering results to define stable and ambiguous patterns. We also proposed to estimate the number of clusters using only a subset of original data that correspond to stable ensemble. Finally, if the pattern is located in an ambiguous region, then SVMs will be used to classify it. The performances of the proposed method have been evaluated against Iris datasets. In the future, we intend to validate our approach using other datasets.

References

1. Kittler, J.: On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20(3), 226–239 (1998)
2. Fred, A., Jain, A.K.: Data clustering using evidence accumulation. In: 16th Intel Conference on Pattern Recognition, ICPR, Quebec, pp. 276–280, (2002)
3. Strehl, A., Ghosh, J.: Cluster ensembles- A knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research* 3, 583–617 (2002)
4. Topchy, A., Jain, A.K., Punch, W.: Clustering ensembles: models of consensus and weak partitions. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27(12), 1866–1881 (2005)
5. Blake, C., Merz, C.: UCI repository of machine learning databases. University of California, Dept. of Information and Computer Science, Irvine (1998)
6. Sayed-Mouchaweh, M.: Semi-supervised classification method for dynamic applications. *Fuzzy Sets and Systems (FSS)* 161(4), 544–563 (2010)
7. Vapnik, V.N.: *The nature of statistical learning theory*. Springer, New York (1995)
8. Corduneanu, A., Bishop, C.M.: Variational Bayesian model selection for mixture distributions. In: Eighth Conf. Artificial Intelligence and Statistics, pp. 27–34 (2001)
9. Vega-Pons, S., Correa-Morris, J., Ruiz-Shulcloper, J.: Weighted partition consensus via kernels. *Pattern Recognition* 43, 2712–2724 (2010)
10. Masson, M.H., Denoeux, T.: Ensemble clustering in the belief functions framework. *International Journal of Approximate Reasoning* (2010) (article in Press)
11. Mimaroglu, S., Erdil, E.: Combining multiple clusterings using similarity graph. *Pattern Recognition* (2010) (article in Press)
12. Kashef, R.S., Kamel, M.: Cooperative clustering. *Pattern Recognition* 43(6), 231–2329 (2010)
13. Ayad, H., Kamel, M.: Cumulative voting consensus method for partitions with variable number of clusters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 160–173 (2007)
14. Forestier, G., Gançarski, P., Wemmert, C.: Collaborative clustering with background knowledge. *Data & Knowledge Engineering* 96, 211–228 (2010)
15. Minaei-Bidgoli, B., Topchy, A., Punch, W.F.: A Comparison of Resampling Methods for Clustering Ensembles. In: International Conference on Machine Learning, Models, Technologies and Applications, pp. 939–945 (2004)
16. He, Z.Y., Xu, X.F., Deng, S.C.: A cluster ensemble method for clustering categorical data. *Information Fusion*, 143–151 (2005)

A Distributed Self-adaptive Nonparametric Change-Detection Test for Sensor/Actuator Networks*

Cesare Alippi, Giacomo Boracchi, and Manuel Roveri

Dipartimento di Elettronica e Informazione

Politecnico di Milano, Milano, Italy

{cesare.alippi,giacomo.boracchi,manuel.roveri}@polimi.it

Abstract. The prompt detection of faults and, more in general, changes in stationarity in networked systems such as sensor/actuator networks is a key issue to guarantee robustness and adaptability in applications working in real-life environments. Traditional change-detection methods aiming at assessing the stationarity of a data generating process would require a centralized availability of all observations, solution clearly unacceptable when large scale networks are considered and data have local interest. Differently, distributed solutions based on decentralized change-detection tests exploiting information at the unit and cluster level would be a solution. This work suggests a novel distributed change-detection test which operates at two-levels: the first, running on the unit, is particularly reactive in detecting small changes in the process generating the data, whereas the second exploits distributed information at the cluster-level to reduce false positives. Results can be immediately integrated in the machine learning community where adaptive solutions are envisaged to address changes in stationarity of the considered application. A large experimental campaign shows the effectiveness of the approach both on synthetic and real data applications.

Keywords: Change detection test, sensor/actuator networks, fault detection.

1 Introduction

Networked embedded systems and sensor/actuator networks designed to work in real-life environments, e.g., water distribution systems, intelligent buildings, critical infrastructure networks, are subject to faults and ageing effects which generally induce a change in the statistical properties of the data coming from the field. Anticipating the detection of a change is a key issue to support intervention and avoid possible fault effects that would induce critical, when not catastrophic, consequences.

Change detection through quantitative observations of process variables is a hot research topic largely addressed in the literature. Among different strategies, refer to [1] for a comprehensive review, of particular interest are Change-Detection Tests (CDTs) [2][3], i.e., statistical techniques aiming at assessing satisfaction of the

* This research has been funded by the European Commission's 7th Framework Program, under grant Agreement INSFO-ICT-270428 (iSense).

stationarity hypothesis for the data generating process. These tests generally assume that observations are all available in a centralized fashion at the decision making unit. This situation, even if acceptable in some cases, is not valid in general since it requires a large data exchange of no interest to the user with a communication cost. The problem amplifies in those remote parts of the network where energy is an issue despite the presence for energy harvesters, i.e., based on photovoltaic cells. To overcome such aspects, intelligence, here to be intended as the change-detection ability and solution adaptation, must be distributed among units which cooperate at different abstraction levels to build up a decision.

In the literature, distributed CDTs, can be grouped as distributed detection tests with quantized observations and local decisions. In both cases, all sensing units communicate with a centralized decision making unit (called data-aggregation center), whose goal is to aggregate data to provide the final decision.

Tests following the former approach (e.g., [4]) rely on the quantization of observations at the sensing units followed by a centralized analysis of such data at the central station. The use of quantized observations, which aims at controlling the power consumption in communication by reducing the bits to be transmitted does not generally provide the expected advantage in real-world applications since the bit savings is a small percentage of the packet size.

Differently, distributed tests following the latter approach (e.g., [5][6]) rely on independent local CDTs at the unit level whose decisions are transmitted to the unit for aggregation and decision. This philosophy fits well with distributed sensor/actuator networks. While tests at the sensing-unit level can be any traditional (centralized) CDT acting on available data, the key issue is how to effectively aggregate local decisions to achieve a final decision with low false positives and negatives. The works present in the literature are straightforward and generally propose to wait for a fixed number k of detections from the sensing units before raising a global variation in the data generating process. Assuming N sensing units in the network, traditional solutions set k equals to 1 (i.e., the distributed CDT detects a variation when at least one sensing unit detects a change) or N (i.e., the distributed CDT detects a variation when all the sensing units detect a change). Other values of k could be considered as well; e.g., [7] suggests $k=N/2$. Obviously, the value of k greatly influences the detection performance of the distributed CDT. In fact, low values of k guarantee low detection delays at the expenses of higher false positives. On the contrary, as k increases, detection delays increase but false positives decrease.

This work proposes a novel distributed change-detection test where each sensing unit performs a CDT and, once a change is detected, a second CDT algorithm is activated at the data-aggregation center receiving data from the network's cluster. If the change is confirmed, the CDT signals a global change in the network, otherwise the local detection is considered to be a false positive and the sensing units providing the wrong assessment are retrained to improve subsequent detection accuracy. We comment that the proposed distributed CDT improves over the distributed tests present in the literature by substituting the voting mechanism on k with a theoretically sound mechanism exploiting correlations among acquired observations. Furthermore, the two-levels distributed architecture allows us for being very reactive and prompt in detections while keeping under control occurrences of false positives.

The paper is organized as follows: Section 2 introduces the problem statement, while the suggested distributed CDT is presented in Section 3. Experiments on synthetic and real applications data are shown in Section 4.

2 Problem Statement

Consider a sensor/actuator network composed of N sensing units that observe the same physical phenomenon, and a data-aggregation center that is connected with all units. The i^{th} sensing unit acquires over time data from a process $X_i: N \rightarrow R$, which generates independent and identically distributed (i.i.d.) observations drawn from an unknown probability density function (pdf). Under specific assumptions, the case of dependent observations w.r.t time could be brought back to this framework by means of suitable transformations (e.g., the i.i.d. assumption may concern the parameters of a model describing the data under suitable hypotheses [9] or innovation [3]). We outline that we do not require independence or the same distribution among observations coming from different sensing units. Let $O_{i,T} = \{X_i(t), t=1, \dots, T\}$ be the sequence of observations acquired by the i^{th} unit up to time T , and assume that (at least) the first $T_0 < T$ observations acquired by all sensing units have been generated by the process in a stationary state. Thus, O_{i,T_0} represents the training set of the i^{th} unit, i.e., a set of observations used to configure the parameters of the test. The proposed approach is nonparametric and, therefore, the pdfs of the data acquired in all the sensing units are unknown, both before and after the change. Moreover, after the change, the pdfs description can be time independent (e.g., abrupt changes) or evolve with time (e.g., drifts). We assume that the X_i s under monitoring change their statistical properties at time instant $t=T^*$ and that all sensing units can, potentially, observe the change. However, no assumptions are made about the effect of this change on each unit, as the magnitude and the profile of the change may vary from unit to unit (e.g., it may even affect only few nodes).

3 Distributed Change-Detection Test

The proposed distributed CDT relies on N units that independently monitor the process by means of unit-level CDTs, and by a data-aggregation center, which validates local detections by analyzing information sent from each node. In particular, the designed solution follows the approach delineated in [8] and combines the ICI-based CDT [10] at the unit-level, with a hypothesis test based on the Hotelling T-square statistic [11] at the data-aggregation center.

The proposed solution is outlined in Algorithm 1: during the training phase the ICI-based CDTs in execution on sensing units are configured using their respective training sequences (line 1). These tests monitor the process by extracting ad-hoc features from the observations: features extracted from O_{i,T_0} are then sent to the data-aggregation center (line 2), as they characterize how the i^{th} unit perceives the stationary process. During the operational life (line 4), for each new observation, each unit independently assesses whether the incoming data have been generated by the process in its initial (stationary) state or not by relying on the ICI-based CDT. When no units detect a change, the process is considered stationary, and each unit waits for the next measurements.

Algorithm 1. Distributed Change-Detection System

-
1. *Each unit*: configure the ICI-based CDT using $\{O_{i,T_0}, 1 \leq i \leq N\}$;
 2. *Each unit*: send feature extracted from O_{i,T_0} to the aggregation center.
 3. **while**(units acquire new observations at time T){
 4. *Each unit* runs the ICI-based CDT at time T;
 5. **let** S_T be the set of units where the ICI-based CDT detects a change at time T;
 6. **if** (S_T is not empty) {
 7. *Each unit in* S_T : run the refinement procedure, sent $T_{ref,i}$ to the aggregation center.
 8. *Aggregation center*: compute T_{ref} out of $T_{ref,i}, 1 \leq i \leq N$, send T_{ref} to each unit.
 9. *Each unit*: send features in $[T_{ref}, T]$ to the aggregation center;
 10. *Aggregation center*: runs the hypothesis test;
 11. **if** (second-level test detects a change){
 12. Change is validated.
 13. *Each unit in* S_T the ICI-based CDT is re-trained on the new process status}
 14. **else**{
 15. Change hypothesis is discarded (false positive);
 16. *Each unit in* S_T : reconfigure the ICI-based CDT to improve its performance }}}
-

Denote as S_T the set of units detecting a variation at time T (line 5): each unit in S_T runs independently the refinement procedure described in [12] to provide an estimate $T_{ref,j}, 1 \leq j \leq |S_T|$ of the change time-instant T^* (line 7). These estimates are sent to the aggregation center and there processed (e.g., by choosing the earliest one, or by computing their mean) to determine a common unique estimate T_{ref} of T^* (line 8).

The estimated change time T_{ref} is then propagated through the network and made available to units. In turn, units send the locally stored extracted features in the $[T_{ref}, T]$ temporal interval (line 9) to the data-aggregation center (an internal shift buffer is made available to locally store incoming observation).

At the aggregation center the hypothesis test aims at reducing false positives by performing a multivariate analysis to assess if there is a statistical evidence that features computed in O_{i,T_0} differs from features in $[T_{ref}, T]$, after the suspected change (line 10). If the aggregation center validates the detection (line 12), each node is retrained to monitor further variations w.r.t. the new process state on the forthcoming observations (line 13). In particular, each node restarts the ICI-based CDT using the features computed in $[T_{ref}, T]$. On the contrary, when the aggregation center does not validate the detection (line 15), each units in S_T is informed to have provided a false positive, and it is retrained using the features estimated from the initial training set O_{i,T_0} , to keep on monitoring changes w.r.t. the initial status (line 16). We provide now further details and comments related to the proposed algorithm.

3.1 ICI-Based Change-Detection Test at the Unit Level

Since in most of practical applications the distribution of the process before and after the change remains unknown, nonparameteric sequential CDTs have to be enforced at the unit level. Among available solutions present in the literature, e.g., see [2][3], the

ICI-based change-detection test [10] has been selected for its low computational complexity and strong theory. Moreover, the test, differently from other nonparametric sequential CDT mechanisms, is endowed with a change-detection refinement procedure able to provide an accurate estimate T_{ref} of the time instant T^* ; such an information is then used in Algorithm 1 (line 7).

Basically, the ICI-based CDT monitors the data generating process by relying on i.i.d. and Gaussian distributed features (in the stationary case); the Gaussian hypothesis is satisfied thanks to ad-hoc transformations. The features we use here are derived from the sample mean and variance (computed on disjoint subsequences of observations) and represent the data transmitted to the aggregation center to validate the unit-level detections (line 9).

3.2 The Change-Detection Test at the Data-Aggregation Center

The test running at the aggregation center level aims at assessing the detections raised by the sensing-units by exploiting group information. In particular, the T_{ref} obtained by processing the local estimates $T_{ref,i}$ allows us for selecting the observations in $[T_{ref}, T]$ stored in each sensing unit, that are representative of the process after the change. A multivariate hypothesis test based on the Hotelling T-square statistic [11], which is the classical technique to inference a change in the mean of a Gaussian multivariate random variable, is executed to assess if there is a statistical evidence (according to a defined significance level α) that the mean features value in O_{T_0} equals that in $[T_{ref}, T]$. The analysis performed at the aggregation center focuses on the feature detecting the change at the sensing units: let F_i^0 and F_i^1 be the sequences of feature values at the i^{th} sensing unit in O_{i,T_0} and $[T_{ref}, T]$, respectively; let n_0 and n_1 be their lengths. The inference is performed on the mean vectors \bar{F}^0 and \bar{F}^1 , defined such that their i^{th} components are $(\bar{F}^0)_i = \sum_{j=1}^{n_0} F_i^0(j)/n_0$ and $(\bar{F}^1)_i = \sum_{j=1}^{n_1} F_i^1(j)/n_1$, with $i=1,..,N$. The covariance matrix Σ is computed by pooling the covariances estimated from the features F_i^0 and from F_i^1 . The Hotelling T-square statistic is

$$S = (\bar{F}^0 - \bar{F}^1)' \left(\left(\frac{1}{n_0} + \frac{1}{n_1} \right) \Sigma \right)^{-1} (\bar{F}^0 - \bar{F}^1),$$

and is distributed as

$$\left(\frac{n_0 + n_1 - 2}{n_0 + n_1 - N - 1} \right) \mathcal{F}(N, n_0 + n_1 - N - 1),$$

where \mathcal{F} denotes the F distribution. This allows us for verifying if the null hypothesis “the difference between $\bar{F}^0 - \bar{F}^1$ equals 0” needs to be rejected with confidence α .

The Hotelling T-square statistics require $n_0 > N$ and $n_1 > N$. For this reason, the length of the training sequence and the minimum amount of samples to be considered between T_{ref} and T should be suitably adapted to the number of sensing units in the network. Large networks could be eventually partitioned, only for change-detection purposes, into smaller clusters of units.

Table 1. Simulations results

			Proposed CDS			Traditional k/N CDS		
			$\alpha=0.05$	$\alpha=0.01$	$\alpha=0.005$	$k=1$	$k=5$	$k=10$
D1	Abrupt $\delta = 1\sigma$	FP(%)	16.0	4.6	4.0	69.3	0	0
		FN(%)	0	0.6	2.6	0	0	0
		MD	4456.6	5289.5	5350.2	4618.9	8523.4	14508.2
	Abrupt $\delta = 5\sigma$	FP(%)	16.0	4.6	4.0	69.3	0	0
		FN(%)	0	0	0	0	0	0
		MD	808.8	793.8	794.2	852.6	1478.1	2179.0
	Abrupt $\delta = 2\sigma$	FP(%)	16.0	4.6	4.0	69.3	0	0
		FN(%)	0	0	0	0	0	0
		MD	449.1	449.1	449.2	442.6	468.9	545.4
D2	Drift $\delta = 1\sigma$	FP(%)	16.0	4.6	4.0	69.3	0	0
		FN(%)	5.3	15.3	23.3	0	0	4.6
		MD	34884.8	39642.7	41709.4	21767.1	37743.6	51967.9
	Drift $\delta = 5\sigma$	FP(%)	16.0	4.6	4.0	69.3	0	0
		FN(%)	0	0	0	0	0	0
		MD	11633.7	12215.4	12861.3	10034.3	14937.8	19263.2
	Drift $\delta = 2\sigma$	FP(%)	16.0	4.6	4.0	69.3	0	0
		FN(%)	0	0	0	0	0	0
		MD	5019.6	4997.4	5181.3	4624.2	6972.7	8703.8
	Abrupt	FP(%)	11.3	2.0	0.6	65.3	0	0
		FN(%)	0	0	0	0	0	0
		MD	461.5	463.8	463.8	457.8	555.9	2643.1

4 Experiments

To validate the effectiveness of the suggested distributed change-detection test we considered both synthetic and real applications applied to a network of $N=10$ sensorial units. Performances of the proposed approach are compared with those of traditional methods (a global detection is raised when at least k sensing units out of N detect a variation [5]-[7]). At the unit level the detection test is the ICI-based CDT shown to be particularly effective in the literature [10]. Three indexes have been considered to assess the performances of the tests:

- False positive index (FP): it counts the number of times a test detects a change when there it is not.
- False negative index (FN): it counts the times a test does not detect a change when there it is.
- Mean Delay (MD): it represents the time delay in detecting a change (expressed in terms of the number of observations).

Application D1 – The i^{th} sensing unit receives 90000 observations extracted from a Gaussian distribution $N(\mu = 1, \sigma^2 = 1)$ then, to reproduce a scenario where each unit has different gain and offset values, these data are scaled by $\sigma_i > 0$ (the gain), and a constant term $\mu_i > 0$ (the offset) is added. Thus, the data generating process at each unit can be described as $N(\mu_i + \sigma_i, \sigma_i^2)$. We considered two kinds of perturbations having magnitude $\delta_i \in \{0.1\sigma_i, 0.5\sigma_i, 2\sigma_i\}$ affecting the mean value at sample $T^* = 30000$: an abrupt change, where the mean suddenly increases of δ_i , and a drift, where the mean increases linearly starting at T^* and reaches $\mu_i + \delta_i + \sigma_i$ at $T=90000$.

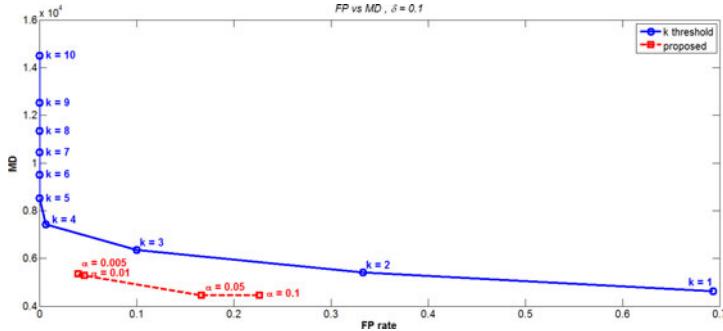


Fig. 1. Mean Delay (MD) w.r.t. false positives for the proposed CDT (with α ranging from 0.005 to 0.1) and the traditional k out of N detection test with $\delta = 0.1\sigma$

Application D2 – Each sensing unit is a photodiode excited by X-rays. Sequences contain 60000 samples and show a perturbation affecting the mean at sample 30000 in the $\delta \in \{0.1\sigma, 2\sigma\}$ range (being σ estimated from samples [0,30000]). As shown in [8], these data are far from being Gaussian distributed.

The ICI-based CDTs in execution on the units have been trained with the first 400 samples for each dataset. According to [12], we experimentally fixed $\Gamma=2.5$ and $\Gamma_{\text{ref}}=3$ (the tuning parameters for the ICI-based CDT and for the refinement procedure, respectively). We considered $\alpha=0.005$, $\alpha=0.01$ and $\alpha=0.05$ for the second-level hypothesis test ($1-\alpha$ is the test confidence).

Table 1 shows the comparison between the proposed and the traditional approach with $k \in \{1, N/2, N\}$; performances are averaged over 150 runs. As far as application D1-abrupt is concerned, the proposed CDT provides prompter detections w.r.t. the traditional approach, yet guaranteeing lower false positives. As expected, as confidence $1-\alpha$ decreases, the CDT increases both false positives and detection delays. The same behavior can be observed when parameter k in the traditional approach. Obviously, the detection delays of both approaches decrease as the intensity of the perturbation δ increases, since the change is more easily detectable. These results are particularly interesting since the proposed approach allows to detect a change in the data generating process even when the variation is detected by just a sensing unit (and then confirmed at the aggregation center), while in the traditional approach the change must be detected by at least k units (which could be critical when the change in the process is perceived only by a subset of the N units).

Experiments on application D1-drift show that both false negatives and the mean delay are higher than those in the abrupt change case since the change is smooth and more difficult to detect. In particular, with $\delta = 0.1\sigma$, the suggested approach provides lower performance than the traditional approach since, as shown in [8], the aggregation center might discard the detections provided by the sensing units due to the lack of statistical evidence in rejecting the stationary hypothesis (this might be caused both by the small magnitude of the perturbation and by an inaccurate estimate of T_{ref}). To reduce both false positives and detection delays, higher values of α should be considered (e.g. $\alpha \geq 0.1$). On the contrary, the proposed approach overcomes the performance of the traditional methods for larger values of δ . Experimental results of

application D2 show that the proposed CDT well behaves even with real non-Gaussian data and that results are in line with those of application D1.

A more detailed comparison between the proposed and the traditional approaches has been performed to evaluate performances when k ranges from 1 to N . Results are given in Figure 1 for application D1-abrupt with $\delta = 0.1\sigma$. Performance improvements are appreciable, obtained at the expenses of negligible increases of false negatives as presented in Table 1: in particular, given a tolerated percentage of false positives, the proposed approach guarantees a lower detection delay once compared to the traditional one. Similarly, at equal values of detection delays, the proposed approach provides lower false positive rates. For example, considering acceptable a FP rate of 5%, the suggested test yields MDs about 25% lower than the traditional approach.

5 Conclusions

This work presents a distributed nonparametric CDT designed to work in networked embedded systems and sensor/actuator networks. The novelty of the proposed approach resides in the distributed two-levels CDT where a change is first detected (ICI-based method) at the unit level and then assessed at the cluster level by exploiting cluster information (Hotelling test). This allows the test for reducing false positives, a common problem which arises in sequential CDTs. Experiments applied to synthetic and real applications show the effectiveness of the proposed approach.

References

1. Chiang, L.H., Russell, E.L., Braatz, R.D.: *Fault Detection and Diagnosis in Industrial Systems*. Springer, London (2001)
2. Lai, T.L.: Sequential analysis: some classical problems and new challenges. *StatisticaSinica* 11(2), 303–350 (2001)
3. Basseville, M., Nikiforov, I.V.: *Detection of Abrupt Changes: Theory and Application*. Prentice-Hall, N.J (1993)
4. Mei, Y.: Information bounds and quickest change detection in decentralized decision systems. *IEEE Trans. on Information Theory* 51(7), 2669–2681 (2005)
5. Tartakovsky, A., Veeravalli, V.: Quickest change detection in distributed sensor systems. In: 6th Int. Conf. on Information Fusion, pp. 756–763 (2003)
6. Tartakovsky, A., Veeravalli, V.: Asymptotically optimal quickest change detection in distributed sensor systems. *Sequential Analysis* 27(4), 441–475 (2008)
7. Yang, D., Qi, H.: An Effective Decentralized Nonparametric Quickest Detection Approach. In: 20th Int. Conf. on Pattern Recognition, pp. 2278–2281 (2010)
8. Alippi, C., Boracchi, G., Roveri, M.: A hierarchical nonparametric sequential change-detection test. In: IEEE 2011 International Joint Conference on Neural Networks (2011)
9. Ljung, L.: *System identification: Theory for the user*. Prentice Hall, N.J (1999)
10. Alippi, C., Boracchi, G., Roveri, M.: Change Detection Tests Using the ICI rule. In: IEEE 2010 International Joint Conference on Neural Networks, pp. 1 – 7 (2010)
11. Johnson, R.A., Wichern, D.W.: *Applied multivariate statistical analysis*. Prentice Hall, N.J (1998)
12. Alippi, C., Boracchi, G., Roveri, M.: Adaptive classifiers with ICI-based adaptive knowledge base management. In: Diamantaras, K., Duch, W., Iliadis, L.S. (eds.) *ICANN 2010. LNCS*, vol. 6353, pp. 458–467. Springer, Heidelberg (2010)

Weighted Mutual Information for Feature Selection

Erik Schaffernicht and Horst-Michael Gross

Ilmenau University of Technology
Neuroinformatics and Cognitive Robotics Lab
98693 Ilmenau, Germany
Erik.Schaffernicht@Tu-Ilmenau.de

Abstract. In this paper, we apply weighted Mutual Information for effective feature selection. The presented hybrid filter wrapper approach resembles the well known AdaBoost algorithm by focusing on those samples that are not classified or approximated correctly using the selected features. Redundancies and bias of the employed learning machine are handled implicitly by our approach.

In experiments, we compare the weighted Mutual Information algorithm with other basic approaches for feature subset selection that use similar selection criteria. The efficiency and effectiveness of our method are demonstrated by the obtained results.

1 Introduction

In supervised learning, there are often tasks that provide plenty of input variables, many of which are not required for the classification or approximation task in question. These irrelevant or redundant features complicate learning processes. The highdimensional decision space increases the problem of overfitting, it may interfere with generalization abilities, and requires more time for the learning.

Feature selection methods are designed to select a sufficiently small subset of meaningful features to mitigate these problems. There are three different types of feature selection approaches: the *Filter* methods, the *Embedded* methods and the *Wrapper* methods [1], [2].

Filter methods operate on the data to find correlation between the variables and the target, independent of any learning machine. Examples include the well known Principal Component Analysis, the linear correlation coefficient, or information theoretic measures like *Mutual Information* [3]. Filter methods are often the cheapest approaches in terms of computational demands.

Embedded methods employ specific learning machines, like neural networks, which learn with all input channels. After the training process is complete, the importance of the inputs can be inferred from the structure of the resulting classifier. This includes e.g. weight pruning in neural network architectures with *Optimal Brain Damage* [4], Bayes Neural Networks [5], or Recursive Feature Selection for SVMs [6].

Wrapper methods [9] employ arbitrary learning machines, which are considered black boxes. The feature selection algorithm is wrapped around the black box. An interesting feature subset is determined by a search strategy and evaluated with the classifier adapted to this subset. The bias of the learning machine is taken into account opposed to the pure feature relevance determined by a filter method. The advantages of the bias implicit view were discussed e.g. in [1].

The essential disadvantage of wrappers that often prevents their use for large data sets are the associated computational costs. Recent developments aim at combining filter and wrapper methods to speed up the wrapper algorithm by a preprocessing with filter methods, while keeping the bias implicit view.

In this paper, we propose a hybrid filter/wrapper algorithm that employs weighted Mutual Information as the filter component for a forward selection.

In the next section, we will discuss Mutual Information, its weighted version and methods to estimate it. Given these foundations, the algorithm for selecting features will be presented and discussed in Section 3. Before experimental results are provided in Section 5, we will discuss related work in Section 4.

2 Weighted Mutual Information

The well known Mutual Information (MI) measures the dependence between two random variables, in the context of feature selection between one feature F_i and the target T (e.g. the class labels). Lower case letter indicate the realization of the random variables.

$$I(F_i, T) = \int_{f_i} \int_t p(f_i, t) \log \frac{p(f_i, t)}{p(f_i)p(t)} dt df_i. \quad (1)$$

If the MI is zero, both variables are independent and contain no information about each other, thus the feature is irrelevant for the target. Higher MI values indicate more information about the target and hence a higher relevance. Simple feature ranking chooses a certain number of the highest ranked features or all features above a threshold as relevant features for the learning machine. More details can be found e.g. in [3].

Weighted Mutual Information

The idea of a weighted form of Mutual Information is mentioned in [10], but it has not gained popularity because the number of meaningful applications is limited. It is defined as

$$wI(F_i; T; W) = \int_{f_i} \int_t w(s_j) p(f_i, t) \log \frac{p(f_i, t)}{p(f_i)p(t)} dt df_i. \quad (2)$$

For each sample s_j , a combination of input value f_i and target value t , a weight $w(s_j) \geq 0$ is imposed. This results in a specific relevance of each unique sample not unlike a prior on how informative a certain combination is. We will use this to weight the influence of the different input samples.

Estimation of Weighted Mutual Information

The main problem with the above Eq. 1 is the estimation of the required probability densities from the available data. The straightforward approach to compute the MI uses histograms for approximation, simplifying the formula to sums instead of integrals or to apply kernel density estimation methods. A comparison between these and more sophisticated methods for estimating the MI can be found in [7] and, specifically applied to the feature selection domain, in [8].

The estimation of the weighted Mutual Information is straightforward only for some of the estimation approaches. Instead of Eq. 2 it is easier to estimate the equivalent formulation

$$wI(F_i; T; W) = \int_{f_i} \int_t w(s_j) p(f_i, t) \log \frac{w(s_j) p(f_i, t)}{w(s_j) p(f_i) p(t)} dt df_i. \quad (3)$$

Effectively, this realizes the weighted Mutual Information by manipulating the probability distributions. Each sample contributes to the probability density according to its weight only (zero weight samples don't contribute anything), which can be compared to the particle representation used in Particle Filters.

Using the histogram approach, which discretizes the data according to some strategy and then replaces the integrals in Eq. 1 with sums over the discrete histogram bins, the use of the weight is trivial. Each sample does not contribute equally to its bin, but according to its weight. Similar is the weighted variant of kernel density estimation, where in practice the sum of pairwise interacting kernels is calculated. The sample weighting is implemented by manipulating the kernel put at the sample's position.

Other popular methods, like Kraskov's k-nearest neighbour estimator [11], which are not based on the Kullback-Leibler divergence formulation, but entropy estimators, are not easily modified to compute the weighted Mutual Information.

3 Feature Selection with Weighted Mutual Information

The basic idea of the feature selection algorithm proposed here is the following: Given a classifier/approximator and its error, the choice which feature to include next to improve the performance should be based on the errors made and not on all available data. This is done by weighting the correct and wrong classified samples differently.

The simple case is if a classifier only produces discrete class information. In the next step, any correctly classified samples are left out for the computation of the MI between samples and target, because they have a weight of zero. Only the wrongly classified samples are used, they have an equal weight.

The use of a continuous predictor allows for a different weighting for each sample according to the residual. For example, a sample that is classified as positive, but which is near the boundary to the negative class will yield a non zero residual despite being in the correct class. But its influence should be smaller compared to a sample that is on the wrong side of the decision boundary.

Algorithm 1. $S = \text{wMI}(X, Y)$

Input: data set of observations X and the corresponding labels Y
Output: final feature set S and the final classifier

 $S \leftarrow \emptyset$
 $W \leftarrow 1$ {Same weight for all samples.}

while stopping criterion not true **do**
 $F_{max} = \arg \max_{F_i} [wI(F_i, T, W)]$ {Find feature with maximum weighted MI}

 $S \leftarrow S \cup F_{max}$ {Add feature to the subset}

 $F \leftarrow F \setminus F_{max}$ {Remove feature from the candidate set}

 $\text{Classifier} \leftarrow \text{TRAINCLASSIFIER}(X, S, Y)$
 $Y' \leftarrow \text{APPLYCLASSIFIER}(\text{Classifier}, X)$
 $W \leftarrow |Y - Y'|$ {Residual for each sample is the new weight.}

 $\text{CHECKSTOPPINGCRITERION}()$
end while

Thereto, we apply the weighted variant of the Mutual Information in a forward selection framework. Starting with an empty feature set, the Mutual Information, more precisely the weighted Mutual Information with an equal weight for all samples, is computed between all available features and the target. The feature yielding the maximal Mutual Information is selected like in a simple ranking algorithm. Then the classifier/approximator is trained with this variable. The resulting residual for each data sample is of interest, because it is used to define the weights for the next selection round employing weighted Mutual Information. The next feature is chosen based on the maximum weighted Mutual Information and the classifier/approximator is retrained including the new feature channel. This repeats until the stopping criterion is met. A pseudocode description of this cycle is given in algorithm 1.

This resembles one of the basic ideas of the well known AdaBoost algorithm [12]. All samples that are misclassified are given a higher importance for the next round, while all correct samples are less important. The reasoning is simple: all correct classified samples are sufficiently explained by the current subset of features and there is the need to find those features that explain the misclassified samples.

The scaling of the values Y for real valued targets can be arbitrary, since the absolute value of the weighted mutual information is not important but its relative value to the other features, which are computed using the same weights.

Taking a look at global function approximators, like MLPs, this is not a problem. They are able to find again the decision surface they found the round before albeit it is now in a subspace of the space spanned by the features including the newly chosen one. The new dimension adds more options to find a better decision surface, but the same result is always achievable.

For classifiers with local activation functions, like RBF networks or nearest neighbour classifiers, it is a bit more complicated, especially for very low dimensional cases. The neighborhood of a sample can change dramatically by the addition of a new feature. Obviously, this effect is less dramatic in higher

dimensional spaces since new features affect the neighbourhood less. But the overall performance of the classifier may decrease in the early stages as a result. The algorithm will try to correct this based on the new residuals and chose features that compensate for the newly introduced errors, but as a consequence the error rates jump up and down.

Finding a good stopping criterion can be difficult but crucial, especially for local approximators. The algorithm starts with an empty feature subset and adds one variable to the final subset in each step until a predefined number of features is reached, or the approximation result does not improve further. If the best new subset improves more than a threshold ε , which can be negative to allow a decreasing performance, the new subset is confirmed, and another round begins, otherwise the algorithm terminates. Other possible stopping criteria are a fixed number of rounds, which equals the number of chosen features in the end, or a certain approximation error of the resulting classifier/approximator.

4 Related Work

The use of *Mutual Information* for feature selection is quite common. Besides the simple ranking approach [3], which cannot handle redundancies at all, a notable representative is the MIFS algorithm [13] and its extensions. The MIFS algorithm approximates the *Joint Mutual Information* by sums over pairwise *Mutual Information* between features. These approaches are pure filter methods and don't take the learning machine into account, and hence, do not compensate for the bias (as in the sense of the bias-variance dilemma) introduced.

The wrapper methods with the basic forward and backward selection methods [9] care for the bias, but require much time to search for good feature sets. Some proposed methods, like floating search algorithms, increase the number of searched subsets by combining forward and backward steps, or add or remove multiple features at once. This increases the required time even further and is not feasible for larger data sets. The other direction tries to reduce the number of tested subspaces, hopefully without missing the interesting ones.

Combining MI-based filters with wrapper methods is one approach. In [14] wrappers are used to refine candidate subsets provided by a incremental filter method based on MI related measures. [15] applies a *Mutual Information* based relevancy and redundancy filter to prune the input variables for a subsequent genetic wrapper search strategy. The *Chow-Liu trees* employed in [16] are used to limit the candidate variables in each deterministic forward selection step. The construction of these trees is again a filterlike preprocessing and operates with *Mutual Information*, too. All of these methods operate with the MI between the input channels and the labels, or the input channels with each other, but do not take into account the actual classifier output.

One of the few methods that uses the output of a learning machine for MI computation in the context of feature selection is presented in Torkkola [17]. The idea is based on the information theoretic learning framework and computes the

Quadratic Mutual Information between the output and the desired target to adjust a feature transformation using a gradient ascent. The method itself is a filter approach, because the learning machine only learns the feature transformation and does not provide the classification or approximation results.

The most similar existing approach is the *Residual Mutual Information* (RMI) algorithm [18]. It computes the basic Mutual Information between the target and the residual produced by a learning machine. In this case, the idea of using the residual is different. It implies, that input variables that have information about the error the classifier or approximator is going to make are useful for reducing that error and hence those variables are chosen.

5 Experiments

For evaluating the algorithm presented in this paper, it is compared to related approaches on data sets from the UCI repository [19] and one larger artificial dataset generated with known properties (200 input variables, low intrinsic dimensionality (7), linear, non-linear and XOR functional dependencies, redundancies and noise on all variables). The used classifier were the $k = 3$ nearest neighbour classifier and a Multi Layer Perceptron with two hidden layers with 20 and 10 hidden units, respectively. These classifiers took the role of the black box for the wrapper as well as final classification instance after the feature selection process. The hyperparameters for the classifiers were fixed for all experiments, since the model selection problem is not considered here. Thus, there were no adjustments for the different data sets or algorithms and there will be a rather large bias in the error. During the feature selection process, a 3-fold cross-validation data split was used to estimate the valid features, while for the final predictor evaluation itself we used a 10-fold cross-validation [20]. All problems are binary classification tasks, hence the *balanced error rate* (BER) was used as error measure.

The proposed method using the weighted Mutual Information (WMI) is compared to different algorithms: basic *Sequential Forward Selection* (SFS), *Mutual Information for Feature Selection* (MIFS with $\beta = 0.15$) and *Chow-Liu trees for feature selection* (CLT-FS) and residual *Mutual Information* (RMI). References for those algorithms are included in the previous section. The results are summarized in Table 1 for the kNN and Table 2 for the MLP.

By looking at the numbers in the tables, it is obvious that the conjecture from section 3 regarding local classifiers is true. The performance of the proposed algorithm with the nearest neighbour approach is moderate at best. When combined with the MLP much better results are achieved using the weighted Mutual Information approach.

The number of required steps of adapting and evaluating a classifier are the least besides the filter and the RMI method. Especially for big data sets the linear dependency on the number features is beneficial compared to the quadratic dependency of the SFS or the logarithmic one for the CLT method [16].

Table 1. The results for the different data sets and feature selection methods using the kNN. The balanced error rate is given in percent. The number of chosen features and the number of classifier training and evaluation cycles) for one crossvalidation step are shown in parentheses.

Data Set	Ionosphere	Spambase	GermanCredit	Breast Cancer	Artifical Data
Features	34	57	24	30	200
Samples	351	4601	1000	569	3000
All	23.78(34/-)	10.84(57/-)	36.33(24/-)	3.55(30/-)	35.36(200/-)
SFS	12.04(5/189)	8.44(12/663)	31.61(7/164)	4.21(6/189)	16.20(8/1764)
MIFS	11.80(5/-)	8.65(19/-)	33.90(6/-)	4.36(5/-)	28.65(3/-)
CLT-FS	12.19(6/39)	15.97(6/76)	34.89(5/28)	4.42(4/30)	25.89(8/202)
RMI	13.82(5/6)	23.62(3/4)	35.45(5/6)	4.49 (3/4)	24.30(4/5)
WMI	11.57(5/6)	10.73(10/11)	33.31(8/9)	4.48(6/7)	29.52(5/6)

Table 2. The results for the different data sets and feature selection methods using the MLP. The balanced error rate is given in percent. The number of chosen features and the number of MLP training and evaluation cycles are shown in parentheses.

Data Set	Ionosphere	Spambase	GermanCredit	Breast Cancer	Artifical Data
Features	34	57	24	30	200
Samples	351	4601	1000	569	3000
All	20.08(34/-)	13.81(57/-)	41.70(24/-)	13.78(30/-)	33.65(200/-)
SFS	18.47(3/130)	17.39(8/477)	39.06(4/110)	13.44(4/140)	20.11(7/1572)
MIFS	24.54(5/-)	16.29(19/-)	37.47(6/-)	12.48(5/-)	26.87(3/-)
CLT-FS	18.12(6/38)	17.26(9/97)	38.52(3/24)	9.37(8/37)	24.08(9/217)
RMI	17.08(5/6)	13.93(54/55)	39.73(15/16)	8.58(5/6)	21.74(6/7)
WMI	16.97(5/6)	16.41(9/10)	39.52 (6/7)	8.03(3/4)	19.29(6/7)

6 Conclusion

In this paper, we introduced an algorithm using the weighted Mutual Information for effective feature selection.

The proposed algorithm aims at a low number of cycles a classifier has to be trained and evaluated during the feature selection process. The number for adapting and testing cycles is only linear in the number of features for the WMI algorithms, while achieving similar results compared to wrapper approaches using more cycles.

Based on the results for the kNN and the MLP, we conjecture, that using the residual is less useful if it is a binary value only. More sophisticated information in the residual can be exploited better.

Based on tests presented, it is hard to determine if using the classifier output in form of the residual (RMI algorithm) or as a weighting for the selection (WMI) is the better overall approach, but at least for classifiers with global activation functions the weighted Mutual Information approach is preferable. More extensive studies are required to answer this question conclusively.

References

1. Kohavi, R., John, G.H.: Wrappers for feature subset selection. *Artifical Intelligence* 97, 273–324 (1997)
2. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. *Journal of Machine Learning Research* 3, 1157–1182 (2003)
3. Torkkola, K.: Information-Theoretic Methods. In: *Feature Extraction Foundations and Applications StudFuzz 207*, pp. 167–185. Springer, Heidelberg (2006)
4. LeCun, Y., Denker, J., Solla, S., Howard, R.E., Jackel, L.D.: Optimal Brain Damage. In: *Advances in Neural Information Processing Systems*, vol. 2, pp. 598–605. Morgan Kaufmann, San Francisco (1990)
5. Neal, R.M.: *Bayesian Learning for Neural Networks*. Springer, Heidelberg (1996)
6. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene Selection fo Cancer Classification Using Support Vector Machines. *Machine Learning* 46 (2002)
7. Khan, S., Bandyopadhyay, S., Ganguly, A.R., Saigal, S., Erickson, D.J., Protopopescu, V., Ostrouchov, G.: Relative performance of mutual information estimation methods for quantifying the dependence among short and noisy data. *Physical Review E* 76, 026209 1–15 (2007)
8. Schaffernicht, E., Kaltenhaeuser, R., Verma, S.S., Gross, H.-M.: On estimating mutual information for feature selection. In: Diamantaras, K., Duch, W., Iliadis, L.S. (eds.) *ICANN 2010*. LNCS, vol. 6352, pp. 362–367. Springer, Heidelberg (2010)
9. Reunanen, J.: Search Strategies. In: *Feature Extraction Foundations and Applications StudFuzz 207*, pp. 119–136. Springer, Heidelberg (2006)
10. Guiasu, S.: *Information Theory with Applications*. McGraw-Hill Inc., New York (1977)
11. Kraskov, A., Stögbauer, H., Grassberger, P.: Estimating Mutual Information. *Physical Review E* 69 (2004)
12. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55, 119–139 (1997)
13. Battiti, R.: Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks* 5(4), 537–550 (1994)
14. Peng, H., Long, F., Ding, C.: Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. *IEEE Trans. Pattern Analysis and Machine Intelligence* 27, 1226–1238 (2005)
15. Van Dijck, G., Van Hulle, M.M.: Speeding up the wrapper feature subset selection in regression by mutual information relevance and redundancy analysis. In: Kollias, S.D., Stafylopatis, A., Duch, W., Oja, E. (eds.) *ICANN 2006*. LNCS, vol. 4131, pp. 31–40. Springer, Heidelberg (2006)
16. Schaffernicht, E., Stephan, V., Gross, H.-M.: An efficient search strategy for feature selection using chow-liu trees. In: de Sá, J.M., Alexandre, L.A., Duch, W., Mandic, D.P. (eds.) *ICANN 2007*. LNCS, vol. 4669, pp. 190–199. Springer, Heidelberg (2007)
17. Torkkola, K.: Feature Extraction by Non Parametric Mutual Information Maximization. *Journal of Machine Learning Research* 3, 1415–1438 (2003)
18. Schaffernicht, E., Moeller, C., Debes, K., Gross, H.-M.: Forward feature selection using Residual Mutual Information. In: *17th European Symposium on Artificial Neural Networks, ESANN 2009*, pp. 583–588 (2009)
19. Newman, D.J., Hettich, S., Blake, S.L., Merz, C.J.: UCI Repository of machine learning databases (1998), <http://www.ics.uci.edu/~mlearn/MLRepository.html>
20. Reunanen, J.: Overfitting in Making Comparisons Between Variable Selection Methods. *Journal of Machine Learning Research* 3, 1371–1382 (2003)

Face Prediction from fMRI Data during Movie Stimulus: Strategies for Feature Selection

Jukka-Pekka Kauppi¹, Heikki Huttunen¹, Heikki Korkala¹,
Iiro P. Jääskeläinen^{2,3}, Mikko Sams^{2,3}, and Jussi Tohka¹

¹ Tampere University of Technology, Dept. of Signal Processing, Tampere, Finland

² Aalto University School of Science, Dept. of Biomedical Engineering and Computational Science, Espoo, Finland

³ Aalto University, Advanced Magnetic Imaging Centre, Espoo, Finland
`{jukka-pekkka.kauppi,heikki.huttunen,jussi.tohka}@tut.fi`

Abstract. We investigate the suitability of the multi-voxel pattern analysis approach to analyze diverse movie stimulus functional magnetic resonance imaging (fMRI) data. We focus on predicting the presence of faces in the drama movie based on the fMRI measurements of 12 subjects watching the movie. We pose the prediction as a regression problem where regression coefficients estimated from the training data are used to estimate the presence of faces in the stimulus for the test data. Because the number of features (voxels) exceeds the number of training samples, an emphasis is placed on the feature selection. We compare four automatic feature selection approaches. The best results were achieved by sparse regression models. The correlations between the face presence time-course predicted from fMRI data and manual face annotations were in the range from 0.43 to 0.62 depending on the subject and pre-processing options, i.e., the prediction was successful. This suggests that proposed methods are useful in testing novel research hypotheses with natural stimulus fMRI data.

Keywords: Natural stimulation, brain imaging, regression.

1 Introduction

Functional magnetic resonance imaging (fMRI) studies based on continuous, complex stimuli such as movies allow investigation of the brain functions in more natural contexts compared with the traditional, highly controlled experimental designs [13]. Because the resulting functional brain data is difficult to interpret due to complex neural spatio-temporal interactions in the brain during the stimulus, powerful and flexible analysis approaches are necessary to better understand this type of data. Multi-voxel pattern analysis (MVPA), aiming at modeling the multivariate relationship between fMRI measurements and the stimulus using machine learning algorithms, is a popular approach in predicting and decoding brain states [4,10]. Typically, MVPA is applied to fMRI data sets acquired during strictly controlled experiments.

In this paper, we investigate the suitability of the MVPA approach to analyze diverse movie stimulus fMRI data. We focus on predicting the presence of faces in the drama movie "Crash" (Paul Haggis, Lions Gate Films, 2005) based on the fMRI measurements of the subjects watching the movie. Compared with the data from controlled experiments, movie data is considerably more complicated to analyze because it embeds a subject to an extremely diverse environment with rapidly changing scenes and emotional content, activating many brain regions simultaneously.

We concentrate on predicting the presence of faces in the movie because this question is already well-established in previous studies using MVPA in controlled experiments (see, e.g., [2,4]) and it is important to investigate if the findings in these experiments can be generalized to more natural conditions [13]. Novel research questions may be investigated using similar tools if the face prediction can be performed reliably. This is not the first time when MVPA is used for natural stimulus fMRI data. In the 2006 and 2007 Pittsburgh Brain Activity Interpretation Competitions (PBAIC) (<http://www.lrdc.pitt.edu/ebc/2006/competition.html>, <http://www.lrdc.pitt.edu/ebc/2007/competition.html>), the task was to predict the mental states of subjects based on the fMRI data collected during watching movie clips (2006 PBAIC) or a virtual reality task (2007 PBAIC). The prediction of the presence of faces was a sub-task in both competitions. However, our "Crash" data is more naturalistic than the 2006 PBAIC data, in which each session consisted of consecutive clips of short movie segments. Moreover, unlike in the PBAIC 2006, where subjects themselves rated the presence of faces, we use the same manually collected face annotation for each subject. Also, the virtual reality task in the 2007 PBAIC is different from movie viewing.

We predict the prevalence of the face in the movie based on the fMRI signal activity from several voxels at a single time point. Because the number of available features P (the number of voxels) exceeds the number of samples N (the number of time points), the most critical part of the prediction algorithm is dimension reduction as in many other fMRI-applications of MVPAs (e.g. [9]). To this aim, we compare four feature selection methods: *stepwise regression (SWR)* [3], *simulated annealing (SA)* (see e.g.,[8]), *Least Absolute Shrinkage and Selection Operator (LASSO)* [3,14] and *Least Angle Regression (LARS)* [1,3].

2 Methods and Materials

2.1 Regression Model and Feature Selection

The face prediction problem could be posed as a classification problem, where the classes are "face" and "non-face". However, we predict a continuous-valued face annotation using linear regression because this approach is well-suited to movie stimulus fMRI data due to rapid and unexpected changes of the movie scenes. For example, continuous-valued annotation allows interpreting the extent of the face prevalence during each TR (repetition time) despite of the relatively low temporal resolution of the fMRI signal. In addition, although the face is present

in the movie frame, it may be partially occluded, shadowed or far away, thus decreasing the "face response" measured by the fMRI. There are also numerous other stimuli in the movie which compete from the attention of the subject.

The training consists of estimating the unknown parameters β of the regression model, which is then used to predict the face-prevalence on independent test data. The simplest approach for estimation is the ordinary least squares (OLS) regression. Denote the annotation time-course (convolved with expected hemodynamic response) by $\mathbf{y} = (y_1, y_2, \dots, y_N)^T$ and the fMRI time series from the k 'th voxel ($k = 1, 2, \dots, P$) by $\mathbf{x}_k = (x_{1k}, x_{2k}, \dots, x_{Nk})^T$. All time series and the annotation time-course are standardized as explained in [1]. The linear model for the annotations explained by the fMRI measurements is $\mathbf{y} = \mathbf{X}\beta + \varepsilon$ with the $N \times P$ matrix $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_P)$, the parameters $\beta = (\beta_1, \dots, \beta_P)^T$ and the residual term $\varepsilon \in \mathbb{R}^N$.

The OLS estimate $\hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$ is found by minimizing the training error $\|\mathbf{y} - \mathbf{X}\beta\|^2$. However, the solution assumes that $N \geq P$, which does not hold in our case. Therefore, the OLS has to be accompanied by a feature selection algorithm that chooses at most P meaningful features and discards the remaining columns from the matrix \mathbf{X} . We will next review the four feature selection methods assessed in this work.

Sequential forward selection (SFS) is an intuitive search heuristic for feature selection [3]. SFS starts with no features in the model and adds features to it until a stopping criterion is satisfied or the whole feature set is included in the model. In each iteration, the best ranked feature is added to the model according to a predefined model selection criterion. In addition to forward steps, the method also performs backward elimination in case some of the features become unnecessary after more features are included in the model.

When applied specifically for regression, the SFS is called *Stepwise Regression* (SWR) [3]. In both steps, statistical hypothesis testing regarding the variable coefficients is performed using the F -statistic. The drawback of SWR is that the method typically converges to a local optimum due to iterative selection procedure. Also the statistical inference procedure faces the multiple comparisons problem because several dependent F -tests are applied to the same data. This complicates interpretation of the p -values.

Simulated Annealing (SA) has been used for feature selection [8] to avoid local minima. SA is a randomized search heuristic with roots in condensed matter physics, where slowed-down cooling is used to reduce the defects of the material by allowing the molecule configuration to reach its global minimum state. The method has been successfully used in various optimization problems with multiple local extrema. SA feature selection is a *wrapper method*, which iteratively evaluate the performance of candidate feature subsets by training a regression model. SA starts with the empty feature subset, and at each iteration attempts to add or remove a random feature from the set. The change in cross-validated prediction error is then used to determine whether the new subset is accepted. All improved results are accepted, while worse solutions are accepted at random with probability $\exp(-\Delta_{\text{error}}/T)$, where $\Delta_{\text{error}} = \varepsilon_{\text{new}} - \varepsilon_{\text{old}}$ is the change in

error and T is the simulated temperature. The temperature is initialized to a high value where almost all configurations are accepted, and it is decreased at each iteration according to the rule $T \leftarrow \alpha T$ with $\alpha < 1$. We added an extra penalty term for the number of features to favor simple solutions. The size of the penalty and the cooling parameter α were selected by cross-validation.

An alternative to the wrapper approach is to embed the feature selection into the performance criterion, resulting in *embedded feature selection methods*, which introduce a penalty for the number of nonzero coefficients in the regression model. LASSO (*Least Absolute Shrinkage and Selection Operator*) regression method enforces sparsity via l_1 -penalty by optimizing the constrained LS criterion [14]:

$$\min_{\beta} \|\mathbf{y} - \mathbf{X}\beta\|^2 \text{ subject to } \|\beta\|_1 = \sum_{j=1}^P |\beta_j| \leq t, \quad (1)$$

where $t \geq 0$ is a tuning parameter. When t is large, this is identical to the OLS solution. With small values of t , the solution becomes shrunken and sparse version of the OLS solution where only a few of the coefficients β_j are non-zero.

The *Least Angle Regression* algorithm (LARS) is another feature selection method applicable for sparse regression problems with very similar regularization path to that of LASSO [1]. LARS iteratively adds the features that are most correlated with the residual of the current model. It increases the coefficient of the selected features towards the OLS solution until the feature no longer has the highest correlation with the residual. Then, LARS chooses a new direction equiangular between the first and the most highly correlated feature and continues moving along this direction until another feature with higher correlation is found. The procedure is continued until all predictors are included in the model. Finally, the resulting β is selected along the regularization path by cross-validation. Also the LASSO solution can be found by modifying the basic LARS algorithm. We used LARS and LASSO implementations from [11].

2.2 Functional MRI Experiment

Functional MRI was measured from 12 subjects while they watched the last 36 minutes of the drama movie "Crash" (Lions Gate Films, 2005; for details, see [5]). Preprocessing of the data included motion correction, detrending, spatial smoothing, and registration of the functional images to a stereotactic template (for details, see [7]). The data for each subject was acquired in two sessions, lasting 14 ($N = 244$ with $TR = 3.4$ s) and 22 minutes ($N = 382$), respectively. For each subject, we used the first session for training and the second session for evaluating the prediction performance. In the feature selection phase, 10-fold cross-validation was used to select the parameter t for LASSO and LARS, p -values for SWR, and cooling and regularization parameters for SA. The test session was not used in any way during the training or feature selection.

The face annotation was collected by manually detecting one or more faces during the periods of 1 s. Because the sampling rate of the face annotation was

higher than that of the fMRI measurements, we integrated the face annotations over a single fMRI measurement. We convolved the resulting annotation with the double gamma model of the hemodynamic response for several hemodynamic lags, and evaluated prediction performances separately for each lag.

Before automatic feature selection, we reduced the number of features (voxels) in the original large feature set including over 200000 voxels across the whole brain. Two different initial feature subsets were formed. The first subset contained the measurements across fusiform cortex, which is expected to be important for the face detection [6]. The region of interest (ROI) was localized using Harvard-Oxford probabilistic cortical atlas from the FSL software package [12] with 50 % threshold. It contained $P = 601$ features. As face detection in the brain may be distributed [2], we chose a second feature subset including voxels across cortex. We further reduced the number of features by preserving only measurements showing the pairwise averaged inter-subject correlations (ISCs) larger than 0.30 across the 12 subjects (see [7]). This reduced the number of features down to 1480.

3 Results

Figures 1 and 2 present the prediction performances of the four methods. One box plot contains the results of 12 subjects, and several box plots are shown for different hemodynamic lags. Correlation coefficient between the face annotation and the model estimate was used as the performance measure in accordance with PBAIC. The best test predictions were obtained using the LARS (Fig. 1) having the median correlation coefficients over 0.50 for both fusiform and cortex feature sets (see e.g. results with the hemodynamic lags of 7-9 s in Figs. 1C-D). The correlations were slightly higher for the fusiform than for the cortex features, even though correlations for the cortical features were higher for the training session (Figs. 1A-B). The highest correlations for the training session were obtained using the hemodynamic lag of 6 s whereas the lag of 8 s provided the best results for the test session.

Figure 2 shows the test prediction performance using LASSO, SWR, and SA with the fusiform feature set. The results of LASSO were almost as good as the results of LARS. Feature analysis between LASSO and LARS indicated that both methods found common features for all the subjects but the solution of the LASSO contained features not present in the solution of LARS and vice versa. The performances of SWR and SA were slightly lower than LASSO and LARS, but they resulted in simpler models with fewer features. Table 1 summarizes the prediction results for the test session using the lag of 8 s, which provided the best prediction performance.

Figure 3 displays the prediction curve of the LARS for the best predicting subject across a part of the test session. The prediction followed the fluctuations of the faces in the movie accurately, indicating that the found feature set was responsible for the face detection during the movie. The anatomical locations of some of the automatically selected features are also shown in Fig. 3. For this

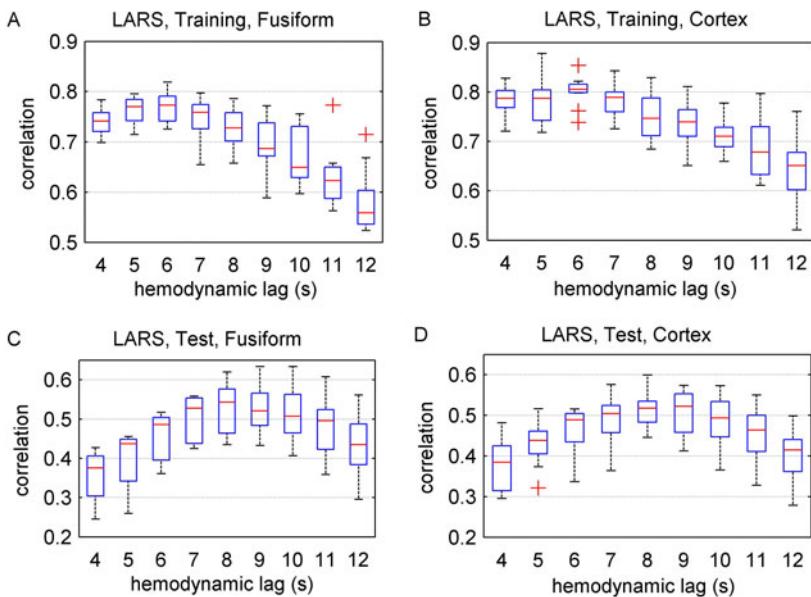


Fig. 1. Correlation coefficients between the face annotation and the model estimate using LARS for 12 subjects: (A) training session results for the fusiform features, (B) training session results for the cortical features, (C) test session results for the fusiform features, and (D) test session results for the cortical features.

Table 1. Test prediction performance and numbers of selected features in the final models. The hemodynamic lag was 8 s. LASSO and LARS provided the best predictions with nearly equal results, but SA and SWR resulted in simpler models with fewer features.

		Prediction Performance				Number of Features			
		SWR	SA	LASSO	LARS	SWR	SA	LASSO	LARS
Fusiform	mean	0.49	0.48	0.53	0.53	12.1	10.6	38.3	31.0
	max	0.59	0.60	0.62	0.62	19	20	60	49
	min	0.33	0.42	0.45	0.43	7	5	19	9
	std	0.08	0.06	0.06	0.06	3.2	4.7	14.7	12.9
Cortex	mean	0.44	0.43	0.52	0.51	18.8	12.3	47.6	42.9
	max	0.54	0.53	0.59	0.60	31	25	67	83
	min	0.35	0.30	0.45	0.45	11	1	21	16
	std	0.07	0.07	0.05	0.04	5.9	6.9	15.2	18.0

subject, we found 25 features located in five brain regions: temporal occipital fusiform cortex (TOFC), occipital pole (OP), inferior lateral occipital cortex (LOC), occipital fusiform gyrus (OFG), and lingual gyrus (LiG).

We analyzed whether exactly the same cortical features were selected across the subjects. Using the criterion that the same feature needs to be found at

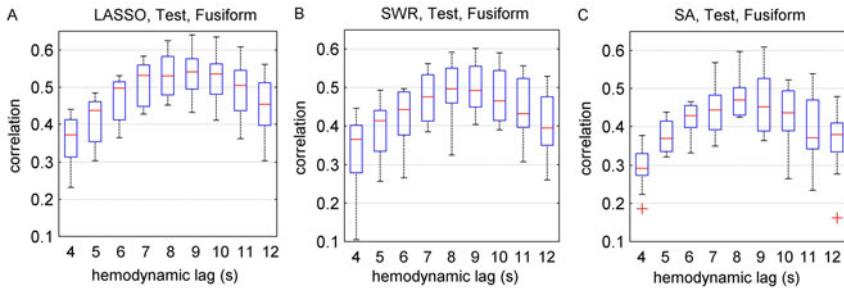


Fig. 2. Test prediction performance for the 12 subjects using the fusiform features

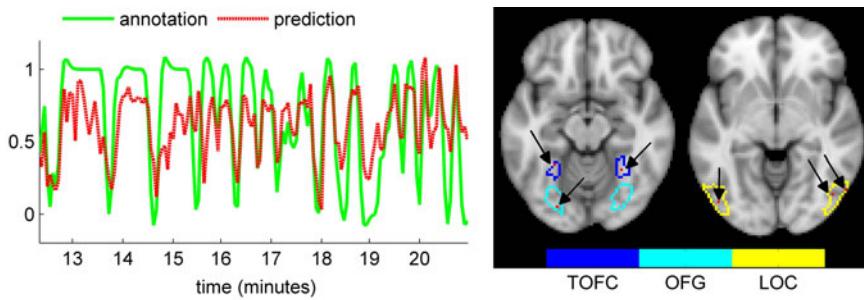


Fig. 3. The best prediction curve shown for the last part of the test session using LARS with the anatomical locations of some of the found cortical features (hemodynamic lag 8 s, correlation coefficient 0.60). Two axial slices show the location of 6 features in three brain regions.

least across 3 out of 12 subjects, we found features in the following brain regions (the number of features in the given brain region is shown in parentheses for LASSO/LARS/SA/SWR): superior LOC (2/3/0/0), inferior LOC (7/8/0/2), LiG (1/2/0/0), TOFC (5/10/0/0), OFG (3/1/0/0), planum temporale (3/1/0/0), and OP (3/7/0/0). Hence, LASSO and LARS solutions were more consistent across subjects than SA and SWR solutions.

4 Conclusion

In this work, we successfully predicted the presence of faces in the eventful drama movie based on the fMRI data using multi-voxel linear regression. Two different preprocessing methods (ROI-based, ISC-based) were used to select initial voxel sets for prediction, and the final subsets were optimized using four automatic feature selection methods (SWR, SA, LARS, and LASSO). The best predictions were obtained with sparse regression models LARS and LASSO, but also SWR and SA provided good results. Importantly, the prediction was successful using

the voxel set that initially contained voxels across several cortical regions, i.e., specific anatomical prior knowledge was not used in the initial selection. This suggests that proposed methods can be useful when testing novel research hypotheses with natural stimulus fMRI data. From neuroscientific point of view, our results support the view that face detection is distributed across the visual cortex, albeit the fusiform cortex has a strong influence on the face detection.

Acknowledgments. Supported by the Academy of Finland, (grants 129657, Finnish Programme for Centres of Excellence in Research 2006-2011, and 130275) and the aivoAALTO project funding of the Aalto University.

References

1. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least angle regression. *Annals of Statistics* 32(2), 407–499 (2004)
2. Hanson, S.J.J., Schmidt, A.: High-resolution imaging of the fusiform face area (FFA) using multivariate non-linear classifiers shows diagnosticity for non-face categories. *NeuroImage* 54(2), 1715–1734 (2011)
3. Hastie, T., Tibshirani, R., Friedman, J.: *The elements of statistical learning: Data mining, inference, and prediction*. Series in Statistics. Springer, Heidelberg (2009)
4. Haxby, J.V., Gobbini, M.I., Furey, M.L., Ishai, A., Schouten, J.L., Pietrini, P.: Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science* 293(5539), 2425–2430 (2001)
5. Jääskeläinen, I.P., Koskentalo, K., Balk, M.H., Autti, T., Kauramäki, J., Pomren, C., Sams, M.: Inter-subject synchronization of prefrontal cortex hemodynamic activity during natural viewing. *The Open Neuroimaging Journal* 2, 14–19 (2008)
6. Kanwisher, N., McDermott, J., Chun, M.M.: The fusiform face area: a module in human extrastriate cortex specialized for face perception. *J. Neurosci.* 17(11), 4302–4311 (1997)
7. Kauppi, J.P., Jääskeläinen, I.P., Sams, M., Tohka, J.: Inter-subject correlation of brain hemodynamic responses during watching a movie: localization in space and frequency. *Frontiers in Neuroinformatics* 4, 5 (2010)
8. Lin, S.W., Lee, Z.J., Chen, S.C., Tseng, T.Y.: Parameter determination of support vector machine and feature selection using simulated annealing approach. *Appl. Soft Comput.* 8, 1505–1512 (2008)
9. Mitchell, T.M., Hutchinson, R., Niculescu, R.S., Pereira, F., Wang, X., Just, M., Newman, S.: Learning to Decode Cognitive States from Brain Images. *Machine Learning* 57(1), 145–175 (2004)
10. Norman, K., Polyn, S., Detre, G., Haxby, J.: Beyond mind-reading: multi-voxel pattern analysis of fMRI data. *Trends in Cognitive Sciences* 10(9), 424–430 (2006)
11. Sjöstrand, K.: Matlab implementation of LASSO, LARS, the elastic net and SPCA (2005), <http://www2.imm.dtu.dk/pubdb/p.php?3897>
12. Smith, S.M., et al.: Advances in functional and structural MR image analysis and implementation as FSL. *NeuroImage* 23(suppl. 1), S208–S219 (2004)
13. Spiers, H., Maguire, E.: Decoding human brain activity during real-world experiences. *Trends in Cognitive Sciences* 11(8), 356–365 (2007)
14. Tibshirani, R.: Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society, Series B* 58, 267–288 (1994)

The Authentication System for Multi-modal Behavior Biometrics Using Concurrent Pareto Learning SOM

Hiroshi Dozono¹, Shinsuke Ito¹, and Masanori Nakakuni²

¹ Faculty of Science and Engineering, Saga University,
1-Honjyo Saga 840-8502 Japan
hiro@dna.ec.saga-u.ac.jp

² Information Technology Center, Fukuoka University,
8-19-1, Nanakuma, Jonan-ku, Fukuoka 814-0180 Japan
nak@fukuoka-u.ac.jp

Abstract. We have proposed the integration of behavior biometrics using Supervised Pareto learning SOM to improve the accuracy of authentication. For small systems such as mobile devices, this method may be heavy, because of the memory usage or computational power. In this paper, we propose the application of Concurrent Pareto learning SOM, which uses a small map for each user. The performance of this method is confirmed by authentication experiments using behavior biometrics of keystroke timings and key typing sounds.

1 Introduction

Recently, biometric authentication attracts more attention as the security system. Biometric authentication is classified to two types. The first type is biometric authentication using biological biometrics such as fingerprint, iris, and vein pattern. The authentication system using biological biometrics generally shows high accuracy. However, it needs special hardware for implementation. Furthermore, biological biometrics can be copied because they are static information. The other type is biometric authentication using behavior biometrics such as keystroke timings, pen patterns, and pressures writing signs or handwritten patterns and walking patterns. For implementing to a computer system or mobile system, the device that is equipped with the system can be used to obtain the behavior biometrics. For example, many recent mobile devices are equipped with touch panels, which can be used to obtain sign or handwritten patterns. And it is difficult to imitate behavior biometrics because it is dynamic information. However, the accuracy of the authentication system using behavior biometrics is less than that of using biological biometrics because of the fluctuation of the behaviors.

For solving this problem, integration of several biometrics is accepted to be effective[1]. We proposed the authentication method using integrated information of multi-modal behavior biometrics. For example, the integration of the pen

speed and pen pressures of handwritten patterns, integration of the features of handwritten patterns and keystroke timing for Tablet PC[2], and integration of keystroke timings and key typing sounds obtained from keyboard and microphone equipped to the notebooks[3] were proposed. For the authentication system, we proposed Supervised Pareto learning Self Organizing Map(SP-SOM) which can integrate the multi modal biometrics naively based on the concept of Pareto optimality. SP-SOM registers the input data of several users and can identify the user from the test input for authentication. However, the number of users may be one or a few for some systems such as mobile devices. For such system, large map that requires many memories and computations is not preferred.

In this paper, we propose the application of Concurrent Pareto learning SOM(CP-SOM) for solving this problem. Concurrent SOM[4] uses the small map for each class, and each map is the classifier of single class, which identifies if the test data belongs to the class or not. The amount of memories and computation will be much less compared with those of SP-SOM that is used for registering the input vectors of several users. For the identification, CP-SOM does not use the Euclid distance to keep the scale free feature of SP-SOM, and it uses the changes of the size of Pareto optimal set, which is defined from the concept of Pareto optimality. Some experiments of authentication using the behavior biometrics of keystroke timings and key typing sounds are made to confirm the performance of this method.

2 Pareto Learning SOM

2.1 Pareto Learning SOM for Multi Modal Vector

The multi-modal vector $(\{\mathbf{x}_1\}, \{\mathbf{x}_2\}, \dots, \{\mathbf{x}_n\})$ is the vector composed of multi kind of vectors and attributes. For example, the keystroke timings and key typing intensities are the features used for the authentication using key typing features. In multi modal vector, each vector and attribute is described in different unit and scale, and the availability for the classification may be different. Conventional Self Organizing Map(SOM) can learn multi modal vector using the simply concatenated vector $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ or concatenated vector with weight values $(w_1\mathbf{x}_1, w_2\mathbf{x}_2, \dots, w_n\mathbf{x}_n)$ as an input vector. Without using weight values, the map is dominated by the largely scaled vectors or easily affected by unreliable vectors. Using weight values, the map heavily depends on the weight values, and it is difficult to select optimal weight values.

Pareto learning SOM(P-SOM) directly uses the multi-modal vector $\mathbf{x} = (\{\mathbf{x}_1\}, \{\mathbf{x}_2\}, \dots, \{\mathbf{x}_n\})$ as an input vector based on Pareto optimality. For each vector, \mathbf{x}_i the objective function is defined as $f_i(\mathbf{x}, U^{jk}) = |\mathbf{x}_i - \mathbf{m}_i^{ij}|$ for unit U^{jk} on the map, where $\mathbf{m}^{ij} = (\{\mathbf{m}_1^{jk}\}, \{\mathbf{m}_2^{jk}\}, \dots, \{\mathbf{m}_n^{jk}\})$ is the vector associated to U^{jk} . The Pareto winner set $P(\mathbf{x})$ for a input vector \mathbf{x} is the set of the units U^{jk} which are the Pareto optimal according to the object functions $f_i(\mathbf{x}, U^{jk})$. Thus, P-SOM is multi winner SOM and all units in $P(\mathbf{x})$ and their neighbors

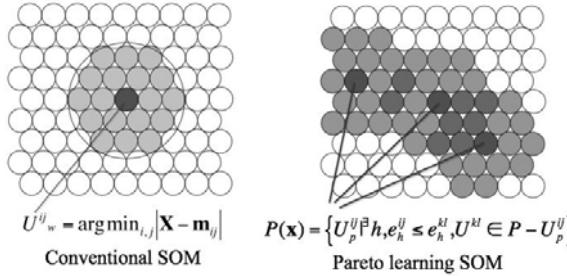


Fig. 1. Difference of the algorithm between SOM and P-SOM

are updated simultaneously. Fig.1 shows the differences of the algorithm between SOM and P-SOM. In the update phase, the units in the overlapped neighbors are updated more strongly, and it plays an important role for the integration of multi modal vectors. P-SOM is the scale free because all vectors in \mathbf{x} are handled evenly independently to the scales of \mathbf{x}_i

For a large number of the vectors in a multi modal vector, too many units may be included in Pareto winner set, because a unit can be Pareto optimal if at least one objective function is superior to others in the Pareto optimal set. For solving this problem, the definition of Pareto optimality is strengthened to M/N Pareto optimality, which requires M superior object functions from N objective functions. However, it causes another problem, which value of M is optimal.

For solving this problem, the adaptive scheme of the size of Pareto optimal set is introduced. The size of Pareto optimal set should be large at the beginning of learning and should be converged to small value at the ending. M of the M/N Pareto optimal set is adaptively adjusted for each input vector in the competitive phase as to decrease the difference between the size of the Pareto optimal set, and the optimal size that is given for each iteration step.

Using the adaptive scheme, the multi modal vector that includes many elements can be applied to P-SOM. Full Pareto learning SOM(FP-SOM) uses the input vector $\mathbf{x} = (x_1, x_2, \dots, x_n)$ which is composed of the set of 1-dimensional independent attributes. The objective function $f_i(\mathbf{x}, U^{ij}) = |x_i - m_n^{ij}|$ is given as the quantization error for each attribute. All attributes described in different scales are handled evenly.

P-SOM can integrate any kind of vector. Thus, the category vector c^i can be introduced as an independent vector to each input vector of P-SOM.

$$\hat{\mathbf{x}}^i = (\mathbf{x}^i, c^i) \quad (1)$$

$$c_j^i = \begin{cases} 1 & \mathbf{x}^i \in C_j \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

The category vector is also used for searching Pareto winner set, and it attracts the input vectors in same category closely on the map corporately with original input vector \mathbf{x} . The category of the given test vector \mathbf{x}_t is determined as $\text{argmax}\{\sum_{U^{ij} \in P(\mathbf{x}_t)} c_k^{ij}\}$ where $P(\mathbf{x}_t)$ is the Pareto optimal set of units for \mathbf{x}_t .

2.2 Concurrent Pareto learnig SOM

Concurrent SOM is a SOM which uses an individual map for each class. For example, the map for each user is organized using the features of a user in the application to authentication system(Fig.2). After organizing the maps, the input vector of a user is authenticated if the quantization error between the input vector and the vector associated with the winner unit for the input vector is smaller than the preset threshold value. Concurrent SOM is considered to be suitable for mobile system, which is used by a single user or a few users because it needs a small map for each user. However, it may be difficult to determine the threshold value for each user.

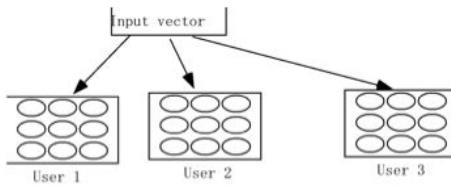


Fig. 2. Concurrent SOM for authentication system

Concurrent Pareto learning SOM(CP-SOM) uses P-SOM or SP-SOM algorithm for the learning of each map. In the application to the authentication system, the map can be applied to authenticate the user. However, P-SOM does not use the threshold value of the quantization error to keep the scale free feature of the attributes values. As the responses of SP-SOM, for the unlearned input vector, the size of Pareto optimal set $|P(\mathbf{x})|$ becomes large compared with learned data, and the size becomes larger for more different input vectors. Additionally, the category value described as $Cv = \sum_{U^{ij} \in P(\mathbf{x}_t)} c^{ij}$ becomes small for

unlearned data. These two responses to the unlearned input vector are used to identify the user by setting the threshold values to the size of Pareto optimal set and category value. The algorithm of CP-SOM used for authentication system is as follows.

CP-SOM Algorithm

Registering the input vectors

1. Initialization of the map and parameters

For each user h , initialize the vector ${}^h\mathbf{m}^{ij}$ which are assigned to unit ${}^hU^{ij}$ on the map using the 1st and 2nd principal components of the input vectors for registering as base vectors of 2-dimensional map. Initialize the size of neighbors S_n , learning rate η and size of Pareto set S_p .

2. Batch competing phase

For each user h , execute the following steps.

(1) Clear all learning buffer of units ${}^hU^{ij}$.

(2) For each input vector ${}^h\mathbf{x}^i = (\{{}^h\mathbf{x}_1^i\}, \{{}^h\mathbf{x}_2^i\}, \dots, \{{}^h\mathbf{x}_n^i\})$, execute the following steps.

(2-1) Set $M = 1$

(2-2) Search for the M/N pareto optimal set ${}^hP = \{{}^hU_p^{ab}\}$. ${}^hU_p^{ab}$ is an element of pareto optimal set hP , if for all units ${}^hU_{kl} \in {}^hP - {}^hU_p^{ab}$, existing g such that $e_g^{ab} \leq e_g^{kl}$ where

$$e_g^{kl} = |{}^h\mathbf{x}_g^i - {}^h\mathbf{m}_g^{kl}| \quad (3)$$

(2-3) if $|{}^hP| \leq Sp$ goto (3), else $M=M+1$ and goto (2-1)

(3) Add ${}^h\mathbf{x}^i$ to the learning buffer of all units ${}^hU_p^{ab} \in {}^hP$.

(4) If current step is last iteration, record hAs (average size of Pareto optimal set), hAm (average of M) and hAc (Average of category value) for all input vectors of user h in the last iteration.

3. Batch update phase

For each user h , execute the following steps.

For each unit ${}^hU^{ij}$, update the associated vector ${}^h\mathbf{m}^{ij}$ using the weighted average of the vectors recorded in the buffer of ${}^hU^{ij}$ and its neighboring units as follows.

(1) For all vectors \mathbf{x} recorded in the buffer of ${}^hU^{ij}$ and its neighboring units in distance $d \leq Sn$, calculate weighted sum \mathbf{S}^{ij} of the updates and the sum of weight values W.

$$\mathbf{S}^{i'j'} = \mathbf{S}^{i'j'} + \eta fn(d)(\mathbf{x} - {}^h\mathbf{m}^{i'j'}) \quad (4)$$

$$W^{i'j'} = W^{i'j'} + fn(d) \quad (5)$$

where ${}^hU^{i'j'}$'s are neighbors of ${}^hU^{ij}$ including ${}^hU^{ij}$ itself, η is learning rate, $fn(d)$ is the neighborhood function which becomes 1 for $d=0$ and decrease with increment of d.

(2) Set the vector ${}^h\mathbf{m}^{ij} = {}^h\mathbf{m}^{ij} + \mathbf{S}^{ij}/W^{ij}$.

Repeat 2. and 3. with changing Sn , η and Sp in pre-defined iterations.

Authentication

For user h , the input vector x_t is authenticated as follows.

(1) Finding the Pareto optimal set

Calculate ${}^hAm/N$ Pareto optimal set hP for input vector x_t using the map of user h .

(2) Applying the threshold

Calculate the category value $Cv = \sum_{U^{ij} \in {}^hP(\mathbf{x}_t)} c^{ij}$

If $|{}^hP| < {}^hAs \times TH_P$ and $Cv < {}^hAc \times TH_C$ the input vector x_t is authenticated as user h .

During registering the input vectors, adaptive scheme for the size of Pareto optimal set is applied with adjusting the M of M/N Pareto optimality in batch competing phase. At the last iteration, the average value of size of Pareto optimal set, M and category values are recorded and are used in the authentication process combined with the threshold values TH_P and TH_C .

3 Experimental Results

3.1 Biometric Authentication Using Keystroke Timings and Key Typing Sound

As the multi-modal behavior biometrics, the keystroke timings, and key typing sounds data, which are obtained from 10 users, typing a phrase "kirakira" in ten times for each user, are used in this experiment. As the behavior characteristic, each data contains 15 keystroke timing data, which are the key pushing times and intervals between keys, and 8 key typing sound data, which is the maximum amplitude of key typing sound for each key, and they are composed as 2 feature vectors of behavior biometrics. As the indexes for evaluating authentication result, FRR, which is the rate for rejecting registered user falsely, and FAR, which is the rate for accepting unregistered user falsely, are used. We have already reported that FRR=0.187, FAR for registered user=0.045 and FAR for unregistered user=0.068 using a single map sized 16×16 organized by SP-SOM using the data of 7 users as registered users and 3 users as unregistered users.

3.2 Experimental Result Using CP-SOM

In this subsection, the experimental results using CP-SOM are shown. The settings of the experiments are as follows. For each user, 5 data are used for registering and 5 remainders are used for authentication. The experiments are made with changing the size of maps, and the threshold values TH_P and TH_C . The number of iterations for learning is 25 batch cycles. The initial size of neighbors and the initial learning rate are set as 1/4 of map size and 0.3 respectively. The initial size of the Pareto optimal set and terminal size are set as 1/4 and 1/2 of the number of units on each map respectively.

Fig.3 shows the average of FRR and FAR with changing the TH_P , without using the threshold for category value. The size of the map is 6×6 . Considering the balance of FRR and FAR, the threshold value at the cross point of FRR and FAR is preferred. At the cross point, FAR=FRR=0.35, and this result is too bad as an authentication system. The size of the Pareto optimal set for registered user and other users are 6.22 and 7.77 respectively, and the difference is too small for classification. For this problem, Concurrent Full Pareto learning SOM(CFP-SOM) which uses the input vectors composed of 23 attributes taken from keystroke timings(15 attributes) and key typing sounds(8 attributes) are applied. Fig.4 shows the results. At the cross point FAR=FRR=0.121 and this result is much improved compared with previous one. The size of the Pareto

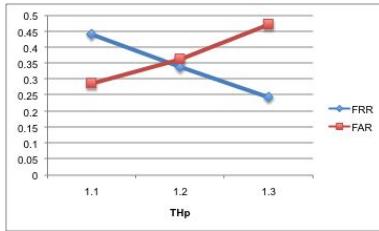


Fig. 3. FRR and FAR of the authentication experiments using CP-SOM

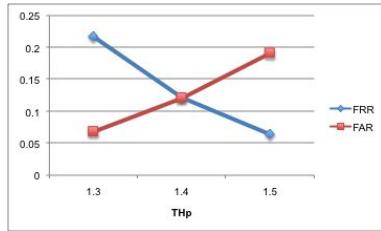


Fig. 4. FRR and FAR of the authentication experiments using CFP-SOM

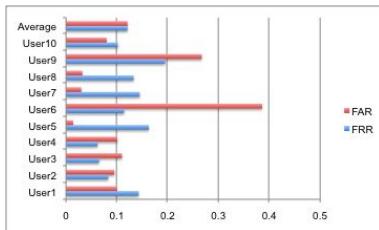


Fig. 5. FRR and FAR of each user at cross point

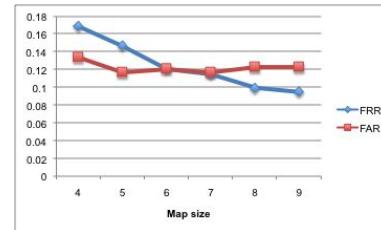
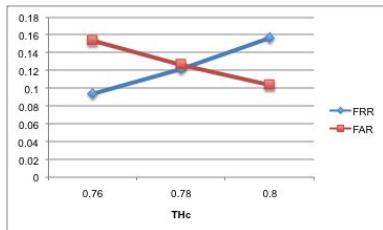
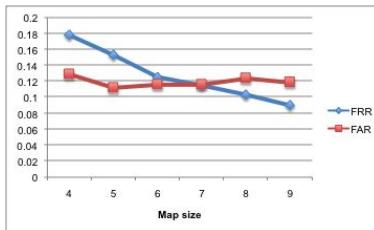


Fig. 6. FRR and FAR with changing map size($TH_p=1.4$)

optimal set for registered user and other users are 17.95 and 27.96 respectively, and the difference becomes large compared with previous case. Fig.5 shows the FRR and FAR of each user at the cross point. The FARs of user 9 and user 6 are too big. They are called as sheep who are not suitable for this authentication method. Excluding these users, the averages of FRR and FAR become 0.112 and 0.070 respectively. In the experiments using SP-SOM with single map, user 9 and user 6 were used as unregistered users. Thus, this result is superior to that of SP-SOM. Fig.6 shows the average of FRR and FAR with changing the map size. TH_p is fixed to 1.4. FAR does not change so much, and FRR decreases gradually. FARs of user 9 and user 6 (sheep) are not also improved. Excluding sheep, the averages of FRR and FAR become 0.090 and 0.061 for the map size 9×9 .

Next, the authentication experiments are made with changing the threshold for category value TH_c . Fig.7 shows the results. The map size is 6×6 . At the cross point, $FRR=FAR=0.125$. This result is almost same as that of changing TH_p . Fig.8 shows the results with changing the map size and setting both thresholds as $TH_p = 1.4, TH_c = 0.76$. These results are the almost same as that without using the threshold TH_c . It is considered that the threshold of the Pareto size and category values affect in almost same way. Thus, it is not necessary to use both thresholds and either of them is needed for classification.

**Fig. 7.** FRR and FAR with changing TH_c **Fig. 8.** FRR and FAR with changing map size($TH_P = 1.4$, $TH_C = 0.76$)

4 Conclusion

In this paper, we proposed the authentication method using Concurrent Pareto learning SOM(CP-SOM). The registered users and others are classified using size of Pareto optimal set or category value as an index without using the Euclid distances. The performance of this method is confirmed by experiments. For mobile devices, CP-SOM requires smaller memories and smaller amount of computations because it uses smaller maps, which are organized using the information of single user.

As the feature works, the method to determine the threshold values for the size of Pareto optimal set and category values is needed. As the biometric authentication system, more experiments will be needed using another behavior biometrics and using the data taken from more users.

References

1. Bolle, R., Connell, J., Pankanti, S., Ratha, S.N.: *Guide to Biometrics*. Springer, Heidelberg (2004)
2. Dokic, S., Kulesh, A., et al.: An Overview of Multi-modal Biometrics for Authentication. In: *Proceedings of The 2007 International Conference on Security and Management*, Lasvegas, pp. 39–44 (2007)
3. Dozono, H.: An Integration Method of Multi-Modal Biometrics Using Supervised Pareto Learning Self Organizing Maps. In: *Proceedings of 2008 International Joint Conference on Neural Networks*, pp. 603–607. IEEE Press, Hongkong (2008)
4. Dozono, H., Nakakuni, M.: Application of Supervised Pareto Learning Self Organizing Maps to Multi-modal Biometric Authentication. *IPSJ Journal* 49(9), 3028–3037 (2008) (in Japanese)
5. Neagoe, V.E., Ropot, A.D.: Concurrent Self-Organizing Maps for Pattern Classification. In: *Proceeding ICCI 2002 Proceedings of the 1st IEEE International Conference on Cognitive Informatics*, pp. 304–312. IEEE Press, Los Alamitos (2002)

Hermite Polynomials and Measures of Non-gaussianity

Jouni Puuronen and Aapo Hyvärinen

Dept of Mathematics and Statistics, Dept of Computer Science and HIIT,
University of Helsinki, Finland
jouni.puuronen@helsinki.fi, aapo.hyvarinen@helsinki.fi

Abstract. We first review some rigorous properties of the Hermite polynomials, and demonstrate their usefulness in estimating probability distributions as series from data samples. We then proceed to explain how these series can be used to obtain precise and robust measures of non-Gaussianity. Our measures of non-Gaussianity detect all kinds of deviations from Gaussianity, and thus provide reliable objective functions for ICA. With a linear computational complexity with respect to the sample size, our method is also suitable for large data sets.

Keywords: ICA, Hermite polynomials, non-Gaussianity.

1 Introduction

Measuring the non-Gaussianity of a random variable is a central problem in the theory of independent component analysis (ICA) and related domains. Most approaches are based either on cumulants such as kurtosis and skewness, or differential entropy. Cumulants are computationally simple but they are very sensitive to outliers which seriously limits their practical utility. Differential entropy is statistically optimal in the context of ICA estimation, but its estimation and computation is very difficult. Various compromises between the two have been proposed, for example in [6].

Cumulant-based non-Gaussianity measures can be motivated by the theory of Hermite polynomials. When a suitable polynomial series is used to approximate the probability density function (pdf) near a Gaussian density, cumulants are obtained naturally as the coefficients in the series [1,2]. The advantage of such an approach could be that we can include an arbitrary number of terms in the expansion and thus it may be possible to obtain very general approximations.

In this paper, we first review some existing theory of Hermite polynomials which gives a more general way of approximating pdf's than what has been used so far in the signal processing or machine learning literature. In particular, we show how the Hermite polynomials can be used to obtain an expansion whose coefficients can be naturally estimated as expectations of damped polynomials, which are robust against outliers. The damping is done by multiplying the polynomials by a Gaussian kernel.

Based on the Hermite polynomial expansion, we propose a family of non-Gaussianity measures which is a) derived in a principled way from a polynomial expansion, b) zero only for the Gaussian distribution, c) robust against outliers, and d) easy to compute since it is essentially obtained by expectations of analytical functions of the data.

2 The Hermite Polynomial Series

2.1 Definition

We use a following definition for the Hermite polynomials:

$$H_n(x) = (-1)^n e^{\frac{1}{2}x^2} D_x^n e^{-\frac{1}{2}x^2}, \quad (1)$$

where D_x is the derivative operator. The orthogonality and formal completeness properties of these polynomials are given by

$$\int_{-\infty}^{\infty} e^{-\frac{1}{2}x^2} H_n(x) H_m(x) dx = \sqrt{2\pi} n! \delta_{nm}, \quad (2)$$

$$\sum_{n=0}^{\infty} \frac{1}{n!} H_n(x) H_n(x') = \sqrt{2\pi} \delta(x - x') e^{\frac{1}{2}x^2}. \quad (3)$$

This means that once a function $f(x)$ on a real axis is given, and once coefficients a_n are defined as follows

$$a_n = \frac{1}{\sqrt{2\pi} n!} \int_{-\infty}^{\infty} e^{(\alpha - \frac{1}{2})x^2} H_n(x) f(x) dx, \quad (4)$$

for all $n = 0, 1, 2, \dots$, we may attempt to recover the function $f(x)$ as a series

$$f(x) = e^{-\alpha x^2} \sum_{n=0}^{\infty} a_n H_n(x). \quad (5)$$

If one substitutes the formula given in (4) into the coefficients a_n in (5), and uses the delta function identity given in (3), the factors depending on α cancel out. Thus, at least formally, these series should work with arbitrary α .

If a choice $\alpha = \frac{1}{2}$ is used, the series may be called a Gram-Charlier series. With a choice $\alpha = \frac{1}{4}$ the series should be called a Gauss-Hermite series. There may be some ambiguity about the terminology in some contexts, but some authors are strict with these conventions [3]. Thus, we have generalized these two well-known series expansions by introducing a general parameter α . It is also possible to interpret this series representation as a Gram-Charlier series of the function $e^{(\alpha - \frac{1}{2})x^2} f(x)$ with any α .

2.2 Convergence of the Series

Many authors mention some conditions that may be used to guarantee the convergence, but the ones given by Szegő are among the most rigorous.¹ Applying Szegő's results, we obtain the following convergence conditions. If a function $f(x)$ is integrable over all finite intervals, and satisfies the condition

$$\int_n^{\infty} e^{(\alpha - \frac{1}{4})x^2} x^{-5/3} (|f(x)| + |f(-x)|) dx = o\left(\frac{1}{n}\right), \quad \text{as } n \rightarrow \infty \quad (6)$$

then the series in (5) converges towards the same number as the limit

$$\lim_{n \rightarrow \infty} \frac{1}{\pi} \int_{x-\delta}^{x+\delta} f(y) \frac{\sin(\sqrt{2n}(x-y))}{x-y} dy \quad (7)$$

with some fixed $\delta > 0$ (or the series may diverge as this limit, whichever the limit does). If f further has the bounded variation property in some environment of x , then

$$\frac{\sin(\sqrt{2n}(x-y))}{x-y} \xrightarrow{n \rightarrow \infty} \pi \delta(x-y). \quad (8)$$

A formal calculation that verifies this delta identity can be computed with a change of variable. A rigorous proof can be carried out using techniques explained in literature of Fourier analysis [5]. In our context, pathological local oscillations are usually assumed to not be present. Thus, if the condition (6) holds, if f has the bounded variation property in some environment of x , and if f is continuous at x , then we know that the series (5) will converge towards the $f(x)$ at the point x .

We now have a prescription for a series representation of a given probability density function p_X . First fix a constant $0 < \alpha \leq \frac{1}{2}$. In order for the series (5) to work optimally for $f(x) = p_X(x)$, we should preprocess a data sample (x_1, x_2, \dots, x_T) so that $E(X) = 0$ and $E(X^2) = \frac{1}{2\alpha}$. Then $p_X(x)$ can be approximated with the series (5), once the integral of (4) is replaced with a sample average, giving us estimates \hat{a}_n . This scaling is not absolutely necessary, but the density function $\sqrt{\alpha/\pi} \exp(-\alpha x^2)$, which is proportional to the first term in the series (5), has the same variance $E(X^2) = \frac{1}{2\alpha}$. Other scalings would make the convergence slower.

However, standardized random variables are preferred for some applications, so we should note that if we define a variable $Y = \sqrt{2\alpha}X$, then $E(Y^2) = 1$ and its pdf will be

$$p_Y(y) \approx \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}y^2} \sum_{n=0}^N \hat{b}_n H_n\left(\frac{y}{\sqrt{2\alpha}}\right), \quad (9)$$

¹ Theorem 9.1.6 [4]. Szegő uses different convention where the Hermite polynomials are $2^{n/2} H_n(\sqrt{2}x)$ (here written using the definition (1)).

where $\hat{b}_n = \sqrt{\frac{\pi}{\alpha}} \hat{a}_n$. The coefficients b_n may equivalently be estimated by the formula

$$\hat{b}_n = \frac{1}{\sqrt{2\alpha n!T}} \sum_{t=1}^T e^{\frac{1}{2}(1-\frac{1}{2\alpha})y_t^2} H_n\left(\frac{y_t}{\sqrt{2\alpha}}\right). \quad (10)$$

Now, we propose to choose the parameter α so that the estimation of the coefficients b_n is robust against outliers. The choice $\alpha = \frac{1}{2}$ would give no robustness, because the exponential dampening term in (10) would vanish, leaving us with nothing more than expectation values of polynomials. The choices $0 < \alpha < \frac{1}{2}$ instead do give more robust formulas for coefficients \hat{b}_n , since the exponential dampening term will take weight away from the outliers.

For sake of example, consider the samples, generated out of a peculiarly shaped distribution, shown in the figure 1. This pdf consists of three pyramids close to the origin, and exponential tails.

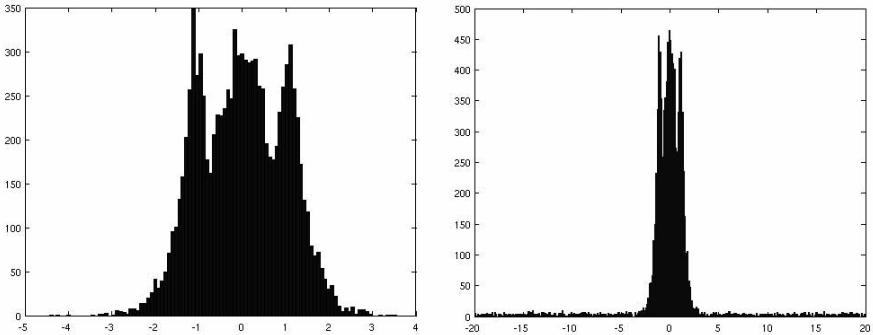


Fig. 1. On the left is a clean sample of 10000 points. On the right is otherwise the same sample but now with 100 outliers.

The figure 2 shows how well the approximation (9) works. We learn two things. Firstly, it would be a mistake to assume that the series approximation could be used only for distributions in which a reasonably small perturbation has been added to the Gaussian distribution. By computing sufficiently large number of terms in the series, the series can be made to converge to all kinds of distributions. Secondly, it would be a mistake to assume that outliers would make the computation of higher order terms impossible. With $\alpha = \frac{1}{4}$, the exponential dampening term provides a very good protection against outliers.

There are many interesting phenomena related to the convergence of these series, but we don't have space for a complete discussion on this topic. We merely note briefly some problems related to the use of too large $\alpha \approx \frac{1}{2}$ and too small $\alpha \approx 0$. The use of too large parameter can result in a complete failure in convergence, possibly due to the failure in satisfying the condition (6), and possibly due to the lack of robustness. For example, if $\alpha > \frac{1}{4}$, the condition (6) is not satisfied

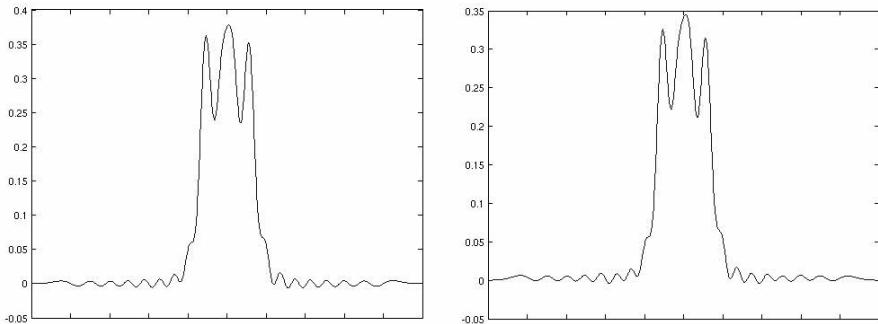


Fig. 2. Samples shown in figure 1 are estimated with approximation (9), with parameters $\alpha = \frac{1}{4}$ and $N = 40$

for the well-known Laplacian pdf, and in this case the series indeed turns out to be divergent. On the other hand, while using a very small parameter does guarantee robustness and pointwise convergence, the convergence can become highly non-uniform, which implies other difficulties for approximation purposes.

3 Measures of Non-gaussianity

3.1 Definition

Once a series representation (9) is obtained for a standardized distribution $p_Y(y)$, the coefficients b_n can be used to obtain a natural measure of non-Gaussianity,

$$J_\alpha = (1 - b_0)^2 + \sum_{n=1}^N n! b_n^2. \quad (11)$$

It is clear that J_α is zero only if Y is Gaussian. Other measures with the same property could be defined from these coefficients too, but the expression (11) is particularly well justified, as can be seen by examining its relationship to the L_2 -norm and negentropy.

In fact, if $\alpha = \frac{1}{4}$ and $N = \infty$, then J_α is proportional to the L_2 -distance between p_Y and the Gaussian distribution. If $\alpha = \frac{1}{2}$, then by using the approximation $\log(1 + t) \approx t$, it can be shown that J_α is approximately the negentropy of Y as in [6]. Both of these claims can be verified by using the orthogonality properties of Hermite polynomials. Thus the quantity J_α can be interpreted as a generalization of these two previously known measures of non-Gaussianity.

The special case $\alpha = \frac{1}{2}$ is nothing new for ICA, since it merely gives the skewness and kurtosis for b_3 and b_4 . Hence it should be emphasized that the quantity J_α should not be used with this parameter value $\alpha = \frac{1}{2}$ if the robustness is an issue.

3.2 Simulations

In order to examine the performance of the quantity J_α , let us compare it with an ad hoc objective function

$$J_{\text{ad hoc}}(Y) = \frac{36}{8\sqrt{3} - 9} \left(E(Y e^{-\frac{1}{2}Y^2}) \right)^2 + \frac{1}{2 - \frac{6}{\pi}} \left(E(|Y|) - \sqrt{\frac{2}{\pi}} \right)^2, \quad (12)$$

which is proposed in [6], and with a mean nearest neighbor (meanNN) objective function [7]

$$J_{\text{meanNN}}(Y) = - \sum_{t=1}^{T-1} \sum_{t'=t+1}^T \log(|Y_t - Y_{t'}|). \quad (13)$$

This meanNN objective is also similar to Renyi-entropy estimators [9]. We used two-dimensional random variables $X = (X_1, X_2)$ in which X_2 is a Gaussian, and X_1 is given different distributions X^a , X^b , and X^c as shown in the figure 3. We then measured $J_{\text{ad hoc}}$, J_{meanNN} and J_α on a one dimensional random variable $Y = w \cdot X$, where $w = (\cos(\theta), \sin(\theta))$, and plotted $J_{\text{ad hoc}}$, J_{meanNN} and J_α as functions of θ . For all choices $X_1 = X^a, X^b, X^c$, we generated samples of 10000 points, and results are shown in figures 4, 5 and 6. The number of coefficients b_n in J_α was $N = 10$.

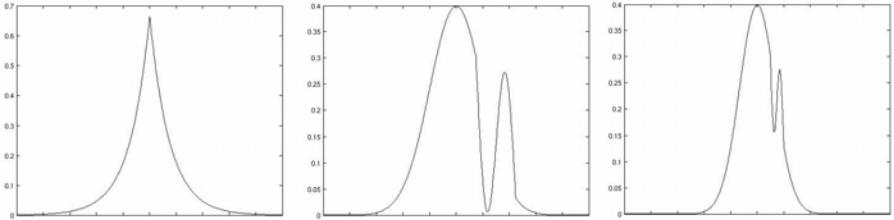


Fig. 3. From left to right the distributions X^a , X^b and X^c , which are used for X_1 , while X_2 remains as a Gaussian. X^a is the Laplacian random variable, X^b is a Gaussian with a notable perturbation, and X^c a Gaussian with a smaller perturbation.

We see that sometimes, like with X_1 as Laplacian and X_2 as Gaussian, our objective J_α merely reproduces the same results that could have been obtained with some ad hoc objective function. On the other hand, if X_1 is almost a Gaussian with some perturbation, J_α can produce better results than some rivals, hence making it a serious tool for ICA.

It should be noted that it is a straightforward exercise to compute an explicit expression for the partial derivatives $\partial J_\alpha / \partial w_k$ (in terms of the derivatives of the Hermite polynomials.) Thus this objective function allows a simple application of the gradient method.

Another relevant remark is that the amount of time consumed for the computation of the coefficients b_n (and the gradient too) grows as $O(T)$ where T is the number of sample points. With meanNN algorithm the time grows as $O(T^2)$.

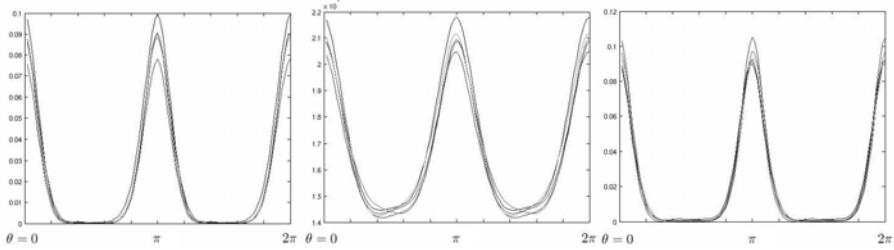


Fig. 4. The quantities $J_{\text{ad hoc}}$, J_{meanNN} and $J_{\alpha=\frac{1}{4}}$ (from left to right) plotted as a function of θ , when a choice $X = (X^a, X_2)$ was used. Samples were generated 5 times and results for all samplings are plotted here simultaneously. All objective functions succeed in finding the Laplacian in directions $\theta = 0$ and $\theta = \pi$.

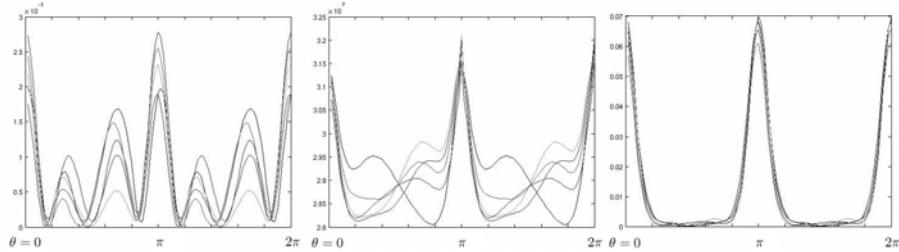


Fig. 5. The quantities $J_{\text{ad hoc}}$, J_{meanNN} and $J_{\alpha=\frac{1}{4}}$ plotted as a function of θ , when a choice $X = (X^b, X_2)$ was used. Samples were generated 5 times and results for all samplings are plotted here simultaneously. All functions have the correct global maxima, but $J_{\text{ad hoc}}$ has a severe local maxima problem.

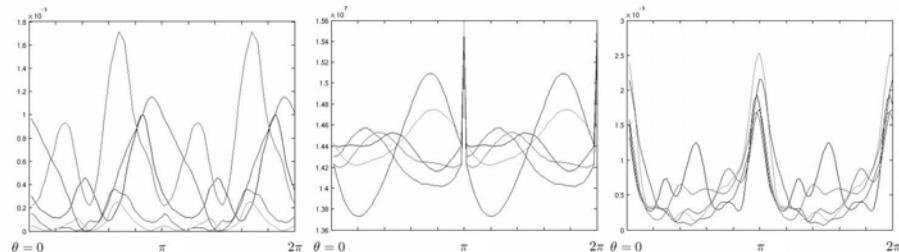


Fig. 6. The quantities $J_{\text{ad hoc}}$, J_{meanNN} and $J_{\alpha=\frac{1}{4}}$ plotted as a function of θ , when a choice $X = (X^c, X_2)$ was used. Samples were generated 5 times and results for all samplings are plotted here simultaneously. The J_{α} suffers from local maxima, but its overall performance is still better than those of $J_{\text{ad hoc}}$ and J_{meanNN} , whose results seem quite random.

4 Conclusions

We have proposed an ICA objective function, which is in a sense a very trivial application of the properties of the Hermite polynomials. Despite the triviality, with right choice of parameters, our objective function can be a very robust and precise measure of non-Gaussianity. Its advantage over ad hoc objective functions is that while ad hoc objective functions may work fine for certain distributions, our objective function measures all kinds of deviations from Gaussianity. Our objective function is also suitable for large data sizes, with only $O(T)$ asymptotic time consumption with respect to the sample size.

There also exists other strategies for ICA. For example, instead of measuring the non-Gaussianity, one can also directly attempt to measure the mutual information. Hulle [8] proposed a method for mutual information approximation using polynomial expansions. His method is not robust against outliers because it uses plain polynomials. We believe that the ideas we propose here for the measurement of non-Gaussianity could also be used to modify the methods of measuring the mutual information into a more robust form.

References

1. M. C. Jones, R. Sibson: What is Projection Pursuit? *J. of Royal Statistical Society, Ser. A* Vol 150, pages 1-36 (1987)
2. P. Comon: Independent component analysis—a new concept? *Signal Processing* Vol 36, pages 287-314 (1994)
3. S. Blinnikov, R. Moessner: Expansions for nearly Gaussian distributions. *Astron. Astrophys. Suppl. Ser.* 130, 193-205 (1998)
4. G. Szego: *Orthogonal Polynomials*. AMS (1939)
5. J. Duoandikoetxea: *Fourier Analysis*. AMS (2001)
6. A. Hyvärinen: New Approximations of Differential Entropy for Independent Component Analysis and Projection Pursuit. *Advances in Neural Information Processing Systems* Vol 10, pages 273-279. (1998)
7. L. Faivishevsky, J. Goldberger: ICA based on a Smooth Estimation of the Differential Entropy. *Advances in Neural Information Processing Systems* Vol 21. (2009)
8. M. M. Van Hulle: Multivariate Edgeworth-based Entropy Estimation. *Neural Computation* Vol. 17, No. 9, pages 1903-1910 (2005)
9. D. Pal, P. Poczos, C. Szepesvari: Estimation of Renyi Entropy and Mutual Information Based on Generalized Nearest-Neighbor Graphs. *Advances in Neural Information Processing Systems* (2010)

Complex-Valued Independent Component Analysis of Natural Images

Valero Laparra¹, Michael U. Gutmann², Jesús Malo¹, and Aapo Hyvärinen²

¹ Image Processing Laboratory (IPL),
Universitat de València, Spain

² Department of Computer Science
Department of Mathematics and Statistics
Helsinki Institute for Information Technology HIIT
P.O. Box 68, FIN-00014 University of Helsinki, Finland

Abstract. Linear independent component analysis (ICA) learns simple cell receptive fields from natural images. Here, we show that linear complex-valued ICA learns complex cell properties from Fourier-transformed natural images, i.e. two Gabor-like filters with quadrature-phase relationship. Conventional methods for complex-valued ICA assume that the phases of the output signals have uniform distribution. We show here that for natural images the phase distributions are, however, often far from uniform. We thus relax the uniformity assumption and model also the phase of the sources in complex-valued ICA. Compared to the original complex ICA model, the new model provides a better fit to the data, and leads to Gabor filters of qualitatively different shape.

Keywords: Complex Independent Components Analysis, Natural Image Statistics, Modeling Fourier phase distribution.

1 Introduction

Natural image statistics has become a very useful tool in order to understand how the visual part of the brain works (see for instance [1] for review). One of the most relevant revelations has been that a set of linear sensors optimized to obtain independent sources from natural image data resembles the Gabor-like receptive fields in the primary visual cortex (V1) [2,3].

In recent years, the advances in natural image statistics have been mainly in describing the statistics of the signals after this linear “simple cell” stage [4,5,6,7,8,9]. A common point of these models is that they focus on the total magnitude of the sensor (simple cell) outputs. Often, a combination of the squared outputs of simple cells is learned, leading to something like complex cells. However, there is evidence that relative magnitude, or phase, of simple cells plays an important role. A simple example about the relative importance of the magnitude and phase can be found in [10]. In this example the magnitude and the phase in the Fourier domain of two images were exchanged, and the images which were perceptually more similar to the originals were the ones that carried the phase information. Moreover there is experimental evidence of phase coupled Gabor-like filters in V1 [12,11]. For this reason, Daugman [13] suggested

that the receptive field in the first stage could be seen as Gabor sensors defined in the complex domain: the real and the imaginary part are essentially the same Gabor filter but with phases in quadrature.

Despite the evidence of the importance of the phase, not too much progress has been made in statistically modeling the phase of the signals after the simple cell step. The contributions in this field are restricted to models with a fixed linear stage, the wavelet transform [14,15]. Although this led to interesting results about the distribution of natural images, the statistics used in this kind of modeling could depend on the particular choice of using the wavelet transform as first linear stage.

Here, we aim at both modeling the phase distribution and learning the first linear stage from the data. For that purpose, we are proposing an extension of complex independent component analysis (cICA) [16]. The proposed extension deals with explicitly modeling the phase of non-circularly symmetric sources as an alternative to [17], which does consider non-symmetric sources but it does not model the lack of symmetry.

The paper falls naturally in two parts. In Section 2, we review cICA and point out its limitations in modeling the phase distribution. Section 3 shows how cICA can be extended to better capture the distribution of the phase variable. The extension includes the version of [16] as special case. Although we focus here on natural images, the extension can be applied to all kinds of data. Conclusions are drawn in Section 4.

2 Complex Independent Component Analysis and Its Limitations

As in Independent Component Analysis (ICA) for real variables, the goal in complex ICA (cICA) [16] is to find a linear transformation W such that, when applied to some vector of signals x , the elements of the output vector $s = W^H x$ are statistically as independent as possible. The difference to real ICA is that W , x , and hence also s are complex valued. Furthermore, instead of the transpose W^T , the transposed, complex conjugate W^H is used.

In ICA, one approach to find such a W is to first whiten the data and then to maximize the kurtosis, or a statistically more robust contrast function. In cICA, the same approach can be taken by appropriately defining whitening and choosing an appropriate contrast function.

For complex variables, the random vector x is white if both the real and imaginary part are white and if the real and imaginary parts are uncorrelated. An equivalent condition is that $\mathbb{E}\{xx^H\} = I$ and $\mathbb{E}\{xx^T\} = 0$. Denoting a column of W by w_i , in [16], cICA can be performed by optimization of J_G ,

$$J_G(W) = \sum_{i=1}^n \mathbb{E}\{G(|w_i^H x|^2)\}, \quad (1)$$

under the constraint $W^H W = I$. Depending on the nature of the sources, J_G needs to be maximized or minimized. The contrast function G must be a smooth even function and x is assumed to be white. Possible candidates include $G(y) = -\sqrt{a + y^2}$ for a small constant a . In the simulations in the next section, we will use this contrast function with $a = 0.1$. Note that the objective function depends only on the moduli $r_i = |w_i^H x|$ of the complex variable $s_i = w_i^H x$, no matter the choice of G . For sparse sources, maximization of this G leads to consistent estimators [16].

An alternative viewpoint of cICA is based on maximum likelihood estimation of the statistical model $s = W^H x$ where x and s are white and $W^H W = I$. Assuming independence of the sources in $s = (s_1, \dots, s_n)^T$, the log-likelihood is

$$\ell(W) = \sum_t \sum_{i=1}^n \log p_{s_i}(w_i^T x_t), \quad (2)$$

where x_t is the t -th observation of x and p_{s_i} is the density of the sources s_i . Since the variables are complex valued, $p_{s_i}(s_i)$ is a bidimensional distribution that can be written as $p_{r\phi}(r_i, \phi_i)/r_i$, where r_i is the modulus and ϕ_i is the phase of s_i . Assuming further that the modulus and the phase are independent and that, importantly for the next sections, the distribution of the phase is a uniform distribution, maximization of ℓ becomes maximization of

$$J_2(W) = \sum_t \sum_{i=1}^n (\log p_r(r_{it}) - \log r_{it}). \quad (3)$$

The term p_r denotes the distribution for the moduli r_i , where we assume that all of them follow the same distribution. Replacing sample average by expectation, we obtain the objective function in Eq.1 with $G(r^2) = \log p_r(r) - \log r$. Note further that the distribution p_q of the squared modulus $q = r^2$ is $p_q(q) = p_r(r)/(2r)$. This means that $G(q) = \log p_q(q) + \log 2$. Hence, the contrast function G used in cICA can be directly related to the distribution of the squared moduli of the complex sources. In particular, we can relate the contrast function $G(q) = -\sqrt{a + q^2}$, where a is a small constant, to the choice of p_q being a Gamma distribution,

$$p_q(q) = q^{k-1} \frac{\exp \frac{-q}{\theta}}{\Gamma(k)\theta^k}, \quad (4)$$

with $k = 1$. Then, $\log p_q(q) = -q + \text{const}$, which is, up to additive constants, the same as the above contrast function when a is small.

2.1 Simulations with Natural Images

We apply cICA on natural images in the Fourier domain. The natural images are 16×16 patches extracted from the data base in [18]. The data x on which we apply cICA are the complex Fourier coefficients. For the visualization, we show the obtained filters, which correspond to: the learned matrix W , combined with the whitening matrix, and the Fourier transform.

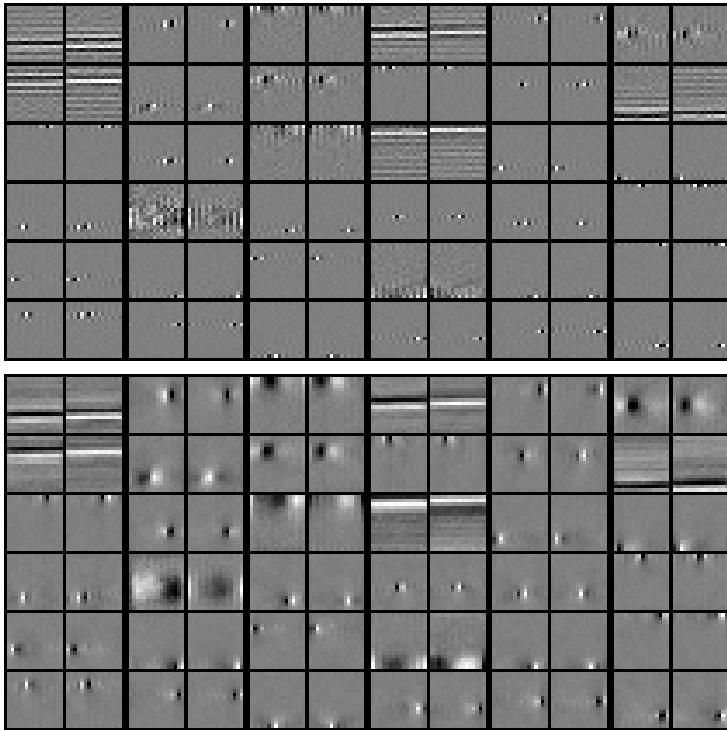


Fig. 1. Filters (see text for details) and features (defined by the pseudoinverse of the filter matrix) obtained with cICA using the algorithm in [16], ordered according to their contribution to the value of J_G in Eq. 1 (first 36 of 126). Filters and features are shown in pairs, with the real part at the left and the imaginary part at the right. Top: complex filters. Bottom: complex features.

Fig. 1 shows the results. The real and the imaginary part of the learned complex filters are shown in pairs from left to the right. Real and imaginary part are Gabor-like filters that are in quadrature-phase. This statistical result is in line with measurements in V1 [12,11] and related empirical models [13].

Complex ICA results essentially replicate those obtained by independent subspace analysis [4]. It is the complex-valued formalism which allows to automatically create two-dimensional subspaces with a linear model formulation as in ordinary ICA.

2.2 Checking Model Assumptions

Here we check whether, for natural images, the obtained complex sources s_i follow the assumption in cICA that the (squared) moduli follow a Gamma distribution and the phases are uniformly distributed.

Fitting gamma distributions to the empirical distributions of the modulus of the sources leads to good fits, see Fig. 2. In contrast, the empirical distributions

of the phases do not follow the model assumptions, as shown in Fig. 3. The clearly visible oscillations in the phases violate the assumption of uniformity in cICA. These roughly bimodal histograms may be modeled by a modified von Mises distribution to account for the two peaks,

$$p_\phi(\phi|k, \mu) = \frac{1}{2\pi I_0(k)} e^{k \cos(2(\phi - \mu))}, \quad (5)$$

where $I_0(k)$ is the Bessel function of order 0. In contrast to the ordinary von Mises distribution, we have here introduced the factor 2 inside the cosine to model the two-peaked distributions seen in Fig. 3. Note that this distribution corresponds to a uniform distribution when the parameter $k = 0$. In Fig. 3 we can see how fitting this distribution to the empirical distribution of the phase is much more precise than fitting a uniform distribution.

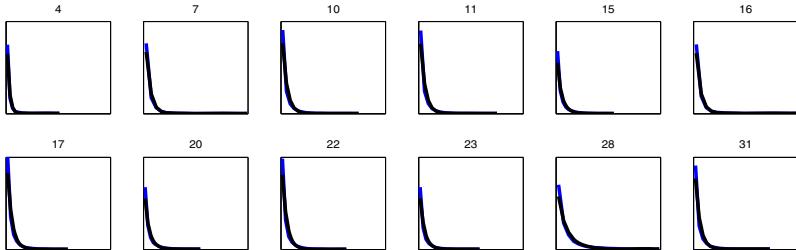


Fig. 2. Selection of distributions of the modulus of the cICA sources (blue) and the fitted gamma distributions (black). The curves are strongly overlapping. Numbers refer to the corresponding filters in Fig. 1 (left to right, top to bottom).

3 Extension of Complex ICA

In this section we propose an extension of cICA. The extension builds on the maximum likelihood approach to cICA in Eq. 2. It will take into account that the distribution of the phase variables can be non-uniform, as found in natural images (Eq. 5 and Fig. 3).

As in the previous section, we write in Eq. 2, p_{s_i} as $p_{r\phi}(r_i, \phi_i)/r_i$, where r_i is the modulus and ϕ_i is the phase of s_i . Also as previously, we assume that the modulus and the phase are independent. However, instead of assuming a uniform distribution for the phases, we assume the distribution in Eq. 5. Here, we can set $\mu = 0$ because this phase localization parameter is redundant: the phase of the oscillations will be determined by the estimated features anyway. Since this distribution includes the uniform distribution, our extension includes the conventional cICA as a special case. With these assumptions, the maximum likelihood principle leads us to maximize the following objective function

$$J_{GQ}(W, k_i) = \sum_i \mathbb{E}\{G(r_i) + Q(\phi_i, k_i)\}. \quad (6)$$

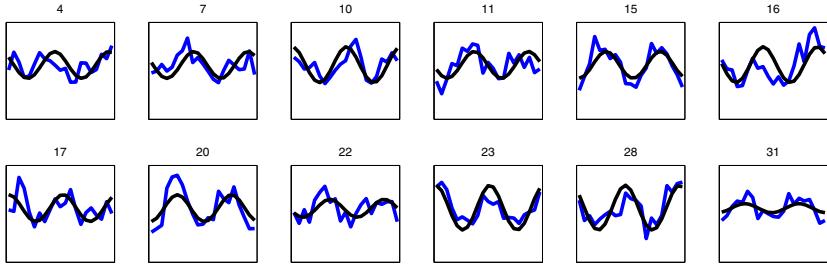


Fig. 3. Selection of distributions of the phases of the cICA sources (blue) and the fitted modified von Mises distributions (black). Numbers refer to the corresponding filters in Fig. 1 (left to right, top to bottom).

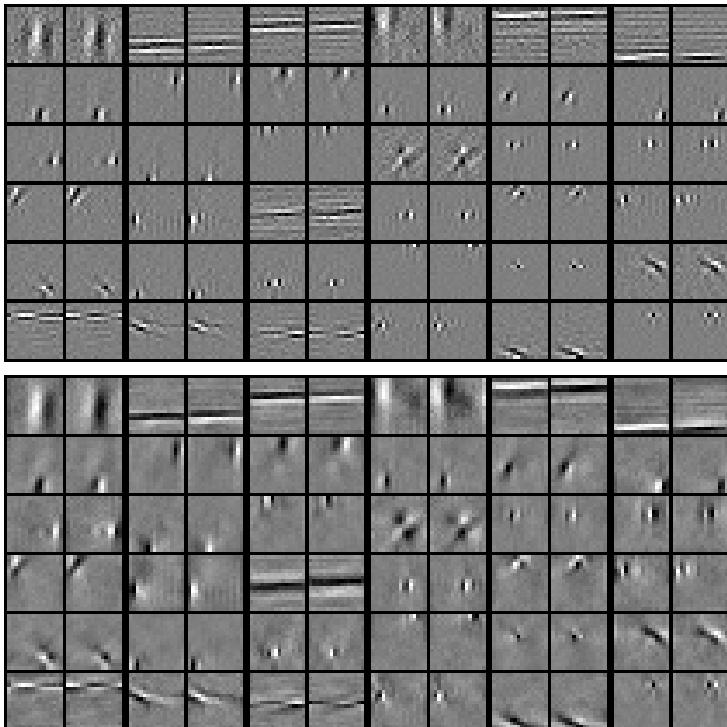


Fig. 4. Filters and features obtained with the extended cICA, ordered according to their contribution to the value of J_{GQ} in Eq. 6 (first 36 of 126). Filters and features are shown in pairs with the real part on the left and the imaginary part on the right. Top: complex filters. Bottom: complex features.

Here, r_i is the modulus of the complex number $w_i^H x$, and ϕ_i is its phase. As before, w_i denotes a column of the matrix W and we have the constraint $W^H W = I$. The function G is, as before, related to the distribution of the

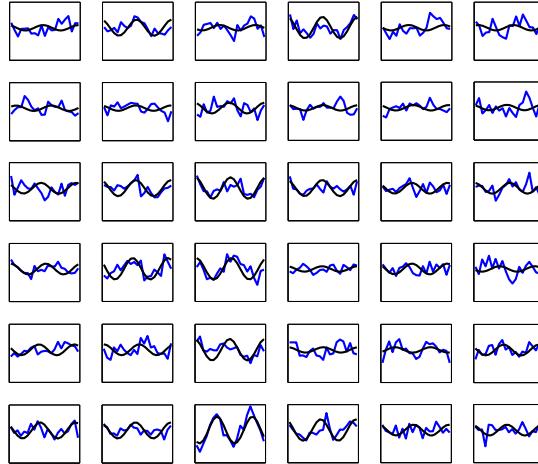


Fig. 5. Phase distributions of the sources obtained using the modified cICA algorithm. The distribution of the phases corresponding to the filters in Fig. 4 are shown. Empirical distributions are plotted in blue and fitted modified Von Mises distribution in black.

squared modulus. A possible choice is $G(y) = -\sqrt{a + y^2}$. The function Q is related to the distribution of the phase and is given by $Q(\phi_i, k_i) = k_i \cos(2\phi_i)$, and depends on the w_i and the shape parameters k_i .

This modification of cICA can also be considered from an information theoretical point of view. The main goal of all ICA-based algorithms is to obtain independent sources, which is equivalent to reduce the mutual information (MI) between them. Therefore, as $MI(s_1, s_2, \dots, s_n) = \sum_i \{h(r_i) + h(\phi_i)\} - h(s_1, s_2, \dots, s_n)$, where $h(\cdot)$ is the entropy. This result can be derived by using the same assumptions as in section 2. Accordingly, we have to reduce the entropy of r_i and ϕ_i , (the joint entropy is invariant under unitary transforms). Note that the uniform distribution is the one with maximum entropy when the domain is bounded. Therefore, anything different to a uniform phase distribution will have less entropy, which means less MI between the variables, and hence more independent sources.

Fig. 4 shows the results when the above extended cICA is applied to natural images (same setup as before). Note how the shape of the filters is more elongated (especially the highest-ranked ones) and spatially more extended than for the classical cICA. In Fig. 5 we can see the distribution of phases of the sources obtained with the proposed algorithm. The distributions are similar to the proposed modified von Mises distribution.

4 Conclusions

In this paper, we have started with modeling natural images with complex Independent Component Analysis (cICA). This led to the emergence of complex

filters where the real and the imaginary parts have the same Gabor-like shape (same orientation and same frequency) but a phase difference of $\frac{\pi}{2}$.

Checking the model assumptions in cICA, we have noticed that the assumption of uniformity of the phases is often violated for natural image data. This led us to formulate an extension of cICA which models also the phase distributions. Simulations with natural images showed that the empirical distribution of the phases provide a good match to the assumptions of the extended model.

Our research has the potential for more extensions. For instance, the assumption of the independence between modulus and phase should be investigated more carefully.

References

1. Simoncelli, E.P., Olshausen, B.A.: Natural image statistics and neural representation. *Annual Review of Neuroscience* 24(1), 1193–1216 (2001)
2. Olshausen, B., Field, D.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature* 381, 607–609 (1996)
3. Bell, A.J., Sejnowski, T.J.: The ‘Independent Components’ of Natural Scenes are Edge Filters. *Vision Research* 37(23), 3327–3338 (1997)
4. Hyvärinen, A., Hoyer, P.: Emergence of phase- and shift-invariant features by decomposition of natural images into independent feature subspaces. *Neural Computation* 12(7), 1705–1720 (2000)
5. Portilla, J., Strela, V., Wainwright, M., Simoncelli, E.: Image denoising using scale mixtures of Gaussians in the wavelet domain (2003)
6. Hyvärinen, A., Hurri, J., Hoyer, P.O.: *Natural Image Statistics*. Springer, Heidelberg (2009)
7. Lyu, S., Simoncelli, E.P.: Nonlinear extraction of independent components of natural images using radial gaussianization. *Neural computation* 21(6), 1485–1519 (2009)
8. Eichhorn, J., Sinz, F., Bethge, M.: Natural image coding in V1: how much use is orientation selectivity? *PLoS computational biology* 5(4) (2009)
9. Malo, J., Laparra, V.: Psychophysically Tuned Divisive Normalization factorizes the PDF of Natural Images. *Neural Computation* 22(12) (2010)
10. Oppenheim, A.V., Lim, J.S.: The importance of phase in signals. *Proceedings of the IEEE* 69(5), 529–541 (1981)
11. Tournyan, J., Felsen, G., Dan, Y.: Spatial structure of complex cell receptive fields measured with natural images. *Neuron* 45(5), 781–791 (2005)
12. Pollen, D.A., Ronner, S.F.: Phase relationships between adjacent simple cells in the visual cortex. *Science* 212(4501), 1409–1411 (1981)
13. Daugman, J.G.: Quadrature-phase simple-cell pairs are appropriately described in complex analytic form. *J. Opt. Soc. Am. A* 10(2), 375–377 (1993)
14. Portilla, J., Simoncelli, E.P.: A Parametric Texture Model Based on Joint Statistics of Complex Wavelet Coefficients. *International Journal of Computer Vision* 40(1), 49–70 (2000)
15. Cadieu, C.: Probabilistic Models of Phase Variables for Visual Representation and Neural Dynamics, Ph.D. thesis, UC Berkeley (2009)
16. Bingham, E., Hyvärinen, A.: A fast fixed-point algorithm for independent component analysis of complex valued signals. *International Journal of Neural Systems* 10, 1–8 (2000)
17. Eriksson, J., Seppola, A.M., Koivunen, V.: Complex ICA for circular and non-circular sources. In: *Proc. EUSIPCO* (2005)
18. Olmos, A., Kingdom, F.A.: McGill Calibrated Color Image Database (2004)

Improving Gaussian Process Value Function Approximation in Policy Gradient Algorithms

Hunor Jakab and Lehel Csató*

Babeş-Bolyai University, Kogălniceanu str 1, Cluj-Napoca, Romania, RO-400084,
Eötvös Loránd University, Pázmány P. Sétány 1/C, Budapest, Hungary, H-1117

{jakabh,lehel.csato}@cs.ubbcluj.ro

<http://www.cs.ubbcluj.ro>

Abstract. The use of value-function approximation in reinforcement learning (RL) problems is widely studied, the most common application of it being the extension of value-based RL methods to continuous domains. Gradient-based policy search algorithms can also benefit from the availability of an estimated value-function, as this estimation can be used for gradient variance reduction. In this article we present a new value function approximation method that uses a modified version of the Kullback–Leibler (KL) distance based sparse on-line Gaussian process regression. We combine it with Williams’ episodic REINFORCE algorithm to reduce the variance of the gradient estimates. A significant computational overload of the algorithm is caused by the need to completely re-estimate the value-function after each gradient update step. To overcome this problem we propose a measure composed of a KL distance-based score and a time dependent factor to exchange obsolete basis vectors with newly acquired measurements. This method leads to a more stable estimation of the action value-function and also reduces gradient variance. Performance and convergence comparisons are provided for the described algorithm, testing it on a dynamic system control problem with continuous state-action space.

Keywords: Reinforcement learning, policy gradient methods, Gaussian processes, value function estimation, control problems.

1 Introduction

Value function approximators are used widely in reinforcement learning (RL) for generalization purposes. One of their main applications is the extension of value-based RL methods to continuous domains. Direct policy search algorithms can also benefit from the availability of an estimated value function. In this article we focus on using Gaussian processes (GP) for the approximation of action-value functions in conjunction with gradient-based policy search algorithms such as the REINFORCE algorithm by [15]. A formal representation of a reinforcement

* We acknowledge the financial support from programs: POSDRU 6/1.5/S/3, and PNCD II 11-039.

learning problem is given using Markov decision processes (MDP) [10]. An MDP is a quadruple $M = (S, A, P, R)$ with the following elements: S is the set of states; A the set of actions; $P(s'|s, a) : S \times S \times A \rightarrow [0, 1]$ the transition probabilities, and $R : S \times A \rightarrow \mathbb{R}$, $R(s, a)$ the reward function. Apart from the MDP describing the agent-environment interaction, we can identify additional elements of a learning system: a *parameterized policy*: $\pi_\theta : S \times A \rightarrow \mathbb{R}$, that is a probability distribution over state-action space. The policy function for each state $s \in S$ and each action $a \in A$ returns the probability $\pi(s, a|\theta)$ of taking action a in state s . We assume that the policy is parameterized with a vector of continuous parameters $\theta \in \mathbb{R}^d$ and d is the number of parameters. The solution of an MDP is the *optimal policy* π^* maximizing the expected *cumulative reward*:

$$\pi^* = \pi_{\theta^*} = \operatorname{argmax}_\pi (J^\pi), \quad \text{with} \quad J^\pi = E_\pi \left(\sum_{t=0}^{\infty} \gamma^t r_{t+1} \right)$$

where J^π is the cumulative reward function, an expectation of the discounted reward; the expectation being computed for the policy π_θ . For simplicity we ignored the index θ from the policy. Gradient-based policy search algorithms find the optimal policy by adjusting the policy parameters based on gradient descent [1]. According to the *policy gradient theorem*, for any MDP the gradient with respect to the objective function J can be expressed as the following average [14]:

$$\nabla_\theta J(\theta) = \int dp(s) \int d\pi(a|s) \nabla_\theta \log \pi(a|s) Q(s, a), \quad (1)$$

where $\int dp(x)$ denotes a weighted average with respect to the respective probability. The above formulation can be considered equivalent to Williams' REINFORCE algorithm where the gradient estimate is calculated as an expectation over trajectories:

$$\nabla_\theta J(\theta) = E_\tau \left[\sum_{t=0}^{H-1} \nabla_\theta \log \pi(a_t|s_t) \sum_{i=0}^{H-t} \gamma^i R(s_{t+i}, a_{t+i}) \right], \quad (2)$$

where τ is a *sampled trajectory*: a sequence of state-action pairs within a *single episode*, known also as roll-out. In eq. (2) the true value function $Q(s_t, a_t)$ is replaced with Monte Carlo samples of the form $\sum_{i=0}^{H-t} \gamma^i R(s_{t+i}, a_{t+i})$ that provide an unbiased but very high-variance estimate. Variance reduction can be achieved if we replace the samples with *values of an approximated action-value function*. In [8] we used GP-s for this purpose. The bottleneck of the method was the re-estimation of the value-function after each update step and the increasing size of the GP. In the following we describe an extended version of the algorithm which uses the KL distance-based sparse on-line updates in order to impose an upper limit on the size of the model. We also propose a method which makes it possible to avoid the complete re-estimation of the action value-function after each gradient update step.

2 Estimating Value Functions with Gpr

Different approaches of using Gaussian processes for estimating value functions have been proposed: [4] use separate GPs for system dynamics and action-value functions. [11] base the value function approximation on estimated system dynamics which makes it possible to perform an arbitrary number of simulated experiments. In [5,7] a generative model is used to approximate the action-value functions, similar to TD-like bootstrapping. [4] modeled both the value and the action-value functions with GPs and proposed a dynamic programming solution that exploited these approximations. They used different GPs for every possible action, therefore the method was feasible only when the action space could be discretized.

In our approach we use as training data the state-action pairs encountered during trajectories, $x_t \doteq (s_t, a_t)$, and the corresponding – possibly – discounted cumulative rewards $\hat{Q}(x_t) = \sum_{i=0}^{H-t} \gamma^i r_{t+i}$ as noisy targets.¹ The model is a GP, completely specified by its mean and covariance function, and it can be used to perform regression directly in a function space, with the resulting $q(s, a) \doteq q(x)$ being the approximation to the action-value function. The elements of the kernel matrix \mathbf{K}_q are $\mathbf{K}_q(i, j) = k_q(x_i, x_j)$, and we denoted k_q to emphasize that the kernel function operates on state-action pairs. Assume that n points have already been processed, therefore we have a GP built on the data set $\mathcal{D} = [(x_i, \hat{Q}_i)]_{i=1,n}$ which is also called the set of basis vectors (BV). GP's provide a fully probabilistic model therefore, to estimate the action-value of a new state-action pair, $x_{n+1} \doteq (s_{n+1}, a_{n+1})$, we compute the predictive mean (3) and variance (4) functions conditioned on the data, given by the posterior GP [12]:

$$\begin{aligned} q_{n+1} | \mathcal{D} &\sim \mathcal{N}(\mu_{n+1}, \text{cov}(q_{n+1})) \\ \mu_{n+1} &= \mathbf{k}_{n+1} \boldsymbol{\alpha}_n \end{aligned} \quad (3)$$

$$\text{cov}(q_{n+1}, q_{n+1}) = k_q(x_{n+1}, x_{n+1}) - \mathbf{k}_{n+1} \mathbf{C}_n \mathbf{k}_{n+1}^T, \quad (4)$$

where $\boldsymbol{\alpha}_n$ and \mathbf{C}_n are the parameters of the posterior GP:

$$\boldsymbol{\alpha}_{n+1} = [\mathbf{K}_q^n + \boldsymbol{\Sigma}_n]^{-1} \hat{Q}, \quad \mathbf{C}_{n+1} = [\mathbf{K}_q^n + \boldsymbol{\Sigma}_n]^{-1}. \quad (5)$$

with $\boldsymbol{\Sigma}_n = \boldsymbol{\sigma} I_n$ the covariance of the observation noise and \mathbf{k}_{n+1} a vector containing the covariances between the new point and the training points:

$$\mathbf{k}_{n+1} = [k_q(x_1, x_{n+1}), \dots, k_q(x_n, x_{n+1})]. \quad (6)$$

The parameters $\boldsymbol{\alpha}$ and \mathbf{C} of the GP can be updated each time a new (x_{n+1}, \hat{Q}_{n+1}) is processed. This is achieved by combining the likelihood of the new data-point and the resulting Gaussian process from the previous step [3]. Replacing $\sum_{i=0}^{H-t} \gamma^i R(s_{t+i}, a_{t+i})$ from eq. (2) with the posterior mean from eq. (3),² we get

¹ We assume that the targets have Gaussian noise with equal variance; one can easily use different *known* noise variance.

² For simplicity we denote the GP predictive mean for a state action pair (s, a) with $Q_{gp}(s, a) = \mu_{n+1}$ where $x_{n+1} = (s, a)$ and prediction is based on the previously visited data-points.

the GP version of the policy gradient algorithm. This modification significantly reduces the variance of the gradient estimates and improves the convergence rate of REINFORCE-based policy gradient algorithms.

2.1 Redundancy in Value Function Approximation

During the learning process we encounter many state-action pairs that lie in a space spanned by data points already processed and do not provide any significant amount of new information. The addition of these points to the basis vector set of points is not necessary since they can be expressed as the linear combination of the previous inputs and an additional error term. In order to fully exploit the probabilistic properties of Gaussian Processes we use a Kullback Leibler distance based sparsification mechanism to avoid the inclusion of insignificant data points into the basis vector set. In the following we briefly describe the steps of the algorithm, see [2] for a more detailed discussion.

When deciding upon the addition of a new data points to the basis vector set we have to assess the amount of information gained or lost. This can be expressed as the Kullback Leibler distance between two GPs $\varepsilon = KL(GP' || GP)$ where GP' contains the new data point and GP is obtained by replacing the data-point with its projection to the space spanned by existing basis vectors. If the length of the distance ε is within a certain threshold the new data point need not be included in the set of the basis vectors, still its effect on the GP parameters can be computed according to the sparse on-line update equations from [2].

To prevent the increase of the GP parameters beyond a certain limit we can restrict the BV set to a certain size. Computing ε –also called score– for all data-points allows us to remove the least important basis vectors and replace them with newly acquired significant measurements. The analytical form of the score is given by:

$$\varepsilon_i = \frac{\alpha^2(i)}{q(i) + c(i)} \quad i = 1 \dots |D|, \quad (7)$$

where $\alpha(i)$ $q(i)$ $c(i)$ are the i -th diagonal elements of the respective matrices. Using sparse on-line updates to the GP parameters we can reduce computational complexity from $O(n^3)$ to $O(N|D|^2)$. In the following we will describe a modified version of the sparse on-line GPR which makes it possible to maintain continuity in our action-value function approximator.

2.2 Maintaining Continuity

The majority of gradient-based policy search algorithms can be divided into two major steps. The first step is the estimation of the gradient with respect to an objective function, and second is the update of the policy parameters according to the gradient estimate. After the gradient update step many policy gradient algorithms require the user to sample new data and restart the value function approximation which is a costly operation. In order to make our algorithm more sample and cost efficient we want to avoid reestimating the value-function from

scratch, by making use of data gathered in the past. In previous work [13,6] on this subject importance sampling is used to reweigh the contribution of old measurements in the value function estimates. In this paper we propose a different approach where we gradually eliminate obsolete data-points from the GP basis vector set based on a combined recency and importance measure. We make two assumptions about the learning process. First, the change in the policy after a gradient-update step is considered to be small enough not to produce a drastically different value function from the previous one. Second the change in the state distribution which is a result of the change in the policy parameters, does not alter the underlying geometrical structure of the state-space. Therefore we can safely assume that the most informative input points will be in the vicinity of those from our previous gradient-estimation step.

After a gradient update step we would like to build upon our previously estimated value function model while simultaneously incorporating new measurements which provide useful information. For this purpose we assign a time variable to every data point in our *BV* set which signifies at which stage of the learning process the data point has been added to the basis vector set.

$$D = \{(x_1, y_1), \dots, x_n, y_n\} \rightarrow \{(x_1, y_1, t_1), \dots, (x_n, y_n, t_n)\} \quad (8)$$

Moreover we slightly modify the expression of the KL distance based scores from (7) by adding a term which penalizes basis vectors that have been introduced in early stages of the learning process. Whenever a new data point is being processed which needs to be included in the *BV* set but the maximum number of basis vectors has been reached we compute a modified score $e'(\cdot)$ for each element:

$$\varepsilon'(i) = \frac{\alpha^2(i)}{q(i) + c(i)} + \lambda g(t(i)) \quad (9)$$

We replace the lowest scoring data point from the *BV* set with our new measurement. Here $g(\cdot)$ is a function of the time variable assigned to each basis vector. Since we want to favor the removal of out-of date basis vectors, this function needs to be monotonically increasing. In our experiments we used an exponential of the form:

$$g(t_i) = e^{c(t_i - \min_i(t_i))} \quad i = 1 \dots |D| \quad (10)$$

The λ term from eq. (9) serves as a trade off factor between loss of information and accuracy of representation, c is a constant. In figure 1 we see how much the choice of λ influences learning performance. If we set λ to be a large number the time-dependent factor from the scores will outweigh the KL distance based factor in eq.(9) leading to the inclusion of all newly acquired measurements into the *BV* set. Setting it too small will allow too many out-of date basis vectors to be present in our representation which leads to inaccurate gradient estimates and a poor policy. Best performance is reached with an intermediate value, which justifies the fact that building on previous estimations is just as important as acquiring new measurements however finding the optimal value is a problem-specific task.

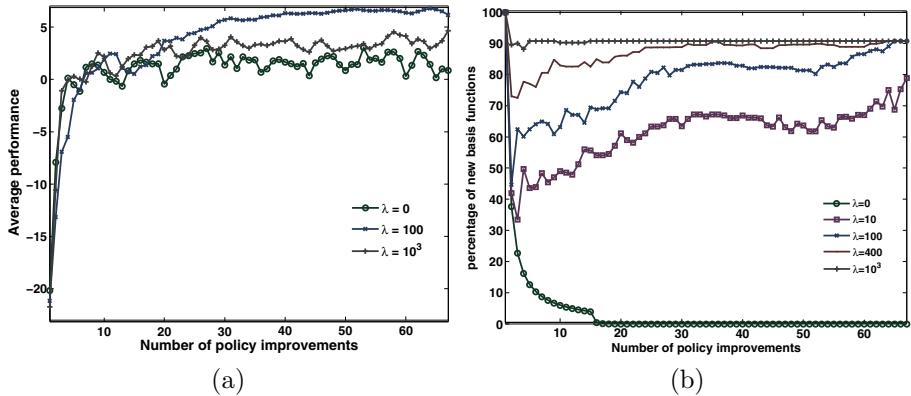


Fig. 1. Effect of λ on (a) performance evolution (b) average percentage of out-of date basis functions present in the action-value function approximation

The subtraction of the minimum assigned time value in eq. (10) is necessary to avoid the increase in importance of the time-dependent term as learning progresses and time variable values increase.

3 Experiments

We tested our proposed method on the classical pendulum balancing task, where both the state and the action spaces are continuous. A state variable consists of the angle and angular velocity of the pendulum, actions are the torques that we can apply to the system, and are limited to a $[-5, 5]$ interval. The Hamiltonian for the pendulum is:

$$H = \frac{1}{2ml}p_\theta^2 - mgl \cos(\theta) \quad p_\theta = ml^2\dot{\theta} \quad (11)$$

In each experiment we performed 200 gradient update steps. In each step for the calculation of the gradient estimates we performed 4 episodes consisting of 50 time-steps. Figure 2 shows the results averaged over 30 experiments for three versions of Williams' episodic REINFORCE algorithm. First we used the method proposed in this paper where λ was set to 100. Additionally we plotted the performance evolution of REINFORCE with standard GPR and as a baseline the standard episodic REINFORCE. We restricted the size of the BV set to contain only 250 basis vectors in order to speed up computation and we used a Gaussian policy consisting of a linear controller and added Gaussian exploratory noise.

The performance comparison in figure 2(a) shows that both GP enhanced versions of REINFORCE achieve better peak performance than the standard algorithm. Our proposed method with continuous GPR value function approximation performs better than the simple GPR approximation in the early phases of the learning process and achieves a slightly higher end performance, although the right choice of λ is crucial. Figure 2(b) shows the evolution of the Gaussian

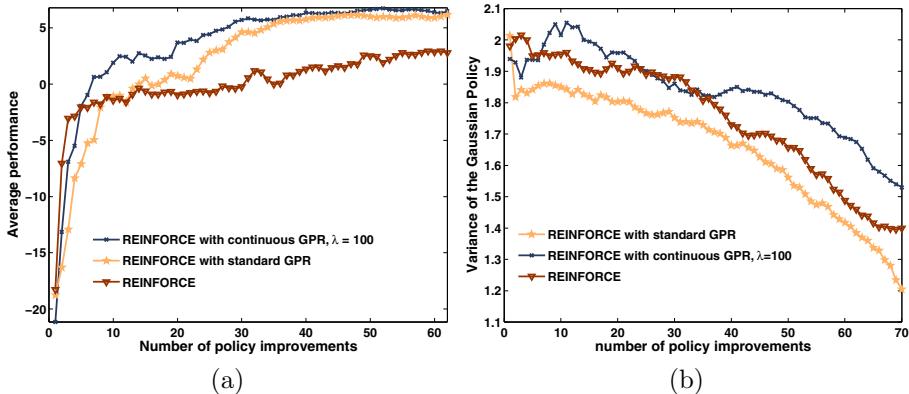


Fig. 2. Experimental results for the pole balancing control problem

policy's variance during the learning process for all three algorithms. The paths are quite similar which means that the difference in performance between the algorithms is not the result of the difference in variance reduction speed, but the quality of their gradient estimates. In our experiments we used equal learning rate in all three of the algorithms however the reduced gradient variance in the GP enhanced versions allow for larger learning rates without becoming unstable. Other variance reduction methods such as time-varying baselines and adaptive step sizes [9] can also be used in conjunction with our method to further increase performance.

4 Conclusions and Further Research

In this paper we presented a new way of maintaining continuity in value-function approximation with Gaussian processes. To facilitate the removal of out-of-date basis vectors we have introduced a measure composed from the KL distance between the original and the constrained GP and a time dependent term. We combined our method with Williams' REINFORCE algorithm which is known to suffer from high gradient variance. Our experiments show that maintaining continuity in value function approximation leads to better long-term performance and more efficient variance reduction. The performance of the value function approximation scheme presented here can be further improved by using geodesic distance based or variable resolution kernel functions. These are the major steps of our future research.

References

1. Baird, L., Moore, A.: Gradient descent for general reinforcement learning. In: Kearns, M.S., Solla, S.A., Cohn, D.A. (eds.) NIPS 1998. Advances in Neural Information Processing Systems, vol. 11, pp. 968–974. MIT Press, Cambridge (1998)
2. Csató, L.: Gaussian Processes – Iterative Sparse Approximation. PhD thesis, Neural Computing Research Group (2002)

3. Csató, L., Opper, M.: Sparse representation for Gaussian process models. In: Leen, T.K., Dietterich, T.G., Tresp, V. (eds.) NIPS, vol. 13, pp. 444–450. MIT Press, Cambridge (2001)
4. Deisenroth, M.P., Rasmussen, C.E., Peters, J.: Gaussian process dynamic programming. *Neurocomputing* 72(7-9), 1508–1524 (2009)
5. Engel, Y., Mannor, S., Meir, R.: Reinforcement learning with Gaussian processes. In: Proceedings of the 22nd International Conference on Machine learning, pp. 201–208, New York (2005)
6. Fan, Y., Xu, J., Shelton, C.R.: Importance sampling for continuous time Bayesian networks. *Journal of Machine Learning Research* 11, 2115–2140 (2010)
7. Ghavamzadeh, M., Engel, Y.: Bayesian policy gradient algorithms. In: Schölkopf, B., Platt, J., Hoffman, T. (eds.) NIPS 2007, Advances in Neural Information Processing Systems, vol. 19, pp. 457–464. MIT Press, Cambridge (2007)
8. Jakab, H.S., Csató, L.: Using Gaussian processes for variance reduction in policy gradient algorithms. In: 8th International Conference on Applied Informatics, Eger, pp. 55–63 (2010)
9. Peters, J., Schaal, S.: Reinforcement learning of motor skills with policy gradients. *Neural Networks* 21(4), 682–697 (2008)
10. Puterman, M.L.: *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, New York (1994)
11. Rasmussen, C.E., Kuss, M.: Gaussian processes in reinforcement learning. In: Saul, L.K., Thrun, S., Schlkopf, B. (eds.) NIPS 2003, Advances in Neural Information Processing Systems, pp. 751–759. MIT Press, Cambridge (2004)
12. Rasmussen, C.E., Williams, C.: *Gaussian Processes for Machine Learning*. MIT Press, Cambridge (2006)
13. Sugiyama, M., Hachiya, H., Towell, C., Vijayakumar, S.: Geodesic gaussian kernels for value function approximation. *Auton. Robots* 25, 287–304 (2008)
14. Sutton, R.S., McAllester, D.A., Singh, S.P., Mansour, Y.: Policy gradient methods for reinforcement learning with function approximation. In: Solla, S.A., Leen, T.K., Müller, K.R. (eds.) NIPS 1999, Advances in Neural Information Processing Systems, pp. 1057–1063. MIT Press, Cambridge (1999)
15. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8, 229–256 (1992)

Application of Nonlinear Neural Network Model for Self Sensing Characteristic in an Ionic Polymer Metal Composite (IPMC) Actuator

Ngoc Chi Nam Doan, Kyoung Kwan Ahn, Quang Truong Dinh, and Jong Il Yoon

School of Mechanical and Automotive Engineering,
University of Ulsan, San 29, Mugeerdong,
Namgu, 680-749, Ulsan, S. Korea
doanngocchinam@hotmail.com,
kkahn@ulsan.ac.kr

Abstract. This paper focuses on a self sensing characteristic of an Ionic Polymer Metal Composite (IPMC) bases on a novel accurate nonlinear black-box model (NBBM) to estimate the IPMC tip displacement. The NBBM is formed by a recurrent multi-layer perceptron neural network (RMLPNN) and a self-adjustable learning mechanism (SALM). The model parameters are optimized by using a set of training data. The ability of NBBM model is evaluated by a comparison of the estimated and real IPMC bending characteristic.

Keywords: IPMC, Identification, NBBM, RMLPNN, SALM.

1 Introduction

In the trend of finding advanced materials, IPMC is gaining great importance in use as both sensors and actuators [1]. A typical IPMC sheet is constructed with a thin ionic polymer membrane and two metal electrode layers outside. When a low electrical field is applied, the transport of hydrated cations leads to bending motions of the IPMC sheet. On the other hand, it has been found that an IPMC has a self sensing characteristics base on non-uniform concentration of ions [2] or variance of surface resistors [3]. Hence, smart sensing methods may be developed for an IPMC actuator. Fig. 1 shows the operating principle of an IPMC.

The aim of this paper is to develop a novel accurate model which is called nonlinear black box model for estimating the IPMC bending behavior using the phenomenon of variant surface resistors. This NBBM is a combination of both a RMLPNN and a SALM to estimate directly the IPMC tip displacement. First of all, an actuator using one type of IPMC sheet is setup to investigate the IPMC characteristics. The model parameters are then optimized by the collected training data. Evaluations are then carried out to evaluate the proposed self sensing methodology.

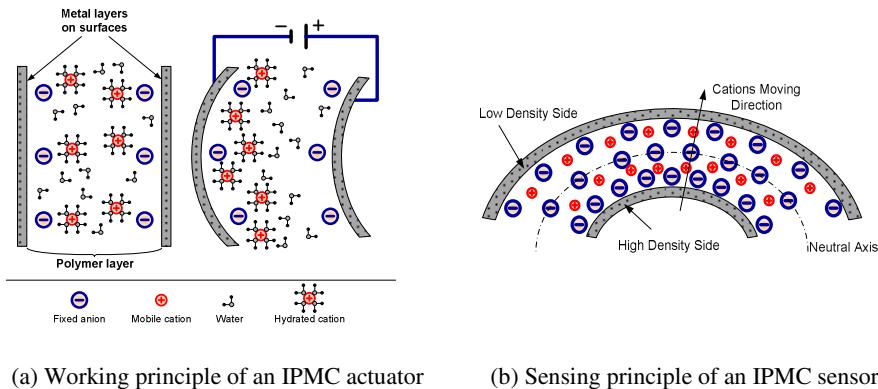


Fig. 1. Fundamental of IPMC as actuators and sensors

2 Experimental Apparatus

A test rig using IPMC is setup as in Fig. 2. The actuator is a sheet of IPMC manufactured by Environmental Robots Inc. and is clamped by two plastic plates. Six electrodes *A*, *B*, *C*, *D*, *E*, and *F* are attached to these plates and also contacted with two sides of the IPMC. A processing system is built on a personal computer within Simulink environment in MATLAB. Two multi-function Advantech cards, A/D 1711 and D/A 1720, are installed on PCI slots of the PC to perform peripheral buses. In addition, a CCD laser sensor, LK-081, from Keyence Corp. is installed on the rig base to measure the IPMC tip displacement. Setting parameters for the IPMC actuator are listed in Table 1.

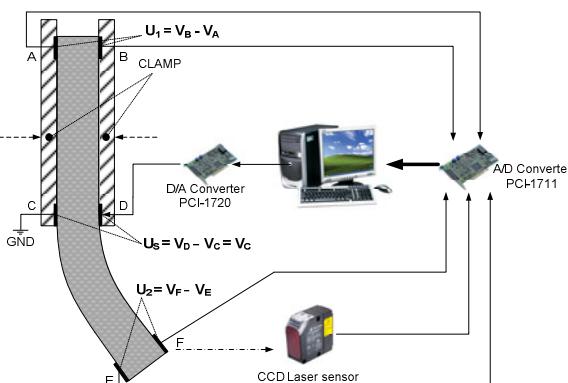


Fig. 2. Experiment setup for the IPMC actuator

Table 1. Setting parameters for IPMC system

Parameters	Specifications
Operating environment	Dry
Size	(40 x 6 x 0.2)mm
Sampling time	0.001s
Driving voltages	3~5V

The IPMC actuation is decided by the voltage signal sent from PC to apply to electrode D while electrode C is connected to the 0VDC (Fig. 2). At the same time, voltages at the other electrodes are measured to combine with the applied voltage to perform the input data. The IPMC tip displacement measured by the laser sensor is the output data.

3 Nonlinear Black-Box Model for the IPMC Actuator Based on RMLPNN and SALM

In order to perform self sensing characteristic of an IPMC, a sufficient estimator should be employed to ‘measure’ the displacement base on the information from the feedback voltage signals. As a promising solution, neural network has been applied for identification of nonlinear dynamics. Therefore, to solve the drawbacks of IPMC identification, the accurate NBBM model, which combines both the RMLPNN and SALM, is designed to model highly nonlinear systems in general and IPMC actuators in particular.

3.1 Multi-Layer Perceptions Neural Network (MLPNN) Structure

A typical MLPNN is shown in Fig. 3. An input vector (u_1, \dots, u_n) within ranges $[-1, 1]$ is standardized from the system input. At each neuron in a hidden layer, a combined value (sh) is defined:

$$sh_j = \begin{cases} \sum_{i=1}^n w_{ji}u_i + w_{j0}b_0 & , j = 1, \dots, q_1 : \text{each node in } 1^{\text{st}} \text{ hidden layer } (h_1) \\ \sum_{i=1}^{q_{l-1}} w_{ji}oh_i + w_{j0}b_{l-1}, j = 1, \dots, q_l & : \text{each node in } l^{\text{th}} \text{ hidden layer } (h_l, l = 2, \dots, k) \end{cases} \quad (1)$$

$(b_0, \dots, b_{k-1} : \text{bias factors})$

In this research, sigmoid activation function, f , is used, consequently, an output from the hidden layer (oh) is computed:

$$oh_j = f(sh_j) = \frac{1}{1 + e^{-sh_j}} \quad (2)$$

By the same way, each network output, \hat{y} , can be obtained as:

$$\hat{y}_p = F(SH_p) = \frac{1}{1 + e^{-SH_p}}, SH_p = \left(\sum_{j=1}^{q_k} W_j o h_j + W_0 b_k \right), p = 1, \dots, m \quad (3)$$

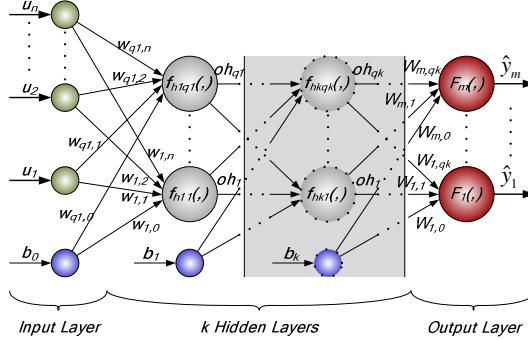


Fig. 3. Structure of a feed-forward MLPNN

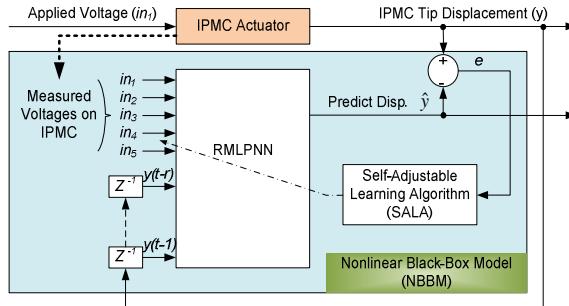


Fig. 4. Block diagram for training the NBBM model

3.2 Nonlinear Black-Box Model for IPMC Actuator

The diagram for training the NBBM is suggested in Fig. 4. In the NBBM, the five measured voltage signals: A , B , D , E , and F are applied as five model inputs (see section 2). There is one hidden layer and one output from the NBBM model which is the predicted IPMC tip displacement. In addition, the model output and its regressed signals are directly sent back to the NBBM as added inputs to exhibit dynamic temporal behavior of the neural system. As seen in Fig. 4, consider a set of the model inputs as:

$$\mathbf{U} = \begin{bmatrix} u_1 \\ \vdots \\ u_{5+r} \end{bmatrix} = \begin{bmatrix} in_1 & \cdots & in_5 & y_{-1} & \cdots & y_{-r} \end{bmatrix}^{-1} \quad (4)$$

Therefore, the model output can be computed based on section 3.1:

$$\hat{y}(t+1) = F(\mathbf{U}, \mathbf{w}, \mathbf{W}) \quad (5)$$

where: \mathbf{w} and \mathbf{W} are weight matrices of the hidden layer and output layer, respectively:

$$\mathbf{w} = \begin{bmatrix} w_{10} & \cdots & w_{1n} \\ \vdots & \vdots & \vdots \\ w_{q0} & \cdots & w_{qn} \end{bmatrix}; \mathbf{W} = \begin{bmatrix} W_0 & \cdots & W_q \end{bmatrix} \quad (6)$$

From (5), the set of weights (\mathbf{w}, \mathbf{W}) are adjustable parameters of the network, and need to be determined. Because the proposed NBBM model is the dynamic network, the dynamic training mechanism, SALM, is required to optimize the NBBM. The proposed SALM is based on the gradient descent method, back propagation algorithms, and a prediction error function which is defined as:

$$E = \frac{1}{2} \sum_{i=1}^m (\hat{y}_i - y_i)^2 \equiv E = \frac{1}{2} (\hat{y} - y)^2, \quad (m=1) \quad (7)$$

The weighting values for the $(k+1)^{th}$ step are updated as follows:

$$\begin{aligned} \bar{W}(k+1) &= \bar{W}(k) + \eta(k+1) \Delta \bar{W}(k+1); \quad \bar{W} \text{ is } w \text{ or } W; \\ \Delta \bar{W}(k+1) &= \alpha(k+1) \Delta \bar{W}(k) - \frac{\partial E}{\partial \bar{W}(k)}; \\ \begin{cases} \eta(k+1): \text{dynamic learning rate tuned by LR Fuzzy} \\ \alpha(k+1): \text{dynamic momentum rate tuned by MR Fuzzy} \end{cases} \end{aligned} \quad (8)$$

For the weights of output layer: the factor, $\frac{\partial E}{\partial \bar{W}(k)}$, in (8) are defined as:

$$\frac{\partial E}{\partial \bar{W}_j} = \begin{cases} -\delta oh_j \text{ for weights: } W_1, \dots, W_q; \\ -\delta b_1 \text{ for weight: } W_0 \end{cases}; \quad \delta = \hat{y}(1 - \hat{y})(y - \hat{y}) \quad (9)$$

here: δ is the search direction value of the neuron i_n output layer at each step

For the weights of hidden layer: the factor, $\frac{\partial E}{\partial \bar{W}(k)}$, in (8) is calculated:

$$\frac{\partial E}{\partial w_{ji}} = \begin{cases} -\delta_j u_i \text{ for weights: } w_1, \dots, w_n; \\ -\delta_j b_0 \text{ for weight: } w_0 \end{cases}; \quad \delta_j = oh_j(1 - oh_j)\delta W_j \quad (10)$$

here: δ_j is the search direction value of j^{th} neuron in the hidden layer at each step

LR Fuzzy: there are two inputs which are within a range $[0, 1]$, $|e(t)|^*$ and $|dw^* / dt|$, are scaled absolute values of the modeling error and weight changing speed. For each of the inputs, five triangle membership functions (MFs) are used and partitioned as in Fig.5(a).

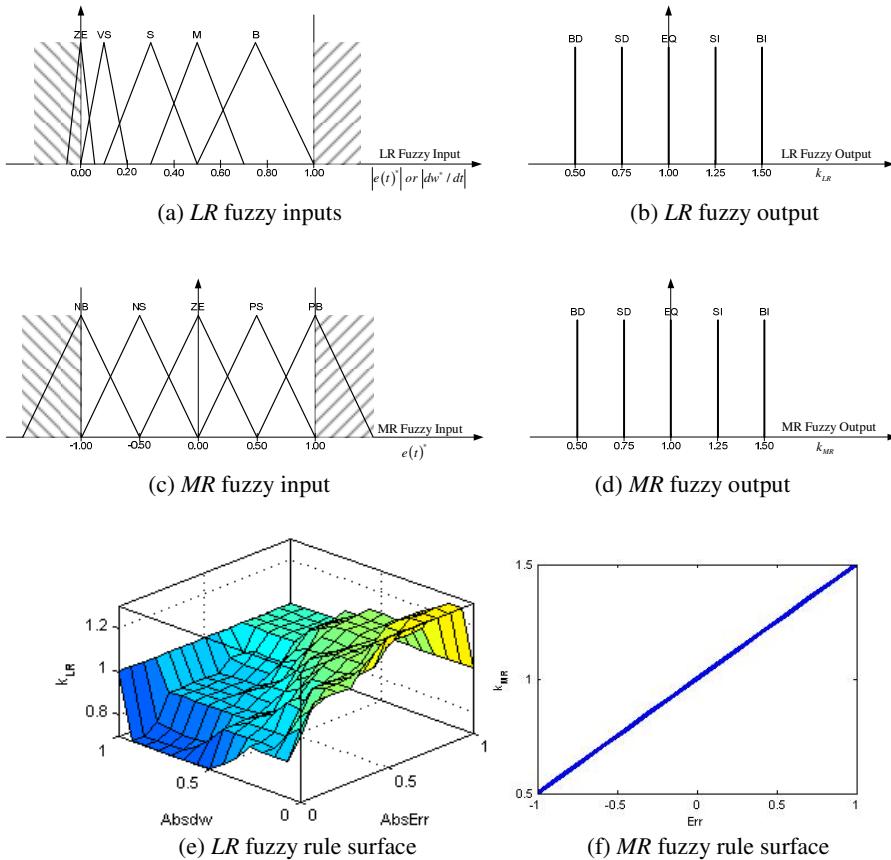


Fig. 5. Fuzzy logic block for *MR* and *LR*

The output of the *LR* fuzzy is a gain, k_{LR} , to compute the learning rate. This output is within a range $[0.5, 1.5]$ with five MFs as shown in Fig. 5(b).

The IF-THEN rules designed for the *LF* fuzzy, where MAX-MIN aggregation and “centroid” defuzzification method are used, is established as in Fig. 5(e). Hence, the learning rate of SALM in (10) is derived:

$$\eta(k+1) = k_{LR}(k+1) \times \eta_0; \text{ constant: } \eta_0 = 0.5 \quad (11)$$

MR Fuzzy: there is one input, $e(t)^*$, is the modeling error scaled into a range $[-1, 1]$ while the output, k_{MR} , is the gain within a range $[0.5, 1.5]$ used to adjust the momentum rate. Five MFs are used to partition the input as well as output as displayed in Fig. 5(c), (d).

The rules designed for the MR fuzzy is established as in Fig. 5(f). Consequently, the output gain, k_{MR} , is used to adjust the momentum rate of the SALM in (10):

$$\alpha(k+1) = k_{MR} (k+1) \times \alpha_0; \text{constant : } \alpha_0 = 0.5 \quad (12)$$

4 Modeling Results

The determination of suitable inputs and number of neurons in the hidden layer in the NNM model is one of the essential issues in practical implementation of the model. To obtain data for the model training process, a series of experiments on the IPMC system is conducted by using an opened loop control system with driving voltage is a square voltage signal of which frequency 0.1 Hz and amplitudes varies from 0.1V to 5V. Consequently, a set of the experimental data with respect to this driving signal as shown in Fig. 6 is used to train the NNM model with different RMLPNN structures.

The number of inputs is varied from 5 to 7 while the neuron numbers in the hidden layer is in a range from 6 to 14. The training results evaluated by the successful training rate [%] for the different NNM structures. As a result, the optimized NNM model in which the 3-layer RMLPNN with 7 neurons in the input layer and 8 neurons in the hidden layer can estimate the IPMC tip displacement with the highest accuracy (91.5%). Fig. 6(a).

Evaluations were done for input signals in Pseudo random binary square (PRBS) form, and sinusoidal form. As the result, the IPMC modeling result using the selected NNM model is displayed in Fig. 7. The result shows that the designed model has enough ability to estimate the IPMC bending behavior.

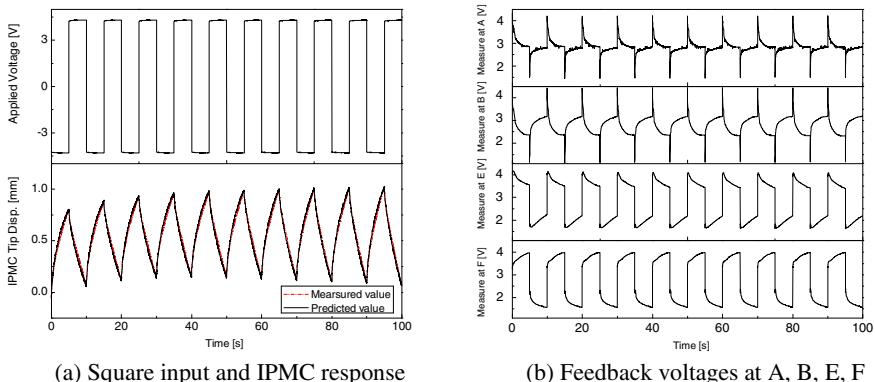
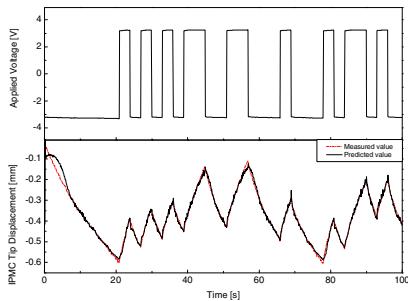
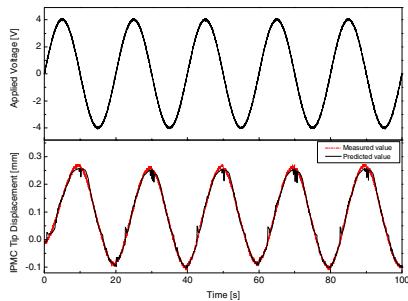


Fig. 6. IPMC training data and estimation result using NBBM model



(a) PRBS input and IPMC response (3[V])



(b) Sinusoidal input and IPMC response (4[V])

Fig. 7. Comparison between the estimated and actual IPMC bending behaviors

5 Conclusion

In this paper, an accurate model, NNM, is newly proposed and designed for IPMC actuators. The model is based on the RMLPNN and optimized by the SALM training method. The modeling results proved that the NNM model can describe well the IPMC dynamic bending characteristic with high precision. The proposed modeling method brings an effective solution for IPMC control applications with self sensing behavior in which the NNM model functions as a processor to ‘measure’ the IPMC tip displacement.

Acknowledgments. This work was supported by Mid-career Researcher Program through NRF grant funded by the MEST No. 2009-0080924.

References

1. C.Z., Tan, X.: A Control-Oriented and Physics-Based Model for Ionic Polymer–Metal Composite Actuators. *IEEE/ASME Trans. on Mech.* 13, 519–529 (2008)
2. Chen, Z., Kwon, K.Y., Xiaobo, T.: Integrated IPMC/PVDF sensory actuator and its validation in feedback control. *J. Sen. and Act. A: Physical.* 144(2), 231–241 (2008)
3. Bandopadhyay, D., Njuguna, J.: Estimation of bending resistance of Polymer Metal Composite (IPMC) actuator following variable parameters pseudo-rigid body model. *J. Mater. Let.* 63(9-10), 745–747 (2009)

Optimal Control Using Functional Type SIRMs Fuzzy Reasoning Method

Takashi Mitsuishi¹ and Yasunari Shidama²

¹ University of Marketing and Distribution Sciences, Kobe 651-2188, Japan

² Shinshu University, Nagano 380-8553, Japan

takashi_mitsuishi@red.umsd.ac.jp

Abstract. A mathematical framework for studying a fuzzy optimal control using functional type SIRMs reasoning method is discussed in this paper. The existence of SIRMs which minimize the cost function of fuzzy control system is proved with continuity of approximate reasoning and topological property of the set of membership functions in SIRMs.

Keywords: Fuzzy reasoning, functional type SIRMs model, optimal control, functional analysis.

1 Introduction

The optimization of fuzzy control discussed in this paper is different from conventional method such as classical control and modern control. We consider fuzzy optimal control problems as problems of finding the minimum (maximum) value of the performance function with feedback law constructed by IF-THEN rules through a fuzzy inference [4]-[6].

Various methods of calculation of control output from input information according to fuzzy set theory are introduced by many researcher. Most of these methods use IF-THEN production rules. Although Yubazaki proposed the single input rule modules connected fuzzy inference method (SIRMs method) that can decrease the number of fuzzy rules and membership functions radically in comparison with the usual fuzzy inference methods [1], functional type SIRMs method that functions are used instead of consequent constants is introduced by Seki [2].

Seki also proved the functional type SIRMs method is special case of T-S fuzzy model, and they are convertible under some conditions [3]. Since we also studied about optimization of T-S fuzzy model [4], in this study, we analyze functional type SIRMs method. We consider the SIRMs as the set of membership functions and consequent functions and cost function of fuzzy logic control in this study as a functional on the set to obtain them topological properties.

2 SIRMs Fuzzy Model

In this study, the following extension of functional type SIRMs model is considered.

$$\begin{aligned}
 \text{SIRM-1} : & \{R_j^1 : \text{if } x_1 = A_j^1 \text{ then } y_1 = g_j^1(x)\}_{j=1}^{m_1} \\
 & \dots \\
 \text{SIRM-}i : & \{R_j^i : \text{if } x_i = A_j^i \text{ then } y_i = g_j^i(x)\}_{j=1}^{m_i} \\
 & \dots \\
 \text{SIRM-}n : & \{R_j^n : \text{if } x_n = A_j^n \text{ then } y_n = g_j^n(x)\}_{j=1}^{m_n}
 \end{aligned} \tag{1}$$

Here, n is the number of SIRMs and $m_i (i = 1, 2, \dots, n)$ are the numbers of single input rules in each SIRM- i . x_1, x_2, \dots, x_n are premise variables. y_1, y_2, \dots, y_n are consequent variables. We write x for the input vector of the premise variables and also the input of this SIRMs connected fuzzy inference model.

Let $A_j^i(x_i) (i = 1, 2, \dots, n; j = 1, 2, \dots, m_i)$ be a fuzzy grade of each fuzzy set A_j^i for i -th input x_i . Let $g_j^i : \mathbb{R}^n \rightarrow \mathbb{R}$ be a function of the consequent output of j -th rule in i -th module. The membership function of fuzzy set A_j^i is written as same character A_j^i in this paper.

For simplicity, we write “if” and “then” parts in the rules by the following notation:

$$\begin{aligned}
 \mathcal{A}^i &= (A_1^i, A_2^i, \dots, A_{m_i}^i), \quad g^i = (g_1^i, g_2^i, \dots, g_{m_i}^i) \quad (i = 1, 2, \dots, n), \\
 \mathcal{A} &= (\mathcal{A}^1, \mathcal{A}^2, \dots, \mathcal{A}^n), \quad g = (g^1, g^2, \dots, g^n).
 \end{aligned}$$

Then, SIRMs (1) is called a fuzzy controller in this study, and is denoted by (\mathcal{A}, g) which is the pair of the membership functions and the consequent functions.

In this study, when an input information $x = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n$ is given to the fuzzy controller (\mathcal{A}, g) (SIRMs (1)), then one can obtain the amount of operation from the controller through the following approximate reasoning calculation.

The inference result of the rule group SIRM- i as a weighted average of the consequent output is calculated by

$$\beta_{\mathcal{A}^i g^i}(x_i) = \frac{\sum_{j=1}^{m_i} A_j^i(x_i) \cdot g_j^i(x)}{\sum_{j=1}^{m_i} A_j^i(x_i)} \quad (i = 1, 2, \dots, n).$$

where $A_j^i(x_i) (j = 1, 2, \dots, m_i; i = 1, 2, \dots, n)$ are the degree of premise part of j -th rule in SIRM- i .

In SIRMs inference method, the importance degrees $d_i (i = 1, 2, \dots, n)$ are introduced to give each SIRMs weight of contribution. In the same way as \mathcal{A} and g , put $d = (d_1, d_2, \dots, d_n)$. Then, the inference result of all rules $\rho_{\mathcal{A} g d}(x)$ is calculated by

$$\rho_{\mathcal{A} g d}(x) = \sum_{i=1}^n d_i \cdot \beta_{\mathcal{A}^i g^i}(x_i).$$

Here, since the inference result depends on the membership functions in the premise parts, the functions of the consequent outputs and the importance degrees, the subscripts \mathcal{A} , g and d are added to the function.

3 Nonlinear Fuzzy Feedback Control

\mathbb{R}^n denotes the n -dimensional Euclidean space with the usual norm $\|\cdot\|$. Let $f(v_1, v_2) : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}^n$ be a nonlinear vector valued function which is Lipschitz continuous. In addition, assume that there exists a constant $M_f > 0$ such that $\|f(v_1, v_2)\| \leq M_f (\|v_1\| + |v_2| + 1)$ for all $(v_1, v_2) \in \mathbb{R}^n \times \mathbb{R}$. Consider a system given by the following state equation:

$$\dot{x}(t) = f(x(t), u(t)),$$

where $x(t)$ is the state and the control input $u(t)$ of the system is given by the state feedback $u(t) = \rho(x(t))$. For a sufficiently large $r > 0$,

$$B_r = \{x \in \mathbb{R}^n : \|x\| \leq r\}$$

denotes a bounded set containing all possible initial states x_0 of the system. Let T be a sufficiently large final time. Then, we have

Proposition 1. *Let $\rho : \mathbb{R}^n \rightarrow \mathbb{R}$ be a Lipschitz continuous function and $x_0 \in B_r$. Then, the state equation*

$$\dot{x}(t) = f(x(t), \rho(x(t))) \quad (2)$$

has a unique solution $x(t, x_0, \rho)$ on $[0, T]$ with the initial condition $x(0) = x_0$ such that the mapping $(t, x_0) \in [0, T] \times B_r \mapsto x(t, x_0, \rho)$ is continuous.

For any $r_2 > 0$, denote by Φ the set of Lipschitz continuous functions $\rho : \mathbb{R}^n \rightarrow \mathbb{R}$ satisfying $\sup_{u \in \mathbb{R}^n} |\rho(u)| \leq r_2$. Then, the following a) and b) hold.

a) *For any $t \in [0, T]$, $x_0 \in B_r$ and $\rho \in \Phi$, $\|x(t, x_0, \rho)\| \leq r_1$, where*

$$r_1 = e^{M_f T} r + (e^{M_f T} - 1)(r_2 + 1). \quad (3)$$

b) *Let $\rho_1, \rho_2 \in \Phi$. Then, for any $t \in [0, T]$ and $x_0 \in B_r$,*

$$\|x(t, x_0, \rho_1) - x(t, x_0, \rho_2)\| \leq \frac{e^{L_f (1+L_{\rho_1})t} - 1}{1 + L_{\rho_1}} \sup_{u \in [-r_1, r_1]^n} |\rho_1(u) - \rho_2(u)|, \quad (4)$$

where L_f and L_{ρ_1} are the Lipschitz constants of f and ρ_1 [5].

In this study, assume that the feedback law ρ of the nonlinear system (2) is constructed based on the SIRMs (1). Then the input of SIRMs x depends on the time. But it is written as follows without distinction.

$$x = (x_1, x_2, \dots, x_n) = (x_1(t), x_2(t), \dots, x_n(t)) = x(t).$$

4 Functional Analytic Properties of Fuzzy Model

4.1 The Set of Membership Functions and Its Properties

In this section, we introduce the set of fuzzy membership functions and study their topological properties. Then we can show that a set of admissible fuzzy controllers is compact and metrizable with respect to an appropriate topology on fuzzy membership functions.

For a sufficiently large $r_2 > 0$, we can have the constant r_1 by proposition 1. Denote by $C[-r_1, r_1]$ be the Banach space of all continuous real functions on $[-r_1, r_1]$ with the norm $\|\mu\| = \max_{x \in [-r_1, r_1]} |\mu(x)|$.

Let $\Delta_{ij} > 0$ ($i = 1, 2, \dots, n; j = 1, 2, \dots, m_i$). We consider the following sets of fuzzy membership functions.

$$F_{\Delta_{ij}} = \{\mu \in C[-r_1, r_1] : 0 \leq \mu(x) \leq 1 \text{ for } \forall x \in [-r_1, r_1],$$

$$|\mu(x) - \mu(x')| \leq \Delta_{ij}|x - x'| \text{ for } \forall x, x' \in [-r_1, r_1]\}$$

The set $F_{\Delta_{ij}}$ above contains triangular, trapezoidal and bell-shaped fuzzy membership functions with gradients less than positive value Δ_{ij} . Consequently, if $\Delta_{ij} > 0$ is taken large enough, $F_{\Delta_{ij}}$ contains almost all fuzzy membership functions which are used in practical applications. We assume that the fuzzy membership functions A_j^i in premise parts of the SIRMs (1) belong to the set $F_{\Delta_{ij}}$ for all $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, m_i$.

Further, for $\Delta'_{ij} > 0$ ($i = 1, 2, \dots, n; j = 1, 2, \dots, m_i$), put

$$G_{\Delta'_{ij}} = \{g \in C[-r_1, r_1]^n : g(x) \in [-r_2, r_2] \text{ for } \forall x \in [-r_1, r_1]^n,$$

$$|g(x) - g(x')| \leq \Delta'_{ij}\|x - x'\| \text{ for } \forall x, x' \in [-r_1, r_1]^n\}.$$

Here, closed interval $[-r_2, r_2]$ is the domain of the control variables y_1, y_2, \dots, y_n which are also output of previous inference method. We consider the functions of consequent part g_j^i ($i = 1, 2, \dots, n; j = 1, 2, \dots, m_i$) are the element of the sets $G_{\Delta'_{ij}}$ respectively.

In the following, we endow the space $F_{\Delta_{ij}}$ and $G_{\Delta'_{ij}}$ with the norm topology. Then, by Ascoli Arzelà's theorem [9], for each $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, m_i$, $F_{\Delta_{ij}}$ is a compact subset of the Banach space $C[-r_1, r_1]$ of real continuous functions on $[-r_1, r_1]$ with the supremum norm $\|\cdot\|_\infty$. Similarly, $G_{\Delta'_{ij}}$ is a compact subset of $C[-r_1, r_1]^n$.

Put

$$\mathcal{F}' = \prod_{i=1}^n \left\{ \prod_{j=1}^{m_i} \left(F_{\Delta_{ij}} \times G_{\Delta'_{ij}} \right) \right\}.$$

Then, by Tychonoff theorem, we can have following proposition.

Proposition 2. \mathcal{F}' is compact and metrizable with respect to the product topology.

Let the n -tuple of importance degrees d join with fuzzy controller (\mathcal{A}, g) . Then it is denoted by (\mathcal{A}, g, d) . We can consider (\mathcal{A}, g, d) as the pair of SIRMs and importance degree and newly call it SIRMs fuzzy controller. In this paper, it is assumed that each $d_i (i = 1, 2, \dots, n)$ is belonging to closed interval $[0, 1]$, and satisfies $\sum_{i=1}^n d_i \leq 1$. Yubazaki does not need this condition [1], but it is needed in this study for the existence of solution of the state equation (2).

Put

$$\mathcal{F} = \prod_{i=1}^n \left\{ \prod_{j=1}^{m_i} \left(F_{\Delta_{ij}} \times G_{\Delta'_{ij}} \right) \right\} \times [0, 1]^n.$$

Then \mathcal{F} is Cartesian product and consists of SIRMs and importance degrees of inference calculations. And it is obvious that $(\mathcal{A}, g, d) \in \mathcal{F}$.

4.2 Admissible Fuzzy Controller

To avoid making the denominator of the expression $\beta_{\mathcal{A}^i g^i}$ in the approximate reasoning equal to 0, for any $\delta > 0$, we consider the set

$$\mathcal{F}_\delta = \left\{ (\mathcal{A}, g, d) \in \mathcal{F} : \forall i = 1, 2, \dots, n, \forall x \in [-r_1, r_1]^n, \sum_{j=1}^{m_i} A_j^i(x_i) \geq \delta \right\}, \quad (5)$$

which is a slight modification of \mathcal{F} . If δ is taken small enough, it is possible to consider $\mathcal{F} = \mathcal{F}_\delta$ for practical applications. We say that an element (\mathcal{A}, g, d) of \mathcal{F}_δ is an admissible fuzzy controller. Then, we have the following

Proposition 3. *The set \mathcal{F}_δ of all admissible fuzzy controllers is compact and metrizable with respect to the product topology.*

4.3 Lipschitz Continuity and Unique Solution of State Equation

In this paper, for any $(\mathcal{A}, g, d) \in \mathcal{F}_\delta$, we define the feedback function

$$\rho_{\mathcal{A}gd}(x) = \rho_{\mathcal{A}gd}(x_1, x_2, \dots, x_n) : [-r_1, r_1]^n \rightarrow \mathbb{R}$$

on the basis of the SIRMs by the approximate reasoning in the section 2. To apply the proposition 1, the following proposition is needed for the existence of unique solution of the state equation (2).

Proposition 4. *Let $(\mathcal{A}, g, d) \in \mathcal{F}_\delta$. Then, the following a) and b) hold.*

- a) $\rho_{\mathcal{A}gd}$ is Lipschitz continuous on $[-r_1, r_1]^n$.
- b) $|\rho_{\mathcal{A}gd}(x)| \leq r_2$ for all $x \in [-r_1, r_1]^n$.

Proof. a) For any $x = (x_1, x_2, \dots, x_n), x' = (x'_1, x'_2, \dots, x'_n) \in [-r_1, r_1]^n$ and any $i = 1, 2, \dots, n$, we have

$$|\beta_{\mathcal{A}^i g^i}(x_i) - \beta_{\mathcal{A}^i g^i}(x'_i)| \leq \frac{m_i^2}{\delta^2} (2r_2 \Delta_i + \Delta'_i) \|x - x'\| \quad (6)$$

Here, $\Delta_i = \max_{j=1,2,\dots,m_i} \{\Delta_{ij}\}$, $\Delta'_i = \max_{j=1,2,\dots,m_i} \{\Delta'_{ij}\}$, Hence the mapping $\beta_{\mathcal{A}^i g^i}$ is Lipschitz continuous on $[-r_1, r_1]^n$. Noting that $d_i \leq 1$ for all $x \in [-r_1, r_1]^n$, it follows from (6) that

$$\begin{aligned} |\rho_{\mathcal{A}gd}(x) - \rho_{\mathcal{A}gd}(x')| &\leq \sum_{i=1}^n |\beta_{\mathcal{A}^i g^i}(x_i) - \beta_{\mathcal{A}^i g^i}(x'_i)| \\ &\leq \frac{1}{\delta^2} \sum_{i=1}^n m_i^2 (2r_2 \Delta_i + \Delta'_i) \|x - x'\| \end{aligned}$$

and the Lipschitz continuity of $\rho_{\mathcal{A}gd}$ is proved. The proof of b) is omitted.

Let (\mathcal{A}, g, d) be a fuzzy controller given by the SIRMs (1). We say that the system (2) is a fuzzy feedback system if the control function $u(t)$ is given by the state feedback $u(t) = \rho_{\mathcal{A}gd}(x(t))$, where $\rho_{\mathcal{A}gd}(x(t))$ is the amount of operation from the fuzzy controller (\mathcal{A}, g, d) for an input information $x(t)$.

It is easily seen that every bounded Lipschitz function $\rho : [-r_1, r_1]^n \rightarrow \mathbb{R}$ can be extended to a bounded Lipschitz function $\tilde{\rho}$ on \mathbb{R}^n without increasing its Lipschitz constant and bound. In fact, define $\tilde{\rho} : \mathbb{R}^n \rightarrow \mathbb{R}$ by

$$\tilde{\rho}(x) = \tilde{\rho}(x_1, \dots, x_n) = \begin{cases} \rho(x_1, \dots, x_n), & \text{if } x \in [-r_1, r_1]^n \\ \rho(\varepsilon(x_1)r_1, \dots, \varepsilon(x_n)r_1), & \text{if } x \notin [-r_1, r_1]^n, \end{cases}$$

where

$$\varepsilon(u) = \begin{cases} 1, & \text{if } u > r_1 \\ -1, & \text{if } u < -r_1. \end{cases}$$

Let $(\mathcal{A}, g, d) \in \mathcal{F}_\delta$. Then it follows from proposition 4 and the fact above that the extension $\tilde{\rho}_{\mathcal{A}gd}$ of $\rho_{\mathcal{A}gd}$ is Lipschitz continuous on \mathbb{R}^n with the same Lipschitz constant of $\rho_{\mathcal{A}gd}$ and satisfies $\sup_{u \in \mathbb{R}^n} |\tilde{\rho}_{\mathcal{A}gd}(u)| \leq r_2$. Therefore, by proposition 1 the state equation (2) for the feedback law $\tilde{\rho}_{\mathcal{A}gd}$ has a unique solution $x(t, x_0, \tilde{\rho}_{\mathcal{A}gd})$ with the initial condition $x(0) = x_0$ [8]. Though the extension $\tilde{\rho}_{\mathcal{A}gd}$ of $\rho_{\mathcal{A}gd}$ is not unique in general, the solution $x(t, x_0, \tilde{\rho}_{\mathcal{A}gd})$ is uniquely determined by $\rho_{\mathcal{A}gd}$ using the inequality (4) of b) of proposition 1. Consequently, in the following the extension $\tilde{\rho}_{\mathcal{A}gd}$ is written as $\rho_{\mathcal{A}gd}$ without confusion.

5 Application to Optimal Control Problem

The performance index of this fuzzy feedback control system is evaluated with the following integral performance function:

$$J = \int_{B_r} \int_0^T w(x(t, \zeta, \rho_{\mathcal{A}gd}), \rho_{\mathcal{A}gd}(x(t, \zeta, \rho_{\mathcal{A}gd}))) dt d\zeta, \quad (7)$$

where $w : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}$ is a positive continuous function. The following theorem guarantees the existence of a SIRMs fuzzy controller (\mathcal{A}, g, d) which minimizes and maximizes the previous performance function (7).

Theorem 1. *The mapping*

$$(\mathcal{A}, g, d) \mapsto \int_{B_r} \int_0^T w(x(t, \zeta, \rho_{\mathcal{A}gd}), \rho_{\mathcal{A}gd}(x(t, \zeta, \rho_{\mathcal{A}gd}))) dt d\zeta$$

has a minimum (maximum) value on the compact space \mathcal{F}_δ defined by (5).

Proof. Since compactness of \mathcal{F}_δ is already derived by proposition 3, it is sufficient to prove that the performance function is continuous on \mathcal{F}_δ . Routine calculation gives the estimate

$$\begin{aligned} \sup_{x \in [-r_1, r_1]^n} |\rho_{\mathcal{A}^k g^k d^k}(x) - \rho_{\mathcal{A}gd}(x)| &\leq \frac{2r_2}{\delta^2} \sum_{i=1}^n \left\{ m_i \sum_{j=1}^{m_i} \left| A_j^{i^k}(x) - A_j^i(x) \right| \right\} \\ &+ \frac{1}{\delta^2} \sum_{i=1}^n \left\{ m_i \sum_{j=1}^{m_i} \left| g_j^{i^k}(x) - g_j^i(x) \right| \right\} + r_2 \sum_{i=1}^n |d_i^k - d_i|. \end{aligned}$$

Assume that $(\mathcal{A}^k, g^k, d^k) \rightarrow (\mathcal{A}, g, d)$ in \mathcal{F}_δ and fix $(t, \zeta) \in [0, T] \times B_r$. Then it follows from the estimate above that

$$\lim_{k \rightarrow \infty} \sup_{x \in [-r_1, r_1]^n} |\rho_{\mathcal{A}^k g^k d^k}(x) - \rho_{\mathcal{A}gd}(x)| = 0. \quad (8)$$

Hence, by b) of proposition 1, we have

$$\lim_{k \rightarrow \infty} \|x(t, \zeta, \rho_{\mathcal{A}^k g^k d^k}) - x(t, \zeta, \rho_{\mathcal{A}gd})\| = 0. \quad (9)$$

Further, it follows from (8), (9) and a) of proposition 1 that

$$\lim_{k \rightarrow \infty} \rho_{\mathcal{A}^k g^k d^k}(x(t, \zeta, \rho_{\mathcal{A}^k g^k d^k})) = \rho_{\mathcal{A}gd}(x(t, \zeta, \rho_{\mathcal{A}gd})). \quad (10)$$

Noting that $w : \mathbb{R}^n \times \mathbb{R} \rightarrow \mathbb{R}$ is positive and continuous, it follows from (9), (10) and the Lebesgue's dominated convergence theorem [10] that the mapping is continuous on the compact metric space \mathcal{F}_δ . Thus it has a minimum (maximum) value on \mathcal{F}_δ , and the proof is complete.

6 Conclusion

Fuzzy logic control which the feedback law is constructed by fuzzy inference method in the nonlinear feedback control was studied. To guarantee the convergence of optimal solution, the compactness of the set of membership functions is proved. And assuming fuzzy inference to be a functional on that set, its continuity is obtained. Then, it is shown that the system has an optimal feedback control by essential use of compactness of sets of fuzzy membership functions. The pair of membership functions, in other words SIRMs, which minimize the integral cost function of fuzzy logic control exists. Two kinds of continuity of

the inference method (approximate reasoning) on the single input rule modules (SIRMs) are proved. One is Lipschitz continuity on the premise variables, the other is the continuity as a functional on the compact set of membership functions. Then it is concluded that the set of membership functions, in other words SIRMs, which gives optimal control to the nonlinear feedback control exists.

Seki proposed the conversion of the SIRMs connected type fuzzy model into the simplified reasoning method [2]. And furthermore, the same result as this study concerning the simplified reasoning method has obtained [4] [6]. So in the future work, the relation between both should be analyzed.

References

1. Yubazaki, N., Yi, J., Hirota, K.: A Proposal of SIRMs (Single Input Rule Modules) Connected Fuzzy Inference Model for Plural Input Fuzzy Control. *Journal of Japan Society for Fuzzy Theory and Systems* 9(5), 699–709 (1997)
2. Seki, H., Ishii, H., Mizumoto, H.: On the generalization of single input rule modules connected type fuzzy reasoning method. In: Proc. of Joint 3rd International Conference on Soft Computing and Intelligent Systems and 7th International Symposium on advanced Intelligent Systems (SCIS&ISIS 2006), pp. 30–34 (2006)
3. Seki, H., Ishii, H.: On the Extension of Functional Type SIRMs Fuzzy Reasoning Method. *IEICE technical report* 106(576), pp. 7–10 (2007) (in Japanese)
4. Mitsuishi, T., Wasaki, K., Ohkubo, K., Kawabe, J., Shidama, Y.: Fuzzy Optimal Control Using Simple Inference Method and Function Type Inference Method. In: Proc. American Control Conference 2000, pp. 1944–1948 (2000)
5. Mitsuishi, T., Kawabe, J., Wasaki, K., Shidama, Y.: Optimization of Fuzzy Feedback Control Determined by Product-Sum-Gravity Method. *Journal of Nonlinear and Convex Analysis* 1(2), 201–211 (2000)
6. Mitsuishi, T., Shidama, Y.: Minimization of Quadratic Performance Function in T-S Fuzzy Model. In: Proc. International Conference on Fuzzy Systems (FUZZ–IEEE 2002), pp. 75–79 (2002)
7. Mitsuishi, T., Shidama, Y.: Continuity of fuzzy approximate reasoning and its application to optimization. In: Orgun, M.A., Thornton, J. (eds.) *AI 2007. LNCS (LNAI)*, vol. 4830, pp. 529–538. Springer, Heidelberg (2007)
8. Miller, R.K., Michel, A.N.: *Ordinary Differential Equations*. Academic Press, New York (1982)
9. Riesz, F., Sz.-Nagy, B.: *Functional Analysis*. Dover Publications, New York (1990)
10. Dunford, N., Schwartz, J.T.: *Linear Operators Part I: General Theory*. John Wiley & Sons, New York (1988)

High-Dimensional Surveillance

Saylisse Dávila¹, George Runger¹, and Eugene Tuv²

¹ School of Computing, Informatics, and Decision Systems Engineering,

Arizona State University, Tempe, AZ, USA

s v george r ger s e

² Intel Corporation, Chandler, AZ, USA

e g e v e o

Abstract. Many systems (manufacturing, environmental, health, etc.) generate counts (or rates) of events that are monitored to detect changes. Modern data complements event counts with many additional measurements (such as geographic, demographic, and others) that comprise high-dimensional attributes. This leads to an important challenge to detect a change that only occurs within a region, initially unspecified, defined by these attributes and current methods to handle the attribute information are challenged by high-dimensional data. Our approach transforms the problem to supervised learning, so that properties of an appropriate learner can be described. Rather than error rates, we generate a signal (of a system change) from an appropriate feature selection algorithm. A measure of statistical significance is included to control false alarms. Results on simulated examples are provided.

Keywords: Feature selection, tree ensembles, process control.

1 Introduction

Many systems generate counts (or rates) of events that are monitored to detect changes. This is a common problem in disciplines such as manufacturing, economics, environmental systems, health, and so forth. In health surveillance, the events might be disease incidents, while in manufacturing, the events might simply be surface flaws or incorrectly soldered components. The discrete nature of such data poses significant challenges to traditional monitoring approaches. Furthermore, modern data typically supplements the counts with a substantial number of important additional attributes (e.g., geographic, demographic, and so forth) that generate categories, or regions in which counts may be organized.

It is relatively easy to detect a change in the overall rate of an event. However, an aggregated count of events is not sensitive to changes in localized regions in attribute space—for example, an increase in asthma in a specific ethnic or age group and/or a specific geographic region. Consequently, a primary challenge of surveillance is to detect a rate change in localized regions of attribute space. Such a localized change requires a more subtle analysis than the aggregated counts of events, particularly when

This material is based upon work supported by the National Science Foundation under Grant 0743160 and the Office of Naval Research under Grant N000140910656.

the number of attributes and or attribute value are more than a few. For example, to monitor asthma events using a weekly count, one might need to define attributes for each of several age groups. Then, the age attribute might need to be crossed with spatial regions to obtain counts in every combination of age group and region. This generates a rapid expansion to the number of regions that limits such an approach.

Our objective is to detect changes through a transform of surveillance to supervised learning. Such an approach can greatly increase the flexibility and robustness of numerous public health monitoring and informatics tasks. We apply computationally-intensive, but feasible, non-linear learners. We focus on the case with high-dimensional attributes where traditional methods are challenged. A change in *any* region (or multiple regions) of this high-dimensional is of interest, and the regions are not pre-specified. Furthermore, a region can be defined from cross-products of either categorical or numerical attributes (or both). We assume case-event data is available. Data from a modern public health database with case-event data is used in example. The background is frequently provided by historical data of the same type of event, but in some cases the background could be generated from a different type of event. We refer to both cases as background. We assume that, in addition to background data, we are given new data that consists of several instances of the event. For public health surveillance, this would commonly consist of a collection of events (such as disease incidents in a week) along with associated attribute information.

Section 2 describes background methods used in surveillance. Section 3 defines the transform to supervised learning, and includes the important topic of signal criteria. Section 4 shows examples of our proposed method on a variety of simulated data sets.

2 Background for Surveillance

Some of the most common methods for event surveillance include: statistical process control, distance and density estimates, and scan statistics. Statistical process control (SPC) has been extensively used in industry to monitor the quality of the manufacturing processes. The connection between SPC and how to monitor events, for data summarized in counts, is clear [3]. One of the major issues with such approach is that in order to monitor for multiple spatial-attribute regions, these regions would need to be defined and each would provide an event rate. Separate control charts for each event type are well-known to increase false alarms and, more importantly, they do not incorporate the valuable information available in the correlations among attributes.

Another approach is to consider an intensity function that defines the mean event rate at values of the spatial locations [10]. One would need to extend from the spatial information to incorporate additional attributes. Then, the intensity can be taken to be a function (\mathbf{x}) for values of the attributes and spatial data specified by a vector \mathbf{x} . Still, developing a monitoring scheme using intensity functions requires several steps. Estimates are needed for the intensity functions. For a nonparametric method, density estimates can be used. These densities must be compared to a background to detect rate changes. For a modeling approach with count data, likelihood and Bayesian generalized linear models can be applied to estimate (\mathbf{x}) . Also, [9] provided an extensive literature review on spatial and temporal models using likelihood and Bayesian methods. This

is also related to point-processes that use probability models (e.g., Poisson) based on random counting measures. A point process is a random measure on a space S (with range the non-negative integers) and (B) represents the number of points in the subset B of S . The probability models can add strong assumptions that we prefer to avoid.

Although intensity estimates can serve other purposes, surveillance objectives can be met without direct estimates over the entire space of attributes. Instead, only the regions of attribute space where background and current intensities differ are of interest. For example, typically one would like to detect increased asthma in a subpopulation (such as children under age ten) in a geographic region. Neither the age nor the region is pre-specified. Consequently, a simpler method can focus to detect a difference between the two sets of data, and this is easier than densities fit to the entire attribute space. From a practical perspective, density estimators are not easy to implement and do not naturally incorporate categorical attributes. Besides, the background and current densities still need to be compared with a decision rule.

Also, event counts can be spatially clustered through distance methods. A well known example was presented by [12]. [11] also monitored a global spatial clustering statistic at each point in time. A measure based on the empirical cumulative distribution function of inter-point distances was presented by [1]. Important attribute information is typically difficult to integrate in distance methods and separate models are needed. These statistics, still, only provide an overall signal of a cluster(s). Extra work is needed to identify the type and location of clusters.

The scan statistic method has been extensively applied to spatial [7], temporal [15], and spatio-temporal [8] surveillance. In the temporal case, the scan statistic resembles the well-known CUSUM control chart. The scan statistic, conceptually, can be applied to one (or few) other attributes other than spatial and temporal data [6]. However, as the number of attributes increases, computations increase exponentially making the scan statistic an ineffective method. Consequently, only spatial and temporal attributes are typically incorporated into the scan statistic. Other attributes need to be integrated through expected values and this requires additional models (such as logistic regression, see below), and it is difficult (and not typical) to allow for any higher-order interactions in these models. Without the simultaneous study of multiple attributes, regions defined by higher-order interactions can be missed.

A method based on logistic regression uses a generalized linear mixed model (GLMM) to estimate the class probability of an event [5]. GLMMs can manage spatial, temporal, and other attributes in numerical and or categorical format. However, as the number of attributes increases, so does the number of scenarios to monitor. With as little as seven three-level categorical attributes, there are over 2000 scenarios to monitor. This dimensional explosion tends to limit the sensitivity of the method. Moreover, when a cluster is spread out across all levels of a attribute, the signal can be easily missed when monitoring each level individually. For example, if there is an increase in asthma that affects all age groups in a given region, monitoring teenagers alone can fail to detect the cluster.

3 Transform to Supervised Learning

Our interest is to detect differences between the rates in different regions over time. An approach is to contrast data from different time periods; that is, to learn differences directly through a transform of the problem to supervised learning. There is typically less subjectivity in supervised learning models than in other alternatives (such as density estimates) so that models are easier to score, evaluate, compare, and refine as needed.

We consider attribute data \mathbf{x}_i from background and new data. The new data might be obtained from an window of time such as hourly, daily, weekly, etc.. For each event in a background and new data, we assign the class label $y_i = 0$ and $y_i = 1$, respectively. A supervised model is then used to predict the class based on the attributes. The model is not expected to predict well if the new and background events have similar attribute information. Conversely, a supervised learner that can detect different rates in any region of the attribute space can be used to signal a change. We do not require the form of a region to be pre-specified and one or multiple regions can be detected with a suitable learner. An example figure that shows a region of greater intensity of events is shown in Figure 1. Note that location data is typically two-dimensional, but the figure only shows one axis for simplicity. A region can be defined by location as well as by any number of attributes.

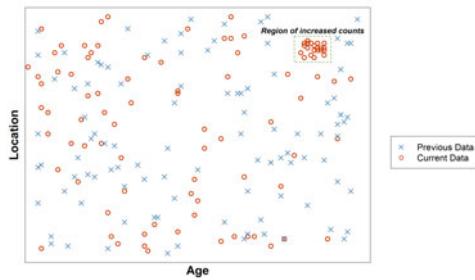


Fig. 1. Example of increased intensity in one local region

Figure 1 illustrates an increased rate for a particular age group and geographical location. This illustrates an interaction between the location and age attributes in the supervised learning model. Consequently, a suitable learner should be sensitive to interactions between attributes. Also, this example illustrates two attributes, but higher-order interactions (and categorical attributes) are of interest.

A transform to supervised learning allows events to be assigned to classes, but an important issue is the criterion to generate a signal from a rate change. Error rates are natural scores from a learner. If error rates are high, the learner cannot distinguish between the new and background data. However, one expects background events to occur in many regions of attribute space. It is, therefore, not surprising that some background events (Class 0) are predicted to be Class 1 events. Of greater interest are the Class 1

events that are predicted correctly. Still, even with an increased rate in a localized region, we might also expect a number of events in the new data that are similar to the background. These cases are not expected to be classified correctly by the learner, and this would attenuate the signal for the Class 1 error rate (particularly when the ratio of random cases to cases in a localized region increases).

An alternative approach is to consider feature selection (FS) [4] as a means to generate a signal. The role of FS for surveillance is to identify attributes that distinguish the background and new data and, thereby, contribute to a region with rate changes. If no attributes are identified, no signal is generated. On the contrary, selected attributes are those that define the region with the rate change. Although a number of FS methods may be used, it is important that the method is sensitive to interactions among the attributes that define the localized regions of interest for surveillance. The number of cases in a region may also be small relative to the number of cases in the new data.

We adopt the FS method based on tree ensembles described by [13]. A single tree is generated by a greedy algorithm that can lead to an unstable model. Ensemble methods construct multiple models (called base learners) and, in a parallel ensemble, the weighted outcome (or vote) is used to classify new data. An exemplar for tree-based ensembles is a random forest model [2]. Tree ensembles can incorporate nonlinear models that can adapt to the characteristics of the background environment; categorical, numerical, and missing data; potentially large interactive effects, invariance to attribute units, insensitivity to outliers, and even a large number of attributes and events. Tree-based ensembles provide embedded importance measures that can identify the feature relevant to a model. For surveillance, it is important to control risks associated with false alarms and missed signals. The FS method by [13] incorporates a statistical criterion (and other enhancements) to ensembles. The approach generates artificial variables from random permutations of each of the actual attributes in the data set. Random permutations break relationships between the class and each attribute. Permutations still maintain the same marginal distribution of the attribute. These artificial attributes are constructed for categorical or numerical attributes in exactly the same manner.

Only attributes with a significant differences (as measured with p -values) from the percentile (typically 90%) of artificial attributes are selected as an important features (and Bonferroni's adjustments may be applied). In addition to artificial attributes, further enhancements include the split weight estimated from out-of-bag (OOB) samples. Furthermore, the effect of identified important attributes are iteratively removed with a serial ensemble model to allow detection of less important attributes. More generally, variables that mask each other can both be identified as important variables and masked sets can be indicated. There are advantages of our chosen FS algorithm for surveillance, but that others can also be applied.

4 Examples

4.1 High-Dimensional Data

The first example illustrates the ability of the method to detect changes even among high-dimensional data sets. The simulated data set includes 10,000 baseline cases randomly generated among 100 attributes. The attributes are uniformly distributed between

0 and 1, but any distribution, or even categorical attributes could be used. New data with single or multiple clusters were generated among multiple attributes with different numbers of events. Clusters within attributes were created by defining smaller ranges in the specific attributes for the events involved in the clusters. For example, one experiment formed a cluster with 20 events within a region defined by two attributes (labeled x_1 and x_2). A second experiment formed three clusters with 20 events each. Attributes x_1 – x_5 were used to define the three clusters, but only attributes x_1 and x_2 were involved in all three clusters. Figure 2 shows that the greatest variable importance scores clearly corresponds to the important attributes in these examples. In Figure 2(a), only attributes x_1 and x_2 stand out as important; whereas in Figure 2(b), attributes x_1 – x_5 stand out as important. Notice, still, that attributes x_1 and x_2 in Figure 2(b) are ranked significantly higher than the rest of the attributes in the cluster. Permuted artificial attributes can be added to generate statistical significance tests. Even so, there was no need to further look at those permutations to select important attributes in this set of experiments. Separation was highly distinct by looking only at the standard importance measures.

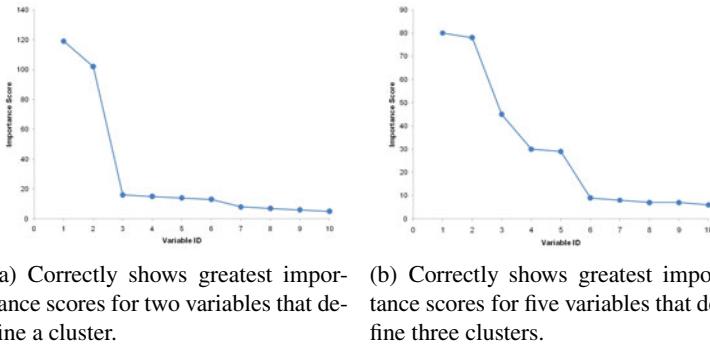


Fig. 2. Variable Importance for clusters among 100 variables (only the top 10 scores shown)

4.2 Data with Multiple Locations

A problem that commonly arises in public health surveillance is how to simultaneously monitor for disease clusters in a large number of cities or even counties. This example consists of 50 categorical geographical locations (e.g., 50 counties) with 3600 baseline events and 1200 new event randomly distributed. The data set also includes 15 numerical attributes uniformly distributed between 0 and 1, and 10 extra categorical attributes. Each of the additional categorical attributes is discrete uniform with 5 to 30 levels.

A disease cluster was superimposed on geographical location by forcing this categorical attribute to level 1 (County 1) in 100 test cases. We generated 10 different replicates for this example. Attribute selection criteria for this example was based on a 0.05 p -value and a 0.9 percentile importance score threshold. Our method accurately detected *County* in all replicates. There were additional attributes detected (false positives) in data sets 1, 4, 6, 7, and 8. Note that the random sampling strategy used can create scenarios where attributes that were not intended to be part of the superimposed

clusters still turned out to be part of them, simply because of the nature of the random sample. Nevertheless, none of these additional attributes was selected more than twice in the 10 different replicates. In addition, the total system time for these experiments was, on average, 44.0 seconds with a standard deviation of 10.0 seconds.

4.3 Disease Clusters with Attribute Interaction

We consider six different experiments with 2,500 baseline cases and 625 test cases (with these parameters chosen similar to an actual case study). The event attributes for these examples include: *latitude*, *longitude*, *gender* (2 values), *age group* (10 values), and *ethnicity* (6 values). Here *latitude* and *longitude* are the geographical coordinates associated with an event and are, thus, numerical. Experiment 0 represents the null case where all data was random within the Arizona state (and state equivalent territories polygon [14]). Experiment 1 generated all data randomly, except for the geographical coordinates of 50 cases in Class 1, which were simulated to be clustered around a random point within the polygon. Experiments 2-3 are defined in the same manner as Experiment 1, but with 100 and 200 cases within the cluster, respectively. Experiment 4 represents a dual cluster of 100 cases in *age* and *space*. Besides modifying the geographical coordinates for 100 cases in Class 1, the *age group* for these cases was fixed to “11 - 20”. Experiment 5 defined one cluster as in Experiment 4. A second cluster in Experiment 5 was defined by setting *ethnicity* “Hispanic” for the 100 cases in the cluster. Each experiment was replicated 10 times, and a planar projection of the geographical coordinates (*x*,*y*) was used for all further analyses.

Table 1. Feature Selection Results with Attribute Interactions

Experiment	<i>age group</i>	<i>gender</i>	<i>ethnicity</i>	<i>x</i>	<i>y</i>
0	0	0	0	0.1	0
1	0	0	0	1	0.9
2	0	0	0	0.9	0.9
3	0	0	0	1	0.9
4	1	0	0	1	1
5	1	0	1	1	1

FS for these experiments was performed using the algorithm by [13] with the following parameters: 100 trees, 100 replicates, and the 0.9 percentile for the artificial importance scores. Each entry in Table 1 is the proportion of replicates in which a given attribute had a *p*-value < 0.01 in 10 independent replicates. Ideally, all entries are 0 or 1 and the results show that the true and false positive rates are from 0.9-1.0 and 0.1-0.0, respectively, in these experiments.

5 Conclusions

The objective to monitor counts is common in many disciplines, but modern data marks these counts a high-dimensional list of attributes. Consequently, a change in any region of attribute space is more difficult to detect. A supervised learner is used that can

handle high-dimensional data of mixed numerical and categorical types, and strong interaction effects. An important question is the criterion to use to generate a signal (with control of true and false positive rates). We derive a criterion from feature selection that can provide a statistically objective decision rule. An advantage of our method against alternatives, such as the spatial scan statistic, is our ability to identify changes in all attributes. Alternatives are challenged by the high-dimensional data, and as data and computationally power increases, there is a need to look throughout the data for signals. Our approach to first identify attributes that contribute to a signal (if any) is expected to reduce the complexity of the problem to many fewer attributes. Then, simpler methods can isolate the locations of changes.

References

1. Bonetti, M., Pagano, M.: The interpoint distance distribution as a descriptor of point patterns, with an application to spatial disease clustering. *Statistics in Medicine* 24(5), 753–773 (2005)
2. Breiman, L.: Random forests. *Machine Learning* 45(1), 5–32 (2001)
3. Fonseca Nobre, F., Sa Carvalho, M.: Spatial and temporal analysis of epidemiological data. *GIS for Health and the Environment* (1995)
4. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. *Journal of Machine Learning Research* 3, 1157–1182 (Mar 2003)
5. Kleinman, K., Lazarus, R., Platt, R.: A Generalized Linear Mixed Models Approach for Detecting Incident Clusters of Disease in Small Areas, with an Application to Biological Terrorism. *American Journal of Epidemiology* 159(3), 217–224 (2004),
p e o or o r s org g o e s r
6. Kulldorff, M., Information Management Services, Inc.: SaTScan™ v8.0 User Guide (2009)
7. Kulldorff, M.: A spatial scan statistic. *Communications in Statistics–Theory and Methods* 26(6), 1481–1496 (1997)
8. Kulldorff, M.: Prospective time periodic geographical disease surveillance using a scan statistic. *Journal of the Royal Statistical Society* 164(1), 61–72 (2001)
9. Lawson, A.B., Browne, W.J., Vidal-Rodeiro, C.L.: *Disease Mapping with WinBUGS and MLwiN*. John Wiley and Sons, Chichester (2003)
10. Lawson, A.B., Waller, L.A.: A review of point pattern methods for a spatial modelling of events around sources of pollution. *Environmetrics* 7(5), 471–487 (1996)
11. Rogerson, P.: Monitoring point patterns for the development of space–time clusters. *Journal of the Royal Statistical Society* 164(1), 87–96 (2001)
12. Tango, T.: A class of tests for detecting ‘general’ and ‘focused’ clustering of rare diseases. *Statistics in Medicine* 14(21–22), 2323–2334 (1995)
13. Tuv, E., Borisov, A., Runger, G., Torkkola, K.: Feature selection with ensembles, artificial variables, and redundancy elimination. *J. Mach. Learn. Res.* 10, 1341–1366 (2009)
14. U.S. Census Bureau Geography Division: Census 2000: State and state equivalent areas in arcview shapefile (.shp) format (July 2001),
p e s s g o v g e o o s
15. Weinstock, M.A.: A generalised scan statistic test for the detection of clusters. *International Journal of Epidemiology* 10(3), 289–293 (1981),
p e o or o r s org g o e s r

A One-Layer Dual Recurrent Neural Network with a Heaviside Step Activation Function for Linear Programming with Its Linear Assignment Application

Qingshan Liu¹ and Jun Wang^{2,*}

¹ School of Automation, Southeast University, Nanjing 210096, China
qsliu@seu.edu.cn

² Department of Mechanical and Automation Engineering
The Chinese University of Hong Kong, Shatin, New Territories, Hong Kong
jwang@mae.cuhk.edu.hk

Abstract. This paper presents a one-layer recurrent neural network for solving linear programming problems. The proposed neural network is guaranteed to be globally convergent in finite time to the optimal solutions under a mild condition on a derived lower bound of a single gain parameter. The number of neurons in the neural network is the same as the number of decision variables of the dual optimization problem. Compared with the existing neural networks for linear programming, the proposed neural network has salient features such as finite-time convergence and lower model complexity. Specifically, the proposed neural network is tailored for solving the linear assignment problem with simulation results to demonstrate the effectiveness and characteristics of the proposed neural network.

Keywords: Recurrent neural networks, linear programming, global convergence in finite time, linear assignment problem.

1 Introduction

Consider a linear programming problem as follows:

$$\begin{aligned} & \text{minimize} && c^T x, \\ & \text{subject to} && Ax = b, \quad x \geq 0, \end{aligned} \tag{1}$$

where $x = (x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n$ is the decision vector, $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$.

Linear programming has widespread application scope in science and engineering, such as filter design, signal processing and power system planning. Over the

* The work described in the paper was supported by the Research Grants Council of the Hong Kong Special Administrative Region, China, under Grants CUHK417608E and CUHK417209E.

past two decades, as parallel computational models for optimization, recurrent neural networks have received substantial attention [1][2]. Specially, recurrent neural networks for linear programming have been well developed. In 1986, Tank and Hopfield [1] proposed a recurrent neural network for solving the linear programming problems for the first time. In 1988, the dynamical canonical nonlinear programming circuit (NPC) with a finite penalty parameter was introduced by Kennedy and Chua [2] for nonlinear programming, which can generate the approximate optimal solutions. Wang [3][4][5] proposed some primal-dual, primal, and dual neural networks for solving linear programming, the shortest path and linear assignment problems. Xia [6] proposed a primal-dual neural network for solving the linear programming problems. Forti et al. [7] investigated the generalized NPC (G-NPC) for non-smooth optimization, which can be considered as a natural extension of NPC. The G-NPC with a finite penalty parameter is limited for optimization problems with bounded feasible region [7]. In [8], we proposed a one-layer recurrent neural network for solving the linear programming problems, which had lower model complexity. However, the model in [8] includes a matrix inversion which needs to be calculated off line, which increases its complexity in some sense.

In view that such as linear assignment [4][9] and shortest path problems [5][10][11], the feasible region of the optimization problems are mostly unbounded. Then the neural network in [7] is not capable of solving these problems. This paper is concerned with a one-layer recurrent neural network for linear programming, in which the feasible region may be unbounded and no matrix inversion is calculated in the model. As a result, the proposed neural network is capable of solving more general linear programming problems. In addition, the global convergence of the recurrent neural network is guaranteed to be achieved in finite time provided that its gain parameter in the nonlinear term exceeds a given lower bound. Moreover, the proposed neural network is utilized for solving the linear assignment problems.

2 Model Description

Consider (1) as the primal problem, then its dual problem is [12]

$$\begin{aligned} & \text{maximize} && b^T y, \\ & \text{subject to} && A^T y \leq c, \end{aligned} \tag{2}$$

where $y = (y_1, y_2, \dots, y_m)^T \in \mathbb{R}^m$ is the dual vector, A , b and c are defined in (1). This section describes the recurrent neural network model for solving the dual linear programming problem (2).

According to the Karush-Kuhn-Tucker conditions [12], y^* is an optimal solution of problem (2), if and only if there exists $z^* \in \mathbb{R}^n$ such that

$$-b + Az = 0, \tag{3}$$

$$\begin{cases} z_i \geq 0, & \text{if } a_i^T y - c_i = 0, \\ z_i = 0, & \text{if } a_i^T y - c_i < 0, \end{cases} \tag{4}$$

where z_i and c_i are the i th components of z and c respectively, and a_i denotes the i th column of A ($i = 1, 2, \dots, n$).

Based on (3) and (4), the dynamic equation of the proposed recurrent neural network model is described as follows:

$$\epsilon \frac{dy}{dt} = -\sigma Ag(A^T y - c) + b, \quad (5)$$

where ϵ is a positive scaling constant, σ is a nonnegative gain parameter, $g(v) = (g_1(v), g_2(v), \dots, g_n(v))^T$ is the heaviside step activation function and its component is defined as

$$g_i(v) = \begin{cases} 1, & \text{if } v > 0, \\ [0, 1], & \text{if } v = 0, \\ 0, & \text{if } v < 0. \end{cases} \quad (i = 1, 2, \dots, n) \quad (6)$$

3 Convergence and Optimality Analysis

In this section, the finite-time global convergence of the proposed neural network is presented, and the optimal solution of problem (2) is guaranteed using the proposed neural network with sufficiently large gain parameter σ over a derived lower bound. In this paper, we denote $\Gamma = \{\gamma : \gamma = (\gamma_1, \gamma_2, \dots, \gamma_n)^T \in \mathbb{R}^n, 0 \leq \gamma \leq 1 \text{ and at least one } \gamma_i = 1\}$ and $A\Gamma = \{A\gamma : \gamma \in \Gamma\}$.

Let $\psi(y) = \sigma e^T (A^T y - c)^+ - b^T y$, where $e = (1, 1, \dots, 1)^T \in \mathbb{R}^n$ and $v^+ = \max\{0, v\}$. Then its generalized gradient is $\partial\psi(y) = \sigma AK[g(A^T y - c)] - b$, where $K(\cdot)$ denotes the closure of the convex hull. Then the neural network in (5) is a gradient system of energy function $\psi(y)$. Since $\psi(y)$ is convex, the minimum point of $\psi(y)$ corresponds to the equilibrium point of neural network (5). Thus, if neural network (5) has a stable equilibrium point, then the minimum point of energy function $\psi(y)$ is guaranteed. Next, we give the finite-time global convergence of the proposed neural network as follows.

Theorem 1. *If the minimum point of $\psi(y)$ is finite, then the state vector of neural network (5) is globally convergent to an equilibrium point in finite time with any $\sigma \geq 0$.*

Proof: From the assumption, the equilibrium point set of neural network (5) is not empty. Denote \bar{y} as an equilibrium point of neural network (5), then it is a minimum point of $\psi(y)$. It follows that $0 \in \partial\psi(\bar{y})$.

Consider the following Lyapunov function:

$$V(y) = \epsilon(\psi(y) - \psi(\bar{y})) + \frac{\epsilon}{2}(y - \bar{y})^T(y - \bar{y}), \quad (7)$$

We have $\partial V(y) = \epsilon(\partial\psi(y) + y - \bar{y})$. Similar to the proof of Theorem 3 in [13], it follows that

$$\dot{V}(y(t)) \leq -\inf_{\eta \in \partial\psi(y)} \|\eta\|_2^2, \quad (8)$$

and the state vector of neural network (5) is globally convergent to an equilibrium point.

Next, we prove the finite-time convergence. Assume that $y(t)$ is not an equilibrium point, so that $0 \notin \partial\psi(y)$. Since $\partial\psi(y) = \sigma AK[g(A^T y - c)] - b$ is nonempty and compact, one gets that $\inf_{\eta \in \partial\psi(y)} \|\eta\|_2^2$ is a positive constant, denoted as β . Then, from (8), we have

$$\dot{V}(y(t)) \leq -\beta.$$

Integrating both sides of the above inequality from 0 to t , it is easily to verify that $V(y(t)) = 0$ for $t \geq V(y(0))/\beta$. From (7), since $V(y(t)) \geq \epsilon\|y - \bar{y}\|_2^2/2$, we get that $y = \bar{y}$ for $t \geq V(y(0))/\beta$. That is, $y(t)$ is globally convergent to an equilibrium point in finite time. \square

From the above analysis, the proposed neural network is guaranteed to reach an equilibrium point in finite time. To obtain the optimal solution of problem (2), the relationship between the optimal solution of problem (2) and the equilibrium point of neural network (5) is presented as follows.

Theorem 2. *Any optimal solution of problem (2) is an equilibrium point of neural network (5) and vice versa, if*

$$\sigma > \frac{\|b\|_2}{\min_{A\gamma \in \Delta} \|A\gamma\|_2}, \quad (9)$$

where Δ is the largest compact set in $A\Gamma \setminus \{0\}$.

Proof: The proof is omitted due to limitation of space. Interested readers may refer to [14] for more details. \square

According to Theorems 1 and 2, the state vector of neural network (5) is globally convergent to an optimal solution of problem (2) in finite time if (9) holds.

4 Linear Assignment Application

In this section, the proposed neural network is utilized for solving the linear assignment problems. In the literature, several recurrent neural networks have been developed for linear assignment (e.g., see [4][9]). The linear assignment problem is to find an optimal solution to the following linear integer programming problem:

$$\text{minimize} \quad \sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij}, \quad (10)$$

$$\text{subject to} \quad \sum_{j=1}^n x_{ij} = 1, \quad i = 1, 2, \dots, n, \quad (11)$$

$$\sum_{i=1}^n x_{ij} = 1, \quad j = 1, 2, \dots, n, \quad (12)$$

$$x_{ij} \in \{0, 1\}, \quad i, j = 1, 2, \dots, n. \quad (13)$$

According to [15], if the optimal solution of the primal assignment problem (10)–(13) is unique, then it is equivalent to a linear programming problem by replacing the zero-one integrality constraints (13) with nonnegative constraints as follows:

$$x_{ij} \geq 0, \quad i, j = 1, 2, \dots, n. \quad (14)$$

Based on the primal assignment problem, the dual assignment problem can be formulated as follows:

$$\begin{aligned} & \text{maximize} && \sum_{i=1}^n (u_i + v_i), \\ & \text{subject to} && u_i + v_j \leq c_{ij}, \quad i, j = 1, 2, \dots, n, \end{aligned} \quad (15)$$

where u_i and v_i denote the dual decision variables.

Let $y = (u_1, \dots, u_n, v_1, \dots, v_n)^T$, $b = (1, 1, \dots, 1)^T \in \mathbb{R}^{2n}$, and $c = (c_{11}, \dots, c_{1n}, c_{21}, \dots, c_{2n}, \dots, c_{n1}, \dots, c_{nn})^T$, then the dual assignment problem (15) can be written as (2) with $A = (M \ E)^T$, where M is a block diagonal matrix with $M = \text{diag}\{e, e, \dots, e\}$ and $E = (I, I, \dots, I)^T$, in which $e = (1, 1, \dots, 1)^T \in \mathbb{R}^n$ and I is the n -dimensional identity matrix. Then the proposed neural network in (5) is capable of solving the dual assignment problem.

Furthermore, for the dual assignment problem (15), the neural network in (5) can be written as the following component form: for $i = 1, 2, \dots, n$

$$\begin{cases} \epsilon \frac{du_i}{dt} = -\sigma \sum_{j=1}^n g(u_i + v_j - c_{ij}) + 1, \\ \epsilon \frac{dv_i}{dt} = -\sigma \sum_{j=1}^n g(u_j + v_i - c_{ji}) + 1. \end{cases} \quad (16)$$

The solution from the dual assignment problem can be easily decoded into that for the primal assignment problem by using the complementary slackness theorem as follows [4]:

$$x_{ij} = h(u_i + v_j - c_{ij}), \quad (17)$$

where $h(z)$ is the output function defined as $h(z) = 1$ if $z = 0$, or $h(z) = 0$ otherwise.

The neural network in (16) has one-layer structure with $2n$ neurons only (same as the number of decision variables in the dual assignment problem (15)). Compared with the primal neural network [4] with n^2 neurons and the primal-dual neural network [11] with $n^2 + 2n$ neurons, the proposed neural network herein has lower model complexity with one order fewer neurons. The proposed neural network has the same model complexity as the dual neural network in [4]. Nevertheless, the parameter selections for the dual neural network therein are not straightforward, whereas the proposed dual neural network herein is guaranteed to converge to exact optimal solutions as long as its single gain parameter in the model is larger than a derived lower bound.

According to the results in Section 3, the proposed neural network is capable of solving the dual linear assignment problem as in the following result.

Corollary 1. *The state vector of neural network (16) is globally convergent to an optimal solution of the dual assignment problem in finite time if $\sigma > \sqrt{n}$.*

Proof: As $b = (1, 1, \dots, 1)^T \in \mathbb{R}^{2n}$, $\|b\|_2 = \sqrt{2n}$. From the definition of A for the dual assignment problem (15), its elements can take 0 or 1 only. For any $\gamma \in \Gamma$, $\|A\gamma\|_2$ gets the minimum value if one element of γ is 1 and the others are 0, where Γ is defined in Section 3. By simple computation, for any $\gamma \in \Gamma$, $\|A\gamma\|_2 \geq \sqrt{2}$. Therefore, according to (9), this corollary holds. \square

5 Simulation Results

We consider a linear assignment problem with $n = 10$ and

$$[c_{ij}] = \begin{pmatrix} 1.6 & 6.7 & -3.6 & 2.8 & 5.6 & -4.1 & 10.5 & 4.2 & -2.8 & 10.9 \\ 1.6 & 9.3 & -5.8 & 5.1 & -4.6 & 10.2 & -1.2 & -2.0 & -2.2 & 8.3 \\ 8.0 & 11.8 & -5.2 & -1.7 & -2.4 & 10.4 & 3.9 & 9.1 & -7.8 & 10.6 \\ 1.4 & 2.1 & 1.2 & -8.4 & 3.2 & -7.7 & 4.3 & -5.7 & 3.5 & -1.8 \\ -4.2 & 4.6 & 7.5 & 0.5 & 1.7 & 7.4 & -3.9 & -2.2 & -5.0 & -2.6 \\ -3.6 & -6.2 & -2.4 & -2.1 & -5.0 & -4.9 & -3.6 & -6.0 & -2.1 & -2.7 \\ 5.3 & 1.0 & -3.5 & 5.5 & 3.4 & 8.3 & 9.1 & 0.5 & 4.6 & -2.7 \\ 5.5 & 2.5 & 10.2 & 10.8 & 1.6 & 8.5 & -5.8 & -4.3 & -2.9 & -3.8 \\ 10.9 & -4.6 & -7.8 & -1.2 & 2.5 & -4.0 & 0.8 & 8.9 & -5.1 & -3.6 \\ 7.4 & -6.6 & 3.7 & 3.3 & 7.8 & 5.5 & 6.0 & 8.7 & 5.1 & -1.2 \end{pmatrix}.$$

According to Corollary 1, a lower bound of σ is $\sqrt{10} \approx 3.1623$. Fig. 1 depicts the convergence of the state variables $(u(t), v(t))$ with $\epsilon = 10^{-5}$ and $\sigma = 3.2$. The state variables of neural network (16) are convergent to an optimal solution of the dual assignment problem (15) in finite time from a random initial point. Fig. 2 shows the convergence of the dual objective function of problem (15) (i.e., $\sum_{i=1}^n \sum_{j=1}^n c_{ij} x_{ij}$ in the primal assignment problem) with three different values of σ . Using (17), the optimal solution to the corresponding primal assignment problem can be easily interpreted as follows:

$$[x_{ij}] = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

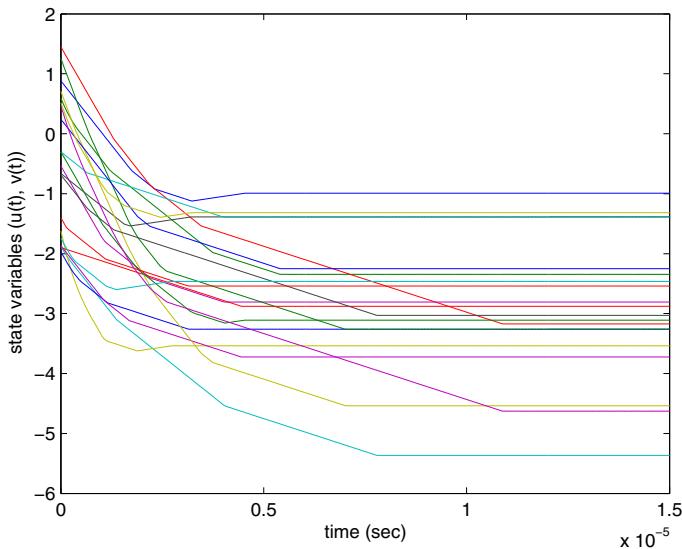


Fig. 1. Transient behaviors of the state variables of neural network (16) in the Example

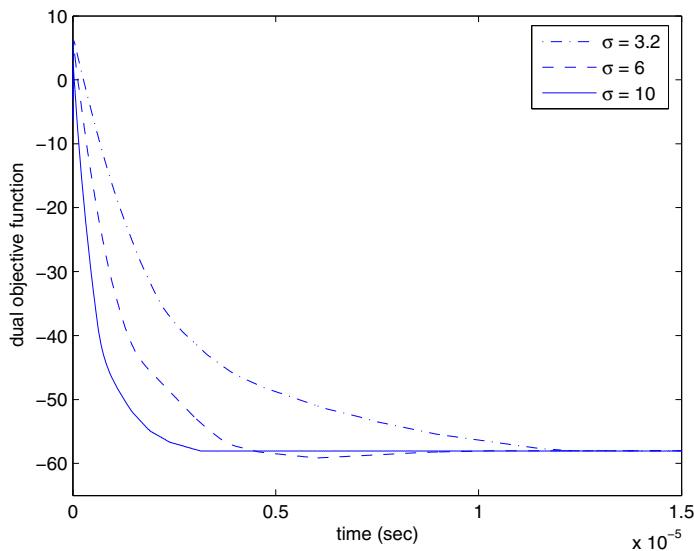


Fig. 2. Global convergence of the dual objective function of neural network (16) in the Example

6 Conclusions

This paper presents a one-layer dual recurrent neural network with a heaviside step activation function for solving linear programming problems. The finite-time global convergence of the proposed neural network to optimal solutions is proven if its single gain parameter σ is larger than a derived lower bound. Furthermore, the proposed neural network is tailored for solving the linear assignment problems. Simulation results are given with a numerical example to illustrate the effectiveness and characteristics of the proposed neural network.

References

1. Tank, D., Hopfield, J.: Simple neural optimization networks: An a/d converter, signal decision circuit, and a linear programming circuit. *IEEE Transactions on Circuits and Systems* 33, 533–541 (1986)
2. Kennedy, M., Chua, L.: Neural networks for nonlinear programming. *IEEE Transactions on Circuits and Systems* 35, 554–562 (1988)
3. Wang, J.: Analysis and design of a recurrent neural network for linear programming. *IEEE Transactions on Circuits and Systems-I* 40, 613–618 (1993)
4. Wang, J.: Primal and dual assignment networks. *IEEE Transactions on Neural Networks* 8, 784–790 (1997)
5. Wang, J.: Primal and dual neural networks for shortest-path routing. *IEEE Transactions on Systems, Man, and Cybernetics-A* 28, 864–869 (1998)
6. Xia, Y.: A new neural network for solving linear programming problems and its application. *IEEE Transactions on Neural Networks* 7, 525–529 (1996)
7. Forti, M., Nistri, P., Quincampoix, M.: Generalized neural network for nonsmooth nonlinear programming problems. *IEEE Transactions on Circuits and Systems-I* 51, 1741–1754 (2004)
8. Liu, Q., Wang, J.: A one-layer recurrent neural network with a discontinuous activation function for linear programming. *Neural Computation* 20, 1366–1383 (2008)
9. Wang, J., Xia, Y.: Analysis and design of primal-dual assignment networks. *IEEE Transactions on Neural Networks* 9, 183–194 (1998)
10. Wang, J.: A recurrent neural network for solving the shortest path problem. *IEEE Transactions on Circuits and Systems-I* 43, 482–486 (1996)
11. Xia, Y., Wang, J.: A discrete-time recurrent neural network for shortest-path routing. *IEEE Transactions on Automatic Control* 45, 2129–2134 (2000)
12. Bazaraa, M., Sherali, H., Shetty, C.: *Nonlinear Programming: Theory and Algorithms*, 3rd edn. John Wiley & Sons, Hoboken (2006)
13. Liu, Q., Wang, J.: A one-layer recurrent neural network for convex programming. In: Proc. IEEE International Joint Conference on Neural Networks. pp. 83–90 (2008)
14. Liu, Q., Wang, J.: Finite-time convergent recurrent neural network with a hard-limiting activation function for constrained optimization with piecewise-linear objective functions. *IEEE Transactions on Neural Networks* 22, 601–613 (2011)
15. Walsh, G.: *An Introduction to Linear Programming*, 2nd edn. John Wiley & Sons, Chichester (1985)

Neural Network Solution of Optimal Control Problem with Control and State Constraints

Tibor Kmet

Department of Computer Science Faculty of Natural Sciences
Constantine the Philosopher University
Tr. A. Hlinku 1, 949 74 Nitra Slovak Republic
tkmet@ukf.sk
<http://www.ukf.sk>

Abstract. A neural network based optimal control synthesis is presented for solving optimal control problems with control and state constraints. The optimal control problem is transcribed into nonlinear programming problem which is implemented with adaptive critic neural network. The proposed simulation methods is illustrated by the optimal control problem of feeding adaptation of filter feeders of Daphnia. Results show that adaptive critic based systematic approach holds promise for obtaining the optimal control with control and state constraints.

Keywords: Optimal control problem with control and state constraints, adaptive critic neural network synthesis.

1 Introduction

It is well known that neural networks are nonlinear models that can approximate smooth time-invariant functions with arbitrary degree of accuracy [4]. There has been a growing number of applications reported in the literature that employ neural networks to optimal control of nonlinear processes which is one of the most active subjects in control theory. There is rarely an analytical solution although several numerical computation approaches have been proposed (for example see [5], [10], [11]) for solving a optimal control problem. Most of the literature that deals with numerical methods for the solution of general optimal control problems focuses on the algorithms for solving discretized problems [1]. In the recent years, the multi-layer feedforward neural networks have been used for obtaining numerical solutions to the optimal control problem [2], [3], [6], [9], [8], [15]. In the present paper we extend adaptive critic neural network architecture proposed by [6] to the optimal control problems with open final time using recurrent neural network. The paper is organized as follows. In Section 2, the optimal control problems with control and state constraints and open final time are introduced. Section 3 presents a algorithm to solve nonlinear optimal control problem with control and state constraints using adaptive critic and recurrent neural networks. In Section 4, we apply the discussed methods to the optimal control problem of feeding adaptation of Daphnia [7]. Conclusions are presented in Section 5.

2 Optimal Control Problem with Control and State Constraints

We consider a nonlinear optimal control problem subject to control and state constraints. Let $x(t) \in R^n$ denote the state of a system and $u(t) \in R^m$ the control in a given time interval $[t_0, t_f]$.

Optimal control problem is to minimize

$$J(u) = g(x(t_f)) + \int_{t_0}^{t_f} f_0(x(t), u(t)) dt \quad (1)$$

subject to

$$\begin{aligned} \dot{x}(t) &= f(x(t), u(t)), \quad x(t_0) = x_0, \quad \psi(x(t_f)) = 0, \quad c(x(t), u(t)) \leq 0, \\ t &\in [t_0, t_f]. \end{aligned}$$

The functions $g : R^n \rightarrow R$, $f_0 : R^{n+m} \rightarrow R$, $f : R^{n+m} \rightarrow R^n$, $c : R^{n+m} \rightarrow R^q$ and $\psi : R^n \rightarrow R^r$, $0 \leq r \leq n$ are assumed to be sufficiently smooth on appropriate open sets. The theory of necessary conditions for optimal control problem of form (1) is well developed (cf. [5], [11]). We introduce an additional state variable $x_0(t) = \int_0^t f_0(x(s), u(s)) ds$ defined by the $\dot{x}_0(t) = f_0(x(t), u(t))$, $x_0(0) = 0$. Direct optimization methods for solving the optimal control problem are based on a suitable discretization of (1). Choose a natural number N and let $t_i \in [t_0, t_f]$, $i = 1, \dots, N-1$, be a equidistant mesh point with $t_i = t_0 + ih$, $i = 1, \dots, N$, where h is time step and $t_f = Nh + t_0$. Let the vectors $x^i \in R^{n+1}$, $u^i \in R^m$, $i = 0, \dots, N$, be approximation of state variable and control variable $x(t_i)$, $u(t_i)$, respectively at the mesh point. *Euler's* approximation applied to the differential equations yields

$$x^{i+1} = x^i + hf(x^i, u^i), \quad i = 0, \dots, N-1. \quad (2)$$

Choosing the optimal variable $z := (x^0, x^1, \dots, x^{N-1}, u^0, \dots, u^{N-1}, h) \in R^{N_s}$, $N_s = (n+1+m)N+1$, the optimal control problem is replaced by the following discretized control problem in the form of nonlinear programming problem with inequality constraints:

$$\text{Minimize } J(z) = G(x^N), \quad \text{where } G(x^N) = g((x_1, \dots, x_n)^N) + x_0^N, \quad (3)$$

subject to $-x^{i+1} + x^i + hf(x^i, u^i) = 0$, $x^0 = x(t_0)$, $\psi(x^N)) = 0$, $c(x^i, u^i) \leq 0$, $i = 0, \dots, N-1$. In a discrete-time formulation we want to find an admissible control which minimize object function (3). Let us introduce the *Lagrangian function* for the nonlinear optimization problem (3):

$$\begin{aligned} L(z, \lambda, \nu, \mu, h) &= \sum_{i=0}^{N-1} \lambda^{i+1} (-x^{i+1} + x^i + hf(x^i, u^i)) + G(x^N) + \\ &+ \sum_{i=0}^{N-1} \mu^i c(x^i, u^i) + \nu \psi(x^N). \end{aligned} \quad (4)$$

and define $H(i)$ and Φ as follows: $H(i) = \lambda^{i+1}(x^i + hf(x^i, u^i)) + \mu^i c(x^i, u^i)$, $\Phi(x^N) = G(x^N) + \nu\psi(x^N)$. The first order optimality conditions of Karush-Kuhn-Tucker [10] for the problem (3) are:

$$0 = L_{x^i}(z, \lambda, \nu, \mu, h) = \lambda^{i+1} + h\lambda^{i+1}f_{x^i}(x^i, u^i) - \lambda^i + \mu^i c_{x^i}(x^i, u^i), \quad (5)$$

$$0 = L_{x^N}(z, \lambda, \nu, \mu, h) = G_{x^N}(x^N) + \nu\psi_{x_N}(x_N) - \lambda^N, \quad (6)$$

$$0 = L_{u^i}(z, \lambda, \nu, \mu, h) = h\lambda^{i+1}f_{u^i}(x^i, u^i) + \mu^i c_{u^i}(x^i, u^i), \quad i = 0, \dots, N-1, \quad (7)$$

$$0 = L_h(z, \lambda, \nu, \mu, h) = \Phi_h + \sum_{i=0}^{N-1} H_h(i), \quad (8)$$

ν must be found so that $\psi(x^N) = 0$. Furthermore, the complementary conditions hold i.e. $\mu^i \geq 0$, $c(x^i, u^i) \leq 0$ and $\mu^i c(x^i, u^i) = 0$, for $i = 0, \dots, N-1$. Herein, the subscript x , u or h denotes the partial derivative with respect to x , u and h , respectively. Eq. (5 – 8) represents the discrete version of necessary condition for optimal control problem (1).

3 Neural Network Solution of Optimal Control Problem

Neural networks are nonlinear models that can approximate smooth time-invariant functions with arbitrary degree of accuracy [4]. The standard network structure for an approximation function is the multiple-layer perceptron (or feed forward network). The multi-layered feed forward network shown in Fig. (1) is training using the steepest descent error backpropagation rule [12]. The adaptive critic neural network shown in Fig. (1) consists of two feed forward neural network: an *action* network the inputs of which are the current states x and outputs are the corresponding control \hat{u} and multiplier $\hat{\mu}$, and the *critic* network for which the current states x are inputs and current costates $\hat{\lambda}$ are outputs. The steps for simultaneous training the action and critic network are as follows:

- | | |
|--|--|
| <p>1) Action network setup.</p> <p>(1.i) Input x^i to the action network to obtain $u^{i,a}$ and $\mu^{i,a}$.</p> <p>(1.ii) Using x^i and $u^{i,a}$ solve state equation (2) to get x^{i+1}.</p> <p>(1.iii) Input x^{i+1} to the critic network to obtain λ^{i+1}.</p> <p>(1.iv) Using x^i and λ^{i+1} solve (7) to calculate $u^{i,t}$ and $\mu^{i,t}$.</p> <p>If $(u^{i,a}, \mu^{i,a}) - (u^{i,t}, \mu^{i,t}) / (u^{i,t}, \mu^{i,t}) < \epsilon_{acnn}$, the convergence criterion for the action network training is met.</p> <p>Else GO TO 1.</p> <p>I/O Input x^i, output $\hat{u}^i = u^{i,t}$ and $\hat{\mu}^i = \mu^{i,t}$.</p> | <p>1) Critic network setup.</p> <p>(1.i) Input x^i to the critic network to obtain $\lambda^{i,c}$.</p> <p>(1.ii) Using x^i and $u^{i,a}$ solve state equation (2) to get x^{i+1}.</p> <p>(1.iii) Input x^{i+1} to the critic network to obtain λ^{i+1}.</p> <p>(1.iv) Using x^i, $u^{i,a}$, $\mu^{i,a}$ and λ^{i+1} solve (5) to calculate $\lambda^{i,t}$.</p> <p>If $\lambda^{i,c} - \lambda^{i,t} / \lambda^{i,t} < \epsilon_{acnn}$, the convergence criterion for the critic network training is met.</p> <p>Else GO TO 1.</p> <p>I/O Input x^i, output $\hat{\lambda}^i = \lambda^{i,t}$.</p> |
|--|--|

Further discussion and detail explanation of this adaptive critic methods for optimal control problem with control and state constraints, fixed terminal time and free terminal condition can be found in [6], [9]. To solve equations (7-8) we

are concerned the following nonlinear projection equation (for detail description see [16]):

$$\alpha F(P_X(u)) + u - P_X(u) = 0, \quad (9)$$

where $\alpha > 0$ is a constant $F : R^l \rightarrow R^l$, $X = \{u \in R^l \mid d_i \leq u_i \leq h_i\}$ and $P_X : R^l \rightarrow X$ is a projection operator defined by $P_X(u) = (P_X(u_1), \dots, P_X(u_l))$

$$P_X(u_i) = \begin{cases} d_i & : u_i < d_i \\ u_i & : d_i \leq u_i \leq h_i \\ h_i & : u_i > h_i \end{cases}$$

which can be solved by the following dynamic model

$$\dot{u}(t) = -\beta(\alpha F(P_X(u)) + u - P_X(u)). \quad (10)$$

A system described by (10) can be realized by a recurrent neural network with a single-layer structure [16]. A asymptotic and exponencial stability of the present neural network in (10) are proven by [16]. Equilibrium points of (10) coincide with solutions of (9). We can state the algorithm to solve optimal control problem using adaptive critic and recurrent neural network.

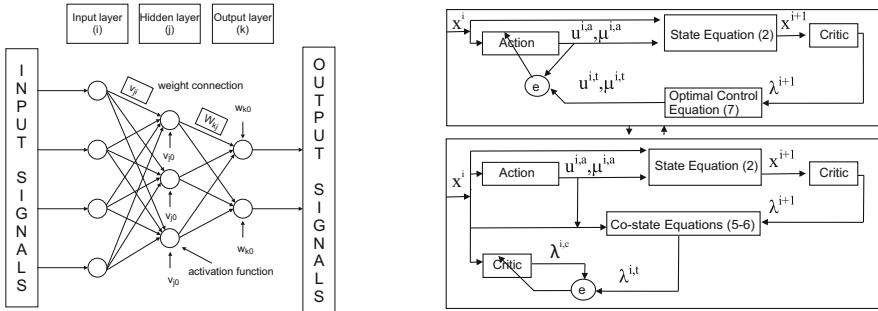


Fig. 1. Feed forward neural network topology with one hidden layer, v_{js} , w_{kj} are values of connection weights, v_{j0} , w_{k0} are values of bias. Architecture of adaptive critic feed forward network synthesis, x^i -input signal to the action and critic network, $u^{i,t}, \mu^{i,t}$ and $\lambda^{i,t}$ output signal from action and critic network, respectively.

Algorithm

- Choose N - number of steps, ϵ_{acnn} and ϵ_{rnn} - stopping tolerance for adaptive critic and recurrent neural network, respectively, ϵ_h - stopping tolerance of algorithm, x^0 -initial value, $\alpha > 0$ and $\beta > 0$.
- Guess time step h .
- (*) For $i = 0, \dots, N - 2$

Compute optimal $\hat{\lambda}^i$, \hat{u}^i , and $\hat{\mu}^i$ using adaptive critic neural network and recurrent neural network with $X = \{(u^i, \mu^i) \in R^{m+q} \mid \mu^i \geq 0\}$,

$F(u^i, \mu^i) = (L_{u^i}(z, \lambda, \nu, \mu, h), -c(x^i, u^i))$ and stopping tolerance ϵ_{acnn} and ϵ_{rnn} .

- For $i = N - 1$

Compute optimal $\hat{u}^{N-1}, \hat{\mu}^{N-1}, \hat{\nu}, \hat{h}$ using (2), (5), (6), (8), recurrent neural network with $X = \{(u^{N-1}, \mu^{N-1}, \nu, h) \in R^{m+q+r+1} | \mu^{N-1} \geq 0, h \geq 0\}$,

$$F(u^{N-1}, \mu^{N-1}, \nu, h) =$$

$(L_{u^{N-1}}(z, \lambda, \nu, \mu, h), -c(x^{N-1}, u^{N-1}), -\psi(x^N), L_h(z, \lambda, \nu, \mu, h))$ and stopping tolerance ϵ_{rnn} .

- If $|h - \hat{h}| < \epsilon_h$ then end.
- Else $h = \hat{h}$ Go to (*).

4 Model of Feeding Adaptation and Neural Network Simulation

The model consists of phosphorus (x_1) as a limiting nutrient for growth of four species of algae of different size ($x_2 - x_5$) and zooplankton (x_6). Similar models of n species of microorganisms competing exploitative for a one, two or more growth-limiting nutrients are used to study continuous culture of microorganisms in chemostat under constant condition [14] without of any predators. Functions occurring in the model are given in Table 1 in ecological and mathematical notation, respectively. The model is described by the following system of ordinary differential equation:

$$\begin{aligned} \dot{x}_1 &= a_7(a_8 - x_1) - \sum_{i=2}^5 \left(\frac{d_1 x_i p_i x_1}{x_1 + s_i} \right. \\ &\quad \left. + r_i f_2 x_i + x_i x_6 C_i \left(1 - \frac{d_4}{a_4 + x_i} \right) \right) \\ \dot{x}_i &= \frac{d_1 x_i p_i x_1}{x_1 + s_i} - r_i f_2 x_i - x_i x_6 E_i - d_2 x_i + a_{i+9} a_7 \\ &\quad \text{for } i = 2 \dots 5 \\ \dot{x}_6 &= x_6 \left(d_3 \sum_{i=2}^5 \frac{C_i x_i}{a_4 + x_i} - a_5 \right) + a_6 \end{aligned} \quad (11)$$

For detail explanation see [7]. It is derived from the models of the series AQUAMOD [13] modified by the inclusion of several “species” of algae. Four species of algae were considered during the computations performed: x_2, \dots, x_5 . Each “species” is represented by a particular algal cell (or colony) volume. The volumes were set arbitrarily to ($V_i = 50, 500, 2500$ and $5000 \mu\text{m}^3$), to approximate the set of “edible” algal sizes commonly occurring in our reservoirs. Table 2 gives the corresponding values used in the present simulations. The description of selectivity E_i is as follows: $E_i(u) = \exp(-0.1 (u - u_i)^2)$, where u is the value of setal density directly related to the algal diameter for which selectivity is maximal and u_i is the diameter corresponding to each algal cell volume V_i .

We are interested in the ability of Cladocera to adapt both the filtration area and filter density to the amount and size structure of the food particles (algae) population. We assume that filtration in aquatic filter feeders is an optimal

Table 1. Values of parameters and size-specific parameters of algae

a_1	0.05	sedimentation rate [day $^{-1}$]	V_i	algal cell volume [μm^3]
a_2	0.6	maximum efficiency of zooplankton assimilation	$u_i = 2 * \sqrt[1/3]{\frac{3*V_i}{4*\pi}}$	diameter corr. to V_i
a_3	0.05	recalculation from units of algae to units of zooplankton	$E_i(u) = \exp(-0.1 * (u - u_i)^2)$	selectivity
a_4	60	half saturation constant for zoopl. feeding [mg.m $^{-3}$ CHA]	$(C_i) Frz(V_i) = a_9 * E_i(u)$ $(p_i) P_{max}(V_i) = 0.5 - 0.05 LOG(V_i)$	forcing function
a_5	0.03	zooplankton mortality [d $^{-1}$]	$(r_i) Resp(V_i) = 0.02 + 0.002 LOG(V_i)$	spec. growth rate [d $^{-1}$]
a_6	0.0002	inflow of zooplankton [m $^{-3}$ C.day $^{-1}$]	$(s_i) KS(V_i) = -5 + 10 LOG(V_i)$	spec. resp. rate of algae [d $^{-1}$]
a_7	0.1	hydraulic loading [d $^{-1}$]		half sat. constant for P [mg.m $^{-3}$ P]
a_8	200	inflow phosphorus		sedimentation func.
a_9	0.9	concentration [mg.m $^{-3}$ P]	$(f_1) Faz = 0.8 + 0.25\cos(t) + 0.12\cos(2t)$	water temperature [$^{\circ}\text{C}$]
a_{10}	120	zooplankton filtration rate [m $^{-3}$ C.day $^{-1}$]	$(f_2) Temp = 12 + 10\sin(t + 220)$ $(f_3) I_0 = 280 + 210\sin(t + 240)$ $f(Temp) = e^{(0.09*Temp)}$	light intensity
$a_{11} - a_{14}$	0.	half saturation constant for light [cal.cm $^{-2}$.day $^{-1}$]	$g(I_0) = \frac{I_0}{T_0 + I_0 KM}$	
		inflow of phytoplankton concentration [mg.m $^{-3}$ CHA]	$d_1 = f * g$ $d_2 = a_1 * f_1$ $d_3 = a_3 * d_4$	

process of maximal feeding strategy. We will investigate two strategies [6], [7]: 1)instantaneous maximal biomass production as a goal function (local optimality), i.e., $\dot{x}_6 = f_6(x, u) \rightarrow \max$ for all t , under the constraint $u \in [u_{min}, u_{max}]$.

2) integral maximal biomass (global optimality), i.e. $J(u) = \int_0^T x_6(t) dt$, under the constraint $u \in [u_{min}, u_{max}]$. In the case of strategy 1, we maximize the following function $J(u) = \sum_{i=2}^5 \frac{E_i(u) d_3 x_i a_9}{(x_i + a_4)}$ under the constraints $u \in [u_{min}, u_{max}]$.

In case of strategy 2, we have the following optimal control problem: to find a function $\hat{u}(t)$, for which the goal function $J(\hat{u}) = \int_0^T x_6(t) dt$ attains its maximum

(i.e. $-\int_0^T x_6(t) dt \rightarrow \min$) under the constraints $c_1(x, u) = u_{min} - u \leq 0$, $c_2(x, u) = u - u_{max} \leq 0$, where T denotes the fixed lifetime of an individual Daphnia and $\psi(x) = 0$. In the adaptive critic synthesis, the critic and action network were selected such that they consist of six and two subnetworks, respectively, each having 6-18-1 structure (i.e. six neurons in the input layer, eighteen neurons in the hidden layer and one neuron in the output layer). The proposed *adaptive critic neural network* and recurrent neural network are able to meet the convergence tolerance values that we choose, which led to satisfactory simulation results. Simulations, using MATLAB show that proposed neural network is able to solve more general nonlinear optimal control problem. Our results are quite similar to those obtained in [7].

The results of numerical solutions (Fig. 2 l,r) have shown that the optimal strategies $\tilde{u}(t)$ and $\hat{u}(t)$ based on short or long-term perspective, respectively, have different time trajectory for different fixed values of *Faz* – sedimentation function, *Temp* – watertemperature, and I_0 – light intensity, (for $t=120$, 210, respectively in Table 1). The numerical results have shown, that for the initial conditions considered $J(\hat{u}(t)) > J(\tilde{u}(t))$, (see Table 2). The higher biomass of zooplankton obtained in the case of integral formulation points towards the assumption that the organisms do better if not reacting only to the immediate

Table 2. Parameters for four "species" of algae and results of goal function evaluations for local and global optimality

V_i	u_i	p_i	s_i	r_i	Value of goal function	$t=120$	$t=210$
					<i>local</i> $J(\tilde{u})$	18.4 (Fig. 2 l)	105.1 (Fig. 2 r)
50	500	2500	5000		<i>global</i> $J(\hat{u})$	27.9 (Fig. 2 l)	178.4 (Fig. 2 r)
4.57	9.85	16.84	21.22				
0.4151	0.3651	0.3301	0.3151				
11.99	21.99	28.98	31.99				
0.023	0.025	0.027	0.028				

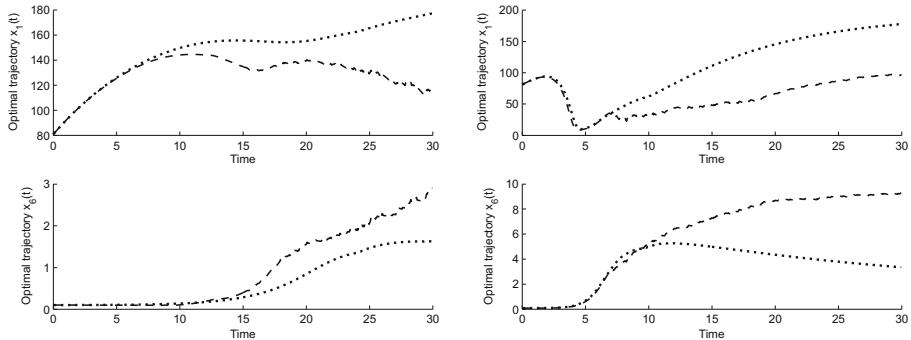


Fig. 2. Simulation results - local optimality (dotted line), global optimality (dashed line) for initial condition $x_1^0 = 80.4$, $x_2^0 = 0.4$, $x_3^0 = 0.3$, $x_4^0 = 0.2$, $x_5^0 = 0.1$, $x_6^0 = 0.1$ and constant environmental condition ($t = 120$ and $t = 210$ in Table 1)

changes, but having developed mechanisms consistent with more long-term consideration.

5 Conclusion

In this paper, the optimal control problem with control and state constraints and with open final time has been investigated. Using adaptive critic approach and recurrent neural network we proposed a methods to solve general optimal control problem with control and state constraints and fixed or open final time. The extended adaptive critic neural network and recurrent neural network are a good solvers for a wide class of optimal control problems. Using MATLAB, a simple simulation model based on adaptive critic and recurrent neural network approach was constructed. On the application side, the proposed neural network is applied to explain feeding adaptation of filter feeders of Daphnia. Simulations shown that the proposed algorithm is effective in solving general optimal control problems.

Acknowledgments. The author is grateful to the referee for his valuable suggestions. The paper was worked out as a part of the solution of the scientific project number KEGA 004UJS-4/2011.

References

1. Buskens, C., Maurer, H.: SQP-Methods for Solving Optimal Control Problems with Control and State Constraints: Adjoint Variable, Sensitivity Analysis and Real-Time Control. *J. Comp. Appl. Math.* 120, 85–108 (2000)
2. Dierks, T., Thumati, B., Jagannathan, S.: Optimal Control of Unknown Affine Nonlinear Discrete-Time Systems Using Offline-Trained Neural Networks with Proof of Convergence. *Neural Netw.* 22, 851–860 (2009)
3. Dierks, T., Jagannathan, S.: Optimal Control of Affine Nonlinear Continuous-Time Systems Using an Online Hamilton-Jacobi-Isaacs Formulation. In: *Proceedings of the IEEE Conference on Decision and Control*, pp. 3048–3053. IEEE Press, New York (2010)
4. Hornik, M., Stichcombe, M., White, H.: Multilayer Feed Forward Networks are Universal Approximators. *Neural Netw.* 3, 256–366 (1989)
5. Kirk, D.E.: *Optimal Control Theory: An Introduction*. Dover Publications, New York (1989)
6. Kmet, T.: Neural Network Simulation of Nitrogen Transformation Cycle. In: *23th European Conference on Modelling and Simulation*, pp. 352–358. ECMS, Madrid (2009)
7. Kmet, T., Straskraba, M.: Feeding Adaptation of Filter Feeders: Daphnia. *Ecol. Modell.* 178, 313–327 (2004)
8. Padhi, R., Unnikrishnan, N., Wang, X., Balakrishnan, S.N.: Adaptive-Critic Based Optimal Control Synthesis for Distributed Parameter Systems. *Automatica* 37, 1223–1234 (2001)
9. Padhi, R., Balakrishnan, S.N., Randolph, T.: A Single Network Adaptive Critic (SNAC) Architecture for Optimal Control Synthesis for a Class of Nonlinear Systems. *Neural Netw.* 19, 1648–1660 (2006)
10. Polak, E.: *Optimization Algorithms and Consistent Approximation*. Springer, Heidelberg (1997)
11. Pontryagin, L.S., Boltyanskii, V.G., Gamkrelidze, R.V., Mischenko, E.F.: *The Mathematical Theory of Optimal Process*. Nauka, Moscow (1983)
12. Rumelhart, D.F., Hinton, G.E., Williams, R.J.: Learning Internal Representation by Error Propagation. In: *Parallel Distributed Processing: Foundation*, pp. 318–362. The MIT Press, Cambridge (1987)
13. Straškraba, M., Gnauck, P.: *Freshwater Ecosystems, Modelling and Simulation, Developments in Environmental Modelling*. Elsevier, Amsterdam (1985)
14. Smith, H.L., Waltman, P.: *The Theory of the Chemostat*. Cambridge Univ. Press, Cambridge (1995)
15. Werbos, P.J.: Approximate Dynamic Programming for Real-Time Control and Neural Modeling. In: *Handbook of Intelligent Control*, pp. 493–525. Multiscience Press, New York (1992)
16. Xia, Y., Feng, G.: A New Neural Network for Solving Nonlinear Projection Equations. *Neural Netw.* 20, 577–589 (2007)

Singular Perturbation Approach with Matsuoka Oscillator and Synchronization Phenomena

Yasuomi D. Sato^{1,2}, Kazuki Nakada¹ and Kiyotoshi Matsuoka¹

¹ Department of Brain Science and Engineering, Graduate School of Life Science and Systems Engineering, Kyushu Institute of Technology,
2-4, Hibikino, Wakamatsu-ku, Kitakyushu, 808-0196, Japan
`{sato-y, nakada, matsuoka}@brain.kyutech.ac.jp`

² Frankfurt Institute for Advanced Studies, Johann Wolfgang Goethe University,
Ruth-Moufang-Str. 1, 60438 Frankfurt am Main, Germany
`sato@fias.uni-frankfurt.de`

Abstract. We study the singular perturbation approach in a pair of Matsuoka nonlinear neural oscillators, which consist of membrane potential (v) and recovery (u) dynamics with a relaxation rate (P). This shows that the u -coupled system of the Matsuoka oscillators would be valid for the modeling of neural firings. The coupled integrate-and-fire model of the improved type with an impulse-like interval results from the u -coupled system, under taking the limit of $P \rightarrow \infty$, without loss of any coupling properties. We simulate systematically synchronization of both the v -coupled and u -coupled systems. We also discuss potential capabilities of the u -coupled system of Matsuoka oscillators.

Keywords: Matsuoka Oscillators, Singular Perturbation Approach, Integrate-and-fire Oscillators.

1 Introduction

It has been addressed that the inhibitory interconnected Matsuoka neural oscillator is a representative of a number of tractable theoretical models for neuronal firings[1,2]. It has frequently been regarded as a good design for central pattern generator (CPG) based controllers for applications in walking robot controls[3]. This Matsuoka oscillator generally consists of two variables for membrane potential (v) and recovery (u) dynamics, which is a very simplified linear model and has logical nonlinearity only in its output. The Matsuoka oscillator is less complex and thus easier to theoretically analyze. This is because the other tractable theoretical models such as the FitzHugh-Nagumo (FHN) model[4] have an additional cubic nonlinearity in the v dynamics, making it more complicated than the Matsuoka oscillator.

The FHN model also consists of the two aforementioned dynamics for v and u and displays a periodic and spontaneous firing phenomenon by controlling its intrinsic parameters. Furthermore, in the FHN model, there seem to be analytically interesting firing dynamics properties. It is shown that the FHN

model is reduced to an integrate-and-fire (IF) model under certain conditions by piecewise-linearizing the FHN model and then using the singular perturbation approach[5,6]. The singular perturbation approach is a mathematical tool to control dynamic time scale for variables v and u . Making the use of the singular perturbation approach, Sato and Shiino[7] analytically showed that the u -coupled FHN model is generalized to the coupled IF one without loss in coupling properties. Meanwhile the v -coupled FHN model can be reduced to the coupled IF model with transformed couplings. This raises a question of whether or not the v -coupled system is valid even if such a coupled system has been considered as natural and typical so far.

In this paper, we study dynamical characteristics found in a coupled pair of the Matsuoka oscillators. We find that the coupled IF model consistently results from the u -coupled system of the Matsuoka oscillator, using the singular perturbation approach. In order to discuss the validity of the u -coupled system of the Matsuoka oscillator, we simulate and analyze synchronous behavior observed in u - as well as v -coupled systems.

2 Coupled Systems of Matsuoka Oscillators

We begin by writing a paired system of the Matsuoka oscillators coupled externally by the counterpart (see Fig. 1(a)) in the general form:

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{F}_i(\mathbf{x}_i) + \epsilon \mathbf{G}_i(\mathbf{x}_{\bar{i}}), \quad (1)$$

where \bar{i} represents a counterpart of the i th neuron. $\mathbf{x}_i = (v_{i1}, u_{i1}, v_{i2}, u_{i2})^T \in \mathbb{R}^4$ and T denotes the transpose. v is a fast variable while u is a slow variable. ϵ represents the strength of the mutual interaction between Matsuoka oscillators. $\mathbf{G}_i(\mathbf{x}_{\bar{i}})$ represents a coupling vector. $\mathbf{F}_i(\mathbf{x}_i)$ is a baseline vector field:

$$\mathbf{F}(\mathbf{x}) = \begin{pmatrix} -v_1 + S - bu_1 - ag(v_2) \\ (-u_1 + g(v_1))/P \\ -v_2 + S - bu_2 - ag(v_1) \\ (-u_2 + g(v_2))/P \end{pmatrix}, \quad (2)$$

$$g(v) \equiv \max(0, v - \theta). \quad (3)$$

We assume that $a = b = 2.5$, $S = 2.0$, $P = 12.5$ and $\theta = 0.01$ so that the Matsuoka model represents an oscillatory system exhibiting spontaneous periodic firing as shown in Figs. 1(b) and 1(c). For the coupling vector $\mathbf{G}_i(\mathbf{x}_{\bar{i}})$, we select two kinds of vectors $(-ag(v_{\bar{i}1}), 0, -ag(v_{\bar{i}2}), 0)^T$ and $(0, ag(v_{\bar{i}1})/P, 0, ag(v_{\bar{i}2})/P)^T$.

3 Singular Perturbation Approach

In this section, we study a singular perturbation approach, in order to show how the Matsuoka oscillator is transformed with two different external interactions of $\mathbf{G}_i(\mathbf{x}_i)$.

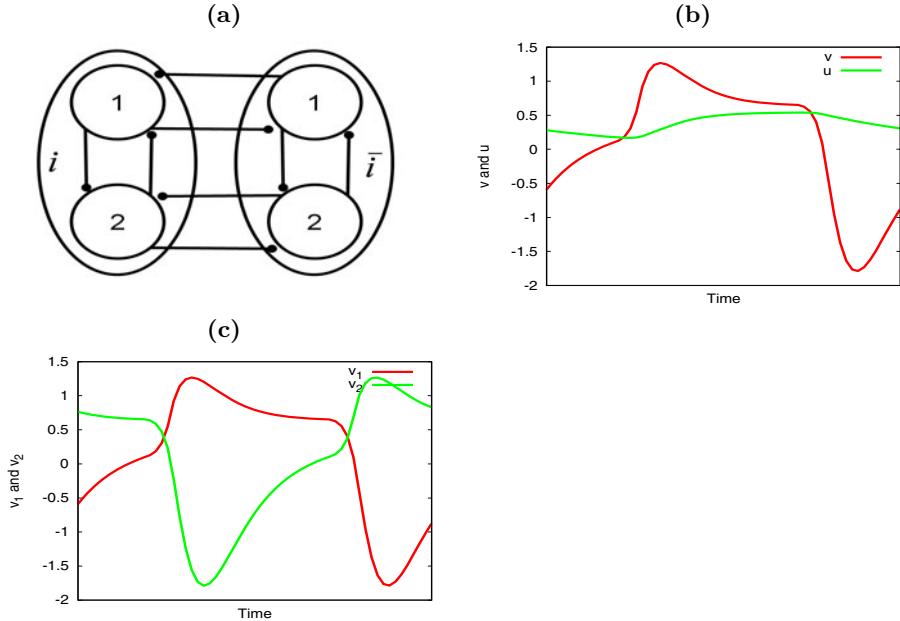


Fig. 1. A network structure of a paired system of the Matsuoka oscillators, i and \bar{i} . (a) One Matsuoka oscillator consists of two mutually inhibiting subunits. Each subunit is also inhibitory coupled by external subunit of the other oscillator. Mutual interactions between external subunits are connected with v - and u -couplings. (b) One subunit in the Matsuoka oscillator consists of v -dynamics and u -dynamics. Each trajectory is drawn by red or light green lines. (c) The v -dynamics in one subunit is mutually inhibiting with the one in the other.

We consider the case of $(0, ag(v_{\bar{i}1})/P, 0, ag(v_{\bar{i}2})/P)^T$ at first. To switch from fast to slow dynamics, we redefine the time scale to $\tau = [1/P]t$ to rewrite the Matsuoka oscillator:

$$\frac{1}{P} \frac{dv_{ij}}{d\tau} = -v_{ij} + S - bu_{ij} - ag(v_{i\bar{j}}), \quad (4)$$

$$\frac{du_{ij}}{d\tau} = -u_{ij} + g(v_{ij}) \underbrace{+ ag(v_{\bar{i}j})}_{\text{coupling term}}. \quad (5)$$

Here \bar{j} represents a counterpart of the j th subunit in the i th neuron. As P becomes larger, the motions in the active and silent phases become slower while the switching motions between them become faster (Fig. 2(a)). Then, taking the limit, $P \rightarrow \infty$, we obtain a simplified version of the Matsuoka oscillator:

$$0 = -v_{ij} + S - bu_{ij} - ag(v_{i\bar{j}}), \quad (6)$$

$$\frac{du_{ij}}{d\tau} = -u_{ij} + g(v_{ij}) \underbrace{+ ag(v_{\bar{i}j})}_{\text{coupling term}}. \quad (7)$$

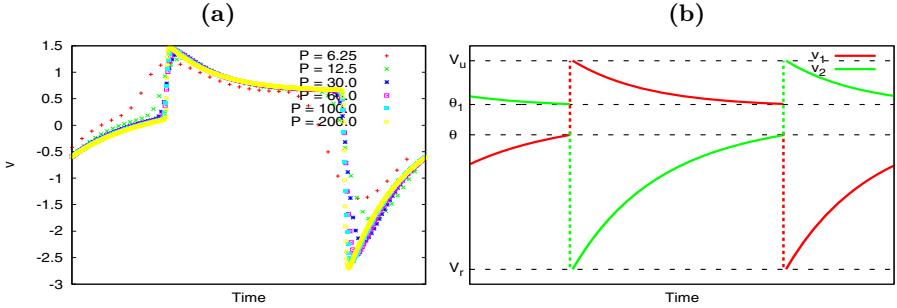


Fig. 2. Singular perturbation approach of the Matsuoka oscillator. (a) Time evolutions of the v -dynamics are changed when P becomes bigger. The temporal duration between points produces a velocity of v . The v motion on the active or inactive phases becomes gradually slower. (b) $P \rightarrow \infty$; v -dynamics for subunits 1 and 2 are represented, respectively, as the red and light green lines. Here $a = b = 2.5$, $S = 2.0$ and $\theta = 0.01$.

Here the transition between the silent and active phases becomes instantaneous because $[du/dt] = 0$, since the time scale is not changed with $P \rightarrow \infty$. The transition motions are thus ignored. Meanwhile the motions are only focused on the active and silent phases. The firing dynamic occurs when v arrives at θ and instantaneously jumps up to V_u , simultaneously and alternately, its counterpart dynamic exponentially decays to θ_1 , after which it is abruptly reset to V_r (Fig. 2(b)).

We continue by differentiating Eq. (6) with respect to τ . Thus, we obtain

$$\frac{du_{ij}}{d\tau} = -\frac{1}{b} \frac{dv_{ij}}{d\tau} - \frac{a}{b} \frac{\partial g(v_{ij})}{\partial v_{i\bar{j}}} \frac{dv_{i\bar{j}}}{d\tau}. \quad (8)$$

Eqs. (6) and (7) becomes a one-dimensional differential model for the v variable with coupling, which is given by

$$\frac{dv_{ij}}{d\tau} + a \frac{\partial g(v_{i\bar{j}})}{\partial v_{i\bar{j}}} \frac{dv_{i\bar{j}}}{d\tau} = -v_{ij} + S - a g(v_{i\bar{j}}) \underbrace{-bag(v_{ij})}_{couplingterm}. \quad (9)$$

To be more precise,

$$\frac{dv_{ij}}{d\tau} = -\gamma v_{ij} + S - a_1 g(v_{i\bar{j}}), \quad (v_{ij} > \theta), \quad (10)$$

$$\frac{dv_{i\bar{j}}}{d\tau} = -v_{i\bar{j}} + a_1 v_{ij} + (1 - a)S - a_1 g(v_{i\bar{j}}), \quad (v_{i\bar{j}} < \theta), \quad (11)$$

where $\gamma = b + 1$ and $a_1 = ab$. Eq. (10) demonstrates v -dynamics on the active phase, or during the firing interval. Eq. (11) shows the dynamics on the inactive phase. When $\gamma \rightarrow \infty$ in one of two subunits, the firing interval vanishes. As a result, we find that a simplified version of the Matsuoka oscillator becomes

approximately equal to the coupled IF oscillators, driven by the other subunit in the Matsuoka oscillator.

Next, the case of $(-ag(v_{\bar{i}1}), 0, -ag(v_{\bar{i}2}), 0)^T$ is treated with a similar approach as mentioned above. The simplified model of the Matsuoka oscillator is obtained as follows:

$$0 = -v_{ij} + S - bu_{ij} - ag(v_{\bar{i}j}) \underbrace{-ag(v_{\bar{i}j})}_{couplingterm}, \quad (12)$$

$$\frac{du_{ij}}{d\tau} = -u_{ij} + g(v_{ij}). \quad (13)$$

Then, by using Eq. (12) and its conversion,

$$\frac{du_{ij}}{d\tau} = -\frac{1}{b} \frac{dv_{ij}}{d\tau} - \frac{a}{b} \left(\frac{\partial g(v_{\bar{i}j})}{\partial v_{\bar{i}j}} \frac{dv_{\bar{i}j}}{d\tau} + \underbrace{\frac{\partial g(v_{\bar{i}j})}{\partial v_{ij}} \frac{dv_{ij}}{d\tau}}_{couplingterm} \right). \quad (14)$$

Eq. (13) is thus rewritten as

$$\begin{aligned} \frac{dv_{ij}}{d\tau} + a \frac{\partial g(v_{\bar{i}j})}{\partial v_{\bar{i}j}} \frac{dv_{\bar{i}j}}{d\tau} &= -v_{ij} + S - ag(v_{\bar{i}j}) \\ &\quad - ag(v_{\bar{i}j}) - a \underbrace{\frac{\partial g(v_{\bar{i}j})}{\partial v_{ij}} \frac{dv_{ij}}{d\tau}}_{couplingterm}. \end{aligned} \quad (15)$$

In conclusion, we have revealed a qualitatively consistent relation between the Matsuoka and the IF oscillators without loss of any coupling properties when the couplings are inserted into the u -dynamics of the Matsuoka oscillator.

Eqs. (9) and (15) lead to a deeper understanding of the dynamical mechanism of the coupled system behaviors of the Matsuoka oscillators. This is largely due to the fact that such behaviors undoubtedly rely on which dynamics in the Matsuoka oscillators are influenced by the external input. In fact, such a difference between two types of the interactions is also observed in investigating synchronous behavior. This will be explained below.

4 Simulation Results

In the previous section of the singular perturbation approach, we found that the u -coupled system of the Matsuoka oscillators is reduced to a unique version of the IF oscillator driven by exponential decays of the other subunit. This indicates that the v -coupled as well as u -coupled systems are theoretically appropriate network models.

In order to show that the u -coupled system is also a theoretically tractable network model, we numerically simulated synchronous behaviors found in coupled systems with mutual interactions between v - or u -dynamics in subunits

(as shown in Fig. 1). Taking into careful account simulation results already reported, we studied coupling strength (ϵ) dependence of the synchronous behaviors ($|\epsilon| \geq 0.1$). Correspondingly, in comparison to simulation results of the v -coupled system, advantages of the u -coupled system were investigated. In these numerical simulations, we set up with $a = 1.5$, $b = 2.5$, $P = 12.5$, $S = 2.0$ and $\theta = 0.01$.

We systematically showed ϵ -dependent synchronous behavior in the u -coupled system. We found very sophisticated synchronization transition when $\epsilon \leq -0.1$. Subunit 1 of one Matsuoka oscillator completely and simultaneously fires to subunit 2 of the other when $\epsilon = -0.1$ (Fig. 3(a)). However when ϵ gradually decreases from it, the simultaneous firings of the subunits 1 and 2 turn into a synchronized state with an arbitrary phase difference (Fig. 3(b)). Further decreases in ϵ induces another synchronous state displaying simultaneous firings of two subunits of i . When $\epsilon = -0.5$, complete synchronization is observed between two subunits of i (Fig. 3(c)). We could not observe any oscillation for $\epsilon \geq -0.54$.

Furthermore, for $0.1 \leq \epsilon \leq 0.43$, complete synchronization are simulated between subunits 1 (or 2) of the Matsuoka oscillator (as shown in Fig. 3(a)). For $0.44 \leq \epsilon \leq 0.53$, they are found between subunit 1 of the one oscillator and

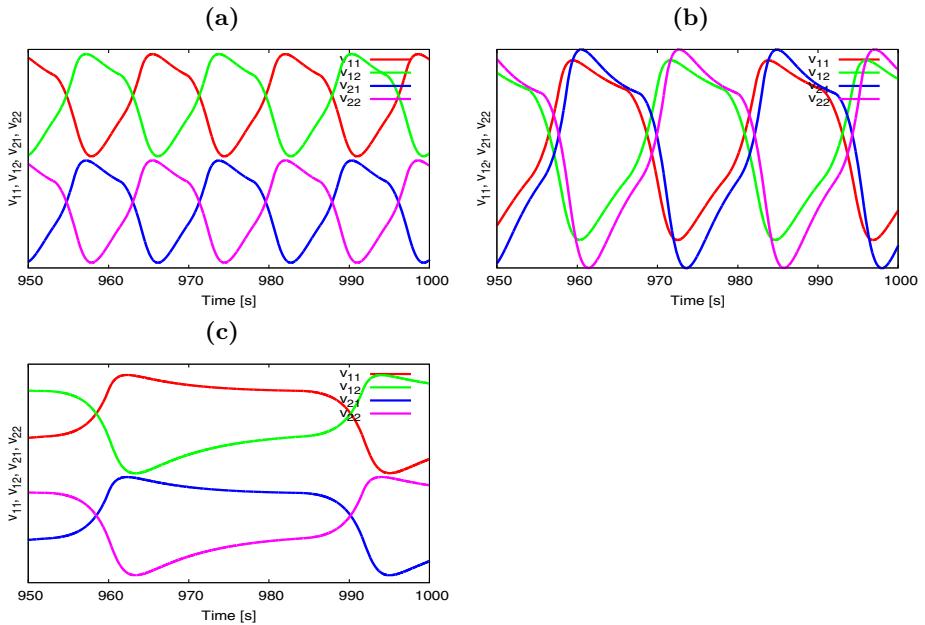


Fig. 3. Synchronous behaviors in a u -coupled pair of the Matsuoka oscillators with $a = 1.5$, $b = 2.5$, $P = 12.5$, $S = 2.0$ and $\theta = 0.01$. (a) Complete synchrony in subunits of v_{11} and v_{21} (or v_{12} and v_{22}) for $\epsilon = -0.1$. (b) Subunits of v_{11} and v_{21} , or of v_{12} and v_{22} are respectively synchronized in the quasi in phase for $\epsilon = -0.25$. (c) Subunits of v_{11} and v_{22} , and of v_{12} and v_{21} respectively fire simultaneously with a certain phase difference for $\epsilon = -0.5$.

subunit 2 of the other (as shown in Fig. 3(c)). However when $0.54 \geq \epsilon$, one of two subunits in the oscillator can no longer reach the threshold θ to generate oscillation.

We studied synchronization phenomena in the v -coupled system. The previous work [1] already showed that subunit 1 in one oscillator and subunit 2 in another are in a complete synchronous state when the coupling is strong ($\epsilon = 1.0$). However, regarding the numerical simulations in this work, such synchronous behavior is calculated over a broad range of ϵ , [0.1, 1.33], including $\epsilon = 1.0$. Synchronization transition as mentioned above cannot be observed. In a coupling strength range of $[-0.1, -1.2]$, only complete synchronization between subunits of the same index are found.

As a result, we found that synchronous behaviors occurred in the u -coupled system smoothly switch or drastically transit, dependent of ϵ . This implies that the u -coupled system may be available and useful for applications to associative memory models or learning control models. In addition, if we deal carefully with ϵ , the u -coupled system can become a stable control model design. Since synchronous behavior in the v -coupled system is very robust against the coupling strength, it is a more stable control model with less functionality, relative to the u -coupled system.

5 Discussion

In this paper, we have shown a difference in inputs to the dynamics in the Matsuoka oscillator. We have also found that such input differences are related to coupling strength-dependent synchronous behavior. In particular, no behavioral results of the u -coupled system and no possibilities for applications are reported so far in studies using the Matsuoka oscillator.

Let us consider another possible application, except the one mentioned in the previous section, specifically, applications of the u -coupled system model design into the related control system. Because one synchronous state continuously shifts to another synchronous state in the u -coupled system in change of the coupling strength, it can be expected that walking motions such as diagonal gait very smoothly switch to pacing or ambling if we consider the implementation of the u -coupled system into quadruped walking robots. We address that the u -coupled system of the Matsuoka oscillator may have highly potential control model design.

Since the key point in finding high potentials in the u -coupled system is to control the relaxation time P , it is also analytically and theoretically interesting to investigate P -dependence of dynamical synchronous behavior in the v - and u -coupled system. We assume that such systems are weakly coupled systems. When the Matsuoka oscillator can be reduced to the one-dimensional phase equation using the phase reduction method [8], we can numerically analyze the stationary synchronous behavior of weakly coupled system of the Matsuoka oscillator and the dependence of P . So-called phase response curves (PRCs) play an important role in such analysis and implicitly predict that the drastic transition of synchronous behavior apparently depends on P .

In conclusion, we can claim that, in theoretical analysis as well as practical applications, very productive and fruitful outcomes are expected. At the very least, this paper will serve as a fundamental step and has shown that the u -coupled system of the Matsuoka oscillator is also theoretically appropriate network model. Compared to analytical and numerical results found in the v -coupled system, more complicated synchronization transitions are observed in the u -coupled system. We discussed possibilities to contain functionally high abilities in the u -coupled system.

Acknowledgments. This work was partially supported by the Grant-in-Aid for Young Scientist (B) No. 22700237.

References

1. Matsuoka, K.: Sustained Oscillations Generated by Mutually Inhibiting Neurons with Adaptation. *Biol. Cybern.* 52, 367–376 (1985)
2. Matsuoka, K.: Mechanisms of Frequency and Pattern Control in the Neural Rhythm Generators. *Biol. Cybern.* 56, 345–353 (1987)
3. Liu, G.L., Habib, M.K., Watanabe, K., Izumi, K.: Central pattern generators based on Matsuoka oscillators for the locomotion of biped robots. *Artif. Life Robotics* 12, 264–269 (2008)
4. FitzHugh, R.: Mathematical models of excitation and propagation in nerve. In: Schwan, H.P. (ed.) *Biological Engineering*, ch.1, pp. 1–85. McGraw-Hill Book Co, N.Y (1969)
5. Somers, D., Kopell, N.: Rapid synchronization through fast threshold modulation. *Biol. Cybern.* 68, 393–407 (1993)
6. Sato, Y.D., Shiino, M.: Spiking neuron models with excitatory or inhibitory synaptic couplings and synchronization phenomena. *Phys. Rev. E* 66, 41903 (2002)
7. Sato, Y.D., Shiino, M.: Generalization of coupled spiking models and effects of the width of an action potential on synchronization phenomena. *Phys. Rev. E* 75, 11909 (2007)
8. Kuramoto, Y.: *Chemical Oscillations, Waves, and Turbulence*. Springer, Tokyo (1984)

A RANSAC-Based ISOMAP for Filiform Manifolds in Nonlinear Dynamical Systems

–An Application to Chaos in a Dripping Faucet–

Hiromichi Suetani^{1,2,3} and Shotaro Akaho⁴

¹ Graduate School of Science and Engineering, Kagoshima University
1-21-35 Korimoto, Kagoshima-shi, Kagoshima 890-0065, Japan

² Decoding and Controlling Brain Information, PRESTO, Japan Science and
Technology Agency

4-1-8 Honcho, Kawaguchi-shi, Saitama 332-0012, Japan

³ Flucto-Order Functions Research Team, RIKEN-HYU Collaboration Research
Center, RIKEN Advanced Science Institute

2-1 Hirosawa, Wako-shi, Saitama 351-0198, Japan

⁴ The National Institute of Advanced Industrial Science and Technology
1-1-1 Umezono, Tsukuba-shi, Ibaraki 305-8568, Japan

Abstract. Trajectories generated from a chaotic dynamical system are lying on a nonlinear manifold in the state space. Even if the dimensionality of such a manifold is much lower than that of the full state space, we need many state variables to trace a motion on it as far as we remain to employ the original coordinate, so the resulting expression of the dynamics becomes redundant. In the present study, we employ one of the manifold learning algorithms, ISOMAP, to construct a new nonlinear coordinate that globally covers the manifold, which enables us to describe the dynamics on it as a low-dimensional dynamical system. Here, in order to improve the conventional ISOMAP, we propose an approach based on a combination with RANSAC for pruning the misconnected edges in the neighboring graph. We show that a clear deterministic relationship is extracted from time series of a mass-spring model for the chaotic dripping faucet using the proposed method.

Keywords: Manifold learning, ISOMAP, kernel methods, RANSAC, chaotic dynamical systems.

1 Introduction

The studies on nonlinear dynamical systems have offered various notions for understanding dynamical behaviors observed in many fields of science including physics, chemistry, biology, and engineering[1]. Especially, chaotic dynamical systems generate complicate non-closed trajectories which are lying on manifolds with high nonlinearity, so-called strange attractors, in their state space. Even if the dimensionality of such a manifold is much lower than that of the full state space, we need many state variables to trace a motion on it as far as we remain

to employ the original coordinate, so the resulting expression of the dynamics becomes redundant.

Recently, manifold learning methods have developed to reduce the dimensionality of data when they are lying closely on a lower-dimensional manifold. It has been shown that there is profound relations between manifold learning and kernel methods[2], e.g., well-known manifold learning algorithms including **ISOMAP**(isometric feature mapping)[3], **LLE**(locally linear embedding)[4], and Laplacian eigenmaps[5] can be unified as versions of kernel **PCA**[6].

In the present study, we employ **ISOMAP** to construct a new nonlinear coordinate which globally covers the manifold so that the original dynamics is replicated by a low-dimensional dynamical system with regard to the new coordinate. On the other hand, when the numbers of samples are limited, the conventional **ISOMAP** often does fail to unfold correctly a manifold like a filiform curve. This is because an isotropic neighborhood used in the conventional **ISOMAP** may wrongly connect pairwise points far different from each other in the sense of geodesic distance. Therefore, in order to improve the conventional **ISOMAP**, we propose an approach based on **RANSAC**(random sample consensus)[7] to prune the misconnected edges from the neighboring graph. We show that a clear deterministic relationship is extracted from time series of a mass-spring model for the chaotic dripping faucet using the proposed method.

2 Mass-Spring Model for Chaotic Dripping Faucet

us consider motion of dripping water. Figure 1 (a) shows an instant when a part of the drop separates from a burette in a real experiment. Although the dripping water is ideally described as an infinite-dimensional dynamical system, i.e., a partial-differential equation of fluid dynamics, if we focus on the case that the dynamics itself is sufficiently low-dimensional, it can be modeled as the following so-called mass-spring system as shown in Fig. 1 (b)[8]:

$$m\ddot{x} + (\dot{x} - v_0)\dot{m} = -kx - \gamma x + mg, \quad (1)$$

$$\dot{m} = Q \text{ (const.)}, \quad (2)$$

where x is the vertical position of the forming drop and m is its mass, g is the gravitational acceleration, k is the stiffness of the spring, and γ is the damping parameter. In this model, the restoring force by the surface extension of water is represented as the spring force in Eq. (1), and the mass of the forming drop linearly depends on time with the rate Q in Eq. (2) because a drop grows with time due to the influx of water from the faucet. Here, v_0 is the velocity of influx of water and there is a relationship between v_0 and Q as $Q = \pi a^2 v_0$ where a is the radius of the faucet. It is also assumed that when the position of the drop reaches critical point x_c , a part with the mass Δm separates from the drop and falls into the ground. In spite of simplicity, this model can

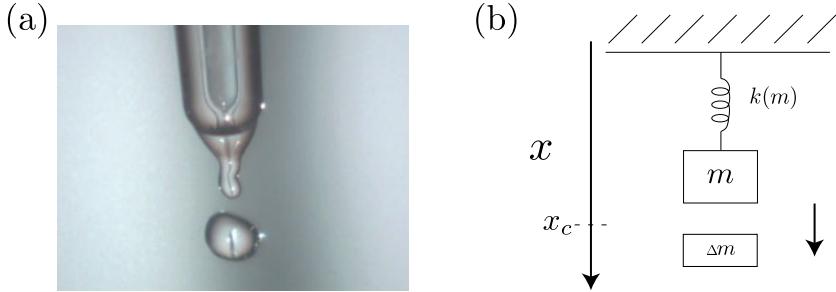


Fig. 1. (a) Snapshot of the dripping water. This picture was taken with VW-9000 (KEYENCE CO., LTD., Japan). (b) Schematic illustration of a mass-spring model.

explain qualitatively many aspects of nonlinear dynamics observed in the real dripping faucet[8].

Furthermore, Kiyono and Fuchikami modified the above phenomenological model by assuming that the stiffness k in Eq. (1) depends on the mass and takes the following form[9]:

$$k(m) = \begin{cases} -11.4m + 52.5 & (m < m_c), \\ 0 & (m \geq m_c). \end{cases} \quad (3)$$

When the mass m amounts to $m_c = 4.61$, the value of the stiffness becomes zero, then the drop undergoes free-fall together with dissipation. Just after a part of the drop separates, the position and velocity are renewed as $x = x_0 = 2.0$, $\dot{x} = \dot{x}_0 = 0$. They fix the critical position at $x_c = 5.5$, loss of the mass at $\Delta m = 0.8m - 0.3$, and a is set to 0.916[9]. Figure 2 shows a trajectory of the above mentioned model with $v_0 = 0.115$ after transient. In Fig. 2 (a) we can see that the trajectory are tracing a strange attractor, i.e., it exhibits chaotic motion. In real experiments, however, it is generally impossible to observe all state variables of the system. In the case of the dripping faucet, time intervals T_n , ($n = 1, 2, \dots$) between successive drippings are measurable in experiments. How to define T_n from $x(t)$ is depicted in Fig. 2 (b). Figure 2 (c) shows the time-delay embedding of the dripping time intervals into the three-dimensional Euclidian space \mathbb{R}^3 as $\mathbf{T}_n = (T_{n-1}, T_n, T_{n+1})$. We denote the manifold on which \mathbf{T}_n are lying as \mathcal{T} . In Fig. 2 (c), we can see that the manifold \mathcal{T} is a very complicate filiform curve with crossing. In fact, however, there is no crossing as shown in Fig. 2 (c'), which is the same \mathcal{T} from another perspective. That is, there is a deterministic causal relationship from \mathbf{T}_n to \mathbf{T}_{n+1} as $\mathbf{T}_{n+1} = f(\mathbf{T}_n)$ in \mathbb{R}^3 . If a new coordinate which globally covers the manifold \mathcal{T} can be constructed from samples of \mathbf{T}_n , we obtain a more simplified expression as $u_{n+1} = g(u_n)$ where u_n is a scalar coordinate along \mathcal{T} .

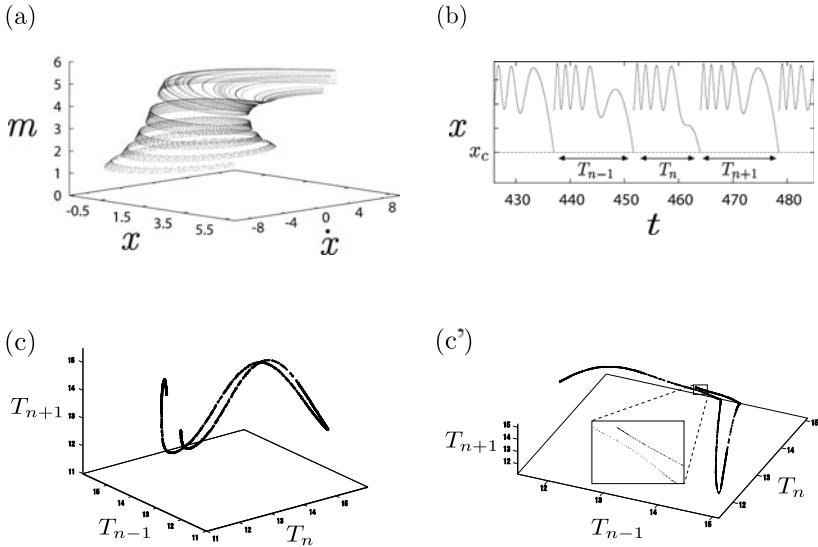


Fig. 2. (a) Strange attractor of the mass-spring model with $v_0 = 0.115$. (b) Time series $x(t)$ and how to define the dripping-time interval T_n is also illustrated. (c) Time-delay embedding of the time series of T_n into the three-dimensional space. (c') The same embedding as (c) from another perspective.

3 Analysis

3.1 Application of the Conventional ISOMAP

ISOMAP[3] is an extension of (classical) MDS(multi-dimensional scalings)[10] by incorporating geodesic distance of the concerned manifold. Here, the geodesic distance between pairwise points are approximated with the shortest path from one to the other on the neighboring graph \mathcal{G} which is constructed by locally connecting among sample points using the Euclidian distance.

Figures 3 shows the results of application of ISOMAP to data on the manifold \mathcal{T} shown in Fig 2 (c)[11]. Here, we use $N = 2 \times 10^3$ samples and $k = 15$ neighborhoods in order to construct \mathcal{G} . Figure 3 (a) shows configurations of sample points onto the plane of ISOMAP. The manifold \mathcal{T} forms a filiform as shown in Fig. 2 (c), so one expects that ISOMAP unfolds it as a straight line. The fact is, however, that samples are distributed in the two-dimensional plane, which implies that the conventional ISOMAP fails. So, the return plot with regard to the $u^{(1)}$ coordinate remains a multi-valued function (Fig 3 (b)).

3.2 RANASC-Based ISOMAP and Its Results

As shown in Fig 2 (c'), there is no crossing in $\mathcal{T} \subset \mathbb{R}^3$. But there is also a region where pairwise points are very close to each other in the sense of Euclidian dis-

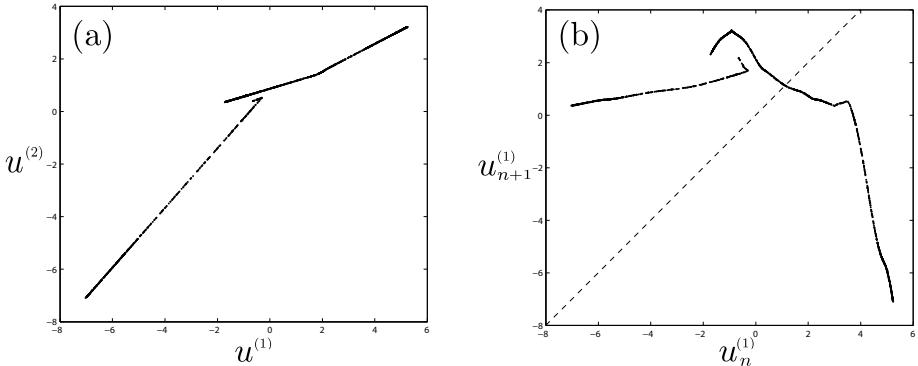


Fig. 3. Results of the application of ISOMAP to data on the nonlinear manifold \mathcal{T} . (a) Configurations of sample points onto the two-dimensional plane. (b) Return plot with regard to the $u^{(1)}$ coordinate.

tance in \mathbb{R}^3 whereas distant from each other in the sense of geodesic distance on \mathcal{T} (the inset of Fig 2 (c')). The conventional ISOMAP employs “isotropic” neighborhoods in constructing the neighboring graph \mathcal{G} , which leads to misconnections between pairwise points that are distant from each other in geodesic distance. This is the reason why the conventional ISOMAP fails to correctly unfold \mathcal{T} . Of course, if sufficiently a lot of samples, say the order of 10^4 , are available, such misconnections can be avoided. But, stationarity that ensures the recording of such many dripping-time intervals is not held in real experiments due to uncontrollable environmental changes. Besides, the calculation of shortest paths on \mathcal{G} becomes computationally heavy against 10^4 samples.

Let us now consider to change from the original neighboring graph \mathcal{G} obtained from isotropic neighborhoods to a new one \mathcal{G}' by pruning the misconnected edges in \mathcal{G} . As shown in Fig. 4, suppose that data samples are lying on the three different curves. Figure 4 illustrates a situation where sample points lying on two different regions with respect to the geodesic distance are very close to each other in the Euclidian space as observed in the inset of Fig. 2 (c'). When we employ the ordinal Euclidian distance to construct the neighborhood of \mathbf{y}_c , sample points within a outer circle(k_1 -neighborhood) are chosen. Because we would like to construct the \mathbf{y}_c ’s neighborhood using only samples on the center curve, so we have to prune the edges that connect \mathbf{y}_c with samples on the other curves. Such an operation can be realized by employing the notion based on RANSAC[7], i.e., drawing straight lines fitted to two sample points chosen randomly, then obtaining the optimal line l_* , as the one which contains the most sample points in its η -neighborhood. In practice, instead of performing the original RANSAC, we select a single point within the inner circle(k_2 -neighborhood) and consider the straight line passing through this point and \mathbf{y}_c . We repeat this process for all points in the k_2 -neighborhood and determine l_* as the one whose η -neighborhood contains the most sample points in the k_1 -neighborhood. After

choosing l_* , we prune the edges connecting between y_c and points which do not contained in η -neighborhood (the symbols “ \times ” in Fig.4), and the remains (the bold lines in Fig.4) are used to construct \mathcal{G}' for the improved ISOMAP.

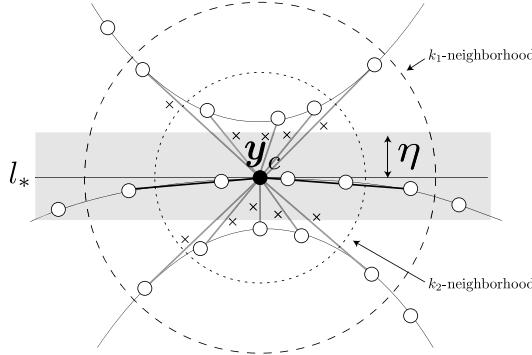


Fig. 4. Schematic illustration for the proposed approach. Edges that connect with points lying on other curves are pruned by the optimal linear subspace l_* .

Figure 5 shows the results of application of our approach (called R-ISOMAP in the following) to data on the manifold \mathcal{T} . In calculation, we use again $N = 2 \times 10^3$ samples, $k_1 = 15$ for constructing the initial neighborhood, and $k_2 = 6$ for drawing lines to determine l_* for each sample point. As same as Fig. 3 (a), Fig. 5 (a) shows the configurations of sample points onto the plane obtained from R-ISOMAP. Here, the spread of points in the direction of the $u^{(2)}$ component is much smaller than that of the $u^{(1)}$ component (about 2.5%). That is, the data on \mathcal{T} is almost explained only by the first component $u^{(1)}$, which implies that R-ISOMAP succeeds to unfold \mathcal{T} to a straight line. Figure 5 (b) shows the resulting return plot with regard to the $u^{(1)}$ coordinate. We can extract a deterministic dynamical system $u_{n+1}^{(1)} = g(u_n^{(1)})$ using only a single variable.

There are benefits to obtain a compact expression such as the one-dimensional map $g(\cdot)$ for the dynamics. For example, the dynamical characteristics such as the topological entropy h_0 and the Lyapunov exponent λ [1] can be easily calculated from the one-dimensional map. We have already checked the values of h_0 and λ and obtained the promising results.

We note here that we have selected the values of the hyper-parameters k_1 and k_2 heuristically whether the relationship between $u_n^{(1)}$ and $u_{n+1}^{(1)}$ in the resulting first coordinate of the R-ISOMAP space looks “deterministic”, i.e., the existence of a function-like form from $u_n^{(1)}$ to $u_{n+1}^{(1)}$ by changing the values of k_1 and k_2 . The proper choice of the hyper-parameters is crucial for obtaining the good performance of R-ISOMAP. In future, we will try to find optimal values for the hyper-parameters using cross-validation (CV). As a measure for performing CV, the average of the conditional variance $\text{Var}(u_{n+1}^{(1)}|u_n^{(1)})$ over the realizations of

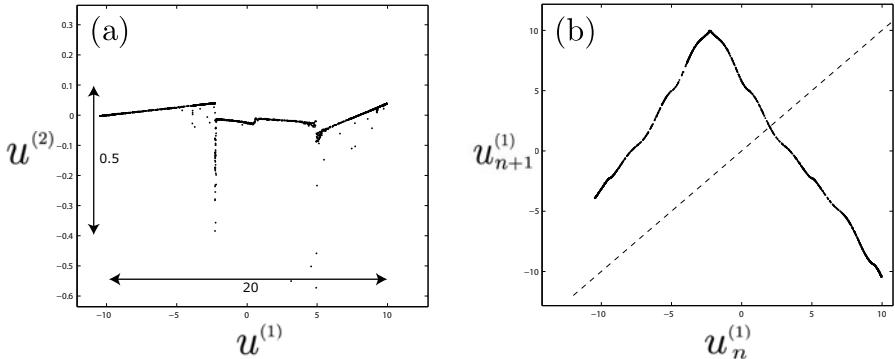


Fig. 5. Results of the application of R-ISOMAP to data on the nonlinear manifold \mathcal{T} . (a) Configurations of sample points onto the two-dimensional plane. (b) Return plot with regard to the $u^{(1)}$ coordinate.

$u_n^{(1)}$ can be useful. If $\langle \text{Var}(u_{n+1}^{(1)}|u_n^{(1)}) \rangle$ takes a smaller value, it means that the relationship from $u_n^{(1)}$ to $u_{n+1}^{(1)}$ becomes more deterministic.

4 Summary and Discussion

We proposed a new version of ISOMAP, R-ISOMAP, with the help of RANSAC-like approach for unfolding filiform manifolds in chaotic dynamical systems. We demonstrated that the proposed approach goes beyond the conventional ISOMAP in supporting the analysis of a mathematical model for the dripping faucet system. In real-world applications, only a portion of state variables is measurable because there are many constraints in experiments. So, it is often seen that the manifolds obtained from the time-delay embedding of such experimental time series may be very complex even if their dimensionality are low, which leads to multi-valuedness in the return plots. For example, the time series of inter-spike intervals is mainly observed in neural systems and its return plot often exhibits a multi-valued function[12]. Besides neural systems, there is a number of such examples, e.g., laser systems[13], passive biped walkers[14], and social activity models[15]. Extracting the deterministic relationship from the observations of these models is also our future subjects. There is a wide applicability of machine learning methods to many real-world problems in nonlinear dynamics such as synchronization in two qualitatively different chaotic oscillators[16].

Acknowledgments. This study is partially supported by Grant-in-Aid for Scientific Research (No. 22740258), the Ministry of Education, Science, Sports, and Culture of Japan. We thank H. Hata and H. Kuroiwa for the help of numerical simulations of the mass-spring model.

References

1. Alligood, K.T., Sauer, T.D., Yorke, J.A.: *Chaos—An Introduction to Dynamical Systems*. Springer, New York (1996)
2. Schölkopf, B., Smola, A.J.: *Learning with Kernels, Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge (2002)
3. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A Global Framework for Nonlinear Dimensionality Reduction. *Science* 290, 2319–2323 (2000)
4. Roweis, S., Saul, L.: Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science* 290, 2323–2326 (2000)
5. Belkin, M., Niyogi, P.: Laplacian Eigenmaps for dimensionality Reduction and Data Representation. *Neural Comp.* 15, 1373–1396 (2003)
6. Ham, J., Lee, D.D., Mika, S., Schölkopf: A Kernel View of the Dimensionality Reduction of Manifolds. In: *Proceedings of the Twenty-first International Conference on Machine Learning (ICML 2004)*, Banff, Canada, pp. 369–376 (2004)
7. Fischler, M.A., Bolles, R.C.: Random Sample Consensus: a Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Comm. ACM* 24, 381–395 (1981)
8. Shaw, R.: *The Dripping Faucet as a Model of Chaotic System*. Aerial Press, Santa Cruz (1984)
9. Kiyono, K., Fuchikami, N.: Dripping Faucet Dynamics by an Improved Mass-Spring Model. *J. Phys. Soc. Jpn.* 68, 3259–3270 (1999)
10. Cox, T., Cox, M.: *Multidimensional Scaling*. Chapman & Halls, London (1994)
11. We use a code of ISOMAP in Matlab Toolbox for Dimensionality Reduction, (http://homepage.tudelft.nl/19j49/Matlab_Toolbox_for_Dimensionality_Reduction.html) by Ph.D. L van der Maatens. We modify his code for implementing our proposing algorithm. We are grateful to him.
12. Feudel, U., Neiman, A., Pei, X., Wojtenek, W., Braun, H., Huber, M., Moss, F.: Homoclinic Bifurcation in a Hodgkin-Huxley Model of Thermally Sensitive Neurons. *Chaos* 10, 231–239 (2000)
13. Hübner, U., Weiss, C.-O., Abraham, N.B., Tang, D.: Lorenz-like Chaos in NH₃–FIR Lasers (data set A). In: Weigend, A.S., Gershenfeld, N.A. (eds.) *Time-Series Prediction: Forecasting the Future and Understanding the Past*, pp. 73–104. Westview Press, Boulder (1993)
14. Goswami, A., Thuiot, B., Espiau, B.: A Study of the Passive Gait of a Compass-like Biped Robot, Symmetry and Chaos. *Int. J. of Robotics Res.* 17, 1282–1301 (1998)
15. Feichtinger, G., Ghezzi, L.L., Piccardi, C.: Chaotic Behavior in an Advertising Diffusion Model. *Int. J. Bifurcation and Chaos* 5, 255–263 (1995)
16. Suetani, H., Iba, Y., Aihara, K.: Detecting Generalized Synchronization Between Chaotic Signals: a Kernel-based Approach. *J. Phys. A: Math. Gen.* 39, 10723–10742 (2006)

Manifold Learning for Visualization of Vibrational States of a Rotating Machine

Ignacio Díaz¹, Abel A. Cuadrado¹, Alberto B. Diez¹, and Manuel Domínguez²

¹ Área de Ingeniería de Sistemas y Automática, Edificio departamental 2, Campus de Viesques s/n 33204, Gijón, Asturias*

idiaz@isa.uniovi.es

² Universidad de León. Instituto de Automática y Fabricación

Abstract. This paper describes a procedure based on the use of manifold learning algorithms to visualize periodic –or nearly periodic– time series produced by processes with different underlying dynamics. The proposed approach is done in two steps: a feature extraction stage, where a set of descriptors in the frequency domain is extracted, and a manifold learning stage that finds low dimensional structures in the feature space and obtains projections on a low dimensional space for visualization. This approach is applied on vibration data of an electromechanical rotating machine to visualize different vibration conditions under two kinds of asymmetries, using four state-of-the-art manifold learning algorithms for comparison purposes. In all cases, the methods yield consistent results and produce insightful visualizations, suggesting future developments and application in engineering problems.

Keywords: manifold learning, dimensionality reduction, vibration analysis.

1 Introduction

Many problems in machine learning involve a –sometimes very– large number of variables. Examples of such problems can be found in image classification, text mining, socioeconomic data analysis or process condition monitoring, just to mention a few. In most cases, relationships –or constraints– among the observed variables arise from physical laws, spatial or geometrical restrictions, redundancy between two or more variables, etc. that make the problem depend on a reduced set of factors that explain the observed behavior. The computation of a minimal set of variables that describe these factors makes it possible to develop efficient data visualization methods with a large explanatory power.

The problem of finding a reduced set of latent variables that explain a large dimensional set is closely related to dimensionality reduction (DR). DR Techniques have been used for a long time. Maybe one of the first and most ever used DR algorithm is principal component analysis (PCA) [7], firstly described

* This work has been financed by the spanish Ministry of Science and Education and FEDER funds under grants DPI2009-13398-C02-01/02.

in 1901 by Pearson [9] and later used in a vast amount of works. Many other approaches have been proposed, such as multidimensional scaling (MDS) methods [15,11,4] that try to preserve inter-point distances, independent component analysis (ICA) [6] that seeks a linear transformation that maximizes independency of components, or self-organizing maps (SOM) that find nonlinear mappings on a low dimensional lattice [8]. In the last decade, however, a renewed interest on DR techniques emerged upon the publication of two methods, namely the local linear embedding and ISOMAP [10,14]. These methods led to a new class of algorithms –called local embeddings– that involve a local optimization –obtaining local models from k nearest neighbors– and a global alignment to obtain global coordinates for the local models, that usually leads to a singular value decomposition (SVD) problem. Some of the most known algorithms in this category are local linear embedding (LLE) [10], ISOMAP [14], laplacian eigenmaps (L-Eig) [1], local tangent subspace alignment (LTSA) [17], or hessian eigenmaps (HE) [5]. A good unifying framework for all these algorithms can be found in [16].

The dynamics of time series in many engineering systems are often produced by an underlying dynamical system usually defined by physical laws (mechanical, electrical, magnetic, etc.) as well as by many other kinds of constraints depending on the system's nature. This often results in highly structured signals, that span low dimensional manifolds in high dimensional signal or parameter spaces and which can also benefit from DR techniques. In this paper we propose the use of embedding methods for efficient visualization of time series dynamics using a two-stage approach, composed of a feature extraction stage followed by a dimensionality reduction stage that uses a local embedding method to capture the low dimensional structure in the feature space. This approach was applied to vibration data of a rotating machine, showing how the vibrational modes span low dimensional latent structures that can be efficiently unfolded by the DR method, allowing to produce a 2D map of vibration states.

2 Low Dimensional Representation of Time Series Dynamics

Frequency domain analysis is a widely used approach for the analysis of electrical and mechanical machinery [13,2,12]. In particular, vibrations bear plenty of information regarding the condition of the machine being monitored, that can be extracted by means of proper algorithms to determine the working condition or to evaluate its performance.

The main reason for using frequency domain approaches is based on the fact that they provide sparser representations for periodical signals than time domain representations. For such signals, the energy appears to be concentrated at certain frequencies, being zero or significantly low for the other ones. This fact, allows us to represent most of the energy of the signal using only a fraction of the frequencies, resulting in a much more compact description, that uses only a few parameters –namely, the energies at certain “harmonics”– to describe the main variations of the signal as a result of a change in the working condition.

Despite frequency domain is an efficient approach for typical signals in electromechanical machinery such as vibrations, the nature of electrical and mechanical phenomena involved usually result in signals with a complex spectrum, where each harmonic may be related to one or more dynamical phenomena that, in turn, may affect several frequencies.

2.1 Feature Extraction

Let's consider a measured a variable $y(t)$ at regular time intervals with a sample period T , producing a sequence $y_k = y(kT)$. Consider overlapped windows of length N each displaced $m < N$ samples apart

$$\mathbf{y}_n = [y_{n \cdot m}, y_{n \cdot m + 1}, \dots, y_{n \cdot m + N - 1}]$$

Fast Fourier transform (FFT) [3] is one of the most commonly used algorithms for spectral analysis and constitutes the basis of many other algorithms. It is just an efficient algorithm for the computation of the discrete Fourier transform (DFT). In this paper a windowed DFT transform is used to avoid Gibbs effect

$$Y_i = \sum_{k=0}^{N-1} w(k) y_k e^{-j2\pi i k / N}, \quad i = 0, \dots, N-1$$

where y_k is a real or complex data sequence, Y_i is a complex sequence describing the amplitudes and phases of the signal harmonics and $w(k)$ is a windowing function -Hanning in this paper. For the n^{th} window \mathbf{y}_n , the energies in bands around p specified center frequencies f_1, f_2, \dots, f_p with predefined bandwidths B_1, B_2, \dots, B_p can be computed by summing up the squares of the harmonics inside the bands, to obtain a p -dimensional feature vector

$$\mathbf{d}_n = [d_{1n}, d_{2n}, \dots, d_{pn}]^T \quad d_{in} = \sqrt{\sum_{\frac{k}{NT} \in [f_i - \frac{B_i}{2}, f_i + \frac{B_i}{2}]} \|Y_k\|^2}$$

Feature vectors can be arranged into a *data matrix* $\mathbf{D} = (d_{in}) = [\mathbf{d}_1, \mathbf{d}_2, \dots]$, where d_{in} represents the energy in the band $\{f_i, B_i\}$ -that is, with center frequency f_i and width B_i -, for window n . The feature extraction process for the k -th buffer along with the DR stage are summarized in the block diagram of Fig. 1.

2.2 Dimensionality Reduction

In many electromechanical systems, the feature vector described above is invariant for a given process condition, that is, significant changes occur if and only if the process condition changes and remains invariant during a steady state condition. Assuming that the process condition depends on one or more quantifiable factors we can establish the hypothesis that there is a one-to-one relationship between the high dimensional feature space and the low dimensional space of

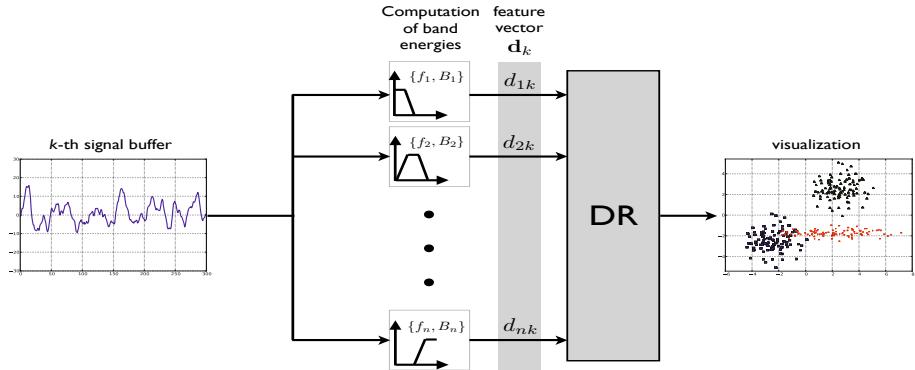


Fig. 1. Block diagram of the feature extraction and DR for one signal buffer. The signal buffer is converted to a 2D point in the visualization space.

factors that define the actual process condition. Under this hypothesis, the problem is to find latent low dimensional structures in the feature space as, precisely, DR techniques are aimed. Thus, DR techniques can be applied to unfold the data on the feature space. This yields a distribution of the projections using a spatial distribution that reflects the factors on which the process conditions depend. This, combined with scatter plots using colors and/or glyphs to represent features or variables with a physical sense, provide insightful information on the behavior of the process.

In this work four state-of-the-art methods, namely, LTSA, LLE, L-Eig, and ISOMAP are tested, showing that all produce insightful results and lead to conceptually similar conclusions.

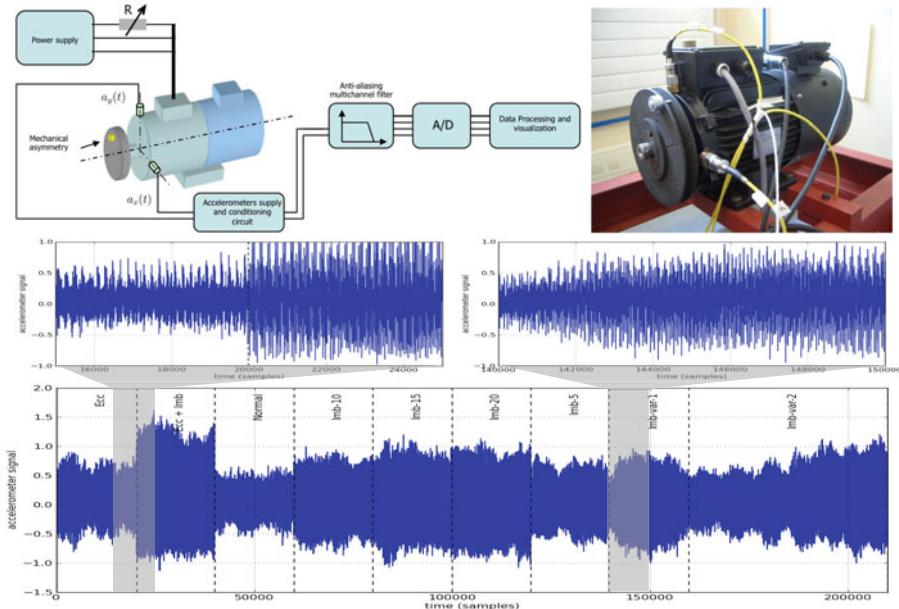
3 Experiments

The test rig consists of one AC asynchronous machine of 4kW and two pole pairs, two accelerometers –only the horizontal was used in this paper– with a bandwidth of 13 KHz and a 100 mV/g sensitivity, a multichannel low-pass filtering box and a data acquisition card. The accelerometer signal was previously filtered using a 4th order Butterworth filter to limit aliasing distortion and later acquired with a 5000 Hz sample rate.

The analyzed faults were induced by two kinds of asymmetries: external vibrations produced by a mechanical eccentricity – a revolving mass bolted to a disc causing an oscillating torque at the rotating speed, near 25 Hz– and an electrical imbalance caused by a variable electrical load –a resistor– in one of the phases. A total of nine experiments were done, running the motor under both kinds of faults and combinations of these. A description of these experiments can be seen in Table 1 and Fig. 2.

Table 1. Description of the 9 experiments

Experiment No.	Label	Description
#1	'Ecc'	Mechanical eccentricity
#2	'Ecc+Imb'	Mechanical eccentricity + Electrical imbalance
#3	'Normal'	No eccentricity, no electrical imbalance.
#4	'Imb-10'	Electrical imbalance (fixed load, 10 Ω)
#5	'Imb-15'	Electrical imbalance (fixed load, 15 Ω)
#6	'Imb-20'	Electrical imbalance (fixed load, 20 Ω)
#7	'Imb-5'	Electrical imbalance (fixed load, 5 Ω)
#8	'Imb-var-1'	Electrical imbalance (variable load, low \rightarrow high \rightarrow low)
#9	'Imb-var-2'	Electrical imbalance (variable load, low \rightarrow high)

**Fig. 2.** Schematic of the rig setup (top) and time plot of the accelerometer signal for the 9 experiments (bottom). The details in the bottom figure show up the change in time series dynamics.

4 Results

The rotation speed of the 2 pole-pair induction machine with no mechanical load is very close to 1500 rpm –25 Hz-. Frequencies associated to mechanical asymmetries will typically be $1\times, 2\times, \dots, n\times$ multiples of 25 Hz. Vibration components produced by electrical imbalance are multiples –specially $2\times$ – of the 50 Hz line frequency. For each of the 9 experiments, 35 dimensional feature vectors containing energies in bands 25 ± 1 Hz, 50 ± 1 Hz, \dots , 875 ± 1 Hz, were computed for 8192-point windows with an overlapping of 5%, by summing up the squared

modules of the FFT harmonics falling inside each band. To represent the process behavior, four manifold learning –LTSA, LLE, laplacian eigenmaps (L-Eig) and ISOMAP– methods were applied, with the parameters described in Table 2.

Table 2. Four DR methods and parameters

Method	Parameters	description
LTSA	$K = 100$	$K = 100$ neighbors –see [17].
LLE	$K = 100$, $\lambda = 0.1$	$K = 100$ neighbors and regularizing factor $\lambda = 0.1$ –see [10].
L-Eig	$\sigma = 0.5$, $\epsilon = 0.5$	Distances $\ \mathbf{x}_i - \mathbf{x}_j\ $ previously normalized to the range $[0, 1]$, the heat kernel parameter is $\sigma = 0.5$ and ϵ -neighborhood $\epsilon = 0.5$ –see [1].
ISOMAP	$K = 100$	$K = 100$ neighbors –see [14].

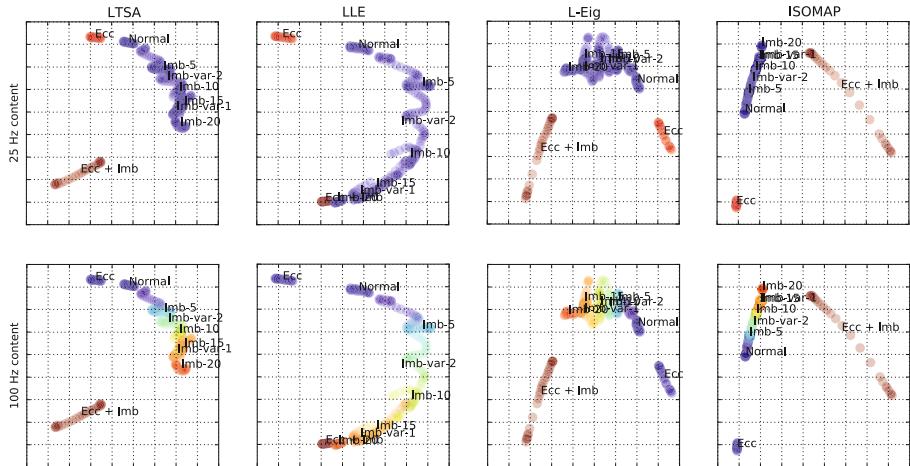


Fig. 3. Projections using the four methods

The projections obtained using the four methods are shown in Fig. 3, using a color scale to represent the values of two dynamic features, namely, the energy in the 25 ± 1 Hz band –which is specially sensitive to mechanical eccentricities– and the 100 ± 1 Hz –sensitive to electrical imbalance. The four columns represent the four DR methods and the two rows represent the features.

The results show that the four DR methods yield similar conclusions. All the methods show in separate regions the four main conditions, namely, Normal, Ecc, Imb-20 and Ecc+Imb. All the projections also show, as expected, smooth continuous transitions between the normal and severe electrical imbalance conditions –Imb-5, Imb-10, Imb-15, Imb-var-1, Imb-var-2–, showing up good consistency with the nature of the fault.

Some minor differences can be observed among the four methods. The combined mechanical and electrical asymmetry Ecc+Imb reveals a small 1D variation of its states in the LTSA, L-Eig and ISOMAP methods, and is reflected as a

small cluster in LLE method. Also, the continuous transition between the different degrees of electrical imbalances is better described in the LTSA, LLE and ISOMAP, while the L-Eig method seems to show a dependency on two factors. A more complete representation of the process dynamics can be obtained including a larger set of features, resulting in spectrum representations. Since every projected point is associated to a vector of energies at frequency bands, glyphs at every projected position can be generated from energies of a whole frequency range, resulting in a visualization like figure 4, that was obtained for projections of the LTSA method showing harmonics from 25 Hz to 200 Hz, in steps of 25 Hz. Such representation gives a comprehensive view of the global behavior of the signal frequency content. Also, it can be seen how the spectrum shape changes continuously from the different conditions, revealing the 1D nature of the vibration conditions induced in the experiments.

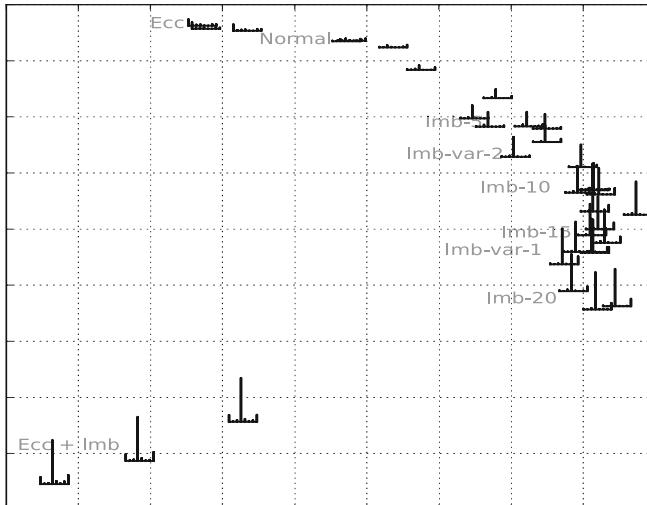


Fig. 4. Map of projected spectra using LTSA. Note that the projections are spatially distributed so that harmonics smoothly change across the vibration conditions.

5 Conclusions

In this paper we have proposed a procedure to explore the dynamic behavior of a process composed of a feature extraction stage based on the frequency domain, and a DR mapping using manifold learning algorithms for visualization. For many kinds of engineering systems, the method is based on the assumption that variations on their operating condition originated by a low number of factors influence in nearly the same way all frequencies of the signal spectrum, leading to highly structured data in a properly chosen feature space. The proposed

method is applied to vibration analysis of a faulty induction motor. The results show that vibration data produced under two different kinds of faults (electrical and mechanical asymmetries) produce low dimensional structures in the feature space that can be efficiently unfolded with state-of-the-art DR methods. The resulting projections can be efficiently represented using visualization methods that provide an insightful view of the changing dynamics, suggesting the potential use of the proposed approach in many problems in which visualization of dynamics is required such as fault detection or industrial process data mining. Finally, the method can be adapted for online monitoring using nonlinear interpolation between the input and projected points to project new data.

References

1. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation* 15(6), 1373–1396 (2003)
2. Benbouzid, M.E.H.: A review of induction motors signature analysis as a medium for faults detection. *IEEE Transactions on Industrial Electronics* 47(5), 984–993 (2000)
3. Cooley, J.W., Tukey, J.W.: An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation* 19(1), 297–301 (1965)
4. Demartines, P., Herault, J.: Curvilinear component analysis: a self-organizing neural network for nonlinear mapping of data sets. *IEEE Transactions on Neural Networks* 8(1), 148–154 (1997)
5. Donoho, D., Grimes, C.: Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences* 100(10), 5591 (2003)
6. Hyvärinen, A.: Survey on independent component analysis. *Neural Computing Surveys* 2, 94–128 (1999), <http://www.cis.hut.fi/aapo/pub.html>
7. Jolliffe, I.: Principal component analysis (1986)
8. Kohonen, T.: Self-Organizing Maps, Springer Series in Information Sciences, 3rd edn., vol. 30. Springer, New York (2001)
9. Pearson, K.: LIII. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series* 6 2(11), 559–572 (1901)
10. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. *Science* 290, 2323–2326 (2000)
11. Sammon, J.W.: A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers C* 18(5), 401–409 (1969)
12. Schoen, R.R., Habetler, T.G., Kamran, F., Bartheld, R.G.: Motor bearing damage detection using stator current monitoring. *IEEE Transactions on Industry Applications* 31(6), 1224–1279 (1995)
13. Tavner, P.J., Penman, J.: Condition Monitoring of Electrical Machines. Research Studies Press Ltd., John Wiley and Sons Inc. (1987)
14. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. *Science* 290, 2319–2323 (2000)
15. Torgerson, W.: Theory and Methods of Scaling. Wiley, Chichester (1958)
16. Zhang, T., Tao, D., Li, X., Yang, J.: Patch alignment for dimensionality reduction. *IEEE Transactions on Knowledge and Data Engineering*, 1299–1313 (2008)
17. Zhang, Z., Zha, H.: Principal manifolds and nonlinear dimension reduction via local tangent space alignment. *SIAM Journal of Scientific Computing* 26(1), 313–338 (2004)

Bias of Importance Measures for Multi-valued Attributes and Solutions*

Houtao Deng¹, George Runger¹, and Eugene Tuv²

¹ School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ, USA
`{hdeng3,george.runger}@asu.edu`

² Intel Corporation, Chandler, AZ, USA
`eugene.tuv@intel.com`

Abstract. Attribute importance measures for supervised learning are important for improving both learning accuracy and interpretability. However, it is well-known there could be bias when the predictor attributes have different numbers of values. We propose two methods to solve the bias problem. One uses an out-of-bag sampling method called OOBForest and one, based on the new concept of a partial permutation test, is called pForest. The existing research has considered the bias problem only among irrelevant attributes and equally informative attributes, while we compare to existing methods in a situation where unequally informative attributes (with or without interactions) and irrelevant attributes co-exist. We observe that the existing methods are not always reliable for multi-valued predictors, while the proposed methods compare favorably in our experiments.

keywords: Attribute importance, feature selection, random forest, cardinality.

1 Introduction

Attribute importance measures for supervised learning are important for improving both learning accuracy and interpretability. There are well known attribute importance measures such as information-based measures, chi-squared, and so forth. However, the bias problem for multi-valued attributes has been recognized for these methods. We refer to the number of distinct values of a attributes as its cardinality. [3] noted that attribute selection with Gini gain measure is biased in favor of those attributes with higher cardinality. [11] showed that there are biases in information-based measures adopted by decision tree inductions. [6] showed that attribute selection biases not only exist in information gain measures such as the Gini index, but also in others such as the distance measure in Relief [5], etc.

For solving the multi-valued problem, [7] introduced a normalization into the attribute selection measure known as the gain ratio. However, attributes with

* This research was partially supported by ONR grant N00014-09-1-0656.

very low information values then appeared to receive an unfair advantage [11,6]. Also [11] experimented with discrete, uniformly distributed attributes with different number of levels. They concluded that chi-squared could be used for the multi-valued problem. [6] proposed a minimum description length principle to alleviate the feature selection bias, but also mentioned that there are still slight decreases in the importance measure with the increasing cardinality.

Recently, a conditional inference framework [4] was proposed to solve the overfitting and attribute selection bias problems. [9] showed that this method (referred to as cForest) demonstrated promising results in both null and power cases. In the null case, all predictor attributes are irrelevant with different cardinality. In the power case, only one predictor attribute is informative, all other attributes are irrelevant with different cardinality [9]. Though these research methods have successfully discovered and alleviated the multi-valued problem to some degree, we observe that there are still some important problems that are unresolved.

A permutation importance measure (PIMP) was introduced by [1]. It permutes the target attribute and a p-value can be used to measure importance. However, PIMP fits the importance score with a prior probability distribution. Though specifying a prior distribution is not necessary, [1] used prior probability distributions in their experiments. One of our proposed algorithms (pForest) also uses permutation importance, but there are substantial differences. pForest permutes the predictors, and more importantly, make use of a partial permutation strategy for better efficiency. Furthermore, pForest does not need prior probability distributions to be specified. Here we focus on non-parametric methods and thus compare our methods to cForest in the later experiments.

Most experiments from the existing research are limited to some idealized situations. For example, [11] considered the multi-valued problem only for irrelevant attributes, while [9] considered irrelevant attributes and only one informative attribute. [6] considered irrelevant and equally informative attributes. However, there can exist both irrelevant and unequally informative attributes with different cardinalities. Furthermore, the informative attributes may interact with each other. Therefore, it is important to consider the multi-valued problem under more realistic scenarios.

We propose two new solutions for these problems. We focus on two-class classification (a common supervised problem), but our methods can be extended. We also focus on tree-based ensembles because of their capability to generate robust models that can handle nonlinearities, interactions, mixed (categorical and numerical) attributes, missing values, attribute scale differences, etc. However, our second method is not limited to a certain type of classifier. It is a meta approach that can be applied to improve feature selection algorithms. Furthermore, we contribute a more comprehensive simulation framework for studying the problem that integrates multiple cardinalities, and where non-equally informative attributes (with or without interactions) and irrelevant attributes co-exist. Such a framework can provide a useful benchmark to compare alternatives. Section 2 briefly summarizes some widely used importance measures. Section 3 describes

our proposed attribute importance methods. Section 4 describes our simulation framework and our experimental results, while Section 5 provides conclusions.

2 Attribute Importance Measures

Random forest (RF) [2] is a commonly-used feature selection tool. It allows for not only nonlinear models, but also attribute interactions. However, it can suffer from the multi-valued problem because it is based on an information criteria. Consequently, a remedy for RF's problem is important.

RF builds an ensemble of decision trees. Each tree is built on a bootstrap sample (random, with replacement) from the original training data. Also, at each node only a subset of attributes is selected from the full set of attributes and the split is calculated only from members of this subset. The objective is to decrease the correlation between trees in the ensemble in order to decrease the final model variance. RF uses the Gini impurity criterion for scoring attribute importance. Denote $Imp(X_k, \tau)$ as the importance of a attribute X_i at a single tree τ , then $Imp(X_k, \tau) = \sum_{t \in \tau} \Delta Gini(X_k, t)$ where $\Delta Gini(X_k, t)$ is the Gini impurity decrease at a node t where X_k is the splitting attribute. The Gini index at node t is defined as $Gini(t) = \sum_j p_j^t (1 - p_j^t)$ where p_j^t is the proportions of cases of class j at node t . The importance of X_k is obtained from the sum of the importance scores from trees $\tau_m, m = 1, \dots, M$ in a RF. For every tree τ in the ensemble, the instances not selected in the bootstrap sample are referred to as out of bag (OOB) and these cases can be considered to be a test sample for tree τ . These samples are used in our proposed importance measure.

A conditional inference framework [4] was proposed to solve the overfitting problem and attribute selection bias problem. [9] used the method to measure importance for multi-valued attributes in a model similar to a RF. In this method, for each node, first the attribute to be split is selected by minimizing the statistical p value of a conditional inference independence test. Then the splitting value is established by an appropriate splitting criterion. The separation of attribute selection and splitting criterion is the key to handle the cardinality bias [4].

3 Attribute Importance from OOBForest and pForest

In this section we introduce new methods to score attribute importance. An OOBForest uses the training samples to find the best splitting value on each attribute in the same manner as for a RF (with the Gini index as the default information measure). But, instead of discarding the OOB samples when building a tree, the OOB samples are used to select the best splitting attribute at a node. That is, the Gini index is recomputed for the OOB samples based on the split value obtained from the training data at each node. Furthermore, the importance measure $Imp(X_i, t)$ uses only the OOB samples. The principle here is similar to a conditional inference framework. The attribute selection criterion and splitting criterion are separated. The role of OOB samples was discussed for model improvements in [10], here we propose to use it to specifically solve

the bias problem in measuring attribute importance. Computationally, the extra work over a RF is to calculate the split score from the OOB samples at each node in the forest. Because the OOB samples for a tree are typically smaller than the original training data less time is needed (approximately 2/3 less) than to generate a second RF (and the basic RF algorithm is fast [2]).

Next consider the pForest. Denote $X_k, k = 1, \dots, K$ as the predictors and T as the target. [8] used permutation tests to obtain the statistical p value for dependency between an X_k and T . Then the inverse of the p value was used as the importance of the attribute. However, this method only measures the dependency of T over a single attribute X_k and the interactions between predictors are not considered. Permutation tests for feature selection was also used by [10]. Their method first randomly permuted each attribute $X_k, k = 1, \dots, K$ and then compared importance score of an attribute to the distribution of scores from the irrelevant variables obtained from the permutations to obtain the corresponding attributes $Z_k, k = 1, \dots, K$.

Our proposed algorithm also uses permutations, but an attribute is only compared to permuted version of itself. Furthermore, we introduce the concept of partial permutations. In each replicate r , by applying an importance method $f(\cdot)$ (such as RF) to $\{X_k, Z_k, T, k = 1, \dots, K\}$, the importance score of $X_k, Z_k, k = 1, \dots, K$, that is, $Imp_r(X_k)$ and $Imp_r(Z_k)$ can be obtained. A feature X_k is compared directly to its permuted version Z_k in each replicate to match the cardinality between X_k and Z_k (and this differs from [10]). Next consider the measure

$$Imp(X_k) = \frac{1}{R} \sum_{r=1}^R I[Imp_r(X_k) > Imp_r(Z_k)] \quad (1)$$

where $I(\cdot)$ denotes the indicator function. It can be seen that equation (1) is proportional to a binomial distribution $B(R, p_k)$, where p_k is the probability that $Imp(X_k) > Imp(Z_k)$. It is not feasible to compute the true p_k over all possible permutations in most practical situations. Therefore, [8] suggested a bounded number of permutations to achieve a significance level of 0.05.

The basic approach described so far is effective to distinguish informative from noninformative attributes. However, to rank informative attributes a more subtle refinement is used. In order to better detect finer importance relationships we propose partial permutations. That is, Z_k is obtained from permuting a fraction of the rows of X_k (a fraction δ selected randomly in each replicate). Consequently, as δ is decreased X_k and Z_k are more similar and it is more difficult for $Imp_r(X_k) > Imp_r(Z_k)$. Our default choice is $\delta = 20\%$ and in our experiments with the default δ , along with $R = 200$ replicates, we can achieve good results. We refer to this partial permutation method to attribute importance, with importance scores obtained from a RF, as the pForest.

Computationally, pForest is more demanding than OOBForest because each replicate requires another RF to be generated. However, the speed of a RF enables even hundreds of replicates to be computed in minutes for moderate data sets. Finally, note that although we focus on decision-tree ensembles, the permutation strategy to solve the multi-valued problem can be applied to any

feature selection method $f(\cdot)$. One would simply replace the score $Imp_r(X_k)$ with another method and still average $I[Imp_r(X_k) > Imp_r(Z_k)]$ over the replicates.

4 Experiments

Similar to [11,6,9], the experiments are setup as simulations so that the "ground truths" for attribute importance are known. The relationships between the predictors and the target are shown in Figure 1. Here T_1 and T_2 are the target attributes with and without interactions present in the model, respectively. All other attributes are predictors. The generation and properties for these attributes are summarized as follows:

- Generate $X_1 \sim Normal(0, 10)$, and then discretize (equal-frequency) into X_2, X_3, X_4, X_5, X_6 with different cardinalities shown in Figure 1. Randomly permute 30% of the rows of X_5 , and 50% of the rows of X_6 . This injects different amounts of noise into X_5, X_6 so that they are unequally informative concerning the target.
- Generate $Y_k, k = 1, \dots, 6$ independent from $X_k, k = 1, \dots, 6$. The generation procedure is similar to the generation of $X_k, k = 1, \dots, 6$.
- Generate $U_1 \sim Uniform(-10, 10)$, and then discretize (equal-frequency) into U_2, U_3, U_4, U_5, U_6 with different cardinalities.
- Generate the binary target T_1 as $P(T_1 = X_4) = 0.95, P(T_1 \neq X_4) = 0.05$.
- Generate the binary target T_2 as $P(T_2 = XOR(X_4, Y_4)) = 0.95, P(T_2 \neq XOR(X_4, Y_4)) = 0.05$ (where XOR is the exclusive or).

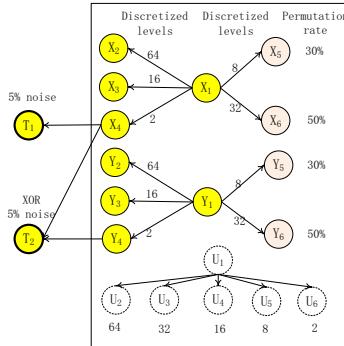


Fig. 1. Relationship between predictors and targets along with cardinalities. Here T_1 and T_2 denote the target attributes for the experiments with and without interactions, respectively.

Two experiments are derived from the relationships among the attributes. First T_1 is the target and $\{X_k, U_k, k = 1, \dots, 6\}$ are the predictors and then T_2 is the target and $\{X_k, Y_k, U_k, k = 1, \dots, 6\}$ are the predictors. In the second experiment the true model for T_2 includes interactions from the XOR function.

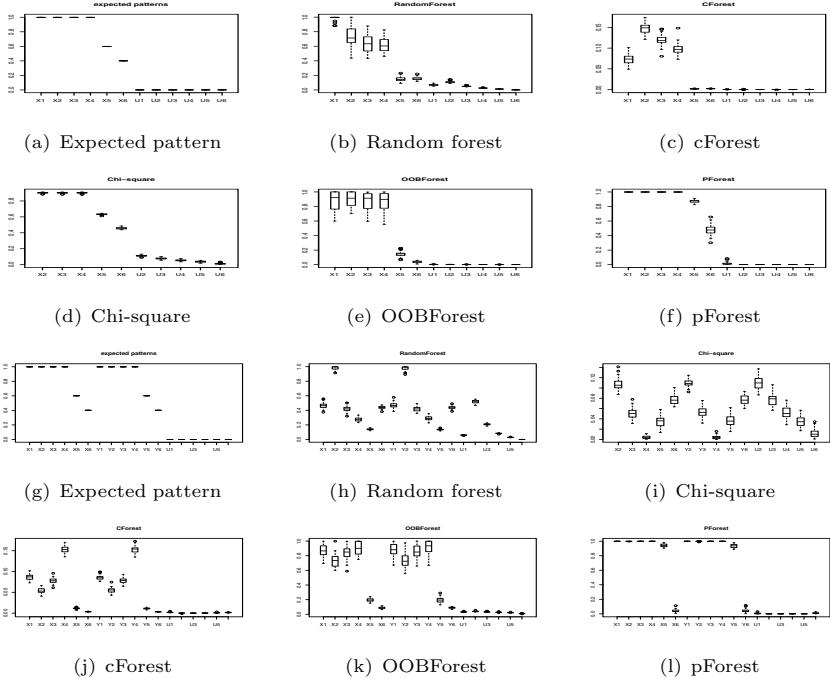


Fig. 2. Feature importance for experiments without (Figures 2(a) to 2(f)) and with (Figures 2(g) to 2(l)) interactions. The X axis represents attributes, and the Y axis provides importance scores. Figures 2(a) and 2(g) illustrate the expected pattern for the no interaction and interaction cases, respectively (only relative expected measures).

In each experiment, 50 replicates of data sets are simulated, with $5120 = 10 * 2^9$ rows of data in each data set (so that all values of an attribute have the same number of rows). For example, for a two-value attribute, values 0 and 1 each have 2560 rows.

By designing such experiments, the order of the importance scores for the predictor attributes is known. In the first experiment: $Imp(X_1) = \dots = Imp(X_4) > Imp(X_5) > Imp(X_6) > Imp(U_1) = \dots = Imp(U_6)$. Therefore, there are four groups and attributes from the same group have equal information regarding T_1 . In the second experiment: $Imp(X_1) = \dots = Imp(X_4) = Imp(Y_1) = \dots = Imp(Y_4) > Imp(X_5) = Imp(Y_5) > Imp(X_6) = Imp(Y_6) > Imp(U_1) = \dots = Imp(U_6)$. Therefore, there are still four groups and attributes from the same group have equal information regarding T_2 . A attribute importance measure should be able to indicate such orders of importance. We applied the original RF, chi-squared, OOBForest [10], cForest [9,4] and pForest to each data set. Each forest used 200 trees. For the pForest test, we set $\delta = 20\%$ and $R = 200$. Because chi-squared works only for categorical attributes, the continuous predictors were removed before chi-squared importance measures were applied.

For the experiment without interactions Figure 2(a) illustrates the expected pattern. For basic RF in Figure 2(b), the importance measure prefers higher attributes for both informative and irrelevant attributes. Also, it can't discriminate between X_5 and X_6 . Furthermore, the continuous attribute X_1 has the greatest importance score among the informative attributes. However, for the irrelevant attributes, the importance scores of the categorical attributes increase as the cardinality increases, and exceed the importance score of the X_1 when the cardinality equals 64. For cForest in Figure 2(c), there is no bias among the irrelevant attributes, which is consistent with the null case in [9]. cForest can also discriminate informative attributes from irrelevant attributes (though the differences between $Imp(X_5)$, $Imp(X_6)$ and the U_k are not obvious), which is also consistent with the power case in [9]. However, for the informative attributes, cForest prefers higher cardinality attributes. Furthermore, it can not discriminate X_5 from X_6 . For chi-squared in Figure 2(d), a concern is that higher-cardinality attributes are preferred for irrelevant attributes. For this experiment, it is able to rank informative attributes higher than irrelevant attributes and there is no obvious multi-valued problems for informative attributes. For both OOBForest in Figure 2(e) and pForest in Figure 2(f) there is no bias in both informative and irrelevant attributes. The expected orders among all predictor attributes are well preserved. Therefore, OOBForest has good performance here.

For the experiment with interactions Figure 2(g) illustrates the expected pattern. For RF in Figure 2(h), the bias is even more severe than in the first experiment. RF cannot even discriminate irrelevant attributes from some informative attributes. The importance of U_2 is only less than X_2 and Y_2 . Therefore, the attribute importance scores from RF are extremely unreliable here. Chi-squared in Figure 2(i) cannot even distinguish between the informative attributes and the irrelevant attributes. This is expected because chi-squared does not consider the interactions and this observation can clearly be extended to other methods (such as information gain), which only consider dependency between a single predictor and the target. For cForest in Figure 2(j) there is no obvious bias among the irrelevant attributes and cForest can also discriminate the informative attributes from the irrelevant attributes (although the importance difference between X_6 and U_i s is not obvious). However, there is multi-valued problem in the informative attributes. In contrast to the previous experiment, cForest now prefers lower cardinality attributes. For OOBForest in Figure 2(k), there is no bias among the irrelevant attributes. The four groups can be discriminated. There are some minor importance differences among the most informative attributes. For pForest in Figure 2(l), there is no bias in both informative and irrelevant attributes. The expected orders among all predictors are well preserved.

From the experiments, RF is not reliable when predictors have different cardinality (prefers high cardinality attributes). cForest performs well for those irrelevant attributes with different cardinality. However, it is not reliable enough for the importance of informative attributes with different cardinalities. Without interactions, chi-squared prefers higher-cardinality ones for irrelevant attributes. More importantly, chi-squared is not reliable when interactions are present. The

results of pForest are much better than RF. OOBForest also performs well in both experiments.

5 Conclusion

The bias of attribute importance measures is an important problem, and the common use of RF for attribute importance is shown to be a concern. We propose one method based on out-of-bag samples, while a second method uses the new concept of a partial permutation test to refine the attribute importance scores. The second method can be easily adapted to other feature scoring algorithms. We use a simulation framework that integrates different cardinalities, and where non-equally informative attributes (with or without interactions) and irrelevant attributes co-exist. Our proposed methods are compared directly to two existing solutions for multi-value bias: chi-squared and a conditional inference framework and we observe that the existing methods are not always reliable for multi-valued predictors, while the proposed methods compare favorably in our experiments.

References

1. Altmann, A., Tolosi, L., Sander, O., Lengauer, T.: Permutation importance: a corrected feature importance measure. *Bioinformatics* 26(10), 1340–1347 (2010)
2. Breiman, L.: Random forests. *Machine Learning* 45, 5–32 (2001)
3. Breiman, L., Friedman, J., Olshen, R., Stone., C.: *Classification and Regression Trees*, Wadsworth, Belmont, MA (1984)
4. Hothorn, T., Hornik, K., Achim, Z.: Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics* 15, 651–674 (2006)
5. Kira, K., Rendell, L.A.: A practical approach to feature selection. In: *Proceedings of the Ninth International Workshop on Machine Learning*, Aberdeen, Scotland, United Kingdom, pp. 249–256 (1992)
6. Kononenko, I.: On biases in estimating multi-valued attributes. In: *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, Montreal, Canada, pp. 1034–1040 (1995)
7. Quinlan, J.R.: Induction of decision trees. *Machine Learning* 1(1), 81–106 (1986)
8. Radivojac, P., Obradovic, Z., Dunker, A.K., Vucetic, S.: Feature selection filters based on the permutation test. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) *ECML 2004. LNCS (LNAI)*, vol. 3201, Springer, Heidelberg (2004)
9. Strobl, C., Boulesteix, A.-L., Zeileis, A., Hothorn, T.: Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics* 8(25) (January 2007)
10. Tuv, E., Borisov, A., Runger, G., Torkkola, K.: Feature selection with ensembles, artificial variables, and redundancy elimination. *Journal of Machine Learning Research* 10, 1341–1366 (2009)
11. White, A.P., Liu, W.Z.: Technical note: Bias in information-based measures in decision tree induction. *Machine Learning* 15(3), 321–329 (1994)

A Computationally Efficient Information Estimator for Weighted Data

Hideitsu Hino and Noboru Murata

Waseda University, Okubo 3-4-1, Shinjuku-ku, Tokyo, 169-8555, Japan
hideitsu.hino@gmail.com, noboru.murata@eb.waseda.ac.jp

Abstract. The Shannon information content is a fundamental quantity and it is of great importance to estimate it from observed dataset in the field of statistics, information theory, and machine learning. In this study, an estimator for the information content using a given set of weighted data is proposed. The empirical data distribution varies depending on the weight. The notable features of the proposed estimator are its computational efficiency and its ability to deal with weighted data. The proposed estimator is extended in order to estimate cross entropy, entropy and KL divergence with weighted data. Then, the estimators are applied to classification with one-class samples, and distribution preserving data compression problems.

Keywords: Information, entropy, quantile, non-parametric.

1 Introduction

In information theory [1], one of the most important quantities is the Shannon information content

$$I_f(x) = -\log f(x), \quad (1)$$

where $f(x)$ is a probability density function (pdf) of a random variable X , and $x \in \mathbb{R}^d$ is its realization (a datum in this study). The Shannon differential entropy is defined by averaging the information content $I_f(x)$ with $f(x)$ as $H(f) = \mathbb{E}_f[I_f(X)] = -\int f(x) \log f(x) dx$, where $\mathbb{E}_f[\cdot]$ is a mean operator. When the information content $I_g(x) = -\log g(x)$ of a datum x from a pdf $g(x)$ is averaged with respect to another pdf $f(x)$, it is called the cross entropy: $H(f, g) = \mathbb{E}_f[I_g(X)] = -\int f(x) \log g(x) dx$. The difference between the cross entropy and entropy is called the Kullback-Leibler (KL) divergence. These quantities play important roles in various literatures. For example, in independent component analysis [2], entropy is one of the criteria often used for separating independent signals from mixed signals [3].

Many attempts have been made to estimate the information content, entropy and KL divergence. A pdf can be estimated by a kernel density method [4] and information is given by negative logarithm of the estimated pdf. Beside these naïve approaches, there are various sophisticated methods [5]. Particularly, the quantile based method presented in this paper is closely related to the k nearest

neighbor (k -NN) methods [6,7]; however, it is not obvious how to deal with weighted dataset in k -NN methods, while we can take weights into account naturally in the quantile based method.

We define the weighed data by

$$\mathfrak{D} = \{\mathcal{D}, \mathcal{W}\}, \mathcal{D} = \{x_i\}_{i=1}^n, \mathcal{W} = \{w_i\}_{i=1}^n, \quad x_i \in \mathbb{R}^d, \sum_{i=1}^n w_i = 1, w_i \geq 0,$$

where, in the pair \mathfrak{D} and \mathcal{W} , \mathcal{D} is a collection of data and \mathcal{W} is a collection of positive valued weights, and each element $w_i \in \mathcal{W}$ is assigned to each datum $x_i \in \mathcal{D}$. With this weighted data \mathfrak{D} , an empirical mean of a function $F(x)$ is defined by $\sum_{i=1}^n w_i F(x_i)$.

The rest of this paper is organized as follows. Section 2 describes the notion of quantile, and derives estimators of the Shannon information content. In section 3, the cross entropy and entropy estimators are derived from the proposed information estimators. In Section 4, two applications of the proposed estimators are given, and examined with artificial and real-world datasets. The last section gives concluding remarks.

2 Information Estimator

2.1 Information Estimator Based on Quantiles

A point $x \in \mathbb{R}^d$ which is the subject of information estimation is called an *inspection point*. Let $\varepsilon(x, x')$ be the Euclidean distance between x and x' , and $b(x, \varepsilon) = \{x' \in \mathbb{R}^d; \varepsilon(x, x') < \varepsilon\}$ be an ε -ball centered at x . Note that the volume of this ε -ball is $|b(x, \varepsilon)| = c_d \varepsilon^d$, where $c_d = \pi^{d/2} / \Gamma(1 + d/2)$ and $\Gamma(x)$ is the gamma function. Denote the probability mass of the ε -ball centered at the inspection point x by $p_x(\varepsilon)$, i.e., $p_x(\varepsilon) = \int_{b(x, \varepsilon)} f(x') dx'$. Sorting $x_i \in \mathcal{D}$ in ascending order of the distance $\varepsilon(x, x_i)$, $x_i \in \mathcal{D}$, we denote the index of the i -th nearest point from x in \mathcal{D} by (i) . Then, we define the *quantile* of $x_{(i)}$ by $\sum_{j=1}^i w_{(j)}$. If the weight $w_i \in \mathcal{W}$ is identical for all i , then the quantile of $x_{(i)}$ is $\sum_{j=1}^i w_{(j)} = i/n$, which is a proportion to whole data size. Conversely, when an inspection point x and a quantile α are specified, the point in \mathfrak{D} is called the α -*quantile point* if the quantile of x is equal to α . We let ε_α be the distance between x and its α -quantile point henceforth. In the dataset \mathfrak{D} , we denote the number of points within a ball $b(x, \varepsilon_\alpha)$ by $\iota(\alpha)_x$.

Assuming that

$$\varepsilon_\alpha \ll 1, \tag{2}$$

we obtain the following approximation formula by Taylor's expansion

$$\begin{aligned} \alpha = p_x(\varepsilon_\alpha) &= \int_{b(x, \varepsilon_\alpha)} f(x') dx' = \int_{b(x, \varepsilon_\alpha)} f(x + (x' - x)) dx' \\ &= |b(x, \varepsilon_\alpha)| (f(x) + o(\varepsilon_\alpha^2)) \sim c_d \varepsilon_\alpha^d f(x), \end{aligned}$$

where o is Landau's little o . Taking logarithm of the above expression, we obtain an information estimator for an inspection point $x \in \mathbb{R}^d$ as

$$I_\alpha(x; \mathfrak{D}) = \log c_d - \log \alpha + d \log \varepsilon_\alpha. \quad (3)$$

We call this estimator a *Quantile Information Estimator (QIE)* for x with weighted data $\mathfrak{D} = \{\mathcal{D}, \mathcal{W}\}$.

When α is fixed, the bias of the QIE depends on the assumption (2). When α goes to zero as $n \rightarrow \infty$ and αn is fixed, say, $\alpha n = M < \infty, M \in \mathbb{R}_+$, then $\varepsilon_\alpha \rightarrow 0$ as $n \rightarrow \infty$. In this case, the estimation bias caused by violation of the assumption (2) goes to zero as $n \rightarrow \infty$, though, there is still a bias caused by unequal weight. As for this bias, the following proposition holds.

Proposition 1. *When $\alpha n = M < \infty, M \in \mathbb{R}_+$ is fixed for any n , the bias of the QIE is given by $\psi(\iota(\alpha)_x) - \psi(M)$, where $\psi(\cdot)$ is the digamma function.*

A proof will be shown in the extended version of this work. Although the QIE is biased, it is easy to use and accurate enough for practical situations. We also note that the assumption (2) can not be enforced by normalizing the data since the information and entropy are not scale invariant.

2.2 Mean Quantile Information Estimator

When we estimate information by the QIE, the value of α should be appropriately determined depending on the given data. Furthermore, to evaluate $I_\alpha(x; \mathfrak{D})$, we have to find the α -quantile point from \mathfrak{D} and its computational cost is of order $O(n \log n)$ because we have to sort the data according to $\varepsilon(x, x_i)$. To avoid these problems, we marginalize α in QIE $I_\alpha(x; \mathfrak{D})$ as

$$\int_0^1 I_\alpha(x; \mathfrak{D}) d\alpha = \log c_d - \int_0^1 \log \alpha d\alpha + d \int_0^1 \log \varepsilon_\alpha d\alpha. \quad (4)$$

In actual calculation of (4), the integration $\int \log \varepsilon_\alpha dx$ is approximated by summation of values at the observed data points. When the observed data are not weighted, this approximated integration is calculated by summation of rectangles with equal width $1/n$ as shown in figure 1 (left). On the other hand, when each datum x_i has its weight w_i , each datum contributes to the summation at the rate of the weight w_i and the approximation is calculated by summation of rectangles with different width w_i as shown in figure 1 (right). As α increases from 0 to 1, each point in \mathfrak{D} becomes the α -quantile point once for each. Then, we obtain an information estimator by integrating $I_\alpha(x; \mathfrak{D})$ with respect to α as

$$I_{MQ}(x; \mathfrak{D}) = \int_0^1 I_\alpha(x; \mathfrak{D}) d\alpha = \log c_d + 1 + d \sum_{i=1}^n w_i \log \varepsilon(x_i, x). \quad (5)$$

We call this estimator a *Mean Quantile Information Estimator (MQIE)*. The derivation of this estimator is similar to that of the estimator proposed in [8], which extends the k -NN entropy estimator to a parameter free estimator by

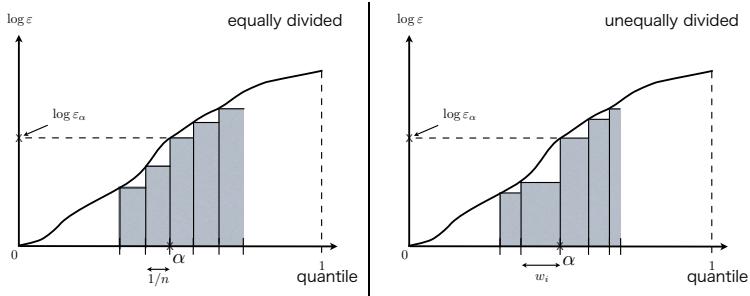


Fig. 1. Difference of quadrature. Left: approximation of integral by equal width rectangles. Right: approximation of integral by unequal width rectangles.

summing up the estimators for all ks . Unlike that work, our quantile based formulation enables us to take weights of observed data into account naturally.

By integrating α out, the estimator inevitably involves ε_α where the assumption (2) does not hold. However, the MQIE requires neither calculation of the α -quantile point nor sorting of the data according to $\varepsilon(x, x_i)$, and its computational cost is on the order of $O(n)$.

3 Entropy Estimator Based on Quantiles

Given two sets of weighted data

$$\mathfrak{D} = \{\mathcal{D}_x, \mathcal{W}\} = \{(x_i, w_i)\}_{i=1}^n, \quad \mathfrak{D}' = \{\mathcal{D}_y, \mathcal{V}\} = \{(y_j, v_j)\}_{j=1}^m, \quad (6)$$

where $x_i \in \mathcal{D}_x$, $y_j \in \mathcal{D}_y$ are sampled from distributions with pdfs $f(x), g(y)$ respectively, we will derive a cross entropy estimator. Let $y_{j(i)} \in \mathcal{D}_y$ be the α -quantile point in \mathfrak{D}' from a point $x_i \in \mathcal{D}_x$. With weighted data \mathfrak{D}' , the QIE of the inspection point x_i is given by $I_\alpha(x_i; \mathfrak{D}') = \log c_d - \log \alpha + d \log \varepsilon(x_i, y_{j(i)})$. Averaging this estimator with respect to $x_i \in \mathcal{D}_x$, we obtain a quantile cross entropy estimator as $H_\alpha(\mathfrak{D}, \mathfrak{D}') = \log c_d - \log \alpha + d \sum_{i=1}^n w_i \varepsilon(x_i, y_{j(i)})$. Furthermore integrating α out taking account of the weight \mathcal{V} for \mathcal{D}_y , we obtain a *Mean Quantile Cross Entropy Estimator* (MQCE) as

$$H_{MQ}(\mathfrak{D}, \mathfrak{D}') = \log c_d + 1 + d \sum_{i=1}^n \sum_{j=1}^m w_i v_j \log \varepsilon(y_j, x_i). \quad (7)$$

To estimate entropy using data $\mathfrak{D} = \{\mathcal{D}_x, \mathcal{W}\} = \{(x_i, w_i)\}_{i=1}^n$, we can use a leave-one-out estimation procedure. Let $\bar{\mathfrak{D}}_i = \{\mathcal{D}_x \setminus x_i, \frac{1}{1-w_i}(\mathcal{W} \setminus w_i)\}$ be weighted data without $\{(x_i, w_i)\}$. Mean quantile information estimator of a datum x_i becomes

$$I_{MQ}(x_i; \bar{\mathfrak{D}}_i) = \log c_d + 1 + d \sum_{j=1, j \neq i}^n \frac{w_j}{1-w_i} \log \varepsilon(x_j, x_i). \quad (8)$$

Then, we define a *Mean Quantile Entropy Estimator (MQEE)* by averaging $I_{MQ}(x_i; \bar{\mathfrak{D}}_i)$ with respect to x_i as

$$H_{MQ}(\mathfrak{D}) = \log c_d + 1 + d \sum_{i=1}^n \sum_{j=1, j \neq i}^n \frac{w_i w_j}{1 - w_i} \log \varepsilon(x_j, x_i). \quad (9)$$

4 Numerical Experiments

Binary Classification by One-Class Samples

We consider to train a binary classifier using only a dataset $\mathcal{D}_x = \{x_i\}_{i=1}^n$ from a class C_1 and classify data from two classes C_1 or C_2 . We use the proposed information estimator as the classifier and verify that it can successfully capture the characteristics of the data for classification. For this purpose, we only consider an equally weighted case, and assign a weight $1/n$ to each $x_i \in \mathcal{D}_x$. That is, we let $\mathfrak{D} = \{\mathcal{D}_x, \mathcal{U}\}$, where $\mathcal{U} = \{1/n, \dots, 1/n\}$. Using the estimator (5), we estimate the information content of every datum $x_i \in \mathcal{D}_x$ to obtain a set of estimates $S_x = \{I_{MQ}(x_i; \mathfrak{D})\}_{i=1}^n$. We introduce a confidence bound parameter $\nu \in [0, 1]$, and denote the upper $100(1 - \nu)\%$ point of S_x by t_x . Then, based on the training dataset \mathfrak{D} , we estimate $I_{MQ}(y_i; \mathfrak{D})$ of each test datum $y_j \in \mathcal{D}_y$ which belongs to C_1 or C_2 , and classify y_i into C_1 if $I_{MQ}(y_j; \mathfrak{D}) < t_x$ and classify y_i into C_2 otherwise.

To evaluate the proposed method, we employ the IDA datasets that are the standard binary classification sample sets originally used in [9]. We let samples from one class C_1 in a training set be $\mathcal{D}_x = \{x_i\}_{i=1}^n$, and test sets which consists of samples from both classes be $\mathcal{D}_y = \{y_i\}_{i=1}^m$.

For the classification problem with one-class training samples, one-class SVMs [10] are widely used. We compare the classification accuracy and computational costs of the proposed method and ν one-class SVM with various kernel functions. We use Gaussian, Laplacian, and polynomial kernels of degree $d \in \{2, 3\}$. The variance parameter for Gaussian and Laplacian kernel were automatically set at appropriate values by the package program [11]. The role of ν in ν one-class SVM is similar to that of the proposed method. For both methods, it is important to set the parameter ν appropriately, and it is usually done through trial and error by system operators. In this experiment, we assume that we can use small number of samples from both classes for ν optimization, and as such data, we use 1/5 of test samples. The remaining 4/5 of the test data are used for accuracy evaluation. We show the sum of false positive and false negative error rates of each method in table 1. From this table, we see that for most datasets, the proposed method show low error rates compared to ν one-class SVMs. This indicates that the proposed information estimator can capture the underlying data structure well. Furthermore, the computational cost of the proposed method is at least 4 times lower than that of ν one-class SVMs.

Application to Distribution Preserving Data Compression

Consider the case where we want to reduce the data size without loosing the characteristics of the data distribution. Learning Vector Quantization (LVQ) [12] is widely used for such purposes. With the input data $\mathcal{D} = \{x_i\}_{i=1}^n$ with class labels, LVQ constructs a set of codebook vectors $\mathcal{C} = \{c_j\}_{j=1}^m$, $m \leq n$ by minimizing

Table 1. Average and standard deviation of total error rates (in % figures). The best results and comparable ones based on the *t*-test with a significance level of 5% are shown in boldface type. Computation time for each method summed over all of the datasets are shown in the bottom of table. For better comparison, we normalized the values of computational time by that of the proposed method.

Data name	Proposed	Gaussian	Laplacian	Polynomial(2)	Polynomial(3)
banana	35.09(7.84)	17.94 (2.47)	38.9(8.31)	43.8(12.67)	43.36(12.98)
breast-cancer	35.75 (13.72)	41.69(10.69)	42.5(11.79)	51.2(16.50)	50.71(14.70)
diabetes	37.66 (9.24)	40.34(8.14)	47.8(15.22)	51.9(14.59)	52.87(14.77)
flare-solar	40.29 (10.18)	43.67(6.99)	44.7(6.23)	46.8(4.58)	47.60(4.07)
german	38.42 (11.60)	44.75(12.92)	45.5(14.99)	48.7(12.87)	44.97(3.12)
heart	32.04 (7.62)	34.80(4.83)	35.6(4.77)	62.2(5.94)	46.27(6.00)
image	37.45(4.58)	29.68 (4.15)	35.17(6.51)	42.9(13.54)	37.95(8.74)
ringnorm	29.59 (26.49)	34.01 (25.26)	32.67 (25.30)	39.2(24.93)	60.12(22.85)
splice	42.43 (9.18)	42.01 (8.42)	42.47 (7.91)	51.3(6.84)	48.14(3.94)
thyroid	28.68 (20.25)	47.97(28.05)	44.22(30.08)	36.5(10.73)	28.57 (12.01)
titanic	27.87 (11.24)	47.24(13.98)	48.16(14.31)	53.1(12.95)	48.79(18.59)
twonorm	18.27(5.73)	17.32 (1.65)	17.36 (2.02)	63.4(1.48)	73.14(5.50)
waveform	24.05 (8.24)	25.94(9.06)	25.27 (8.86)	59.9(16.40)	60.68(17.54)
time	1.00	5.40	7.69	4.24	7.60

the expected distortion between x_i and corresponding codebook c_j , but the distribution of the original data and that of codebook vectors might be completely different. Now, we assign a weight for each codebook vector c_j and optimize the weight $\mathcal{W} = \{w_j\}_{j=1}^m$ for the codebook vectors to let the distribution of $\mathfrak{C} = \{\mathcal{C}, \mathcal{W}\}$ be close to that of $\mathfrak{D} = \{\mathcal{D}, \mathcal{U}\}$ as much as possible. We consider the KL divergence between \mathfrak{C} and \mathfrak{D} , which is estimated by the difference between the MQCE (7) and MQEE (9) as:

$$D_{KL}(\mathfrak{C}, \mathfrak{D}) = \frac{d}{n} \sum_{j=1}^m \sum_{i=1}^n w_j \log \varepsilon(c_j, x_i) - \frac{d}{n} \sum_{j=1}^m \sum_{k \neq j} \frac{w_k}{1-w_j} \log \varepsilon(c_j, c_k). \quad (10)$$

In the experiments shown below, we minimized $D_{KL}(\mathfrak{C}, \mathfrak{D})$ with respect to \mathcal{W} by a quasi-Newton method (the BFGS method).

We first show a simple experimental result using artificial data. We sampled data $\mathcal{D}, |\mathcal{D}| = 1000$ from 5 mixture of a 2 dimensional Gaussians with different means ($\mu = (0, 0), (2.5, 0), (0, 2.5), (-2.5, 0), (0, -2.5)$) and the same covariance matrix $\Sigma = \begin{pmatrix} 0.3 & 0 \\ 0 & 0.3 \end{pmatrix}$. The mixture parameters are $(0.1, 0.2, 0.4, 0.1, 0.2)$. The sampled data are shown in figure 2 (a). Then, we applied LVQ to the sampled data \mathcal{D} to get the codebook vectors $\mathcal{C} = \{c_j\}_{j=1}^{35}$ of size $m = 35$ (see figure 2 (b)). Each cluster in figure 2 (a) is compressed to 7 codebook vectors. Then, we optimize the weight $\mathcal{W} = \{w_j\}_{j=1}^{35}$ for codebook vectors, and show the sum of optimized weights correspond to each cluster in figure 2 (c). It is also shown in the same graph that the case of the equal weights for every codebook vectors. From this graph, we see that we could recover the original mixture ratio $(0.1, 0.2, 0.4, 0.1, 0.2)$ by weight optimization.

We next consider more practical use of the weight optimization for codebook vectors. We compress the original data \mathcal{D} to the small sized codebook vectors \mathcal{C} by LVQ, and only store the codebook vectors to conserve the storage resources.

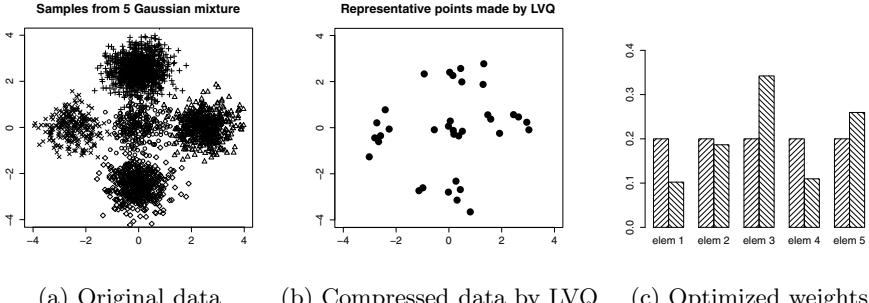


Fig. 2. An example of the distribution preserving data compression. (a): Original data from Gaussian mixture with mixture ratio (0.1, 0.2, 0.4, 0.1, 0.2). (b): Representative points obtained by LVQ. (c): Optimized weights for representative points. The left barplots show equal weight for 5 elements. The right barplots show sum of weights of 7 points for each element.

Table 2. Specifications of the IDA dataset (left), and average and standard deviation of misclassification rates (in percent) of k -NN classification with full data, LVQ compressed data, LVQ compressed data with optimized weights (right).

Data name	specifications				Result		
	dim	# train	# test	# sets	Original	LVQ	LVQ+weight
banana	2	400	4900	100	11.44(0.54)	24.62(4.22)	18.66 (2.44)
breast-cancer	9	200	77	100	27.53(4.213)	26.91(5.01)	26.86(4.83)
diabetes	8	468	300	100	27.02(2.00)	26.64(2.22)	26.33(2.26)
flare-solar	9	666	400	100	35.46(2.01)	39.44(2.58)	35.87 (2.76)
german	20	700	300	100	25.62(2.45)	26.39(2.26)	26.43(2.16)
heart	13	170	100	100	17.55(3.47)	18.30(3.81)	18.43(3.78)
image	18	1300	1010	20	5.17(0.70)	13.10(2.22)	12.40 (1.51)
ringnorm	20	400	7000	100	45.03(1.18)	47.59(1.91)	46.73 (2.37)
splice	60	1000	2175	20	26.43(2.02)	21.21(4.24)	21.58(2.42)
thyroid	5	140	75	200	8.71(2.72)	23.77(4.05)	21.27 (4.43)
titanic	3	150	2051	100	22.91(0.84)	24.61(2.12)	22.79 (0.67)
twonorm	20	400	7000	100	3.42(0.23)	3.26 (0.28)	3.56(0.44)
waveform	21	1000	1000	100	12.09(0.56)	15.73(2.36)	14.27 (1.77)

Thereafter, using the stored data, we classify the new coming data by k -NN classifier. Let x_{te} be the test point, and $\mathcal{C}^{te} = \{\mathcal{C}_1^{te}, \mathcal{C}_2^{te}\}$ be an index set of codebook vectors which consists of k nearest codebook vectors from the test point x_{te} . \mathcal{C}_1^{te} and \mathcal{C}_2^{te} are subsets of \mathcal{C}^{te} correspond to codebook vectors with class label C_1 and C_2 , respectively. The test point is classified as class C_1 if $\sum_{i \in \mathcal{C}_1^{te}} w_i > \sum_{i \in \mathcal{C}_2^{te}} w_i$ and as class C_2 if $\sum_{i \in \mathcal{C}_2^{te}} w_i > \sum_{i \in \mathcal{C}_1^{te}} w_i$. We employ the IDA datasets again for evaluation. The original data are compressed to one-tenth by LVQ and the weight for the codebook vectors are optimized by the BFGS method. In this experiment, we set $k = 7$. From table 2, we see that the k -NN classification accuracy is improved with the proposed method. In the

k -NN classification, there are some test instances which are misclassified if every codebook vectors have the same weights. We infer that some of such instances are correctly classified when we assign appropriate weights for the codebook vectors.

5 Conclusion

In this study, we proposed a computationally efficient estimator for the Shannon information content for weighted data, and extended it to the (cross) entropy and KL divergence estimators. To the authors' knowledge, there is currently no estimation method for the information content with weighted data. In binary classification problem with one class samples, the proposed estimator MQIE is applied to calculate outlier score and it showed superior performance to the conventional one-class SVM with various kernel functions. As another example of applications, we could compress data preserving the original data distribution by optimizing weight in the KL divergence estimate.

Acknowledgements. Part of this work was supported by JSPS Grant-in-Aid for Research Activity Start-up No.22800067.

References

1. Cover, T.M., Thomas, J.A.: Elements of information theory. J. Wiley, Chichester (1991)
2. Hyvärinen, A., Karhunen, J., Oja, E.: Independent Component Analysis. J. Wiley, Chichester (2001)
3. Learned-Miller, E.G., Fisher, J.W.: ICA using spacings estimates of entropy. *JMLR* 4(7-8), 1271–1295 (2004)
4. Wand, M.P., Jones, M.C.: Kernel Smoothing. Chapman & Hall/CRC (1994)
5. Beirlant, J., Dudewicz, E.J., Györfi, L., Meulen, E.C.: Nonparametric entropy estimation: An overview. *International Journal of the Mathematical Statistics Sciences* 6, 17–39 (1997)
6. Kozachenko, L.F., Leonenko, N.N.: Sample estimate of entropy of a random vector. *Problems of Information Transmission* 23, 95–101 (1987)
7. Wang, Q., Kulkarni, S.R., Verdú, S.: Divergence estimation for multidimensional densities via k-nearest-neighbor distances. *IEEE Trans. Inf. Theor.* 55(5), 2392–2405 (2009)
8. Faivishevsky, L., Goldberger, J.: ICA based on a Smooth Estimation of the Differential Entropy. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) *NIPS* 2008, pp. 433–440. MIT Press, Cambridge (2008)
9. Rätsch, G., Onoda, T., Müller, K.R.: Soft margins for adaboost. *Machine Learning* 42(3), 287–320 (2001)
10. Schölkopf, B., Platt, J.C., Shawe-Taylor, J.C., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. *Neural Computation* 13(7), 1443–1471 (2001)
11. Karatzoglou, A., Smola, A., Hornik, K., Zeileis, A.: kernlab - An S4 package for kernel methods in R. *Journal of Statistical Software* 11(9) (2004)
12. Kohonen, T., Hynninen, J., Kangas, J., Laaksonen, J., Torkkola, K.: LVQ PAK:the learning vector quantization program package. Technical report., Helsinki University of Technology (1996)

i f r r

i i i r fr N r N r

ws	d	d				
n	r	o	o	o	0	S

A st t N ra n wor off r goo g n ra a on r or anc no
ro n an o co x con ro ow r n ra n wor a
ng ar x r n r o co ca a a ca nc on a
ar n r n ar o n r an o o rco a on r
x rac on o a n ro o a r r n a no
o o r x rac on w c r c r n a o own ann r
a R c r r D c on D agra agra r c r a
ow ar ng o no w c ar a o rco wo ro r n
n D c on r a r x rac on ro o r r ca
on an o ran ng rag n a on o or r c ng r
arc ac n ng r g on n w c n wor ow ar
a or r n r nar r o o r or anc ar
r or

n u c i n

l	tw	s	ff	d	l	z	t	s	st	ss	d
d	l	l	t	t	l	w	t	t	l	st	t
t	d	st	d			s	s	t	l	ss	t
st	t	s	d	d			t	sl	t	t	ds
	l	tw	s			s	d		ll	l	t
t	ld		d	st	d	l	l	ss	w	s	d
l		s		s	2	2	d	d	w	t	l
t	t			l	t	t	t	ds	s	w	t
t	st		l	tw			d	o	l	2	d
d	t	l		t	l	t	l	st	t	t	l
	t	t			s	d	t	t	l	ss	t
st		l	tw			l	t		t	d	lds
t	d	w	d	w	t		t	sts			d
s	l	s	l	l		l	d	s	s	ffi	t
t	t			s	l	t	l	dd	t	l	s
s	t	s	l	ss		t	s	w		d	s
t	t		s	t	ts	t		t		t	7
o	N		ar	xac		con	on	or	an	con	on
con	on	o	o	N	ar	r			or		an

0 J orow an J ra a

w ds t t ts t s l d s s t t
t l s l s t l t d t t s t t s t t s l t t
st s t t d t d o o o l 2 d d 2 t ds l
s t l t t t t s l ss S st t t
l ss t t l tw d d t s t t s t t s t 5
t t ds ll d l 2 ls s t t s t t s t
l l t s w t dl t d t tw t l t
t s s s ll t w t t tw t l s
s d 2 2 s d t d w t
s l s s d s s d w w l s d d l s t
d t d d d s

An i w n u c i n

ss l t t s s s ts t ts
w ss d t tw t l ss U d t s t s w
l ss t tw s t l s l t l
tw ll ts l t t ts l d s t
t l t t l dst d l l s
t ss t l tw d s t s l t
tw w t l t t t t s s l d s
t d d l s sl l t s ld t s d t
t t ll t t s t s t s s
t t s s d t d st t
t s l d s t t l t t s l
w s l d t t d t d t s t
t w s t s l t d w s t d l d d s t
d d d d s s d d w s l t t d
l tw s s d t d s d d st t
d ll w d t t s l s t s s 5

uc O cisi n i s

d d d d s s l t l t s s d t st t
d s d s l t l t s t d t d t
s l s d S t d s d t t d s l
d t d t dl l st d t s t ll d d d
d d s s t ll s s l t d
s t s t t t d d s w t sts t
l s d d l d s w d t t t l ss d ff s
t t s d t d l s t s t d t d t

o Down In c on o R c r r D c on D agra

s s l s t l s t l t d s t s w
ffi t d t t s s s t s d s t t t
d t st d t s s t s t t t d
s s w t t l l t t s
l s t t t s ll w t st l t t l
s t s l t d ls l t l t l
t tw t s s s s t l t t d t t
d s t d s t t s t t s t ll s d
t s s ll s t t s s ll ll s t t s
t t s d t s l t d l t d w s d s s
d s st t l t t t ddl ts l t l ts d
t t s t s t s t ll w t s
t t s t t s l t w s t d t s z s t d s
s ls s s t t d t s ts t s t l
t d s l t s d l s t s t s d l t s
t d s d t l t d t s d l t s t t t
s s s d t s s s wl d s t t t
s 7 s s ts t s t t d w d d t
l t w t l tw l s s

sc ip i n f A i

t l t t t l ls t t s t t ll l d
tw l ss t s s l l st d s l t t
d s d s d t d w s s w l t
t t t l l t d s l t st t s l tt
t s d t d s d t w t t
t s s l t d l t d s d s d
S d w l s t d d t t s l t d l t t
d s t s l t d s d d s d s l t t
s t t s l t t d s t l d t t l
w st t s d st w st s l tt % d
d s w d t w d t s t t s l d² t
st s w s t t ll d w t w l ss
S d w st s l tt w t tw t t s d t d
d l w d s t tw t t t d s t l s l d
t st d t s t s s t t t st t s ll ws
w d s t t d s s t l t t t s l w s t t s l d
t d ff t s t l t t s l w s t t s l d
st t d l st s s ll ws t t t t t t
t t t d s d t w d s t t t t

In a x r n w a a

J orow an J ra a

r	h	1	d	w	d	t	s						
u		a a	data w		k a	r an	N n	anc					
su		c on	agra ca		ng data								
<i>currentLayer</i> $\leftarrow \{\text{rootNode}(\text{data})\}$													
w	a	no		o	a	r an	r ar	r no	do				
<i>splitFeature</i> \leftarrow		n		a	r								
<i>newLayer</i> $\leftarrow \{\}$													
fo	<i>node</i> \in <i>currentLayer</i> do		<i>newLayer</i> \leftarrow <i>newlayer</i> $\cup \{\text{split}(\text{node}, \text{splitFeature})\}$										
d fo													
fo	<i>node</i> \in <i>newlayer</i> do		c c	r na	on con	on or	<i>node</i> r	o	<i>node</i>				
d fo													
fo	<i>n1, n2</i> \in <i>newLayer</i> , <i>n1</i> \neq <i>n2</i> do		<i>n1, n2</i> = $\underset{n1, n2}{\text{argmin}} \text{distance}(n1, n2)$ and $\text{distance}(n1, n2) < \text{maxdistance}$ do										
co				anc	<i>n1</i> <i>n2</i>								
d fo				rg	<i>n1</i> w	<i>n2</i> r co		anc					
d													
<i>currentLayer</i> \leftarrow <i>newLayer</i>													
d													

$$\text{pathActivation}(\text{path}) = \sum_{f \in \text{ea es es e n a h}} f \cdot \text{Weight}_f.$$

t l t t d s t

2

$$\text{partActivation}(\text{node}) = \text{pathActivation}(\text{diagram root} \rightarrow \text{node}).$$

ll t t l s ll t s w t t t s

$$\text{activation}(\text{leaf node}) = \sum_{f \in \text{ea es}} f \cdot \text{Weight}_f + \text{bias} =$$

$\forall_{\text{node}} \text{pathActivation}(\text{diagram root} \rightarrow \text{node})$

$+ \text{pathActivation}(\text{node} \rightarrow \text{leaf node}) + \text{bias}.$

t	t	d	st	t	ll	d	s	s	l	l	w	l	s
t	s	s t	t	s	tw	d	s	n1	d	n2	s	l	t l
t	t	t s d	s s1	s2	t d	t	n1	d	n2	s	t	l	s ld
s			t	s	n1	w	t	n2	w	d	t	s t	
t					t	n1	d	n2	w	l	s t	d	t t
t	t	s t			t	t l	t	t	s	ll t	d	s	l s t
		d	st	t	s	l	d	s			d	w t	
w	l	ss l	l	s d	st	ll	d	st	t	w s	l	d	t ll
t	s l	d	t t s	d s t	s l	d							
	st	t t	d			t	s l t	s	st	s	t	t t t	
	w dt	t d	t			d	s t	s	l l	l s l	t d		

o Down In c on o R c r r D c on D agra

t	t	s	l s	s	w	d	t	s	l t	t	d	s	t	s
d d	$O(n \cdot k)$	w	n s t			t	s	l s	d k s t					
l		l s	t	t	w	ss	t	t k s s	ll t					
t	w dt	t d		$s O(n)$			d	l	l w	l s t	t			
d s	$O(width^2)$	t	d s l t t	t	t	t	t s l t		$O(f \cdot n)$	t				
w	$f \cdot s t$		t s	t t l	l	t	s t	s	$O(f(f \cdot n + n^2))$					
	$O(f \cdot n^2)$	$f \ll n$												

p i n **su s**

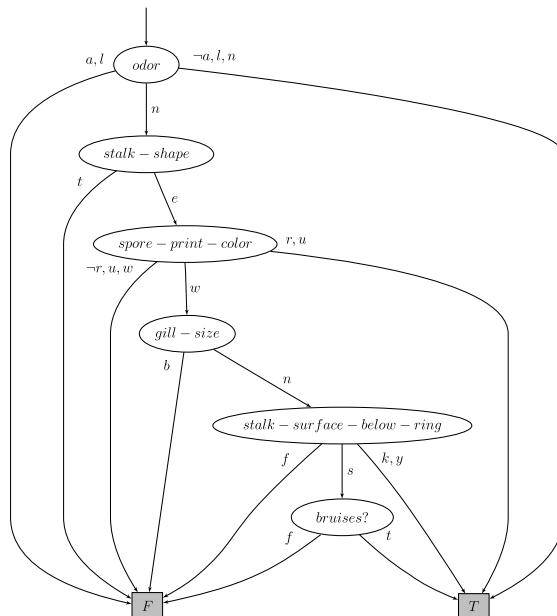
s d	t d	s	t s t d	d t s	ts	t	U	s t						
s d		S t	sts 22	t	t	t		d t	t	s				
d t s t	d t	tt	d t s t	t	d	tl	s	ts	l					
l s		l tt	t s	d s t	t z d	t	s tt	t s s						
t s		l t t		w t d	l t	t s	t							
t t s t		t 2	l s w	s t d	t t	tw	s							
d	t d	tl	ss s	l s s s	t l tt	d t s	t w							
	d l s s s		d 2	w		tt	t s w							
d s s	l s	tl	t s	t d	d w		d t	t						
t t	t s	d t s t	S	l s	t	w	tt	t s						
w	d	t t	d t	t	t	t	t	t						
d w	d l	t	tt	t s	2									
t s t	s d d t s t s	t	ss											

ro r o a a

Da a	ra n	a r	a											
		no	con	o a n	r									
on														
on														
on														
o ng	0													
G r an	000 0													
roo	0		0											
r	0000 0													

l	t	s	d t s t s s w		s l t s									
s t d	l 2	st l	s s w t											
t l	d w t	lt t	s d d		l tw	t								
t	l tw t l		tl s		t s w	t d								
l w	ss l d t		t s l t s	t d s										
t	5	d	s t d l	w t										
t d		l t t l	t w s t	w t t	d ff	t								
l s	t d	t s ld	0.2 0.4	d 0.6	d t t	st								
d st	t s ld	d d s lt			s z	d t	t							

J orow an J ra a



F In c c on agra or roo a a

2 R o x r n

		N wor NN		J		R DD		R DD		ro		NN		R DD	
		wg	acc	acc		acc	acc	r	acc	rg	co n		w o	NN acc	
Da a															
on		0	0	0			0	0	0					0	
on		0	0	0			0	0	0					0	
on		0	0	0			0	0	0					0	
o ng		0	0	0					0					0	
G r an		0	0	0			0	0	0					0	
roo		0	0	0	0	0	0	0	0			0		0	
r	0	0	0	0	0	0	0	0	0					0	

r r or on non cr a a

o Down In c on o R c r r D c on D agra

d s d l t t t d w d t s
tl s l t s w ls t ts
w t s s t s ll s l tw s d s s
s lts s w t t s ll s ll t d s t s w
t d d s w l tw st d t l
s s ll ss d s d t l
w s t d s ll s ll t d s t s w
t t l st t s t d t ll w

C nc usi n

s t d l t t t d s d t l t t s
l st d t l d s t w s d
d s d s w l st t s t d t st t t
l d s t d t t d d s t s l tt
t s d t t t s t l s lts s st t t t
s t d s s l d t s ts s d tt
t st t t d d s t l t d s t s
t ls w l t d d t d t s d t s
t s d t s s s l ss t w t t st t

f nc s

n r n R n n ro c on o nar c on agra c no I n r
o o n ag n http www it p e p e h te i e ht
n r w R D r c J c S r an cr q o c n q or
x rac ng r ro ran ar ca n ra n wor now g a S

r an R Gra a a gor or oo an nc on an a on I
ran ac on on o r 00
orow J ra a J x rac ng r ro nn a c on agra I
ran ac on on N ra N wor 0 i g 9 MN
ra n S a J ng a ng an q r o x rac r ro
ran n ra n wor In roc ng o n In rna ona on r nc
on ac n arn ng organ a ann San ranc co
ra n S a J x rac ng r r c r r r na on o ran n
wor In anc n N ra In or a on roc ng S 0 I
r a r g
c oa r ogona arc a r x rac on or or ran
n ra n wor a rac ca an ffic n a roac I ran ac on on N ra
N wor 00
a a Iran In r a D cr a on o on n o a
r or a ca on arn ng 0 0
ran nc on I ac n arn ng r o or 0 0
http h i e i i e
0 R g n ra on ro n ra n wor I ran ac on on S
an an rn c

J orow an J ra a

a ran o G a r ng r R ann n I
a a n ng o war n a SIG DD x ora on N w
r 0 00
an J a n an n n J ng r x rac on o ro
co r n o r c o c r n 00
http www e e be etew p p b i tie p
o a R o o n c on o o o r a onc c ongra r ng
an a on In roc ng o w Na ona on r nc on r ca
In g nc I r can oca on or r ca In g nc no
ar S o
o a R o c on r gra an o own r n ng In
roc ng o In rna ona Jon on r nc on r ca In g nc
o 0 0 organ a ann r Inc San ranc co
r nan R S a ar G a ac ar a arc c n q or r x
rac on ro ran n ra n wor a rn R cogn on r 0 0

n o a gor an Da a S r c r n SI D gn
n S r ng r r ag N w or Inc S ca c
a n an n n J D c on agra n ac n
arn ng an r ca on r a cr r a a x r S w
ca on 00
ra Sang o ann nc n ng n cr on ng
r nc o n r r c or r c on gra ac n arn ng 0
0 0 0 0
n an J R rogra or ac n arn ng organ a ann San ran
c co
0 S ono R a n R c r n r a n wor r x rac on or
a a w x a r I ran ac on on N ra N wor 0
00
S ono R S o c r r n a on o n r a n wor o r

r n S a a J o orn ra o I n ng J Jong D
D ro S a an S r D a ann R a an r S
onon n o I r g r J ca R c ac ow c R c
a a D n n J ang J on ro
a r or anc co ar on o ff r n arn ng a gor c r

A Framework for Application-Oriented Design of Large-Scale Neural Networks

David Bouchain, Florian Hauser, and Günther Palm

Institute of Neural Information Processing
Ulm University
89069 Ulm
Germany

{david.bouchain,florian.hauser,guenther.palm}@uni-ulm.de

Abstract. Tools for simulations of neural networks exist aplenty. They range from simulators for detailed multi-compartment neurons, over packages for precise reconstruction of small biological networks, to simulators for large-scale networks with stochastic connectivity properties. However, no frameworks for constructing large-scale, dedicated networks exist. Based on the design principles used for our previous work, we introduce a C++ framework which is specifically tailored to simplify the construction of large networks with specific cognitive functionalities.

1 Introduction

Ever since the first neural simulations were performed in the 1950s, simulation tools and frameworks have been created in all varieties. Today's simulators range from biophysical single-neuron simulation with great physiological detail in multi-compartment neuron models (e.g. [2]), over tools for the creation of small networks with reasonable biological detail (e.g. NEURON [6] or BIOSIM [1]), to simulators for analysis of stochastic properties of large-scale neural networks (NeST, [4]). Simulators like BIOSIM are especially suited for the creation of networks with dedicated functionality, in order to simulate real networks found using intracellular recordings from various animals. Spiking neural network simulators like NeST are used for the reproduction of stochastic spike patterns in large networks. But there are no simulators for large-scale networks with specific connection matrices. Networks with thousands or even tens of thousands of neurons have connection matrices so large that constructing these matrices on synapse level is a hopeless undertaking.

We introduce a novel simulation framework which is especially designed for the construction of large-scale dedicated networks, i.e. large-scale neural networks with specific cognitive functionalities. Our previous simulations used this task-oriented approach to construct large-scale networks that are able to parse sentences adhering to a simple regular grammar ([7]). Here, we explain the concept used in those simulations on a more general level in order to provide the rationale for creating a new simulation framework.

2 Pattern-Based Networks

The deterministic construction of the connection matrices in large-scale networks requires a series of assumptions about the functionality of cortical networks. The most important one of these assumptions is that cortical networks are neural associative memories with binary synapses and sparse patterns. This paradigm has been introduced in [8] and served as a foundation for the language understanding network mentioned above. This pattern-based type of network construction implies that the neural activity isn't determined by single synaptic connections, but rather by the auto associative mapping between the neurons of a pattern. The abstraction from single synaptic connections to pattern-based connectivity schemes is an abstraction from Hebbian cell assemblies ([5]) and serves as the fundamental design principle of our networks.

The second abstraction stems from the view of the cortex as a set of areas which are interconnected through excitatory long-range connections ([3]). The graph which is formed by viewing the areas as nodes and the (bidirectional) long-range connections as edges forms the basis of the network design.

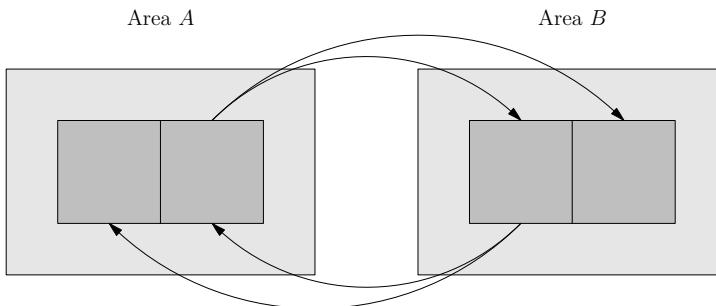


Fig. 1. Aspect-based interconnection scheme between two areas. Both areas have one pattern, each of which consists of two aspects. One aspect in area A projects onto the whole pattern in area B . Similarly, one aspect in area B projects onto the whole pattern in area A .

The simplest such network, shown in Fig. 2 consists of two bidirectionally connected areas A and B . Our pattern-centric view is that activity in area A represents an aspect of the activity in area B , and vice versa. We know that most neurons project an axon (long-range connection) to only one other area ([3]). Thus, if there is a third area C , which is also bidirectionally connected with area A , the set of neurons in area A which project onto B is disjoint from the set of neurons in A which project onto C .

The subgroup of neurons of one pattern which all project long-range connections into another area is what we call an *aspect*¹. An aspect is the fundamental

¹ Technically, this is an oversimplification. Correct would be “aspect instance”, where an aspect could be “color”, and the aspect instance would be “blue”.

unit of computation in our networks. Each aspect can be part of arbitrarily many patterns, and each pattern, all of which are the same size, can consist of a certain number of aspects.

The network design consists of three major steps. First, one needs to design the long-range connectivity graph in order to determine the areas needed in the system. The second step is to determine how the aspects and patterns are to be stored in the areas. The third step is to decide how to map aspects and patterns between two connected areas.

The simplest and most straightforward method of pattern storage is auto association. In order to incorporate spatiotemporal aspects of cortical processing, however, hetero association within areas is needed. This can be accomplished by defining maps between patterns, which are then stored as hetero associative connections between the corresponding neurons. The long-range connectivity graph defines semantic relationships between areas and is not affected by the choice of pattern storage method.

Our previous work with this kind of construction method resulted in a large network of about 20 areas, which was able to syntactically parse and semantically understand input sentences if they complied to a simple regular grammar ([7]). A graphical representation of the network activity at some point in the simulation is shown in Fig. 2. The primary goal of the framework we introduce here is a generalization of the construction principles used, as well as a simplification of the pattern specification.

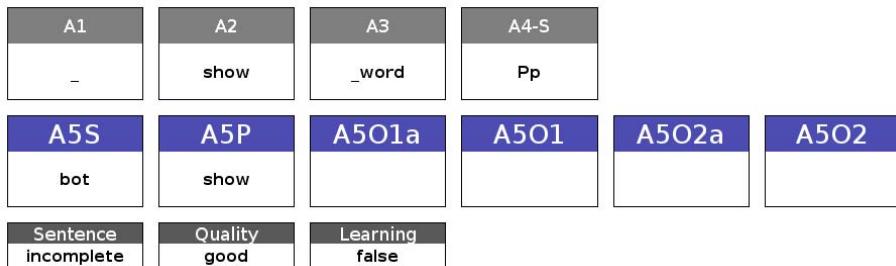


Fig. 2. A graphical representation of network activity during simulation of the network presented in [7]. The boxes represent areas, and the words show all patterns which have a certain minimum amount of their neurons activated.

3 A C++ Framework for Network Construction

We now introduce the framework which we have developed specifically for the construction of such large-scale pattern-based networks. The design of the library revolves around *population templates*, *area templates* and a *patch template*.

3.1 Population model

Populations are created using the following model. Each population has a certain number of neurons of one type, as well as at least one matrix type for external input, and optionally one feedback matrix. For each external matrix type, the number of matrices of this type are specified as a template parameter. If the feedback matrix is present, another template parameter specifies the internal population delay in time steps. We provide templates for up to three different input matrix types. The templates in this group are accordingly named `one_forward`, `one_auto`, `two_forward`, `two_auto`, `three_forward`, and `three_auto`. For example, a population type with one external matrix type and a feedback matrix with delay 1 might be defined like this:

```
typedef one_forward< my_neuron, my_input_synapse, 2,
                    my_auto_synapse, 1 > my_population_type;
```

The population templates provide the core functionality of our neural simulations, because they are instantiated with neuron and synapse models that provide the dynamics calculations and spike generation mechanisms. A neuron model is a collection of one data structure (neuron type) and two function objects which both act on the data structure. The first of these function objects determines the spike generation mechanism and is a `struct` containing a method with the following signature, which should return `true` if a spike is generated:

```
template< typename Neuron, typename Params >
bool operator()( Neuron&, Params const& );
```

The `Params` parameter is used to pass a global parameter `struct`, which contains all parameters needed by the function objects, to all simulation algorithms. The second function object provides the neuron dynamics calculation and is a `struct` containing again the `operator()` with a slightly different signature:

```
template< typename Neuron, typename Params >
void operator()( Neuron&, Params const& );
```

A synapse model consists of one data structure (the synapse type) and three function objects which all act on a synapse and a neuron, and are defined using a `struct` containing an `operator()` with the following signature:

```
template< typename Synapse, typename Neuron, typename Params >
void operator()( Synapse&, Neuron&, Params const& );
```

The first function object determines the synaptic dynamics calculation. The second function object, called the `pre_spike_type`, defines what happens to a specific neuron and a specific synapse if the synapse receives a presynaptic spike. The third function object, called the `post_spike_type`, defines what happens to a specific neuron and a specific synapse if the neuron generates a postsynaptic spike. Each population is instantiated with exactly one neuron model, and one or more synapse models, one for each input matrix type, and one for the optional feedback matrix.

Using this kind of population model, we can simulate all kinds of spiking neural and synaptic dynamics, from McCulloch-Pitts models to detailed spiking networks with synaptic plasticity. The advantage of requiring the models to be specified as template parameters is that user-defined models can be used transparently. Additionally, it enables the C++ compiler to apply various optimizations, especially inlining. The models are fixed at compile time and can't be changed afterward.

3.2 Area Templates

One or more populations can then be combined to create an area, which handles the spike pattern communication between the populations. For example, if we build a simple associative memory, areas need only one neuron population with a feedback matrix. If we want to be able to specify inhibition, we need at least two populations in an area. We have so far implemented two area templates: `single_area`, which needs the neuron model, the external synapse model and the feedback synapse model as well as the internal delay as template parameter. And `ex_in_area`, which contains two populations (excitatory and inhibitory) and in addition needs the inhibitory neuron model and the inhibitory synapse type as template parameters. A corresponding instantiation might look like this:

```
typedef ex_in_area< my_ex_neuron, my_ex_synapse,
                    my_in_neuron, my_in_synapse,
                    4, // number of external matrices
                    1 > my_area_type;
```

3.3 Patch Template

The `patch` template can be used to combine areas into a modular network and provides the long-range connectivity infrastructure. Areas can be created at run time and are stored using a readable name. The `patch` template is constructed in a transparent manner, so that areas can be directly manipulated through their own interface. The template parameters are the area type and the long-range connection delay. A corresponding definition might look like this:

```
patch< my_area_type, 10 > my_patch;
```

Areas can be created and connected:

```
my_patch.create_area( "A" );
my_patch.create_area( "B" );
my_patch.connect_bidirectional( "A", "B" );
```

Finally, a simulation step can be executed using the `update()` method and the parameter structure:

```
parameters p;
my_patch.update( p );
```

Note that our framework doesn't automatically provide any kind of output. Simple text-based output can be generated by reading the output pattern of a specific population, in this case the excitatory population of area A , and using standard stream operators for console output:

```
my_patch.select_area( "A" );
std::cout << my_patch.selected_area().output( 0 ) << std::endl;
```

3.4 Pattern Storage

The framework provides a class called `area_description` which can be used to provide a logical description of aspects and patterns as well as their inter relations. Another class, `pattern_indexes` is constructed by the global function `resolve()` to create an allocation of neuron indexes to which the logical patterns are assigned. For example:

```
area_description ad;
ad.aspect( "a" ); // define an aspect
// ...
ad.pattern( "A" ); // define a pattern
// ...
ad.pattern_map( "A", "B" ); // local map definition
pattern_indexes pi;
resolve( ad, pi,
        4, // neurons per aspect
        5 ); // aspects per pattern
```

Various methods are provided for storing a set of aspects and patterns in an area. Patterns or aspects, or both, can be simply allocated (which means that the area is enlarged according to the number of neurons needed) or stored as auto associations. If the area type has an inhibitory population, inhibitory interneurons can be allocated (one per aspect) which either inhibit the whole network, or all neurons except for the aspect currently being stored.

Maps can be constructed by using simple hetero association, or by additionally removing the inhibitory connections between the mapped patterns. For example:

```
construct_auto_patterns_inhibit_other(
    pi, my_patch.selected_area() );
embed_chain_clear_inhibition(
    pi, my_patch.selected_area() );
```

When the areas are constructed using these functions, patterns can be activated by artificially invoking the spike generation mechanism in a population:

```
my_patch.selected_area().activate( 0, pi.patterns[ "A" ] );
```

4 A Simpler Description

Usually, one would not want to specify the patterns and chains at compile time. Therefore, we're currently developing a small regular description language which can be used to construct networks at run time, and whose features map directly onto the features explained above.

This language consists of area and cortex definitions. Areas contain aspects, patterns and chains. Aspect definitions can optionally depend on a pattern. For example, an aspect *a* can be defined as a subpattern of pattern *A* like this:

```
pattern A;
aspect a( A );
```

A pattern can also have one or more source aspects, which means that the pattern is a super set of the aspects:

```
aspect b, c;
pattern B[ b, c ];
```

Areas can also contain named chain definitions, which is an abbreviation for a series of pattern maps:

```
chain one: A - B - C;
```

Cortex definitions instantiate area types and define maps between patterns, aspects, or chains using the long-range connections. A complete description file might look like this:

```
area some_area {
    pattern one, two, three;
    chain test: one - two - three;
}

cortex example {
    some_area A, B;
    map A.test B.test;
    map B.test A.test;
}
```

5 Conclusion and Future Work

We have introduced a C++ library which serves as a fundamental platform for constructing large, dedicated networks. These networks are constructed by pattern specification and automatic resolution into neuron index allocations.

The abstraction from single synaptic connections to pattern-based network construction provided by our framework and the description language for constructing such networks are fundamental steps towards the ability to rapidly

construct large-scale dedicated networks designed for specific cognitive functionalities. There is currently no other simulation framework with this pattern-based approach. Additionally, the design as a template library without tight integration into graphical display methods makes it suitable for use within larger applications, which use these cognitive networks as part of their core functionality, e.g. for robot control.

References

1. Bergdoll, S., Koch, U.T.: BIOSIM — A biological neural network simulator for research and teaching, featuring interactive graphical user interface and learning capabilities. *Neurocomputing* 8, 93–112 (1995)
2. Bower, J.M., Beeman, D.: *The Book of GENESIS: Exploring Realistic Neural Models with the General Neural Simulation System*. Springer, Heidelberg (1998)
3. Braitenberg, V., Schütz, A.: *Anatomy of the Cortex*. Springer, Heidelberg (1991)
4. Gewaltig, M.-O., Diesmann, M.: *Nest*. Scholarpedia 2, 1430 (2007)
5. Hebb, D.O.: *The Organization of Behavior*. Wiley, Chichester (1949)
6. Hines, M.L., Carnevale, T.N.: The neuron simulation environment. *Neural Computation* 9, 1179–1209 (1997)
7. Markert, H., Knoblauch, A., Palm, G.: Modelling of syntactical processing in the cortex. *BioSystems* 89, 300–315 (2007)
8. Palm, G.: *Neural Assemblies: An Alternative Approach to Artificial Intelligence*. Springer, Heidelberg (1982)

A Dynamic Field Model of Ordinal and Timing Properties of Sequential Events

Flora Ferreira^{1,*}, Wolfram Erlhagen^{1,3}, and Estela Bicho²

¹ Dept. Mathematics, University of Minho, 4800-058 Guimarães, Portugal
fjferreira@dei.uminho.pt, wolfram.erlhagen@math.uminho.pt

² Dept. Industrial Electronics, University of Minho, 4800-058 Guimarães, Portugal
estela.bicho@dei.uminho.pt

³ Donders Institute, BCB, Radboud University Nijmegen, The Netherlands

Abstract. Recent evidence suggests that the neural mechanisms underlying memory for serial order and interval timing of sequential events are closely linked. We present a dynamic neural field model which exploits the existence and stability of multi-bump solutions with a gradient of activation to store serial order. The activation gradient is achieved by applying a state-dependent threshold accommodation process to the firing rate function. A field dynamics of lateral inhibition type is used in combination with a dynamics for the baseline activity to recall the sequence from memory. We show that depending on the time scale of the baseline dynamics the precise temporal structure of the original sequence may be retrieved or a proactive timing of events may be achieved.

Keywords: Dynamic Field Model, Serial Order, Interval Timing, Prefrontal Cortex.

1 Introduction

Virtually every aspect of our everyday routine actions is embedded in a sequential context. We see this in tasks like getting dressed, playing games, setting the dinner table or cooking a meal. The capacity to hold the temporal order of a short sequence of events in memory is of primary importance to our ability for efficient high-level planning when precise order of what has just happened and what is about to happen is of the essence. Very often a fluent and successful task execution requires not only a judgment about the ordinal sequence structure but also a metrical judgment that involves the analysis of elapsed time. Our ability to adjust behavior to temporal regularities in the environment in the range of seconds or minutes is called interval timing [1]. It manifests for instance when we start preparing an action in anticipation of a salient perpetual event that will happen in the nearer future.

Experimental evidence from physiological and behavioral studies suggests that the neural mechanisms supporting both ordinal and interval properties of time

* The work was supported by the PhD Grant SFRH/BD/41179/2007 from FCT.

are closely related (for discussion see e.g. [2,3,4]). It has been postulated that temporally predictable changes in self-sustained activity of prefrontal “delay” cells which are known for their role in working memory and action planning may be used by the brain to measure elapsed time [5].

In this paper we present a computational model based on dynamic neural fields [6] that aims at testing the idea of shared mechanisms for temporal and ordinal coding of sequential processes. More concretely, the model builds on three key neuroscientific findings. 1) Averbeck et al. [7] described firing patterns of cells in prefrontal cortex (PFC) in a sequential task that are consistent with the notion of a parallel processing of planned serial movements (see also [8]). They reported that prior to sequence onset neural population activity reflected each of all forthcoming goal-directed actions with a pre-activation gradient encoding serial order. 2) Various studies that directly investigated the neural basis of timing report ensemble activity showing a monotonic relationship between peak activity and elapsed time [3,9]. 3) Separate subpopulations in PFC represent intended future goal-directed actions and already accomplished goals [10].

Dynamic neural field models explain the capacity to hold an item of information “on-line” in short term memory as the result of strong recurrent interactions within neural populations that can sustain a persistent “bump” activity in the absence of external input [6,11,16]. The model extends previous mathematical results on the existence of multiple bumps of equal strength [12] to implement a working memory of sequential events in which varying levels of self-sustained peak activity are correlated with the relative position of each item.

In what follows, we begin with a description of the model consisting of three coupled neural fields. We discuss mathematical details of the model in section 3. A report of a series of simulation results is given in section 4, followed by a discussion of results and related work.

2 Model Description

In the experiments designed to test the neural processing of serial order in goal-directed behavior typically simple reaching or saccadic eye movements towards objects are used. The temporal order may be defined by the spatial object position (that is, movement direction and amplitude) or object features like color, weight or size. The central idea of dynamic field models is that suprathreshold population activity of neurons tuned to these continuous dimensions represents specific parameter values. Fig. 1 presents an overview of the model architecture which is inspired by the experimental findings summarized in the Introduction. The self-sustained activation pattern in field u_{SM} stores all items of a sequence with a strength of activation that decreases from item to item as a function of elapsed time since sequence onset. This activation gradient is achieved by combining a field dynamics that guarantees the evolution of a single, self-stabilized bump in response to a localized transient input representing a perceived event with a state-dependent threshold accommodation dynamics for the firing rate function [13]. Through excitatory connections, neurons in u_{SM} project their

activation to corresponding neurons in field u_{SR} . This input leads to a sub-threshold pre-activation of neural populations that mirrors the primacy gradient of strengths in u_{SM} . Sequence recall starts with a continuous increase of the baseline activity which brings neural populations closer to the threshold for the evolution of a self-stabilized bump. The order and timing of sequence elements can be retrieved by using the baseline dynamics to first trigger a suprathreshold response of the population with the highest pre-activation which then becomes suppressed due to inhibitory feedback from field u_{PE} . Self-stabilized population activity in this field is initially driven by the activation dynamics of the corresponding population in u_{SR} and may thus be described as a memory for already achieved events or goals.

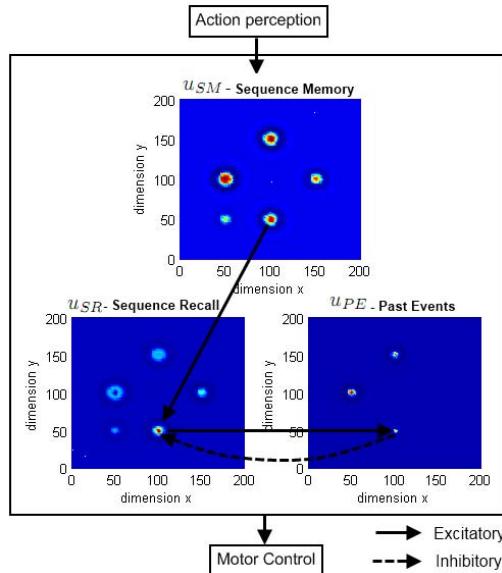


Fig. 1. The architecture of the field model

3 Model Equations

The dynamics in the sequence memory field is governed by the following equation [6]:

$$\tau_{u_{SM}} \dot{u}_{SM}(x, t) = -u_{SM}(x, t) + h_{SM}(t) + \int w(x - x') f(u_{SM}(x', t)) dx' + S(x, t) \quad (1)$$

where $u_{SM}(x, t)$ represents the activity at time t of a neuron encoding dimension x . The parameter $\tau_{u_{SM}} > 0$ defines the time scale of the field. The firing rate function f is taken as the Heaviside step function with threshold 0. $S(x, t)$

represents the time-dependent, localized input to the field. It is taken as a Gaussian centered at position x with amplitude $S_{strength} > 0$ and standard deviation $\sigma_{input} > 0$. The connection function $w(x)$ determines the coupling between neurons within the field. It is well known that coupling function of lateral inhibition type used by Amari in his original work (1977) do not support stable multi-bump solutions [12,16]. Using a coupling function with oscillatory rather than monotonic decay, Laing et al (2002) have shown that multiple regions of suprathreshold activity can persist in a field. To model working memory for several events separated in time we adapt the connection function w used in [12]:

$$w(x) = Ae^{-b|x|} (b \sin |\alpha x| + \cos(\alpha x)), \quad (2)$$

where the parameter $b > 0$ controls the rate at which the oscillations in w decay with distance. The parameters A and α are added to control the amplitude and the zero crossings of w . The variable $h_{SM}(t)$ defines the baseline level of activation which we chose to vary with time. Note that by including $h_{SM}(t)$ in the definition of the firing rate function $f = f(u - h_{SM}(t))$ it becomes clear that changing the baseline level is equivalent to changing the threshold of f . Following the idea of the phenomenological model for threshold accommodation in dynamic fields discussed by Coombes and Owen [13], we apply the following state-dependent dynamics for the baseline activity:

$$\dot{h}_{SM} = (1 - g(u_{SM}(x, t))) (-h_{SM} + h_{SM_0}) + kg(u_{SM}(x, t)), \quad (3)$$

where g is chosen as the Heaviside step function, h_{SM_0} defines the level to which h_{SM} relaxes without suprathreshold activity at position x and $k > 0$ measures the growth rate when it is present. As the result of the baseline or threshold dynamics a primacy gradient of strengths is establish when at different points in time bumps evolve in response to external input. The standard deviation σ_{input} is chosen to guarantee that the population evolves a single localized activity pattern centered at the stimulated field position. A minimum spatial distance between successive input presentations ensures that a multi-bump solution may stabilize.

The dynamics of the “sequence recall” field u_{SR} and the “past events” field u_{PE} are governed by the following equations, respectively:

$$\begin{aligned} \tau_{u_{SR}} \dot{u}_{SR}(x, t) = & -u_{SR}(x, t) + h_{SR}(t) + \int w_{SR}(x - x') f(u_{SR}(x', t)) dx' \\ & - \int w(x - x') f(u_{PE}(x', t)) dx' + u_{SM}(x), \end{aligned} \quad (4)$$

$$\begin{aligned} \tau_{u_{PE}} \dot{u}_{PE}(x, t) = & -u_{PE}(x, t) + h_{PE} + u_{SR}(x, t) f(u_{SR}(x', t)) \\ & + \int w(x - x') f(u_{PE}(x', t)) dx'. \end{aligned} \quad (5)$$

Since like u_{SM} also u_{PE} has to represent multi-bumps as stable solution we use the same connection function (2). The baseline activity $h_{PE} < 0$ is constant. The situation is different for u_{SR} where a single localized activity pattern represents

a particular event during sequence recall. To ensure the existence of a 1-bump solution we use a connection function of lateral inhibition type given by:

$$w_{SR}(x - x') = w_{excite} e^{\left(-\frac{(x-x')^2}{2\sigma_{excite}^2}\right)} - w_{inhib}. \quad (6)$$

The baseline activity $h_{SR}(t)$ evolves continuously in time described by the equation:

$$\tau_{h_{SR}} \dot{h}_{SR} = 1, \quad h_{SR}(t_0) = h_{u_{SR_0}} < 0, \quad (7)$$

where $\tau_{h_{SR}}$ controls the growth rate of h_{SR} .

4 Simulations Results

Fig. 2 shows a snapshot of a simulation of a one-dimensional version of the sequence model. To give a concrete example, the order of a sequence of goal-directed reaching movements may be defined by object color. There are 5 reaching movements to different objects that are successively executed. In the model, the series of localized inputs to the “sequence memory” field triggers a self-stabilized pattern consisting of 5 peaks (Fig.2, left). Due to the threshold accommodation dynamics (dashed line) the peak amplitudes reflect the temporal order of events. Fig.2 (right) shows the activation patterns in the “sequence recall” field (solid line) and the “past events” field (dashed line) at a time when all representations in u_{SR} are below threshold because the activation peak at $x = 100$ in u_{PE} has just suppressed the representation of the first event and the representation of the second event at $x = 60$ is just about to reach the threshold.

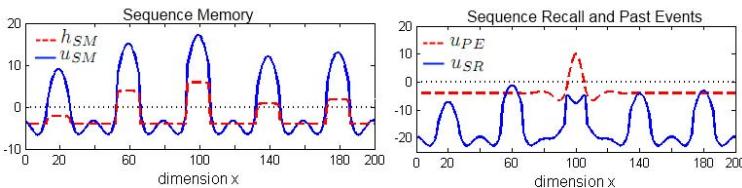


Fig. 2. A snapshot of a simulation of a one-dimensional version of the sequence model is shown. The following parameters value were used: For u_{SM} , $\tau_{u_{SM}} = 20$, $h_{SM_0} = -4$, $k = 0.01$, $S_{strength} = 8$, $\sigma_{input} = 4$, $x_{input} \in \{20, 60, 100, 140, 180\}$. For u_{SR} , $\tau_{u_{SR}} = 20$, $\tau_{h_{SR}} = 100$, $h_{SR_0} = -17$, $w_{excite} = 16$, $\sigma_{excite} = 4$ and $w_{inhib} = 0.01$. For u_{PE} , $h_{PE} = -4$ and $\tau_{u_{PE}} = 40$. The connection function parameters $b = 0.15$ and $\alpha = 0.3$ were equal for all fields, $A = 2$ for u_{SM} and $A = 3$ for the others fields.

To directly compare the timing of events during encoding and recall, Fig. 3 compares the time courses of the population activity in the “sequence memory” field (top) and the “sequence recall” field (bottom) for two different choices of the time scale for the baseline dynamics in u_{SR} . Time $t = 0$ indicates the start of the sequence and the first event (i.e., first object reached) is perceived at

about $t = 200$. Note that the events are irregularly spaced in time. As can be seen in the bottom figures, all subpopulations in the recall field appear to be from the beginning on pre-activated with a strength reflecting the rank order of execution. This model behavior reflects nicely the main findings about parallel processing of serial order in the neurophysiological study of Averbeck and colleagues (2002). If the time scale of the baseline dynamics is chosen as inversely related to the parameter k controlling the growth rate of the threshold accommodation dynamics, $\tau_{h_{SR}} = 1/k$, the recall dynamics nearly perfectly reproduces the timing of events (bottom left). If the time scale for the baseline dynamics is chosen to be faster, $\tau_{h_{SR}} < 1/k$, proactive timing of events can be observed (bottom right). The total execution time decrease but the proportion of total time for each event of the sequence remains essentially invariant. It is beyond the scope of this article to compare model prediction directly with theoretical and experimental results about interval timing. It is however worth mentioning that the model behavior is in line with key principles [1]. When people are asked to speed up or slow down the execution of a movement sequence they do so with near constancy in the relative timing. Moreover, assuming that noise may affect the growth rate of the baseline shift from trial to trial the model predicts that the variability of time estimation grows proportionally with interval duration.

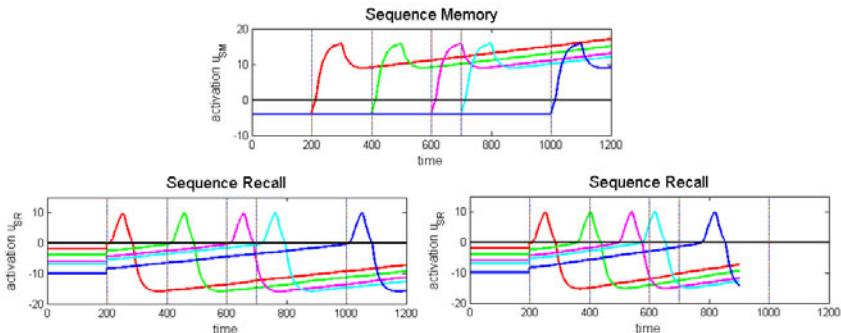


Fig. 3. The time course of the maximal activation of each subpopulation in Fig. 2 (left) is shown for the sequence encoding (top) and the sequence recall (down) fields. For the recall $\tau_{h_{PE}} = 100$ (left) and $\tau_{h_{PE}} = 70$ (right) were chosen.

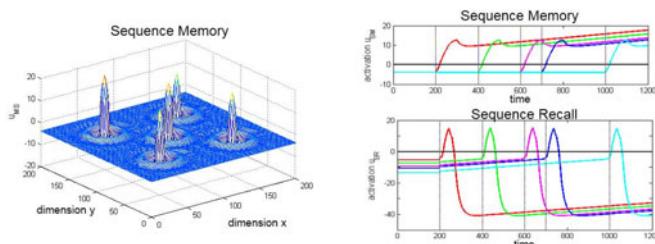


Fig. 4. Example of a model simulation in two dimensions. Self-stabilized bumps in u_{SM} (left) and the time course of activity in u_{SM} and u_{SR} (right) are shown.

It is straightforward to generalize the model to higher dimensions. Fig.4 shows an example of dynamic fields spanned over two dimensions representing for instance movement direction and amplitude. Stable multi-bump solutions in response to transient inputs exist for the two-dimensional analogue of the coupling function (2).

5 Discussion

We have presented a dynamic field model of sequential events that implements the idea of closely related neural systems for controlling the interval and ordinal dimensions. Serial order is stored in working memory by assuming that memory strength for each event decreases as a function of elapsed time between sequence onset and the event. During recall, the ordinal and temporal structure is recovered from the memory list by applying a simple dynamics for the baseline activity of the decision field. An interesting feature of the model is that a speeding up of the baseline dynamics leads to a proactive timing of events. For a cognitive agent such a mechanism may be important for instance to timely prepare the next action or to allocate attention. The field model shares key features like parallel response activation and activation gradient with “Competitive Cuing” models [14,15] that have been applied to a wide variety of serial order problems (mostly concerning the ordinal dimension). Compared to connectionist implementations, the dynamic field approach offers advantages because it allows us to rigorously understand the existence and stability of activation patterns and their dependence on external inputs [6,12,13]. This understanding may guide the development of complex cognitive models.

One of our future goals is the validation of the sequence model as part of an existing dynamic field for human-robot interaction [17]. In the context of robotics applications it is important to stress that due to the self-stabilized properties of the field dynamics the model runs autonomously without feedback from the environment. This can be used by the robot for instance to “mentally” simulate the timing of sequential events. However, it is also possible and technically straightforward to include in the model sensory feedback as additional input necessary to trigger event representations in u_{SR} or u_{PE} [18].

In the present implementation the field model does not allow us to handle repeated items in a sequence. One possibility to overcome this limitation is to postulate the existence of a dynamic control signal that allows the activation gradient to vary during the course of sequence generation [15]. A neural substrate for such a control signal is not known and it would require an extra learning process to establish links between the signal and the sequence elements. For a dynamic field model a more parsimonious solution would be to add time as a continuous dimension along which items may be discriminated, just like perceptual or motor dimensions [19]. The existence of neural populations that are tuned to specific intervals of elapsed time might be interpreted as supporting this view [9].

References

1. Machado, A., Malheiros, M.T., Erlhagen, W.: Learning to Time: A Perspective. *J. of the Experimental Analysis of Behavior* 92, 423–458 (2009)
2. Dominey, P.F.: A shared system for learning serial and temporal structure of sensori-motor sequences? Evidence from simulation and human experiments. *Cognitive Brain Research* 6, 163–172 (1998)
3. Janssen, P., Shadlen, M.N.: A representation of the hazard rate of elapsed time in macaque area LIP. *Nature Neuroscience* 8(2), 234–241 (2005)
4. Staddon, J.E.R.: Interval timing: memory, not a clock. *Trends in Cognitive Sciences* 9, 312–314 (2005)
5. Lewis, P.A., Miall, R.C.: Remembering the time: a continuous clock. *Trends in Cognitive Sciences* 10, 401–406 (2006)
6. Amari, S.: Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics* 27, 77–87 (1977)
7. Averbeck, B.B., Chafee, M.V., Crowe, D.A., Georgopoulos, A.P.: Parallel processing of serial movements in prefrontal cortex. *Proc. Natl. Acad. Sci.* 99, 13172–13177 (2002)
8. Tanji, J., Shima, K., Mushiaki, H.: Concept-based behavioral planning and the lateral prefrontal cortex. *Trends in Cognitive Sciences* 11, 528–534 (2007)
9. Genovesio, A., Tsujimoto, S., Wise, S.P.: Neuronal activity related to elapsed time in prefrontal cortex. *J. Neurophysiology* 95, 3281–3285 (2006)
10. Genovesio, A., Brasted, P.J., Wise, S.P.: Representation of future and previous spatial goals by separate neural populations in prefrontal cortex. *J. Neuroscience* 26, 7305–7316 (2006)
11. Erlhagen, W., Schöner, G.: Dynamic field theory of motor preparation. *Psychological Review* 109, 545–572 (2002)
12. Laing, C.R., Troy, W.C., Gutkin, B., Ermentrout, G.B.: Multiple bumps in a neuronal model of working memory. *SIAM J. on Applied Math.* 63, 62–97 (2002)
13. Coombes, S., Owen, M.R.: Exotic dynamics in a firing rate model of neural tissue with threshold accommodation. *AMS Cont. Math.* 440, 123–144 (2007)
14. Grossberg, S.: Behavioral contrast in short term memory: Serial binary memory models or parallel continuous memory models. *J. Math. Psych.* 17, 199–219 (1978)
15. Houghton, G.: The problem of serial order: A neural network model of sequence learning and recall. In: Dale, R., Mellish, C., Zock, M. (eds.) *Current Research in Natural Language Generation*, pp. 287–319. Academic Press, London (1990)
16. Laing, C., Troy, W.: Two-bump solutions of Amari-type models of neuronal pattern formation. *Physica D* 178(3), 190–218 (2003)
17. Bicho, E., Louro, L., Erlhagen, W.: Integrating verbal and non-verbal communication in a dynamic neural field architecture for human-robot interaction. *Front. Neurorobot.* 4, 5 (2010), doi:10.3389/fnbot.2010.00005
18. Sandamirskaya, Y., Schöner, G.: An embodied account of serial order: How instabilities drive sequence generation. *Neural Networks* 23(10), 1164–1179 (2010)
19. Farrell, S., McLaughlin, K.: Short-term recognition memory for serial order and timing. *Memory & Cognition* 35, 1724–1734 (2007)

Robot Trajectory Prediction and Recognition Based on a Computational Mirror Neurons Model

Junpei Zhong, Cornelius Weber, and Stefan Wermter

Department of Computer Science, University of Hamburg,
Vogt Koelln Str. 30, 22527 Hamburg, Germany
{zhong,weber,wermter}@informatik.uni-hamburg.de
<http://www.informatik.uni-hamburg.de/WTM/>

Abstract. Mirror neurons are premotor neurons that are considered to play a role in goal-directed actions, action understanding and even social cognition. As one of the promising research areas in psychology, cognitive neuroscience and cognitive physiology, understanding mirror neurons in a social cognition context, whether with neural or computational models, is still an open issue [5]. In this paper, we mainly focus on the action understanding aspect of mirror neurons, which can be regarded as a fundamental function of social cooperation and social cognition. Our proposed initial architecture is to learn a simulation of the walking pattern of a humanoid robot and to predict where the robot is heading on the basis of its previous walking trajectory.

Keywords: Recurrent Neural Network, Parametric Bias, Mirror Neurons, Robot Walking Pattern.

1 Introduction

Mirror neurons are a kind of premotor neurons that exist in primates (e.g. monkeys). They have been detected in monkey's premotor area (F5), but the existence of mirror neurons in humans is also evident in the premotor cortex and the inferior parietal lobule (IPL) [8]. According to the experiments done with monkeys, mirror neurons are involved in learning by imitation and social cognition. Moreover, in cooperation with environment affordances, they have the ability to indicate motor actions by activating corresponding neurons in the superior temporal sulcus (STS), indicating the code of action [10].

One class of computational models of mirror neurons focuses on the action imitation property, for instance, the modular action approaches by Demiris [4,3] and Wolpert [18,9]. These models emphasize *how* to generate motions by decentralized automatic modules of the action parts using mirror neurons. The core of these architectures are multiple forward-inverse models, which compete for control based on the selection of likelihoods to imitate actions. Also the selection of a controller can be regarded as action recognition to some extent.

With the further discovery that the mirror neuron system takes part in the recognition of *what* the agent is doing by both the observations from the executors the executions of own actions by the primate itself, further implementations refer to the characteristics incorporating different network patterns to deal with multimodal inputs, in parallel to the vision, somatosensory or auditory stimuli in the human brain, so that the network discharges as a goal-directed action layer. This idea was realized by an association network based on the Helmholtz machine, in which goal-directed codes were associated with vision and language representations, as well as the output of motor actions [7,17]. The learned association enabled neurons of the hidden layer to behave like mirror neurons.

Another network model called RNNPB advocated by Tani et al. [15] can generate and recognize temporal actions by the self-organizing property of an additional layer, called parametric biases units (PB Units). This model deals with temporal inputs with the PB units acting as additional biases to compensate the back-propagation error (Figure 1(a)). Three running modes (learning, observing and action generation) functionally simulate different imitation stages of mirror neurons in the human brain. With the ability of recognizing and generating the desired action from previous training, the PB units behave in a similar manner as mirror neurons in human brain [2].

Recent research [16,11] further suggests that it is not necessary to split *how* and *what*. In fact, as an integrated process, all mirror neurons receive information from visual inputs and represent both *how* the agent does an action and *what* the agent is doing, supporting that these two levels can be switched flexibly.

To integrate the two levels of understanding of mirror neurons and their related processes, the target of our project is to establish a computational model that enables a robot to recognize and understand the action sequence of another robot without communication, and then to act by mimicking or imitation [1]. Due to PB units' property of recognizing temporal input sequences, PB units should be beneficial to robot action understanding. With a prototype architecture of PB units, we will focus on robot trajectory prediction and recognition in the following sections, and test their ability to understand *what* the robot is doing. In the remainder of this paper, we will present a new combined version of recurrent network with PB units and an experiment for predicting and recognizing robot trajectories.

2 SRNPB for Robot Trajectory Recognition

The Simple Recurrent Network with Parametric Biases (SRNPB) is based on an Elman network [6] with additional PB units. In a SRN, the hidden layer is fully connected with its previous state, so that it is not only updated with the external input but also with activation from the previous forward propagation. With the recurrent input from the hidden layer, rather than from the output layer as in the RNNPB [15], the short term memory processed in the Elman Network is attractive for sequential processing, because the error which is back-propagated

in the recurrent process is smaller. Furthermore, the recurrent connections in the hidden layer can be seen as related to the horizontal connections in the human cortex. By directly delivering the past history of the hidden layer instead of relying only on the output memory as in the RNNPB, the SRNPB should learn the temporal data with less effort from the network dynamic uncertainty and the learning capability is expected to improve. A layer of parametric bias units is also connected to the hidden layer. Figure 1(b) shows the architecture of the SRNPB. It is based on an Elman Network [6], with an additional layer connecting to the hidden layer as the parametric biases, which serve as adaptable biases learned by the back-propagation through time (BPTT) algorithm [13]. There exist three running modes in SRNPB. In the learning mode, all connection weights and PB values are updated by BPTT. In the recognition mode, only the PB units are updated. The PB value and manually set in generation mode. The determined synaptic weights are common for all learning patterns, but the parametric biases are the sigmoid function values (Equation 2) of the interval values of the PB units. The outputs of the PB units also act as a compensation for the network output error.

Learning mode: The learning is performed supervised and off-line; when providing the training stimulus (positions) for each pattern, the weights are updated with BPTT from layer to layer. Similarly, the internal values of the PB units are also updated. In the updating of PB units, we refer to one entire learning cycle (all sequences) as an epoch e . In each epoch, the k th PB unit u updates its internal value based on the summation of the back-propagation error from the whole sequence (Equation 1 and 2).

$$u_{k,e+1}^{PB} = u_{k,e}^{PB} + \eta_l \sum_{t=1}^T \delta_{k,t}^{PB} \quad (1)$$

$$p_{k,e} = \frac{1}{1 + e^{u_{k,e}^{PB}}} \quad (2)$$

$\delta_{k,t}^{PB}$ represents the back-propagation error of the PB unit u_k at time-step t , and η_l is the learning rate of the PB unit, $p_{k,e}$ is the output value of parametric units which are transferring into the hidden layer via full connectivity. T represents the length of the whole training sequence, so the update value of the internal value is the summation of the error based on the whole sequence multiplied by a specific updating rate.

Action recognition mode: This mode is responsible for the recognition by updating the PB units according to the past observation. The information flow is mostly the same as in the learning mode, except that the synaptic weights are not updated. The generated error between target and prediction is only back-propagated and updated into the PB units. If a trained sequence is presented to the network, the activation of the PB units will converge to the values that were previously shown in the learning mode.

The internal values of PB unit are updated by:

$$u_{k,t+1}^{PB} = u_{k,t}^{PB} + \eta_r \sum_{step=t-l}^t \delta_{k,step}^{PB} \quad (3)$$

where η_r is the updating rate. The internal values of the PB units are integrated with a specific time length l . The reason of doing a summation is that we expect it

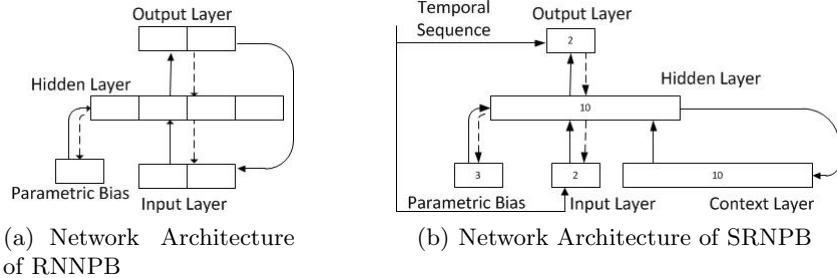


Fig. 1. Comparison of architectures of RNNPB and SRNPB

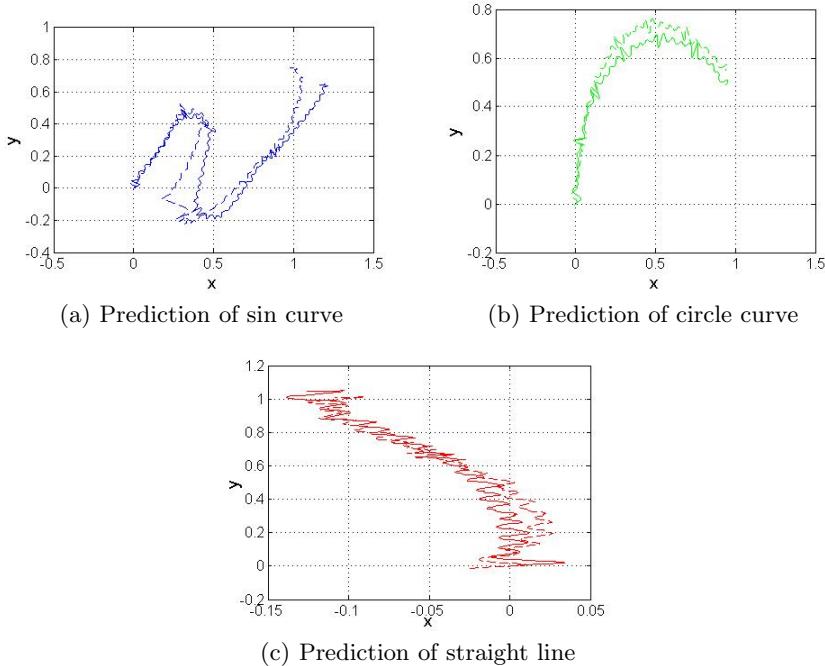


Fig. 2. Prediction of three curves: Prediction experiments were done for three types of trained sequences. Solid lines represent the true positions and the dashed line represent the predictions. We can see the predicted sequence and the target sequence were quite close in the above figures.

can smooth the local fluctuations of the error to decrease its effect and maintain the PB unit at a stable value.

Action generation mode: After learning and after the synaptic weights are determined, the SRNPB can act in a closed-loop, in which the output prediction can be applied as the input for the next time step. The network can automatically generate a trajectory by externally setting the PB unit, representing action codes (*what to do*). Although action generation has not been tested in the experiment, we regard it as an important feature to give commands for robot behavior in future experiments.

3 Experimental Results

As a foundation of robot action understanding, the recognition and prediction of robot walking trajectories are the objectives of the following experiments. For effectiveness, we use the Webots simulator [12] to collect the trajectory data. Our NAO robot is controlled in the Webots simulator to walk along pre-defined trajectories. From the supervisor function within Webots, three kinds of trajectories, that is a straight line, a sine curve and a half circle were recorded

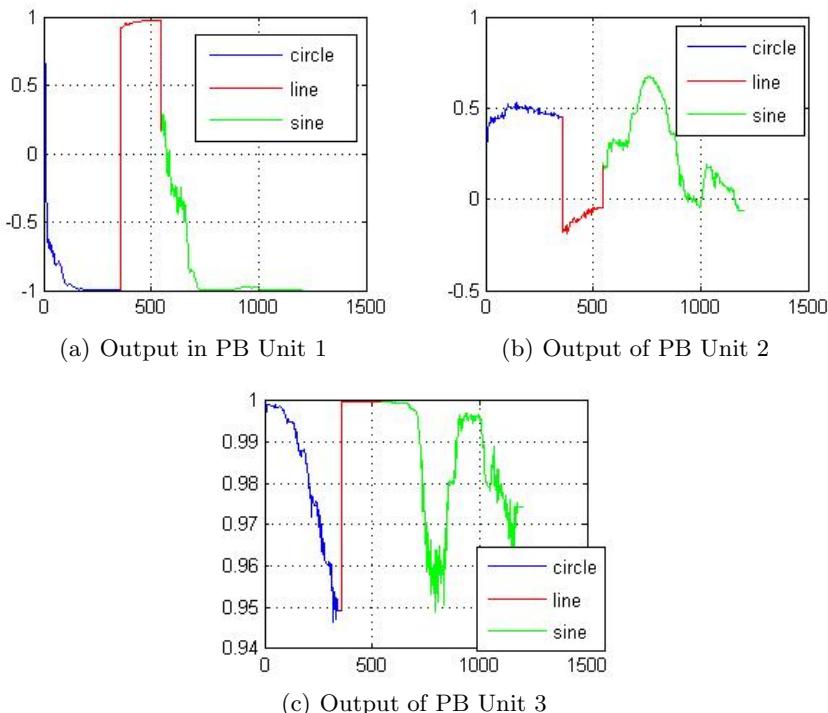


Fig. 3. Three sequences were fed into the network to demonstrate the recognition in PB units

using x and y coordinates. Different combinations of these curves with different parameters make the robot walk in various trajectories. As an initial controlled experiment with known trajectories, we select three trajectories: 1. sine curve: $y = 0.5\sin(\frac{2\pi}{3} * x)$; 2. a half circle curve with 0.57m radius; 3. straight line.

The reason why we use these kinds of trajectories is that they can be combined into different kinds of trajectories, e.g. into a walking trajectory when doing obstacle avoidance, by changing their parameters, i.e. frequency and amplitude in sine curve and radius in half circle. We train the network with three types of input sequences. The expectation is that the generalization ability of PB units can recognize similar trajectories with different parameters. As denoted in the remarks in Figure 1(b), for all simulations we use the same network: 2 input nodes, 10 hidden nodes, 10 context nodes, 2 output nodes. Additionally, we use 3 PB nodes in the experiment. The empirically determined network parameters are: $l = 30$, $\eta_l = 0.01$, $\eta_r = 0.5$, and the learning rate of connection weights in back-propagation is defined by $\eta_{BP} = 0.01$.

After training, we input the walking records from the above three trajectories respectively and attempt to predict position one step ahead given the previous inputs. As shown in Figure 2, after several time-steps, the network can basically

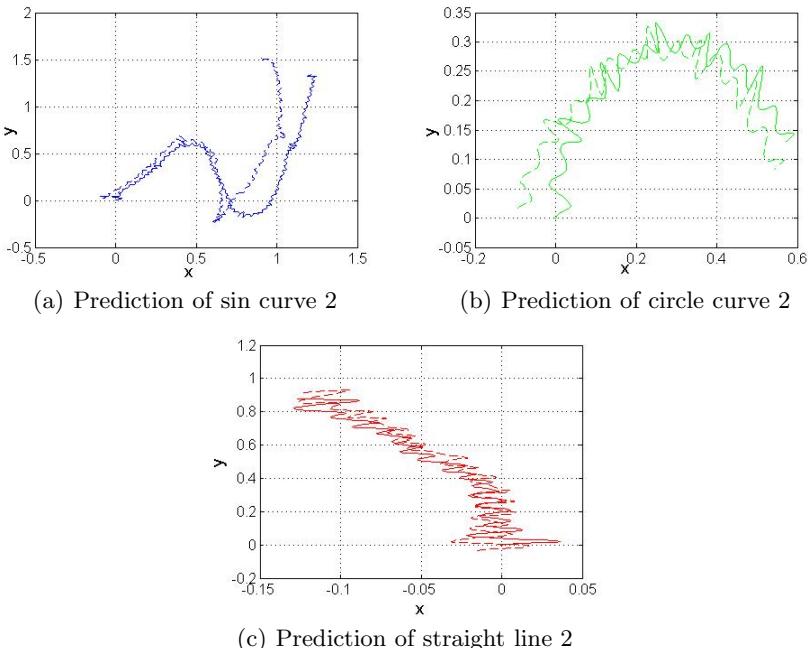


Fig. 4. Prediction of three untrained curves: The errors between the predicted curves and untrained curves were larger than those in Fig. 2, but the trend of the similar curves can also be predicted

predict the learned trajectories. Furthermore, we inspect the values in the PB units while we continuously feed three types of input sequences into it. As shown in Figure 3 internal values in the PB units can reflect different input patterns of the whole learning sequence.

Secondly, we attempt to input another three different types of similar patterns, but with different parameters in order to test the generalization ability for other untrained trajectories. Figure 4 shows the prediction results of the network. Although some errors occur, the generalization of the network still successfully predicts the trend of the curves: 1. sine curve: $y = \sin(\frac{\pi}{2} * x)$; 2. half circle curve with 0.3m radius; 3. straight line. The Table 1 below shows the RMS error between the true value and prediction.

Table 1. Root Mean Square Error of two curve sets predictions

RMSE	sine	line	circle	sine2	line2	circle2
x coordinate	0.0714	0.0052	0.0077	0.2655	0.0187	0.0427
y coordinate	0.0829	0.0066	0.0108	0.1884	0.0094	0.0744

4 Conclusion and Future Works

In this paper, the Simple Recurrent Network with Parametric Biases (SRNPB) is developed for trajectory prediction simulation. The SRNPB is an Elman network with parametric biases, which can predict the robot trajectory as temporal sequence and recognize the trajectory type. After training, the network not only shows the prediction and recognition ability of the robot walking trajectory, but also the generalization ability during prediction of unknown trajectories records. The internal values in PB units can be interpreted as the different discharging rate of mirror neurons indicating the action codes. To conclude, the temporal property is an important feature for action understanding and imitation, therefore, a recurrent network could be a basic prototype building block for modeling mirror neurons to understand *what* the action sequence means [14].

In future research, the generalization ability will be further used as information for *how* to generate similar actions. Also the future architecture may incorporate multi-modal inputs as a representation of environment affordances, so that the network will integrate multiple temporal sequences from both auditory and vision inputs.

Acknowledgments. This research has been partially supported by the EU projects RobotDoc under 235065 from the FP7, Marie Curie Action ITN and KSERA under n 2010-248085 for Research and Technological Development.

References

1. Cakmak, M., DePalma, N., Arriaga, R., Thomaz, A.L.: Computational benefits of social learning mechanisms: Stimulus enhancement and emulation. In: Proceedings of the 2009 IEEE 8th International Conference on Development and Learning, DEVLRN 2009, pp. 1–7. IEEE Computer Society, Los Alamitos (2009)
2. Cuijpers, R., Stuijt, F., Sprinkhuizen-Kuyper, I.: Generalisation of action sequences in RNNPB networks with mirror properties. In: Proceedings of the European Symposium on Neural Networks, ESANN (2009)
3. Demiris, Y., Johnson, M.: Distributed, predictive perception of actions: a biologically inspired robotics architecture for imitation and learning. *Connection Science* 15(4), 231–243 (2003)
4. Demiris, Y., Khadhouri, B.: Hierarchical attentive multiple models for execution and recognition of actions. *Robotics and Autonomous Systems* 54(5), 361–369 (2006)
5. Dinstein, I., Thomas, C., Behrmann, M., Heeger, D.J.: A mirror up to nature. *Current Biology* 18(1), R13–R18 (2008)
6. Elman, J.: Finding structure in time. *Cognitive science* 14(2), 179–211 (1990)
7. Elshaw, M., Weber, C., Zochios, A., Wermter, S.: A mirror neuron inspired hierarchical network for action selection. *Proc. NeuroBotics*, 89–97 (2004)
8. Fogassi, L., Ferrari, P.F., Gesierich, B., Rozzi, S., Chersi, F., Rizzolatti, G.: Parietal lobe: from action organization to intention understanding. *Science* 308(5722), 662 (2005)
9. Haruno, M., Wolpert, D., Kawato, M.: Hierarchical mosaic for movement generation. In: International Congress Series, vol. 1250, pp. 575–590. Elsevier, Amsterdam (2003)
10. Jellema, T., Baker, C., Oram, M., Perrett, D.: Cell populations in the banks of the superior temporal sulcus of the macaque and imitation. The imitative mind: Evolution, development, and brain bases, 267–290 (2002)
11. Lange, F.P.d., Spronk, M., Willems, R.M., Toni, I., Bekkering, H.: Complementary systems for understanding action intentions. *Current biology* 18(6), 454–457 (2008)
12. Michel, O.: Webots: Professional mobile robot simulation. *Journal of Advanced Robotics Systems* 1(1), 39–42 (2004)
13. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning internal representations by error propagation, pp. 673–695. MIT Press, Cambridge (1988)
14. Sugita, Y., Tani, J.: Learning semantic combinatoriality from the interaction between linguistic and behavioral processes. *Adaptive Behavior* 13(1), 33 (2005)
15. Tani, J., Ito, M., Sugita, Y.: Self-organization of distributedly represented multiple behavior schemata in a mirror system: reviews of robot experiments using RNNPB. *Neural Networks* 17(8-9), 1273–1289 (2004)
16. Thioux, M., Gazzola, V., Keysers, C.: Action understanding: how, what and why. *Current biology* 18(10), R431–R434 (2008)
17. Wermter, S., Weber, C., Elshaw, M., Gallese, V., Pulvermuller, F.: A mirror neuron inspired hierarchical network for action selection. *Biomimetic Neural Learning for Intelligent Robots*, 162–181 (2005)
18. Wolpert, D.M., Kawato, M.: Multiple paired forward and inverse models for motor control. *Neural Networks* 11(7-8), 1317–1329 (1998)

A Sentence Generation Network That Learns Surface and Abstract Syntactic Structures

Martin Takac*, Lubica Benuskova, and Alistair Knott

Dept. of Computer Science, University of Otago
PO Box 56, Dunedin 9054, New Zealand
takac@ii.fmph.uniba.sk

Abstract. In this paper we present a connectionist model of sentence generation based on the novel idea that sentence meanings are represented in the brain as sequences of sensorimotor signals which are replayed during sentence generation. Our model can learn surface patterns in language as well as abstract word-ordering conventions. The former is achieved by a recurrent network module; the latter by a feed-forward network that learns to inhibit overt pronunciation of predicted words in certain phases of sensorimotor sequence rehearsal. Another novel element of the model is adaptive switching of control based on uncertainty (entropy) of predicted word distributions. Experiments with the model show that it can learn the syntax, morphology and semantics of a target language and generalize well to unseen meanings/sentences.

Keywords: sentence generation, language acquisition, neural network.

1 Introduction

Sentence generation can be viewed as the problem of encoding a “message” as a sequence of words. A *simple recurrent network* (SRN) has proven to be particularly successful in learning sequential dependences [1]. When trained for predicting the next word in the sequence, it can implicitly form syntactic categories and learn probability distributions conditioned by grammatical rules of the target language [2]. However, pure SRN models have difficulty generalizing to patterns that were rarely or never seen during training, even though they conform to abstract grammatical rules [3,4]. As a workaround, models were suggested that separate rules from their content (words) and learn sequences of more abstract elements, e.g. semantic roles [3], abstract word classes [5] or multi-word phrasal units [6].

In this paper, we present a model of sentence generation that combines learning surface patterns, such as idioms or fixed expressions, with learning of abstract rules. The key novel idea of the model is its representation of sentence meaning as a *sequence* of semantic representations, rather than as a static assembly of active units (Sect. 2).

* Also at Dept. of Applied Informatics, Faculty of Mathematics, Physics and Informatics, Comenius University, Mlynská dolina, 842 48 Bratislava, Slovakia.

The presented model is also interesting from an architectural point of view. It consists of several modules: a content-blind control network for learning abstract rules, a context-independent vocabulary, and a SRN for learning surface patterns. These modules need to be mutually coordinated and employed in different phases of sentence generation. Control passing among the modules is driven by another neural network that is trained to use the *entropy* of the different modules (i.e. their degree of confidence in their predictions) to select which module is in control of processing.

In the rest of the paper we introduce the architecture in more detail (Sect. 3), describe an experiment exploring the model’s ability to acquire different word-ordering conventions and surface patterns (Sect. 4) and present the results of this experiment (Sect. 5).

2 Meanings Represented as Sensorimotor Sequences

Declarative sentences typically describe *episodes* – events or states. We focus on concrete episodes that can be described by transitive sentences (e.g. *John kisses Mary*). The semantic structure of an episode can be modelled as a collection of thematic roles (e.g. AGENT, PATIENT, ACTION) with associated fillers. A connectionist model must employ a scheme for binding semantic objects to particular roles. The scheme we use is motivated by the embodied view on cognition, namely that high-level semantic representations of concrete episodes are delivered by sensorimotor (SM) routines. In our model, the experience of a transitive episode involves a canonical sequence of SM operations – a *deictic routine* [7] (Table 1, for an extensive body of evidence see [8]). Each operation takes place in an initial context, generates a reafferent signal and establishes a new context. We also assume that experienced episodes can be stored in working memory as prepared SM sequences that can be internally replayed. In our model, in order to express an episode verbally, a speaker needs to internally replay the episode’s stored SM sequence, in a mode where the replayed signals generate linguistic side-effects. In this account, the syntactic structure of a sentence is in part a reflection of the structure of the underlying SM routine.

Table 1. The time course of signals occurring during the replay of a deictic routine ‘an agent grasps a cup’ from working memory

Sustained signals	Transient signals			
	Initial context	Operation	Reafferent signal	New context
<i>plan</i> _{attend_agent,attend_cup,grasp}	<i>C</i> ₁	<i>attend_agent</i>	<i>agent_rep</i>	<i>C</i> ₂
<i>plan</i> _{attend_agent,attend_cup,grasp}	<i>C</i> ₂	<i>attend_cup</i>	<i>cup_rep</i>	<i>C</i> ₃
<i>plan</i> _{attend_agent,attend_cup,grasp}	<i>C</i> ₃	<i>grasp</i>	<i>agent_rep</i>	<i>C</i> ₄
<i>plan</i> _{attend_agent,attend_cup,grasp}	<i>C</i> ₄		<i>cup_rep</i>	

3 Architecture

The complete model of language production consists of several functional modules that work together: an **episode rehearsal network**, which replays a working memory episode representation to generate a sequence of SM signals; a **word production network**, which maps individual SM signals onto word forms; a **control network**, which determines the points during episode rehearsal when these word forms should be pronounced; and a **word sequencing network** which learns surface regularities in word sequences, and several other components (Fig. 1). We will now explain each module in turn.

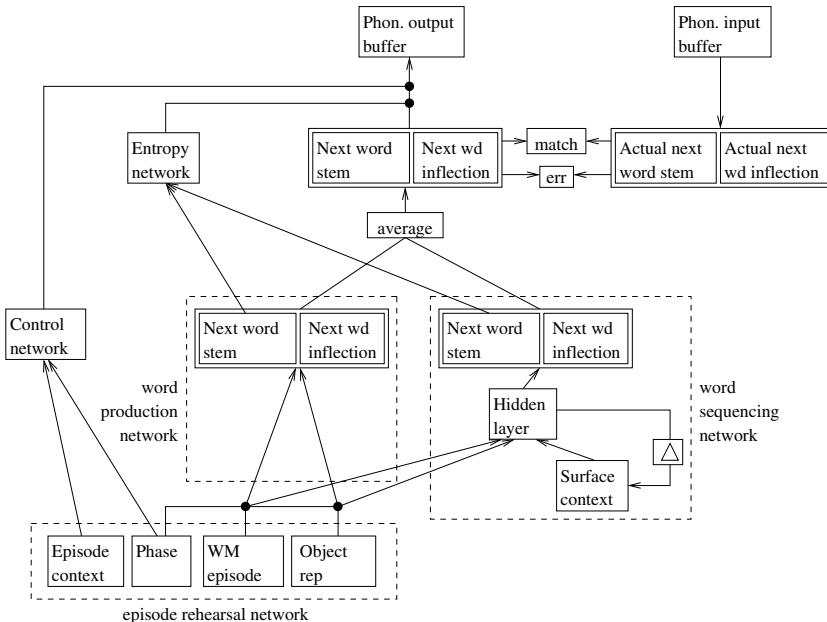


Fig. 1. The complete model of language production. Besides gating overt pronunciation, the control network and the entropy network coordinate mode switching between episode rehearsal and surface word sequencing. Localist linear neurons in the ‘Next word stem’ and ‘Next word inflection’ blocks are combined using the softmax function and represent probability distributions.

3.1 The Episode Rehearsal Network

The **episode rehearsal network** (Fig. 2) consists of four parts. The **working memory (WM) episode** area stores a prepared SM sequence (a plan). The plan is tonically active during the whole rehearsal of a particular episode, but generates transient activity in two areas (‘context’ and ‘current object’) when it is replayed. As well as supporting the rehearsals of SM sequences, WM episode

representations also provide the semantics of inflected verbs in our model. They encode a planned motor action, but also planned actions of attention to the agent and patient, which we assume can surface as agreement inflections on the verb stem. The semantics of nouns come from the ‘current object’ area. We model the syntactic differences between nouns and verbs using the different temporal dynamics of current object and WM episodes in the episode rehearsal network.

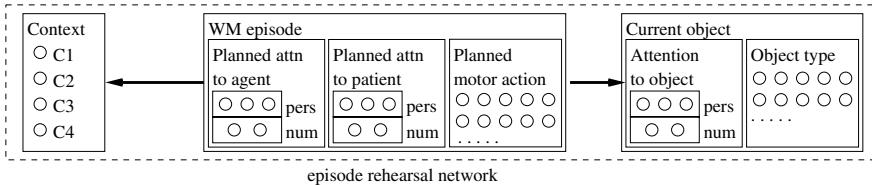


Fig. 2. The episode rehearsal network. The ‘Phase’ part (Fig. 1) is not shown. Each block with circles uses 1-hot localist coding.

The **current object** area holds a transient representation of the currently attended object. During the course of episode rehearsal, this area alternately holds representations of the agent and the patient. Besides a person and number information (in the same format as in the WM episode), it codes the type of the currently attended object.

The **context** and **phase** (Fig. 1) areas hold a representation of the current stage during episode rehearsal. This representation helps to drive the episode rehearsal process. In our simulation there are four possible contexts (see Table 1), each of which has two phases (see Sect. 3.2). The thick arrows in the diagram reflect the fact that the sequence of transient representations in the current object and context areas are generated by a WM episode representation.

3.2 The Word Production Network

The episode rehearsal network provides input to the **word production network** (WPN), which learns a context-independent lexicon in the form of a mapping from single concepts to single words. It also provides input to the word-sequencing network that will be described in the next section.

The WPN consists of one layer of linear perceptrons taking input from all the units in the WM episode and the current object parts of the episode rehearsal system. The input connections are gated by inhibitory links from a cyclic pattern generator (depicted as ‘Phase’ in Fig. 1) so that at any time input comes either wholly from the current object or wholly from the WM episode. During episode rehearsal, the pattern generator cycles through two phases in each context, providing first an opportunity to read out the current object representation (phase *a*), and then the tonically active WM episode (phase *b*) – i.e. to read out first a noun and then a verb.

The WPN is trained on the utterances of mature speakers, paired with episode representations stored in the episode rehearsal network. We are simulating an

infant who experiences episodes in the world and who also hears mature speakers talking. The mature utterances the system hears are stored in a **phonological input buffer** as sequences of target words in exactly the same format as words generated by the WPN.

3.3 The Control Network

For a given semantic input from the episode rehearsal network, a word/inflection output is computed via two independent paths – the WPN and the word-sequencing network (WSN). We envisage their joint averaged output (JAO) to be a premotor articulatory plan that may or may not be overtly pronounced, depending on the decision of other module – the **control network** that gates the connection from the joint output to the phonological output buffer.

The episode rehearsal network gated by the phase generator delivers a structured sequence of semantic signals to the WPN and WSN. For a transitive episode, the sequence is AGENT (in context/phase $C1a$), WM EPISODE ($C1b$), PATIENT ($C2a$), WM EP. ($C2b$), AGENT ($C3a$), WM EP. ($C3b$), PATIENT ($C4a$). Note that this sequence contains multiple occurrences of each concept. Different languages have different word-ordering conventions (e.g. English has the SVO – Subject Verb Object word order, while Māori has VSO); the model has to learn on which occasion the JAO should be pronounced/withheld. This is the task of the control network – a feed-forward network with one hidden layer, which is trained on the match between the predicted word (JAO) and the current word in the phonological input buffer. The desired output is binary (1 for ‘pronounce’ in case of match, 0 for ‘withhold’ in the case of a mismatch).

Note that the control network is content blind in the sense it takes no input from actual semantic concepts, just from the ‘context’ and ‘phase’ parts of the episode rehearsal system. Hence it has a potential to learn abstract syntactic rules in terms of contexts/phases when the overt pronunciation should be suppressed. For example, for SVO language, pronunciation should be suppressed in all context/phases but $C1a$, $C1b$, $C2a$; for VSO in all but $C1b$, $C3a$, $C4a$.

3.4 Learning Surface Patterns

The model described so far can learn a lexicon and a set of abstract word-ordering conventions for a given language. However, languages also contain surface patterns such as idioms or fixed expressions – sequences of words that occur together with particularly high frequency and that contribute their meaning collectively rather than individually, e.g. *Winnie the Pooh*. Other surface patterns take the form of statistical tendencies, where in some context a particular word is more likely to occur than other words. Idioms violate the one-to-one correspondence between concepts and words; hence we need to extend the model with a device that can generate more than one word for a particular semantic concept present at the input. In our model, this is the **word-sequencing network** – a variant of a SRN. Input and output-wise, it mimics the WPN (Fig. 1). Both networks are trained by the ‘actual’ next word replayed from the phonological input buffer.

However, the WSN has a hidden layer with recurrent connections, which enables it to learn commonly occurring sequential patterns in its training data.

3.5 The Entropy Network

Since the WSN is able to produce more than one word for any given semantic input, the model needs to decide when to pass control back to the episode rehearsal network, i.e. when to deliver the next semantic signal.

Imagine the WSN is to produce an idiomatic expression – say *Winnie the Pooh*. This expression describes a single semantic signal. Given this signal, the WSN should begin by predicting the word *Winnie* with high confidence, and then, after copying back the surface context, the word *the* and then (after another copy operation) the word *Pooh*, both with high confidence. But after this point, the network can no longer be so confident. Like a regular SRN, it can at best predict the category of the following word (e.g. predicting a higher likelihood for verbs), but not a particular content word. This indicates that the episode rehearsal network should deliver the next semantic signal.

As a (inverse) measure of confidence, we use the *entropy* in the word stem part of the WSN output. If the predicted word has high entropy (many competing alternatives), it should not be overtly pronounced and the control should be passed back to the episode rehearsal network. An exact threshold for the entropy can be task dependent and can change in time, so we use an adaptive feed-forward network (called the **entropy network**) that learns the right value. It takes as its input the entropies of the WSN and WPN outputs and is trained on the same Boolean ‘match’ signal as the control network. Besides control passing, the output of the entropy network has a gating function similar to that of the control network, i.e. to suppress the overt pronunciation of words predicted with low confidence.

3.6 Sentence Generation in the Trained System

As already mentioned, sentence generation in our conception involves replaying a particular episode from working memory, generating a sequence of semantic signals in the episode rehearsal system, from which a sequence of words is produced in the phonological output buffer. The trained model alternates between two modes of iteration. In one mode, the episode rehearsal system is in control. This system iterates through the sequence of SM signals until it reaches a context at which the control network allows a word to be overtly pronounced. In the other mode, the WSN is in control. At each iteration, the WPN and WSN jointly predict a probability distribution for the next word given the currently active SM signal. If they can confidently predict the next word, the word is pronounced, the WSN updates its surface context layer and the model carries on in this mode until the networks can no longer confidently predict the next word.

3.7 Training the System

During training, the model alternates between the same two modes as during generation. In the first mode, episode rehearsal advances (and the control

network is trained) until a context/phase is reached in which the control network gives the ‘pronounce’ signal. Then the network switches into the word sequencing mode. As long as the WPN and WSN predict the next word with sufficient confidence and it matches the actual word in the phonological input buffer, they keep predicting (based on a changing surface context), being trained, and advancing the phonological input buffer. If the prediction does not match or has a low confidence, the actual word stays in the phonological input buffer, the surface context is not copied and the model switches back to the episode rehearsal mode. Details of the training algorithm are given in [9].

4 Experiment

The model we have just described was trained on an artificial language with an English vocabulary (105 words), morphology featuring number (Sg, Pl) inflections of nouns, number and person (1st, 2nd, 3rd) inflections on verbs, subject-verb agreement, irregular plurals (leaves, fish, teeth, women, etc.) and personal pronouns, and the SVO word order.¹ The language consisted of 127088 transitive sentences, out of which roughly 80% were regular transitive sentences such as *Mice bite-3pl dog-sg*, the rest contained continuous **idioms** such as *Mia-sg lick-3sg ice cream-sg* (13%) and idioms interleaved with a noun phrase, such as *Daddy-sg kiss-3sg me good bye* (6.4%).

The model was trained on a sample of 4000 randomly selected sentences paired with their meanings (sequences of semantic signals) for 25 epochs. After each training epoch, the weights were frozen and the model was tested for sentence generation on a set of 4000 previously unseen meanings. All results were averaged over 10 runs with different initial random weights of connections and different training/test samples of the target language.

To test the ability of the model to acquire all possible word-ordering conventions, we created another five target languages with the same vocabulary, morphology and similar idioms, but with different basic word-ordering (SOV, VSO, VOS, OVS, OSV) and ran 10 runs for each target language in the same way as for the SVO language.

5 Results

The control network was able to learn correct word-ordering rules with 100% success for all the six word-orders. We also recorded the network’s overall *generation accuracy*, measured as the proportion of correctly generated sentences. We considered an utterance to be correctly generated for a given meaning, if all thematic roles were expressed with semantically appropriate words, the sentence

¹ We conducted a preliminary study of a pure SRN (enhanced with spatially represented semantic roles) on an analogous language production task [10]. Although successful in generating certain types of unseen sentences, this network has a problem in principle – it cannot generalize across semantic roles, as argued in [3].

was syntactically correct (i.e. it complied with the transcription rules) and all the words had correct morphology (inflections). Averaged over all six word-orders, the models achieved 96.6 % (SD=2.7 %) accuracy on training sets and 94.1 % (SD=4.3 %) accuracy on test sets. Given that they were trained on 3 % of target sentences, the model achieved good generalisation ability.

6 Conclusion

The main goal of this paper was to introduce a novel connectionist architecture for sentence generation which is able to learn both abstract grammatical rules and surface patterns in a language. The experiments reported here show that our network can generate regular sentences but also sentences containing a variety of idiomatic surface structures. The main technical innovation which permits this is our use of sequences to represent sentence meanings (episodes). From the perspective of embodied cognition, this is helpful in connecting semantic representations to the sensorimotor system. From the perspective of syntax, it is helpful in supporting a rich model of patterns in language.

Acknowledgments. This research was supported by VEGA 1/0439/11 grant and a BuildIT postdoctoral fellowship grant for Martin Takac.

References

1. Elman, J.: Finding Structure in Time. *Cognitive Science* 14, 179–211 (1990)
2. Elman, J.: Distributed Representations, Simple Recurrent Networks, and Grammatical Structure. *Machine Learning* 7, 195–225 (1991)
3. Chang, F.: Symbolically Speaking: A Connectionist Model of Sentence Production. *Cognitive Science* 26, 609–651 (2002)
4. Marcus, G.F.: Rethinking Eliminative Connectionism. *Cognitive Psychology* 37(3), 243–282 (1998)
5. Pulvermüller, F., Knoblauch, A.: Discrete Combinatorial Circuits Emerging in Neural Networks: A Mechanism for Rules of Grammar in the Human Brain. *Neural Networks* 22(2), 161–172 (2009)
6. Dominey, P., Hoen, M., Inui, T.: A Neurolinguistic Model of Grammatical Construction Processing. *Journal of Cognitive Neuroscience* 18(12), 2088–2107 (2006)
7. Ballard, D., Hayhoe, M., Pook, P., Rao, R.: Deictic Codes for the Embodiment of Cognition. *Behavioral and Brain Sciences* 20(4), 723–767 (1997)
8. Knott, A.: Sensorimotor Cognition and Natural Language Syntax. MIT Press, Cambridge (in press)
9. Takac, M., Benuskova, L., Knott, A.: Mapping Sensorimotor Sequences to Word Sequences: A Connectionist Model of Language Acquisition and Sentence Generation. Technical report OUCS-2011-01, University of Otago, New Zealand (2011)
10. Takac, M., Knott, A., Benuskova, L.: Generation of Idioms in a Simple Recurrent Network Architecture. Technical report OUCS-2010-02, University of Otago, New Zealand (2010)

A Perceptual Memory System for Affordance Learning in Humanoid Robots

Marc Kammer^{1,2}, Marko Tscherepanow^{1,2}, Thomas Schack^{1,3},
and Yukie Nagai^{1,4}

¹ CITEC, Cognitive Interaction Technology, Center of Excellence

² Applied Informatics, Faculty of Technology

³ Neurocognition and Action, Faculty of Psychology and Sport Sciences
Bielefeld University, Universitätsstraße 25, 33615 Bielefeld, Germany

⁴ Graduate School of Engineering, Osaka University,

2-1 Yamadaoka, Suita Osaka, 565-0871 Japan

mkammer@cit-ec.uni-bielefeld.de

Abstract. Memory constitutes an essential cognitive capability of humans and animals. It allows them to act in very complex, non-stationary environments. In this paper, we propose a perceptual memory system, which is intended to be applied on a humanoid robot learning affordances. According to the properties of biological memory systems, it has been designed in such a way as to enable life-long learning without catastrophic forgetting. Based on clustering sensory information, a symbolic representation is derived automatically. In contrast to alternative approaches, our memory system does not rely on pre-trained models and works completely unsupervised.

Keywords: Cognitive robotics, artificial memory, life-long learning, affordances.

1 Introduction

How humanoid robots can be enabled to learn complex real-world tasks is still an open research question. Recently, the concept of affordances has become a popular paradigm in teaching robots. The psychologist Gibson [6] defined the term affordances as action opportunities an observer becomes aware of by looking at an environment or at an object; for example, a car affords to drive and a ball affords to kick. The learning of affordances requires a robot to memorize certain types of objects, actions, effects as well as their relationships [5].

Since real-world environments are usually dynamic, an agent acting in them needs to adapt continuously. This requires the ability of life-long learning and the stable memorization of relevant information. Although important, there are only few investigations (e.g., [2]) on memory systems in cognitive robots that enable robots to learn, *inter alia*, affordances.

In this paper, we present the basis of a biologically inspired and distributed perceptual memory system for cognitive robots. It enables the stable, unsupervised, and incremental learning of perceptual information as well as the automatic generation of symbolic object representations. A symbolic and therefore discretized representation of perceptual information is a necessary requirement for learning affordances [13].

The rest of the paper is organized as follows: First, we will give an overview of biological and technical background information that forms the theoretical foundation of the introduced memory approach. Second, we present a prototypical implementation of the developed memory and will show first evaluation results using a real-world data set. Finally, we summarize the results and discuss challenges, possible improvements and potential future application scenarios.

2 Memory as a Basis for Learning in Cognitive Robotics

We claim that memory is a necessary requirement for any form of learning; as pointed out by Baxter and Browne in [2](p. 1), “cognition is inherently memory-based”. In biological organisms even the most basic acquisition of knowledge is already a form of learning and it is difficult, if possible at all, to define where the process of memorization ends and the process of learning starts [10].

Neuroscientific and biological research offers a rich foundation of theories and models which can, if not in detail but in principle, be used to simulate cognitive capabilities such as memory by technical means. For instance, human memory can be divided into several subsystems regarding neural correlates, temporal aspects, and content [12].

In order to meet the capacity and time constraints of life-long learning systems, stored information needs to be organized efficiently, which is summarized by the term *cognitive economy* [7]. In particular, the amount of data must be reduced by mechanisms such as categorization to meet the storage and time requirements. In [4] the principle of cognitive economy is satisfied by using Self-Organizing Feature Maps to create a heteroassociative memory which learns categories based on prototype representations. But the network structure is not incremental and therefore limited in its capability to incorporate novel data.

A further problem arising in life-long learning systems is the *stability-plasticity dilemma* [8]: How can a system retain old memories but still learn new information? The memory system introduced in [11] approaches the stability-plasticity dilemma by adopting a biologically inspired hierarchical visual pathway processing. However the solution does not allow explicit symbolic access to the learned entities, which would be beneficial for applying reasoning methods in the context of learning affordances.

In [15], Sun discusses the technical representation of memorized information. He favors a combination of subsymbolic and symbolic representations, as realized within the cognitive architecture CLARION [16]. The CLARION architecture simulates human mental processes, which works in simulation but not in a real world environment.

Hanheide and his colleagues [9] presented a perceptual memory system for learning faces of interaction partners in a real-world scenario. It uses pre-trained models for face detection and feature extraction. Known interaction partners are classified by means of support vector machines following an one-vs-rest approach. If unknown persons appear, the corresponding face patches are used to train a new classifier. The one-vs-rest approach requires the extracted face patches of all known interaction partners to be stored, which is not compatible with cognitive economy and puts the scalability of the system in question.

Although related works from the field of affordance learning do not focus on memory aspects in an overall cognitive architecture as investigated in [2], they incorporate at least a very simple form of memory to represent objects from the real world. For example [13] and [18] use X-Means clustering for a perceptual discretization and memorization to apply a high level reasoning, which limits the number of objects that can be used severely.

As the goal of our architecture is the incremental and online learning of affordances using only few training samples the architectural memory has to fulfill the above mentioned criteria. Therefore, we introduce a perceptual memory system based on Adaptive Resonance Theory (ART) [8] networks, which learn online, incremental, unsupervised and constitute a solution of the *stability-plasticity dilemma*. Similar to the CLARION architecture, we use a subsymbolic feature processing mechanism to form a symbolic feature representation at the top level of the memory. A detailed explanation of the developed architecture is given in the next section.

3 Our Perceptual Memory System

We met the above-mentioned requirements by creating a distributed, hierarchical, incremental perceptual network structure, composed of different ART networks. These ART networks are capable of unsupervised incremental online learning and can be trained on few training samples. As mentioned before, the long term goal of this work is the creation of an interactive learning architecture that shall be used in an online affordances learning scenario. Each displayed constituent is described in this section.

Our so far preliminary cognitive architecture is able to detect objects that are introduced into a static scene by observing changes. Detected objects are rotated according to their first principal axis to mimic the process of *mental rotation* [14] and scaled to a common size. These normalized images are further processed according to Fig. 1. The normalized object patches are split into five partitions, as indicated by the vertical lines in Fig. 1. For each of these partitions several histograms reflecting the frequency of the occurrence of specific colors are determined. The colors are defined by ranges in the hue (H) plane of the HSV color space. Each range was defined manually and corresponds to the color impression it triggers in the human perceptual system. Finally, the histograms of all partitions that correspond to an individual color are concatenated and fed as input into an ART network. These networks learn efficient sub-symbolic

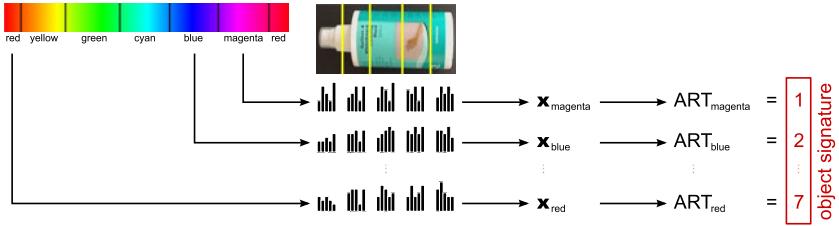


Fig. 1. Processing of the normalized object patches. After splitting an image into five disjunct partitions, histograms are computed for several color ranges. These color-specific histograms are concatenated and fed into ART networks. Finally, the indices of the best-matching node of each ART network are concatenated to yield object-specific symbolic signatures.

representations of presented input samples which are referred to as categories. For each presented object patch, the indices of the best-matching categories form an object-specific signature, which we regard as symbolic representation of an object. The common activation of all ART networks therefore provides a distributed, unique representation of an object image i , which we term *object signature*, referring to it as \mathbf{s}_i .

$$\mathbf{s}_i = (s_{i1}, s_{i2}, \dots, s_{in}) \quad (1)$$

Such a distributed feature representation allows the system to identify certain features in reasoning processes as important or irrelevant. This is important, for example in the task of affordance learning. In order to measure the dissimilarity between two object signatures \mathbf{s}_i and \mathbf{s}_j , we apply the hamming distance $\Delta(\mathbf{s}_i, \mathbf{s}_j)$.

$$\Delta(\mathbf{s}_i, \mathbf{s}_j) = \frac{1}{n} \sum_{k=1}^n d(s_{ik}, s_{jk}) \quad , \text{with } d(s_{ik}, s_{jk}) = \begin{cases} 1 & \text{if } s_{ik} \neq s_{jk} \\ 0 & \text{if } s_{ik} = s_{jk} \end{cases} \quad (2)$$

The hamming distance $\Delta(\mathbf{s}_i, \mathbf{s}_j)$ counts the number of positions at which two signatures differ. The results are normalized to the interval $[0, 1]$, where 0 means that both object signatures are identical and 1 means that both object signatures differ at all positions.

4 Experimental Results

We compared three different ART networks: Fuzzy ART [3], TopoART [17] and Hypersphere ART [1]. All of these networks allow for stable and incremental learning of new objects. But they differ in their activation functions and their sensitivity to noise. The goal of the evaluation process of our perceptual memory architecture is to identify the best parameters for each used ART-network, as

well as to evaluate the performance of each ART-network in the given scenario and if it is suited or not.

We used the described preliminary cognitive architecture to create a real world data set by recording 25 different objects with 10 variants for each object, which resulted in an overall set of 250 images. This set was split in disjunct training and test sets, each containing 125 images. Using a real world setup resulted in different variants for each object. In each recording attempt, the position of the object might vary as well as the lighting conditions or even noise can lead to different object appearances. Figure 2 shows five of the 25 different objects with all of its variants.

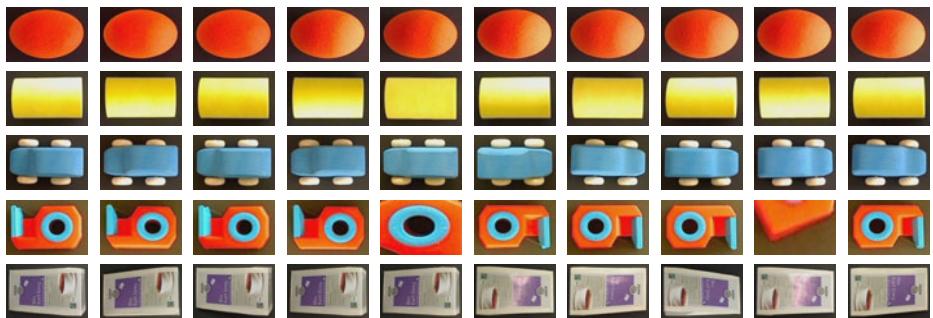


Fig. 2. Recorded and normalized images of five exemplary objects. For each object (rows) ten different images (columns) were acquired and processed automatically. In some very rare cases, the images were incorrectly cropped.

The best parameter values for ρ (all networks) and β_{sbm} (only TopoART) were determined by grid search. Due to the small amount of available data, all networks were trained in fast-learning mode ($\beta=1$) and the noise reduction mechanism of TopoART was disabled ($\varphi=1, \tau \geq 1$). The selected parameters were used in the test phase.

As evaluation criteria, two different measures denoted by d_w and d_b were used. The first measure d_w , given in (4), represents the dissimilarity each object has within its variants. A low value is desirable, as it indicates that the signature of the variants of the objects are similar to each other. θ and ϑ denote the number of all objects and their variants, respectively. In our experiments, $\theta = 25$ and $\vartheta = 5$ were used for training as well as testing. Then the number of all hamming distances an object can have within its own variants is $\varrho = \frac{1}{2}\vartheta \cdot (\vartheta - 1)$. Then, the mean over all ϱ hamming distances for one object o is given by d_w^o .

$$d_w^o = \frac{1}{\varrho} \cdot \sum_{i=1}^{\vartheta} \sum_{j=i+1}^{\vartheta} \Delta(s_i^o, s_j^o) \quad (3)$$

If an specific object o has three variants $\vartheta=3$, for example, an overall of $\varrho = 3$ hamming distances can be calculated which results to $d_w^o = \frac{1}{\varrho}(\Delta(s_1^o, s_2^o) +$

$\Delta(s_1^o, s_3^o) + \Delta(s_2^o, s_3^o)\). The mean of d_w^o over all objects is denoted by d_w .$

$$d_w = \frac{1}{\theta} \sum_{o=1}^{\theta} d_w^o \quad (4)$$

The second measure d_b , given in (6), represents the dissimilarity each object variant has to all variants of the other objects and is therefore termed the between object similarity. A high value indicates a high dissimilarity of variants between different objects which is preferable. The mean dissimilarity of a specific variant i of an object o to all variants of other objects is denoted by $d_b(s_i^o)$.

$$d_b(s_i^o) = \frac{1}{(\theta-1) \cdot \vartheta} \sum_{\substack{o'=1 \\ o' \neq o}}^{\theta} \sum_{j=1}^{\vartheta} \Delta(s_i^o, s_j^{o'}) \quad (5)$$

d_b denotes the mean dissimilarity averaged over all objects and their variants.

$$d_b = \frac{1}{\theta \cdot \vartheta} \sum_{o=1}^{\theta} \sum_{i=1}^{\vartheta} d_b(s_i^o) \quad (6)$$

For parameter optimization, d_w should be low, while for d_b high values are desired. This relationship is captured by the goal function G .

$$G = d_b - d_w \quad (7)$$

For the maximum of G , each individual object is represented by a set of similar variant signatures, while the signatures of different objects differ strongly. The chosen parameters at the maximum of G are therefore optimal.

Figure 3 depicts an exemplary evaluation of the FuzzyART network, showing d_w , d_b , and the goal function G averaged over ten training trials depending on the vigilance parameter ρ . In addition, it compares the goal functions for all ART networks.

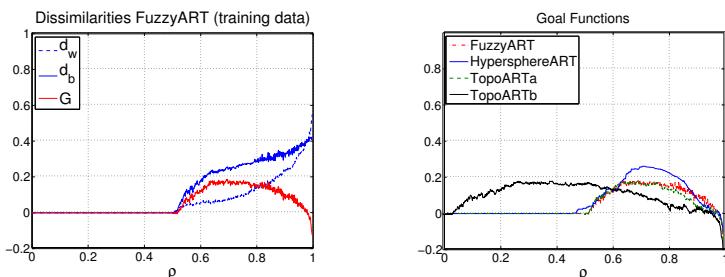


Fig. 3. Training results of an exemplary FuzzyART network (left) and the corresponding goal functions G of all ART networks (right) depending on ρ

Table 1. Test results and their standard deviations calculated with the optimal parameter settings for each network. The settings for TopoART were independently optimized for its components TopoART *a* (I) and TopoART *b* (II).

	FuzzyART $\rho = 0.696$	HypersphereART $\rho = 0.71$	TopoART <i>a/b</i> (I) $\rho = 0.694 \& \beta_{sbm} = 0$	TopoART <i>a/b</i> (II) $\rho = 0.388 \& \beta_{sbm} = 0$
d_w	0.0598857	0.0221714	0.0626286 / 0.096	0 / 0.0626286
d_b	0.135429	0.117171	0.129229 / 0.240762	0 / 0.129229
σ_w	0.305495	0.216834	0.318894 / 0.393548	0 / 0.318894
σ_b	0.0097925	0.0155369	0.010532 / 0.0145529	0 / 0.010532

Finally, the estimated parameters were used in the test phase to calculate the hamming distances on the separated test set. The results summarized in Table 1 indicate that HypersphereART is best suited for the given setup as its difference of $d_b - d_w$ has the overall highest value which mean that variants of the same object are represented by similar signatures and that the signatures between different objects have a greater deviation.

5 Conclusion and Outlook

In this paper, we presented a distributed, incremental perceptual memory system tailored to the task of learning affordances in an interactive real-world scenario. In order to fulfill the requirement of stable life-long learning, we applied different ART networks (Fuzzy ART, Hypersphere ART, and TopoART) to learn sub-symbolic object representations. Furthermore, we introduced the idea of using a distributed but common activation of different ART networks to create a bottom-up symbolic object representation based on the respective best-matching nodes.

In our experimental setup, Hypersphere ART performed best. Furthermore, the results of Fuzzy ART and TopoART are very similar to each other. Therefore, we conclude that the Euclidean distance used for activating nodes in Hypersphere ART is more suited to the task at hand than the city block distance utilized by Fuzzy ART and TopoART.

Our future research will focus on additional perceptual input features for categorizing objects and enriching the object signature, e.g. by shape and size information. Also especially a recognition test, based on an enlarged object signature and a distance measurement, as for example the used hamming distance has to be investigated. Another future investigation is the comparison of the ART networks with networks that suffice most of the required criteria but do not belong to the ART family, as for example growing self organizing maps or growing neural gas.

Acknowledgements. This research was funded by the German Research Foundation (DFG), Excellence Cluster 277, Cognitive Interaction Technology, Research Area D (Memory and Learning).

References

1. Anagnostopoulos, G.C., Georgopoulos, M.: Hypersphere ART and ARTMAP for unsupervised and supervised incremental learning. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN), vol. 6, pp. 59–64 (2000)
2. Baxter, P., Browne, W.: Memory as the substrate of cognition: A developmental cognitive robotics perspective. In: Johansson, B., Sahin, E., Balkenius, C. (eds.) Proceedings of the International Conference on Epigenetic Robotics (EpiRob), pp. 19–26 (2010)
3. Carpenter, G.A., Grossberg, S., Rosen, D.B.: Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks* 4(6), 759–771 (1991)
4. Chartier, S., Giguère, G., Langlois, D.: A new bidirectional heteroassociative memory encompassing correlational, competitive and topological properties. *Neural Networks* 22(5-6), 568–578 (2009)
5. Şahin, E., Çakmak, M., Doğar, M.R., Uğur, E., Üçoluk, G.: To afford or not to afford: A new formalization of affordances toward affordance-based robot control. *Adaptive Behavior* 15(4), 447–472 (2007)
6. Gibson, J.J.: The theory of affordances. *Perceiving, Acting, and Knowing*, 67–82 (1977)
7. Goldstone, R.L., Kersten, A.: Concepts and categorization. In: Healy, A.F., Proctor, R.W. (eds.) *Comprehensive Handbook of Psychology. Experimental psychology*, vol. 4, pp. 599–621. Wiley, Chichester (2003)
8. Grossberg, S.: Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science* 11, 23–63 (1987)
9. Hanheide, M., Wrede, S., Lang, C., Sagerer, G.: Who am I talking with? A face memory for social robots. In: IEEE International Conference on Robotics and Automation, pp. 3660–3665. IEEE Computer Society Press, Los Alamitos (2008)
10. Kandel, E.R.: The molecular biology of memory storage: a dialogue between genes and synapses. *Science* 294(5544), 1030–1038 (2001)
11. Kirstein, S., Wersing, H., Körner, E.: A biologically motivated visual memory architecture for online learning of objects. *Neural Networks* 21(1), 65–77 (2008)
12. Markowitsch, H.J., Halligan, P.W., Kischka, U., Marshall, J.C.: *Functional neuroanatomy of learning and memory*, pp. 724–741. University Press (2003)
13. Montesano, L., Lopes, M., Bernardino, A., Santos-Victor, J.: Learning Object Affordances: From Sensory–Motor Coordination to Imitation. *IEEE Transactions on Robotics* 24(1), 15–26 (2008)
14. Shepard, R.N., Metzler, J.: Mental rotation of three-dimensional objects. *Science*, 701–703 (1971)
15. Sun, R.: Robust reasoning: Integrating rule-based and similarity-based reasoning. *Artificial Intelligence* 75(2), 241–295 (1995)
16. Sun, R., Zhang, X., Mathews, R.: Capturing human data in a letter counting task: Accessibility and action-centeredness in representing cognitive skills. *Neural Networks* 22(1), 15–29 (2009)
17. Tscherepanow, M.: TopoART: A topology learning hierarchical ART network. In: Diamantaras, K., Duch, W., Iliadis, L.S. (eds.) *ICANN 2010. LNCS*, vol. 6354, pp. 157–167. Springer, Heidelberg (2010)
18. Uğur, E., Şahin, E., Oztop, E.: Affordance learning from range data for multi-step planning. In: Proceedings of the International Conference on Epigenetic Robotics (EpiRob), pp. 177–184 (2009)

Probabilistic Proactive Timeline Browser

Antti Ajanki¹ and Samuel Kaski^{1,2}

¹ Aalto University School of Science

Department of Information and Computer Science

Helsinki Institute for Information Technology HIIT

PO Box 15400, 00076 Aalto, Finland

² University of Helsinki, PO Box 68, 00014 University of Helsinki, Finland

{antti.ajanki,samuel.kaski}@tkk.fi

Abstract. We have developed a browser suitable for finding events from timelines, in particular from life logs and other timelines containing a familiar narrative. The system infers the relevance of events based on the user's browsing behavior and increases the visual saliency of relevant items along the timeline. As recognized images are strong memory cues, the user can quickly determine if the salient images are relevant and, if they are, it is quick and easy to select them by clicking since they are salient. Even if the inferred relevance was not correct, the timeline will help: The user may remember if the sought event was before or after a saliently shown event which limits the search space. A user study shows that the browser helps in locating relevant images quicker, and augmenting explicit click feedback with implicit mouse movement patterns further improves the performance.

Keywords: Image retrieval, implicit relevance feedback, interaction, machine learning.

1 Introduction

One of the main reasons why image retrieval is difficult is that it is hard to formulate effective queries. In contrast to texts, images cannot be easily automatically decomposed into low level units which the user could combine to form queries. Two common image retrieval approaches are retrieval based on textual metadata, such as image captions, and content-based image retrieval (CBIR). The content-based approach [2] is typically an iterative process where the retrieval system returns a set of images and the user is required to grade their degree of relevance. The retrieval system then updates its estimate of the desired image features and retrieves a new set of images.

Explicit feedback in the form of clicking the relevant images or giving rankings is rather accurate but requires judgement calls by the user and is therefore either laborious or scarce. Use of implicit feedback has been suggested as an alternative. The idea is to measure, as a by-product of normal use, data about how the user interacts with the system, and infer relevance from the measurements that are indirectly related to relevance. Implicit feedback has been shown to be useful in

text [9,10] and image retrieval [4,7,8], but not very accurate as the only source of feedback.

When the images are ordered and the order is familiar, new forms of retrieval and feedback become possible. This is the case for life logs [5] and other narratives where the images are strong memory cues for recalling past events. We are in the process of developing an image and event retrieval method that utilizes the capability of humans to very effectively recognize familiar events from cues associated with the event, in our case images.

Our search interface helps in locating images in two ways: first, the images are displayed on a timeline to allow using the temporal neighborhood as a rough search cue. Secondly, the search interface suggests potentially relevant images by making them more salient. The relevance is inferred from explicit and implicit feedback using probabilistic inference. Previous information re-finding methods (e.g. [3]) require manually entering the remembered details as a search query. Our hypothesis is that automatic relevance prediction and visualization of the relevance estimates decreases the effort required to find the correct images.

Unlike a typical CBIR interface, where a small number of images estimated to be relevant is shown at once, our interfaces shows all the images on screen. The screen space is allocated proportional to the estimated relevance. This has the advantage that even the less relevant images are easily accessible, so that the user can correct the relevance prediction by selecting an image whose current estimated relevance is low.

The idea of visualizing the relevance predictions as the size of the images is motivated by Dasher, a predictive text-entry system [11]. Dasher shows all letters of the alphabet entering from the right edge of the screen. The letters can be selected by mouse or by gaze. The sizes of the incoming letters are determined by a language model that predicts likelihood of the next letter given letters selected so far. In our case, the sizes of the images are similarly modulated according to their estimated relevance to make it easier to spot other relevant images.

In the remaining of this paper, we introduce our search interface and the relevance prediction model that combines explicit and implicit feedback. Then we report results of a user study, where the dynamic interface was compared to baseline interfaces which do not try to predict the relevance. We also studied how much the relevance prediction performance improves when explicit feedback (mouse clicks) is combined with implicit mouse movement features, such as time duration of hovering over an image.

2 Timeline Browser with a Relevance Estimator

We introduce a browser for finding images that are ordered on a timeline. We utilize the memory of humans: shown images work as recall cues; seeing an image brings back memories from the time the image was taken or seen.

The browser includes a mouse-operated fish-eye lens. When the mouse is moved over an image, the image and its close neighbors are grown to allow

inspecting the images more closely. The size of the images is restored when mouse moves away from the images.

The sizes of the images are also affected by their estimated relevance. Relevance is estimated from explicit feedback (mouse clicks) and implicit browsing patterns. Images estimated to be relevant are shown in a larger size. The relevances are recomputed and sizes are updated dynamically after each click. Our hypothesis is that emphasizing the relevant images makes it easier and quicker to find the correct images. Figure 1 shows snapshots of the interface.



Fig. 1. The three interface variants from the experiments. From top to bottom: our new proactive interface where image sizes are proportional to the estimated relevance, zooming interface that magnifies the image on which the cursor is located and its neighborhood, and a scrollable simple timeline. The images are snapshots from a Creative Commons licensed movie by Adam Wojtanek.

2.1 Relevance Prediction

To be able to emphasize potentially relevant images, we need to estimate the relevance based on clicks and implicit feedback. Next, we will introduce a probabilistic generative model for this purpose.

We assume that relevance is reflected in a latent (potentially multidimensional) variable \mathbf{z} . The latent variable \mathbf{z}_i of image i is assumed to be generated by a linear regression from image's observed content feature vector \mathbf{f}_i to the latent space. The values in the unobserved regression matrix \mathbf{Q} capture the user's "implicit" query by weighting the content dimensions appropriately. This prior distribution constrains images that are similar with respect to the latent query \mathbf{Q} to have similar relevances. We further assume a user model where the image

click counts \mathbf{y} are drawn from a multinomial distribution. The weights of the multinomial are softmax-normalized values of the latent variables \mathbf{z} . The plate diagram of the model is shown on the left-hand side of Fig. 2.

We also consider the case where implicit mouse movement feedback, in addition to the clicks, is available for the relevance prediction. The implicit browsing behavior on an image i is encoded as a feature vector \mathbf{x}_i . We assume that the relevance of an image is related to what is common between the explicit clicks and the implicit mouse movements. Therefore, we consider a model where both the click \mathbf{y} and movement features \mathbf{x} are generated by the common latent relevance variables \mathbf{z} . The linear mapping from \mathbf{z} to \mathbf{x} is a parameterized by a latent matrix \mathbf{W} . The model structure, where two observed variables are generated by a common latent variable, is similar to the Bayesian CCA (see [6]). This variant of the model is depicted on the right-hand side of Fig. 2.

To summarize, we make the following distributional assumptions:

$$\begin{aligned} \mathbf{z}_i | \mathbf{Q}, \mathbf{f} &\sim N(\mathbf{Q}\mathbf{f}_i, \sigma^2 \mathbf{I}) \\ \mathbf{x}_i | \mathbf{W}, \mathbf{z} &\sim N(\mathbf{W}\mathbf{z}_i, \sigma_x^2 \mathbf{I}) \\ \mathbf{y} | \boldsymbol{\alpha}, \mathbf{z} &\sim \text{Multinomial}([\dots, \frac{\exp(\boldsymbol{\alpha}^T \mathbf{z}_i)}{\sum_j \exp(\boldsymbol{\alpha}^T \mathbf{z}_j)}, \dots]) \end{aligned}$$

Columns of \mathbf{W} and \mathbf{Q} and the vector $\boldsymbol{\alpha}$ are drawn from Gaussian distributions with zero mean and diagonal variance.

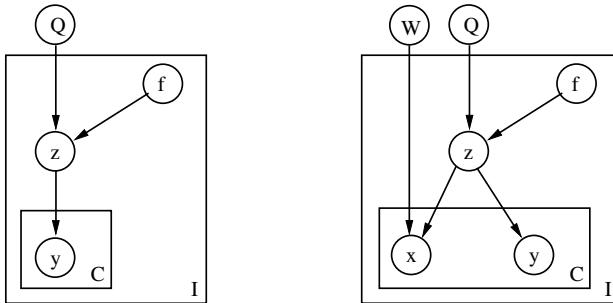


Fig. 2. Plate diagram of the relevance prediction model. The simpler model on the left uses only information about which documents were clicked to give explicit feedback for the prediction. The model on the right includes mouse movement features \mathbf{x} , as well. The plate I is over the images and the plate C is over the feedback rounds.

We use adaptive MCMC for inferring the relevances. The sampler is initialized from the MAP estimate.

3 Experiments

We tested the model in a video recall task. Three test subjects were asked to view a 17 minute video and, after viewing, recall certain events from it. To aid the recall, the test subjects were able to browse a timeline of snapshots from the video. We recorded the mouse movements and clicks during the browsing. The test subjects completed six recall tasks where they were asked to find certain number of scenes with a specific set of people in specific places. Specifically, they were asked to select one image per suitable shot by clicking it and, after selecting enough images, to shortly describe out loud what they remember about each selected scene. This video recall setup was chosen to simulate retrieval from a personal database, where the content is already somewhat familiar beforehand.

The test subjects were randomly assigned to one of three interface conditions (Fig. 1) in each task. The first condition was the interface described in Sec. 2, which dynamically changes the size of the images to reflect the predicted relevance. We will refer to this interface as *proactive* below. To quantify the effect of altering image saliency during the search, we had a second interface (called *zooming*) which was otherwise identical to the first except that the sizes of the images were not changed in according to the relevance predictions. The third interface (called *scrollable*) was a simple baseline where images were shown in a static size on a scrollable window. The order of the tasks and interface conditions was balanced between the test subjects.

The video was compressed into a series of snapshots taken at 5 second intervals. The timeline, which the users saw, included every second snapshot. The rest of the snapshots, which were not shown, were held out as a test set. There were 104 and 105 images in the timeline and hold out sets, respectively.

The image content was encoded as binary indicators of specific people, objects and locations in the image. For this experiments, the features were constructed by manually tagging the images. Similar features could be constructed without manual tagging using face and object recognition algorithms with the expense of a lower accuracy. For example, in lifelogging type of applications a wearable recording device can recognize people and objects in the image [1].

3.1 Comparing the Interfaces

An interface is efficient if it allows a quick access to the relevant images without having to view too many non-relevant images. We measure the effort required to find the relevant images among the non-relevant ones using the standard information retrieval measure of mean average precision. We consider the viewed images as an ordered list, where the clicked images are labeled as positive and all others as negative. The average precision is the average of the precisions computed at each positive rank. In the proactive and zooming interfaces, an image is considered viewed when the user moves the mouse over it. In the scrollable interface, an image is considered viewed when it is scrolled into the view.

Figure 3 shows the mean average precisions for the three interfaces in each task. We are interested in comparing the zooming and the proactive interfaces,

which behave equally until the first click. Therefore, the figure displays values computed after discarding all views up to and including the first clicked image.

The proactive interface attains higher mean average precision than the zooming interface in all tasks. This shows that modulating the saliency according to the relevance estimates helps the users to locate the relevant images faster.

The performance of the scrollable interface has the largest variance between the tasks. In task number 2 it is clearly better than the alternatives. The reason is that the scrolling interface is initially showing the first images on the timeline and there happened to be a few scenes which are relevant in the second task, in the beginning of the video. In five tasks the scrollable interface is the worst by a large margin. They correspond to the typical case where one has to scroll through many images before finding relevant ones, whereas in the other two interfaces it is possible to easily skip over a sequence of images.

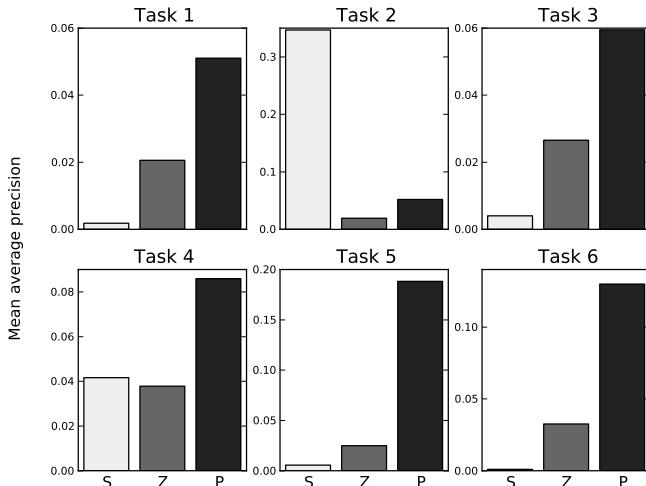


Fig. 3. Mean average precisions in image viewing sequences after the first click for the scrolling (S), zooming (Z) and proactive (P) interfaces. Higher values mean that fewer non-relevant images were viewed between the relevant (clicked) images. The proactive interface is better in all tasks than the zooming interface, which is identical except for the prediction.

3.2 Integrating Explicit and Implicit Feedback

We also study if implicit feedback from the mouse movement patterns can improve inference over the explicit clicks. We encode the implicit feedback as a feature vector for each image. The features are listed in Table 1. The evidence from the explicit and implicit feedback is combined using the model from Sec. 2.1.

We compared the performances of the combined feedback and the explicit-only models. They were trained using data collected with the zooming and proactive interfaces. The data from the scrollable condition is discarded because, in that condition, the mouse is mostly used to interact with the scrollbar and less with the images.

Table 1. Implicit mouse movement features designed for capturing the browsing patterns

Description	Type
Number of visits to the image	Integer
Total hover duration on the image (ms)	Continuous
Was this image already visited during the last 15 visits?	Binary
Both left and right neighbors visited during the last 4 visits	Binary
Average hovering duration on the previous 2 images (ms)	Continuous

To select the dimensionality of the latent variable z we train models of different dimensionality on the observations excluding the last click of a task, and compare the log-likelihoods when predicting the last click. The log-likelihoods averaged over the tasks are plotted in Fig. 4. If the dimensionality is too low (less than 6), the model is not flexible enough to model both the features x and the clicks y , and hence the performance of the combined feedback variant is much worse than that of the explicit-only model. When the dimensionality is 6, including the implicit feedback improves the performance slightly. Combined feedback model with $\dim(z) = 8$ has the best log-likelihood. In summary, the performance is improved when the implicit feedback is combined with explicit feedback.

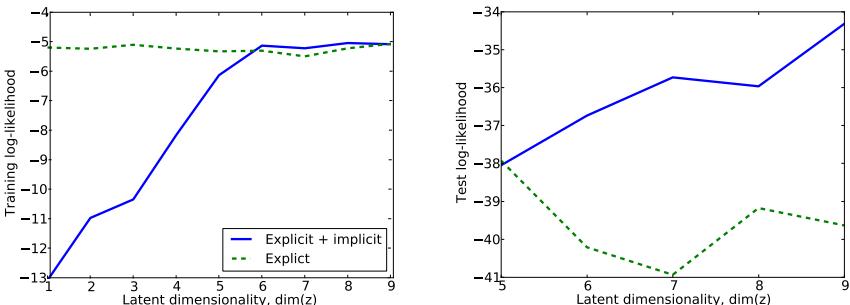


Fig. 4. Average log-likelihood of the clicks y for training (on left) and testing (on right) sets as a function of $\dim(z)$, the dimensionality of the latent variable. Larger values mean better prediction performance.

4 Discussion

We have introduced an image browser that modulates the saliency (size) of images according to their predicted relevance. The relevance is estimated online by observing explicit and implicit mouse interaction patterns. We performed a small scale user study where we showed that changing the image size in proportion to the relevance predictions helps in finding the relevant images with less effort. We also showed that complementing explicit feedback with implicit mouse movement patterns improves the relevance prediction further.

Acknowledgements. The authors belong to Adaptive Informatics Research Centre at Aalto University. This work has been funded by Aalto MIDE programme (project UI-ART) and by the PASCAL2 Network of Excellence, ICT 216886.

References

1. Ajanki, A., Billinghurst, M., Gamper, H., Järvenpää, T., Kandemir, M., Kaski, S., Koskela, M., Kurimo, M., Laaksonen, J., Puolamäki, K., Ruokolainen, T., Tossavainen, T.: Contextual information access with augmented reality. In: Proc. MLSP 2010, pp. 95–100. IEEE Press, New York (2010)
2. Datta, R., Joshi, D., Li, J., Wang, J.Z.: Image retrieval: Ideas, influences and trends of the new age. *ACM Comput. Surv.* 40(5), 1–5, 60 (2008)
3. Dumais, S., Cutrell, E., Cadiz, J., Jancke, G., Sarin, R., Robbins, D.C.: Stuff I've seen: a system for personal information retrieval and re-use. In: Proc. SIGIR 2003, pp. 72–79. ACM Press, New York (2003)
4. Faro, A., Giordano, D., Pino, C., Spampinato, C.: Visual attention for implicit relevance feedback in a content based image retrieval. In: 6th Symposium on Eye-Tracking Research & Applications, pp. 73–76. ACM Press, New York (2010)
5. Gemmel, J., Bell, G., Lueder, R.: MyLifeBits: a personal database for everything. *Commun. ACM* 49, 88–95 (2006)
6. Klami, A., Kaski, S.: Probabilistic approach to detecting dependencies between data sets. *Neurocomput.* 72, 39–46 (2008)
7. Kozma, L., Klami, A., Kaski, S.: GaZIR: Gaze-based zooming interface for image retrieval. In: Proc. The Eleventh International Conference on Multimodal Interfaces and The Sixth Workshop on Machine Learning for Multimodal Interaction (ICMI-MLMI), pp. 305–312. ACM Press, New York (2009)
8. Oyekoya, O., Stentiford, F.: Perceptual image retrieval using eye movements. In: Zheng, N., Jiang, X., Lan, X. (eds.) IWICPAS 2006. LNCS, vol. 4153, pp. 281–289. Springer, Heidelberg (2006)
9. Puolamäki, K., Ajanki, A., Kaski, S.: Learning to learn implicit queries from gaze patterns. In: Proc. ICML 2008. pp. 760–767. Omnipress, Madison (2008)
10. Puolamäki, K., Salojärvi, J., Savia, E., Simola, J., Kaski, S.: Combining eye movements and collaborative filtering for proactive information retrieval. In: Proc. SIGIR 2005, pp. 146–153. ACM Press, New York (2005)
11. Ward, D.J., MacKay, D.J.: Fast hands-free writing by gaze direction. *Nature* 418, 838 (2002)

Person Tracking Based on a Hybrid Neural Probabilistic Model

Wenjie Yan, Cornelius Weber, and Stefan Wermter

University of Hamburg, Department of Informatics, Knowledge Technology
Vogt-Kölln-Straße 30, D - 22527 Hamburg, Germany
{yan,weber,wermter}@informatik.uni-hamburg.de
<http://www.informatik.uni-hamburg.de/WTM/>

Abstract. This article presents a novel approach for a real-time person tracking system based on particle filters that use different visual streams. Due to the difficulty of detecting a person from a top view, a new architecture is presented that integrates different vision streams by means of a Sigma-Pi network. A short-term memory mechanism enhances the tracking robustness. Experimental results show that robust real-time person tracking can be achieved.

Keywords: Person detection, particle filter, neural network, multimodality.

1 Introduction

Artificial neural networks (ANN) are widely used to model complex behavior and are applied in different fields, such as computer vision, pattern recognition, and classification. They can also be used to overcome the major challenge of real-time person tracking in a complex ambient intelligent environment. In this paper we present a novel approach of indoor person tracking using a single ceiling-mounted camera with a fish-eye lens.

A few person tracking systems based on ceiling mounted cameras have been proposed previously [6],[12]. However, it is hard to get a robust tracking ability based on a single feature. A person observed from the top view produces very different shapes at different locations thus it is difficult to be recognized by fixed patterns. Motion provides a good tracking indicator but cannot provide information when a person does not move. The color obtained from the clothes can be a reliable tracking feature, but we have to learn the color information first from other information. Different vision information in combination, however, can be used to detect and localize a person's position reliably.

A hybrid knowledge-based architecture tackles the specific challenges that arise from this setup by integrating different vision streams into a Sigma-Pi network [13]. A person can be localized using a particle filter based on the output of this network. The system architecture and the used methods are presented in section 2 and 3. The experimental results are shown in section 4. A discussion is presented in section 5 and section 6 concludes this article.

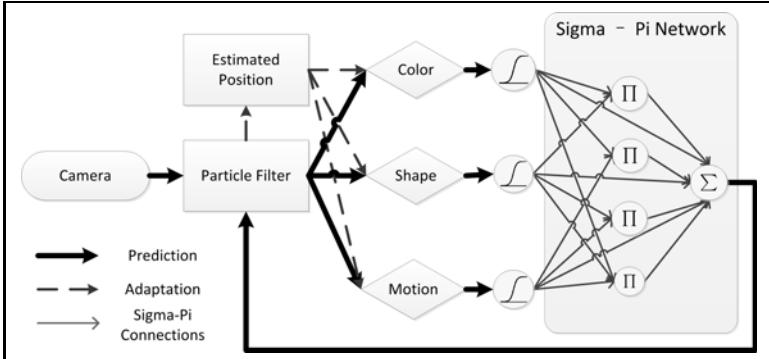


Fig. 1. System architecture

2 Methods

Our model is illustrated in Figure 1. A Sigma-Pi network integrates shape, motion and color streams and passes its output to a particle filter, which provides robust object tracking based on the history of previous observations [10]. The work flow can be split into two parts: *prediction* and *adaptation*. In the prediction phase (see arrows in Figure 1), each particle segments a small image patch and evaluates this patch using three visual cues. The activities of visual cues are generated via activation functions and scaled by their connection weights, which are called reliabilities here. Through the Sigma-Pi network, the weights of particles are computed and then the position of the particles will be updated. In the adaptation phase, the reliability weights of the Sigma-Pi network will be adapted. The estimated position of the person will be validated again using the visual cues (see arrows for adaptation) and weights will be calculated based on the validation results. With the collaborative contribution of each cue, the tracking performance can be improved significantly.

2.1 Particle Filter

Particle filters are an approximation method that represents a probability distribution with a set of particles and weight values. A particle filter is usually integrated in partially observable Markov decision processes (POMDPs) [5]. A POMDP model consists of unobserved states of an agent s , in our case the position of the observed person, and observations of the agent z . A transition model $P(s_t|s_{t-1})$ describes the probability that the state changes from s_{t-1} to s_t at time t . If the agent executes the action a_{t-1} , $P(s_t|s_{t-1}, a_{t-1})$ can be estimated based on the transition model. For simplicity, let us assume here that we do not know anything about the person's actions. Based on the Bayesian formulation, the agent's state can be estimated according to an iterative equation:

$$P(s_t|z_{0:t}) = \eta P(z_t|s_t) \int P(s_{t-1}|z_{0:t-1})P(s_t|s_{t-1})ds_{t-1} \quad (1)$$

where η is a normalization constant, $P(z_t|s_t)$ is the observation model and $P(s_t|z_{0:t})$ is the probability of a state given all previous observations from time 0 to t . In a discrete model, the probability of the state s_t can be computed recursively from the previous distribution $P(s_{t-1}|z_{0:t-1})$:

$$P(s_t|z_{0:t}) \approx \eta P(z_t|s_t) \sum_i \pi_{t-1}^{(i)} P(s_t|s_{t-1}^{(i)}) \quad (2)$$

In the particle filter, the probability distribution can be approximated with a set of particles i in the following form:

$$P(s_t|z_{0:t}) \approx \sum_i \pi_{t-1}^{(i)} \delta(s_t - s_{t-1}^{(i)}) \quad (3)$$

where $\pi^{(i)}$ denotes the weight factor of each particle with $\sum \pi^{(i)} = 1$ and δ denotes the Dirac impulse function. The mean value of the distribution can be computed as $\sum_i \pi_{t-1}^{(i)} s_t$ and may be used to estimate the state of the agent if the distribution is unimodal.

At the beginning of the tracking, the particles are placed randomly in the image. Then a small patch surrounding them is taken and probed to detect the person with the visual cues. Where the sum of weighted cues returns large saliencies, the particles will get larger weight values, raising the probability of this particle in the distribution and showing that a person is more likely to be in this position. In order to keep the network exploring, 5% particles are assigned to random positions in each step.

2.2 Sigma-Pi Network

In the tracking system, the weight factor $\pi^{(i)}$ of particle i will be computed with a Sigma-Pi network [13]. The activities of the different visual cues are set as the input of the Sigma-Pi network and the weights are calculated with the following equation:

$$\begin{aligned} \pi^{(i)} = & \sum_c^3 \alpha_c^l(t) A_c(s_{t-1}^{(i)}) + \sum_{c_1 > c_2}^3 \alpha_{c_1 c_2}^q(t) A_{c_1}(s_{t-1}^{(i)}) A_{c_2}(s_{t-1}^{(i)}) \\ & + \alpha_{c_3}^c(t) A_{c_1}(s_{t-1}^{(i)}) A_{c_2}(s_{t-1}^{(i)}) A_{c_3}(s_{t-1}^{(i)}) \end{aligned} \quad (4)$$

where $A_c(s_{t-1}^{(i)}) \in [0, 1]$ is the activity of cue c at the position of particle i , which can be thought of as taken from a saliency map over the entire image [4]. The network weights $\alpha_c^l(t)$ denote the linear reliability and $\alpha_{c_1 c_2}^q(t)$ and $\alpha_{c_3}^c(t)$ are the quadratic and cubic combination reliabilities of the different visual cues. Compared with traditional multi-layer networks, the Sigma-Pi network contains the correlation and higher-order correlation information between the input values. The reliability of some cues, like motion, are non-adaptive, while others, like color, need to be adapted on a short time scale. This requires a mixed adaptive framework, as inspired by models of combining different information [11], [2]. An

issue is that an adaptive cue will be initially unreliable, but when learned may have a high quality in predicting the person's position. To balance the changing qualities between the different cues, the reliabilities will be evaluated with the following equation:

$$\alpha(t) = (1 - \epsilon)\alpha(t - 1) + \epsilon f(s'_t) + \beta \quad (5)$$

where ϵ is a constant learning rate and β is a constant. $f(s'_t)$ denotes an evaluation function and is computed by the combination of visual cues' activities:

$$f_c(s'_t) = \sum_{i \neq c}^n A_i(s'_t) A_c(s'_t) \quad (6)$$

where s'_t is the estimated position and n is the number of the reliabilities. In this model n is 7 and contains 3 linear and 4 combination reliabilities. The function is large when more cues are active at the same time, which leads to an increase of the cue's reliability α .

3 Processing Different Visual Cues

The image patches segmented by the particles are evaluated by the visual cues. Three independent cues, *motion*, *shape* and *color* are used in this model to extract different features from the image.

Motion detection is a method to detect an object by measuring the difference between images. We use here the background subtraction method [7] that compares the actual image with a reference image. Since the background stays mostly constant, the person can be found when the difference of image is larger than a predefined threshold. Considering that the background may also change, as when furniture is being moved, the background is updated smoothly using a running average. When the new input image remains static for a longer time, for example a person sits in a chair, the background will be converted to the new image, the person will merge into the background and then he will not be detected anymore. In this case, the shape and color cue will allow the system to find the person.

Color is an important feature for representing an object. Because the color of objects and people does not change quickly, it is a reliable feature for tracking. The image is converted to the HSV color space to reduce the computation effort [8]. Using a histogram backprojection algorithm [9], a saliency value image is generated that shows the probability of the pixels of the input image that belong to the example histogram. For each particle, the pixel values of the probability image inside of the segmentation window are accumulated. The higher the value is, the more this image segment matches the histogram pattern.

Since *shape* contains information irrelevant of the light condition as well as the surface texture, it represents significant features of an object. We extract here SURF features [1] for describing the image objects. Because the shape of a person from the top view changes significantly and is hard to be described

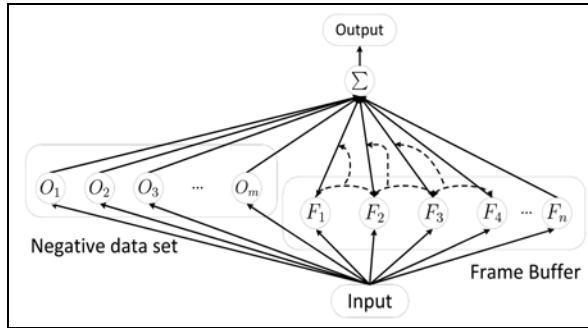


Fig. 2. Shape cue

by static patterns, a short time memory mechanism is conceived to track the person based on previous features. A feature buffer stores the image features of the last 10 frames. The correlations between the new input image feature and the features in the 10 frames are calculated. Based on these values, the output activity is calculated via an activation function. If the change of the person's shape is continuous and slow, the features of neighboring frames in the buffer should be similar. Weights of the buffer images are calculated using the matching rates between the adjacent frames. Features from a negative background data set such as sofas, tables and chairs have a negative contribution to the shape cue, which helps the particles to avoid the background.

4 Experimental Results

The environment for testing the tracking system is shown in Figure 3. The camera image is calibrated and subsampled to the resolution 320×240 , which allows real-time processing. Image material from 6 videos have been tested. The experiment aims to detect and locate a person or a mobile robot under static condition in the image as well as to track their motion trajectories when moving. One person will be tracked in the experiment. Different image noises, for example when changing the furniture's position, changing the person's appearance and also disturbance by another person are tested. 50 particles were used for the person tracking and therefore only a small part of the images is being processed. This accelerates the system in comparison with a search window method.

4.1 Tracking a Person

The mechanism of the person tracking system is demonstrated in Figure 3. The particles are initialized at random positions in the image. When a person enters the room, the weight values of the nearby particles will increase so that the particles move towards the person. The shape feature as well as the color histogram will adapt themselves at the same time. When a person does not move, when sitting as in Figure 4, the motion cue is missed but then the shape and color memory will recover the system to detect the person.

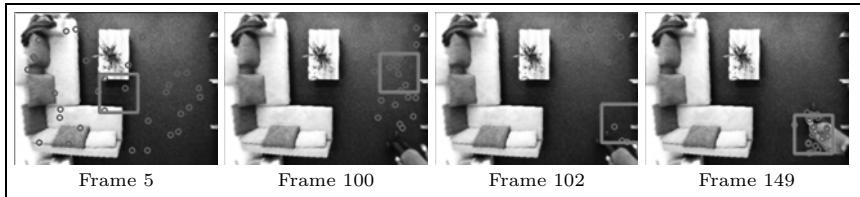


Fig. 3. Tracking a person moving into the room

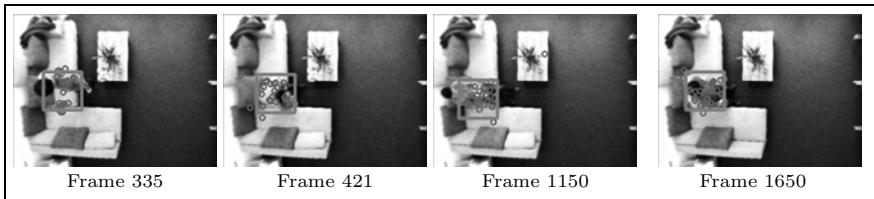


Fig. 4. Tracking a sitting person

4.2 Changing Environment

The disturbance of a changing environment, for example a moving table in the room (Figure 5) will automatically be corrected by the negative feedback of the shape cue. Although the particles may follow the motion cue, the shape of the table from the background model returns a negative feedback to the shape cue, which helps the particles to go back to the person. The experimental results are summarized in Table 1. 90.28% of the images on average are tracked correctly. In comparison, the success rate of tracking a person based on single motion detection could reach only 69% on average.



Fig. 5. Person tracking during change of environment

5 Discussion

We have presented a new hybrid neural probabilistic model that adapts its behavior online based on different visual cues. The model is to some extent indicative of a human's ability of recognizing objects based on different features. When

Table 1. Experiment results

Name	Total Frames	Missing	Mismatch	Success rate (%)
Person Moving 1	2012	19	22	97.96
Person Moving 2	2258	169	12	91.98
Person Moving and Sitting 1	1190	78	21	91.68
Person Moving and Sitting 2	980	22	130	84.18
Change Environment 1	1151	89	30	89.66
Change Environment and Distracter Person 1	1564	157	141	80.94
Total	9155	534	356	90.28

some of the features are strongly distorted, detection recovers by the integration of other features. The particle filter parallels an active attention selection mechanism, which allocates most processing resources to positions of interest. It has a high performance of detecting complex objects that move relatively slowly in real time. Accordingly, our model has potential as a robust method for object detection and recognition in complex conditions. It may in the future be better if the system tracks a person not only based on these three cues, but also on some further features. Also, non-visual sensors could be used such as a microphone, which provides auditory data to enhance the tracking accuracy.

The short-term memory enables the system to localize objects rapidly without a-priori knowledge about the target person. We have experimented with a multilayer perceptron network based on moment invariant features [3] that was trained to recognize a person. However, due to the variety of the person's shape observed from the top view and its similarity to the furniture, this method was not efficient to distinguish the person from the background. Nevertheless, we are considering to include a person-specific cue in the future.

6 Conclusions

In this paper we have presented a novel approach for real-time person tracking based on a ceiling-mounted camera. A hybrid probabilistic algorithm is proposed for localizing the person based on different visual cues. A Sigma-Pi architecture integrates the output of different cues together with corresponding reliability factors. Advantages of this system are that the feature pattern used for one cue, such as the color histogram, can adapt on-line to provide a more robust identification of a person. With this short-term memory mechanism, the system processes images from an unstructured environment as well as moving objects in a real ambient intelligent system. We are planning to generalize this architecture with a recurrent memory neural network and improve the quality of visual cues to obtain higher tracking precision and extend the functions for detecting the pose of a person.

Acknowledgements. This research has been partially supported by the KSERA project funded by the European Commission under the 7th Framework Programme (FP7) for Research and Technological Development under grant agreement n°2010-248085, and the EU project RobotDoc under 235065 ROBOT-DOC from the 7th Framework Programme, Marie Curie Action ITN.

References

1. Bay, H., Tuytelaars, T., Van Gool, L.: Surf: Speeded up robust features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006)
2. Bernardin, K., Gehrig, T., Stiefelhagen, R.: Multi-level particle filter fusion of features and cues for audio-visual person tracking. In: Stiefelhagen, R., Bowers, R., Fiscus, J.G. (eds.) RT 2007 and CLEAR 2007. LNCS, vol. 4625, pp. 70–81. Springer, Heidelberg (2008)
3. Hu, M.-K.: Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory* 8(2), 179–187 (1962)
4. Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20(11), 1254–1259 (1998)
5. Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101(1-2), 99–134 (1998)
6. Nait-Charif, H., McKenna, S.J.: Activity summarisation and fall detection in a supportive home environment. In: Proceedings of the 17th International Conference on Pattern Recognition, vol. 4, pp. 323–326 (2004)
7. Piccardi, M.: Background subtraction techniques: a review. In: IEEE International Conference on Systems, Man and Cybernetics, vol. 4, pp. 3099–3104 (2004)
8. Sural, S., Qian, G., Pramanik, S.: Segmentation and histogram generation using the hsv color space for image retrieval. In: International Conference on Image Processing, vol. 2, pp. 589–592 (2002)
9. Swain, M.J., Ballard, D.H.: Color indexing. *International Journal of Computer Vision* 7, 11–32 (1991)
10. Thrun, S.: Particle filters in robotics. In: Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI), vol. 1 (2002)
11. Triesch, J., Malsburg, C.: Democratic integration: Self-organized integration of adaptive cues. *Neural Computation* 13(9), 2049–2074 (2001)
12. West, G., Newman, C., Greenhill, S.: Using a Camera to Implement Virtual Sensors in a Smart House. In: From Smart Homes to Smart Care, pp. 83–90 (2005)
13. Zhang, H.B., Muhlenbein: Synthesis of Sigma-Pi neural networks by the breeder genetic programming. In: Proceedings of the First IEEE Conference on Evolutionary Computation, vol. 1, pp. 318–323 (June 1994)

Gaze- and Speech-Enhanced Content-Based Image Retrieval in Image Tagging

He Zhang¹, Teemu Ruokolainen¹, Jorma Laaksonen¹,
Christina Hochleitner², and Rudolf Traunmüller²

¹ Department of Information and Computer Science
Aalto University School of Science, Espoo, Finland

{he.zhang,teemu.ruokolainen,jorma.laaksonen}@aalto.fi
² celum gmbh., Linz, Austria

{christina.hochleitner,rudolf.traunmueller}@celum.com

Abstract. We describe a setup and experiments where users are checking and correcting image tags given by an automatic tagging system. We study how much the application of a content-based image retrieval (CBIR) method speeds up the process of finding and correcting the erroneously-tagged images. We also analyze the use of implicit relevance feedback from the user's gaze tracking patterns as a method for boosting up the CBIR performance. Finally, we use automatic speech recognition for giving the correct tags for those images that were wrongly tagged. The experiments show a large variance in the tagging task performance, which we believe is primarily caused by the users' subjectivity in image contents as well as their varying familiarity with the gaze tracking and speech recognition setups. The results suggest potentials for gaze and/or speech enhanced CBIR method in image tagging, at least for some users.

Keywords: Content-based image retrieval, automatic image tagging, gaze tracking, speech recognition.

1 Introduction

Automatic tagging is a useful but still not fully reliable means for associating keyword-type information to unannotated images. In the current state of art, human effort is still needed for checking and correcting the tags [8,1]. The tag correction process can be seen as a special case of content-based image retrieval (CBIR) where the goal is to quickly correct the erroneously-tagged images.

We present the results of a study that has addressed the image tag correction and assignment task from a multitude of viewpoints. 1) we have studied whether neural-network-based CBIR techniques can speed up finding of erroneously-tagged images. 2) we have used implicit relevance feedback from the user's gaze tracking patterns to boost the performance of CBIR compared to using only explicit feedback from pointer clicks. 3) we have applied automatic speech recognition for providing corrected image tags. Section 2 describes the CBIR system and Section 3 the automatic speech recognizer used in the experiment. The results are presented in Section 4 and conclusions drawn in Section 5.

Table 1. PicSOM’s visual features extracted from images [11]

Feature	Tiling	Dim.
DCT coefficients of average color in rectangular grid	global	12
CIE L*a*b* color of two dominant color clusters	global	6
Histogram of local edge statistics	4×4	80
Haar transform of quantized HSV color histogram	global	256
Average CIE L*a*b* color	5	15
Three central moments of CIE L*a*b* color distribution	5	45
Histogram of four Sobel edge directions	5	20
Co-occurrence matrix of four Sobel edge directions	5	80
Magnitude of the 16×16 FFT of Sobel edge image	global	128
Histogram of relative brightness of neighboring pixels	5	40
Histogram of interest point SIFT features	global	256

2 Gaze-Enhanced Content-Based Image Retrieval

The typical setting for interactive content-based image retrieval is that the user identifies images that are the most relevant for his current search task (or visually the most similar ones to the target image he has in mind) and passes this information to the system as relevance feedback on the images the system has retrieved. One can differentiate between positive and negative relevance feedback. The former is given for the relevant images and the user is expecting to see more similar ones, whereas the latter is given for the non-relevant images the user does not want to view. Based on the relevance feedback the system is then able to find more similar images and present them to the user (see for example [3] for an extensive survey).

2.1 PicSOM CBIR System

PicSOM¹ [7] is a content-based image retrieval system developed since 1998, first at the Helsinki University of Technology and then at the Aalto University. PicSOM uses the principles of *query by example* and *relevance feedback* in implementing iterative and interactive image retrieval.

The unique approach used in PicSOM is to have several Self-Organizing Maps (SOMs) [6] in parallel to index and determine the similarity of images. These parallel SOMs have been trained with separate data sets obtained by using different feature extraction algorithms on the same objects. The extracted image features and their dimensionalities are listed in Table 1.

As the SOM maps visually similar images near to each other, this motivates to spread the relevance feedback given for the viewed images to their neighboring images on the map surface. Images marked as relevant are first given positive and those marked as non-relevant are given negative values on the map surface. These relevance values are then smoothed and spread around with low-pass filtering. Images with the largest resulting relevance scores are then shown to the user.

¹ <http://www.cis.hut.fi/picsom>

Table 2. Eye movement features collected in PinView [5]

Number	Name	Description
1	numMeasurements	log of total time of viewing the image
2	numOutsideFix	total time for measurements outside fixations
3	ratioInsideOutside	percentage of measurements inside/outside fixations
4	speed	average distance between two consecutive measurements
5	coverage	number of subimages covered by measurements ¹
6	normCoverage	coverage normalized by numMeasurements
7	pupil	maximal pupil diameter during viewing
8	nJumps1	number of breaks ² longer than 60ms
9	nJumps2	number of breaks ² longer than 600ms
10	numFix	total number of fixations
11	meanFixLen	mean length of fixations
12	totalFixLen	total length of fixations
13	fixPrct	percentage of time spent in fixations
14	nJumpsFix	number of re-visits (regressions) to the image
15	maxAngle	maximal angle between two consecutive saccades ³
16	firstFixLen	length of the first fixation
17	firstFixNum	number of fixations during the first visit
18	distPrev	distance to the fixation before the first visit
19	durPrev	duration of the fixation before the first visit

¹The image was divided into a regular grid of 4×4 subimages, and covering a subimage means that at least one measurement falls within it.

²A sequence of measurements outside the image occurring between two consecutive measurements within the image.

³A transition from one fixation to another.

2.2 Using Gaze Patterns as Implicit Relevance Feedback

The PinView project² has studied the use of gaze patterns as a form of on-line implicit relevance feedback in interactive CBIR [2]. Based on the analyzed gaze patterns we calculate for each viewed image a 19-dimensional feature vector as specified in Table 2. Based on these features, relevance predictions for the images are obtained with a simple logistic regression model created with separate training data.

In the PicSOM system, the gaze-based implicit relevance estimates are combined with the click-based explicit relevance feedback values. In this process the gaze-based regressor outputs are always in the range of $[0, 1]$ and the larger the value, the more probably the image is relevant. These values are then summed with the $+1$ and -1 values given for the clicked and non-clicked images, respectively. The combined relevance values are finally placed in the SOM units and spread to their neighbors with low-pass filtering similarly to PicSOM's normal operation.

² <http://www.pinview.eu/>

2.3 CBIR-Assisted Image Tag Correction

Let us consider a CBIR setup where the viewed images are such that an automatic image annotation system has assigned all of them some particular tag or keyword based on their visual properties. The considered images are thus visually quite similar to each other, but due to imperfections in the assignment, there are bound to be semantic differences or tagging errors among them. The burden of a user who needs to check and correct the automatically assigned tags would be eased if the wrongly-tagged images could be found as early as possible.

This can be understood as a complementary setting for the conventional interactive CBIR setting. Now the relevant images are not those that resemble the target image, but those that are semantically different from the other, correctly-tagged ones. Nevertheless, CBIR techniques can be used to speed up retrieving of such images. This time, the search will be driven more by the negative relevance feedback, now given to the correctly-tagged images. The system will then retrieve more and more images that are different from the typical correctly-tagged images and are thus more likely to be the wrongly-tagged ones.

3 Automatic Speech Recognition

Automatic speech recognition (ASR) aims at providing a textual transcript for a given uttered speech signal. The acoustic and language models and the lexicon have been trained earlier with available training corpora. During recognition, spectral feature vectors are first extracted from the speech signal at regular time intervals. Then, given the acoustic information and the probabilities provided by the models, the decoder finds the best transcript hypothesis. Finally, the best hypothesis is output as the recognition result.

The speech recognition system used in the experiments has been developed in the Department of Information and Computer Science at Aalto University. The speech signal is sampled using 16 kHz sampling rate and 16 bits. The signal is then represented with 12 MFCC (mel-frequency cepstral coefficients) and the log-energy along with their first and second differentials. Above features are calculated in 16 ms windows with 8 ms overlap. Cepstral mean subtraction (CMS) and a maximum likelihood linear transformation, which is estimated in training, are applied to the features.

For the acoustic model, we use state-clustered Hidden Markov triphone models that have 5062 states modeled with 32 Gaussians. State durations are modeled with gamma probability functions described in [10]. The Hidden Markov model was trained with Wall Street Journal (WSJ) speech corpus; the WSJ corpus consists of dictated newswire articles spoken by American English speakers.

The speech recognition system supports the n -gram models [9] as the default language model type. Therefore, the language model based on the keyword list of the experiment is created in the following straightforward manner: each keyword is considered a word in the vocabulary and each word is assigned an equal probability. In other words, we model the keyword list as a unigram model with equal probability assigned to each word.

4 Experiments

The experiments were conducted to validate the relative performance of different variants of the system and to verify how well the system works in practice.

Data. We used the *train* subset of the PASCAL Visual Object Classes Challenge 2007 (VOC2007) data set [4] with a total of 2501 annotated images that cover 20 overlapping categories. To ease the burden of users, we randomly selected 16 categories and divided them into two groups:

1. correctly-tagged: *car, dog, bicycle, person, motorbike, train*
2. wrongly-tagged: *sheep, horse, aeroplane, boat, bus, bottle, dining table, potted plant, sofa, tv-monitor*

Experiment setup. We recruited 18 test subjects both males and females from several departments at the Aalto University. The mean age of the test subjects was 27.2 years old, ranging from 23 to 34 with good balances in between. Almost none of the users had experiences in image tagging and only one user had experiences in gaze tracking.

Each subject was asked to perform six tagging tasks. For every task, the user had to check and correct the tags of one particular category. Before each task, the system randomly selected 40 images of that category and another 40 images of the ten categories from the wrongly-tagged class. Thus half of the images were always tagged correctly. During each task, the system showed a total of 40 images, contained in five image pages each having eight images. After each task, the user was asked of his or her subjective opinions whether the corresponding variant facilitated the tagging task, and whether it was reliable and fast enough.

Feedback modalities. The following relevance feedback modality types or variants of the system were compared:

1. *Baseline*: The user corrects the image tag by selecting the corresponding category name from the drop-down menu under the image. No CBIR or speech recognition techniques are used.
2. *Explicit*: The user clicks the pointer over the wrongly-tagged image and speaks the desired category name into the microphone. Only explicit relevance feedback from pointer clicks are used.
3. *Implicit*: The tag correction is similar as in *explicit*. However, the user's eye movements are unobtrusively recorded by a Tobii eye tracker³. Both explicit pointer relevance and implicit gaze relevance feedback are used.

For the baseline variant all the 40 images presented to the user were randomly chosen, whereas for the other two variants only the eight images in the first page were random while the images in the remaining four pages were selected by the relevance feedback information.

The evaluation and results of image retrieval. The measure of performance is the number of images that the user corrects in one tagging task, which gives reflection on how well the system retrieves wrongly-tagged images. Table 3a

³ <http://www.tobii.com/>

gives the quantitative performance of the three variants for each user. Although the relative performance of the variants varies between users, it is clear that the explicit and implicit feedback variants are better than the baseline (see *t*-test values in Table 3c). The difference between the explicit variant using only clicks and the implicit variant using also gaze is not significant (with a pairwise *t*-test *p*-value of 0.1901), though the former on average ranks higher than the latter. The worst implicit result was that of user 8 with whom the gaze tracking obviously failed. For users 10 and 12, the implicit variant of the system retrieved about 17% more of the wrongly-tagged images than the explicit variant did.

Table 3b gives the quantitative performance for each tagging category averaged over all the users. Similarly, the performances of the explicit and implicit variants are better than that of the baseline type (see *t*-test values in Table 3d), except for the *motorbike* category. The reason is probably because of the overlapping categories of the images in the VOC2007 database. For example, an image tagged as *motorbike* usually contains a person riding on it, which might cause users to tag it as *person*. Again the difference between the explicit variant and the implicit variant is not significant (with a pairwise *t*-test *p*-value of 0.6747), though the former slightly outperforms the latter. However, for the *person* category, the implicit variant of the system retrieved about 20% more of the wrongly-tagged images than the explicit variant did.

The evaluation and results of speech recognizer. Speech recognition accuracy is commonly reported with the word error rate (WER) defined as the number of erroneously recognized words divided by the total number of spoken words. In our experiments, obtaining the WER would have required a manual annotation of the speech recorded during the experiments, i.e. which word was uttered at which time instance, followed by a (partly) manual search for the corresponding recognized words from the speech recognizer's output.

Due to tediousness of such task, we instead investigated other means of evaluation; particularly, we consider the increase in the number of correctly labeled images after the annotations divided by the total number of annotations conducted by the users. This gives us an efficiency score with maximum value 1.0 in case each annotation results in a correctly labeled image; value zero in case there is no change in the number of correctly labeled images; and negative values in case the annotation results in a decrease in correctly labeled images.

Averaged over all the 18 users with four experiments using speech each, we obtain a mean efficiency of 0.36. This value means that, on average, a successful labeling required roughly three annotation trials. The highest efficiency achieved in an individual experiment was 0.60 and the lowest 0.05; the corresponding numbers of trials per successful labeling are roughly 1.5 and 20, respectively, indicating a large variance in user-wise performance. However, according to our observations, the reason for the varying performance need not be technical in nature. Instead, we suggest that the most important factor leading to performance variance is the lack of user experience with speech recognizers.

The evaluation of user experience. A close examination of the qualitative feedback from the users indicates that most of the test subjects (between 66%

Table 3. (a) The rounded average numbers of images that each user corrected when using the three variants of the system. The best performance(s) are marked in bold for each user. (b) The rounded average numbers of images corrected for each category averaged over 18 users. (c) Pairwise *t*-test *p*-values for variants in user-wise experiment. (d) Pairwise *t*-test *p*-values for variants in category-wise experiment

(a)				(b)			
User	Baseline	Explicit	Implicit	Category	Baseline	Explicit	Implicit
1	16	23	24	car	20	23	22
2	22	21	22	dog	22	28	26
3	26	25	26	bicycle	23	27	25
4	19	27	27	person	23	26	31
5	24	27	23	motorbike	26	24	23
6	23	27	26	train	22	26	23
7	25	25	26	Average	23	26	25
8	23	29	19				
9	23	26	24				
10	18	24	28				
11	15	26	22				
12	25	22	26				
13	22	28	27				
14	22	28	24				
15	27	25	22				
16	28	29	27				
17	27	28	25				
18	26	24	27				
Average	23	26	25				

		Explicit	Implicit
Baseline	0.0080	0.0812	
Explicit	—	0.1901	

		Explicit	Implicit
Baseline	0.0218	0.1665	
Explicit	—	0.6747	

and 75%) believed that all the variants help to facilitate the tagging tasks, though they had to spend extra efforts in adapting to the eye tracker and microphone. As for reliability, about 82% of the test subjects considered the explicit variant with speech input to be the most reliable one, while 56% marked the implicit variant, and only 50% marked the baseline variant to be reliable. As for speed, the implicit variant with gaze tracking received the highest vote of 64%, followed by the explicit variant of 56% and the baseline variant of 43%.

5 Conclusions

We investigated the use of neural-network-based CBIR system enhanced by gaze feedback modality and speech input for an image tagging task. Three relevance feedback variants were evaluated combined with automatic speech recognition being applied for providing the corrected image tags. Our results showed that both the explicit variant using pointer clicks and the implicit variant using gaze tracking patterns can to some extent speed up the search and correction of wrongly-tagged images, compared to the baseline variant with drop-down menus.

Based on the quantitative evaluation, the explicit variant generally outperformed the other two, whereas from the qualitative evaluation, the implicit variant was believed to have the highest speed among the three and was considered more reliable than the baseline, even though most of the test subjects reported discomfort in sitting still in front of the eye tracker and difficulties in adapting to the microphone. These results imply, in addition to the quantitative evaluation of the tagging system, that people's subjective characteristics and preferences vary with respect to willingness and ability to interact with novel multimodal interfaces. This will need to be taken into account in future studies.

Acknowledgments. The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007–2013) under *grant agreement* n° 216529, Personal Information Navigator Adapting Through Viewing, PinView.

References

1. Ames, M., Naaman, M.: Why we tag: motivations for annotation in mobile and online media. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 971–980. ACM, New York (2007)
2. Auer, P., Hussain, Z., Kaski, S., Klami, A., Kujala, J., Laaksonen, J., Leung, A.P., Pasupa, K., Shawe-Taylor, J.: Pinview: Implicit feedback in content-based image retrieval. In: Diethe, T., Cristianini, N., Shawe-Taylor, J. (eds.) Proceedings of Workshop on Applications of Pattern Analysis. JMLR Workshop and Conference Proceedings, vol. 11, pp. 51–57 (2010)
3. Datta, R., Joshi, D., Li, J., Wang, J.Z.: Image retrieval: Ideas, influences, and trends of the new age. ACM Computing Surveys 40(2), 1–60 (2008)
4. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results (2007), <http://www.pascal-network.org/challenges/VOC/voc2007/index.html>
5. Klami, A., Kaski, S., Pasupa, K., Saunders, C., de Campos, T.: Prediction of relevance of an image from a scan pattern. PinView FP7-216529 Project Deliverable Report D2.1 (December 2008), <http://www.pinview.eu/deliverables.php>
6. Kohonen, T.: Self-Organizing Maps, 3rd edn. Springer Series in Information Sciences, vol. 30. Springer, Berlin (2001)
7. Laaksonen, J., Koskela, M., Oja, E.: PicSOM—Self-organizing image retrieval with MPEG-7 content descriptions. IEEE Transactions on Neural Networks, Special Issue on Intelligent Multimedia Processing 13(4), 841–853 (2002)
8. Lerman, K., Jones, L.: Social browsing on flickr. CoRR abs/cs/0612047 (2006)
9. Manning, C., Schütze, H.: MITCogNet: Foundations of statistical natural language processing. MIT Press, Cambridge (1999)
10. Pylkkönen, J., Kurimo, M.: Duration modeling techniques for continuous speech recognition. In: Eighth International Conference on Spoken Language Processing, ISCA (2004)
11. Viitaniemi, V., Laaksonen, J.: Evaluating the performance in automatic image annotation: example case by adaptive fusion of global image features. Signal Processing: Image Communications 22(6), 557–568 (2007)

Modelling Hypothetical Wage Equation by Neural Networks

Jaakko Talonen and Miki Sirola

Aalto University School of Science,
P.O. Box 15400 FI-00076 Aalto, Finland
{talonen,miki}@cis.hut.fi
<http://www.cis.hut.fi/talonen>

Abstract. In this paper, a hypothetical wage equation is modelled using quarterly data from United Kingdom. Wage and price data have a great importance for the overall features of large-scale macro models and for example for the different policy actions. The modelled feature in this paper is the real wage, the differential of a nominal wage and a price index. In the variable selection phase, the stationary properties of the data were investigated by augmented Dickey-Fuller tests (ADF). The main idea in this paper is to present a neural network model, which has a better fit than conventional MLR model.

Keywords: Wage equation, Macroeconomics, Time Series Modelling, Neural Networks.

1 Introduction

The United Kingdom is made up of England, Wales, Scotland and Northern Ireland. In the economic literature, prices, inflation expectations, productivity and various measures of the situation on the labour market are often used as variables in wage formation models [1]. Six different variables were selected for the wage model. Five of them were downloaded from Eurostat [2], Source OECD [3] and Office for National Statistics UK databases [4]. The change rates of *nominal wage rate*, *consumer price index*, *unemployment rate*, *nominal income*, *employment* and *producer price index* are analyzed. From Fig. 1 it is possible to see that wages and price indices have been doubled in three decades. From the year 1995 the unemployment rate shows a tendency to fall in United Kingdom and some other European countries until the last economic crisis. This paper considers these questions in a model of wage and for example employment dynamics of the United Kingdom.

Selected modelling methods in this paper are multilinear regression (MLR) and Feed-Forward back-propagation Neural Network [5] (FF). MLR is a conventional way to solve this type of problem in macroeconomics. FF is selected, because it is used frequently in time series forecasting [6,7]. The selection of economic meters for interpretative and dependent variables in the model was performed by Microsoft Excel and PcGive in OxMetrics. Matlab was used in data analysis, visualization and modelling.

2 Description of Used Methods

2.1 Unit Root Tests

If a non-stationary series, Y_t must be differenced d times before it becomes stationary, then it is said to be integrated of order d . It is written $Y_t \sim I(d)$. The majority of economic and financial series contain a single unit root, $I(1)$ [8]. Complications arise, when variables are cointegrated in different orders, which means that two or more time series do not share a common type of stochastic drift. For example, sometimes price indices must be differenced two times before it becomes stationary. The concept of cointegration is used to construct the model. First, it is sensible to investigate the stationary properties of the individual series. The early work on testing for a unit root in time series was done by Dickey and Fuller.

$$Y_t = \rho Y_{t-1} + \epsilon_t, \quad (1)$$

The time series Y_t converges to a stationary time series, if the coefficient of Y_{t-1} is $|\rho| < 1$. [9] The stationary properties of the variable is tested by ADF with null hypothesis $H_0 : \gamma = 0$ and alternative hypothesis $H_1 : \gamma < 0$, where $\gamma = \rho - 1$. The same reasoning can be extended for a generic $AR(p)$ process as

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \sum_{i=1}^p \delta_i \Delta y_{t-i} + \epsilon_t, \quad (2)$$

where α is a constant, β the coefficient on a time trend and p the lag order of the autoregressive process [10]. By including lags, the ADF formulation allows for higher-order autoregressive processes. Too few lags will leave autocorrelation in the ϵ_t . Models are estimated with a range of values for p and the optimal lag length p is determined by Akaike Information Criterion (AIC) [11]. The series contains at least one unit root, if H_0 is rejected.

2.2 Multilinear Regression

Since its introduction by C. F. Gauss in the early 1800s, the least-squares parameter matching technique has penetrated to all fields of research and practical engineering work, and it still seems to be among the only ones that are routinely used [12]. Common problems are cointegration, discussed in the previous section and multicollinearity, especially when there is a lot of data available.

Multilinear regression (MLR) model from X to estimated Y can be written as

$$F_{MLR} = (X^T X)^{-1} X^T Y. \quad (3)$$

The MLR solution to modeling relationships between variables is exact and optimal in the sense of the least squares criterion. Available data is divided into two sections: training section consists of 2/3 of the data points and testing set the rest.

2.3 Feed-Forward Back-Propagation Neural Network

A proposed alternative method for MLR in our experiments is a Feed-Forward back-propagation Neural Network (FF) with one hidden-layer. A major problem for the method is that it can learn only static input-output mapping [5]. The model dynamics can be learnt with a correct variable selection. Data is divided into three sections: training, validation and testing set. Each section has 1/3 of the data points. The training mode begins in our experiments with randomly selected weights and those are optimized in training. In each epoch, the NN adjusts the weights in the direction that reduces the error. The weights gradually converge to the locally optimal set of values. Many epochs are usually required before training is completed, but then the probability of over-training is higher. In our experiments a limit for mean squared error (MSE) and Maximum validation checks were set to avoid over-training.

3 Experiments

3.1 Variable Selection

A general model for economical time series in a matrix form is defined as

$$Y = \beta X + \gamma dt_t + \epsilon_t, \quad (4)$$

where X is dependent variables. It has m columns where first column is constant. Since quarterly data is used, it is adjusted by three mean centered seasonal dummies dt_t . ϵ_t is the error term. Variables and data base definitions are listed below. These are used to derive dependent variables X :

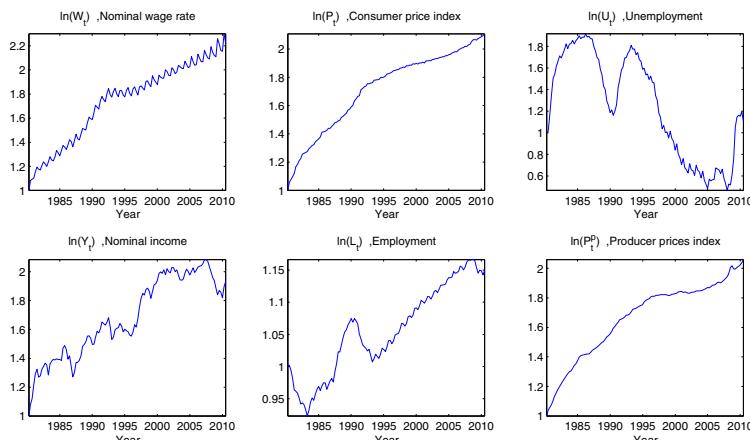


Fig. 1. A natural logarithm of each variable is derived. These six variables or differences are needed on the wage equation.

- nominal wage rate (w_t): Auxiliary indicators to National Accounts, Unit labour cost, Quarterly data (namq_aux.ulc). [2]
- consumer price index (p_t): Available data from 1996 Q1 [3]. Available data from 1979 Q1 [4].
- unemployment rate (u_t): The longest period was available in source OECD [3].
- nominal income (y_t): Gross domestic product (GDP) and main components, price indices, GDP at market prices, price index, 2000=100 (based on euro) [3].
- employment (l_t): The main Economic Indicators - Labour force statistics - Employment, 1956-2010 [3].
- producer price index (p_t^P): Data from 1979 Q1 was available [4].

Databases are in update, so new (until 2010 Q2 or Q3) measurements were collected. Collected data was not seasonally adjusted, because seasonal dummies are added in the modelling phase. The length of stored data varies, and in the end it was managed to collect 122 samples, which is enough for this analysis. So, for example training area for MLR is 1980 Q2 to 2000 Q4 and testing area is from 2001 Q1 to 2010 Q2. All variables are preprocessed by taking natural logarithms. Because of logs, there is no matter on the unit of the variables. Variables were mean centered and first measurement was set to value one by adding $1 - x_{i,1}$ to all x_i measurements. This procedure helps to understand changes and relation between other model variables in the measured data, see Fig. 1. Some difficulties have to be taken into account. It is possible that some variables are probably $I(2)$ processes.

All variables were unit root tested. The results are shown in Table 1. When original variable was tested, in addition to constant and seasonals, the trend was added. The null hypothesis is rejected with higher absolute t-ADF values, see 5% criteria values in Table 1 and 2.

The nominal wage rate w_t was classified barely as $I(2)$, see Table 1, and it means that the model results are more spurious, because the model will have non-stationary dependent variables. An interesting remark was that the estimated number of unit roots varied depending on the different p_{max} values, for example on u_t , see two rows for u_t in Table 1. The fact is that ADF test is just an empirical test. Maybe with the Philips-Perron test [13] different result would be achieved. Surprisingly *employment* is stationary, $l_t \sim I(0)$, even various p_{max} values was tested, see two rows for l_t in Table 1. Different cointegration terms were tested to be sure that all terms in the model are $I(1)$, as the theory of cointegration expects. The results are shown in Table 2.

In the economic literature, a hypothetical wage equation, typically resulting from wage negotiations between employers and labor unions is defined different ways. So, based on unit root tests and the logical explanation of derived variables, a hypothetical wage equation in this paper is given by

Table 1. The unit root test results are based on the ADF test, where the number of selected lags is based on the AIC information criteria. (* = ADF test results varied with a different range of values for p).

Variable	t-ADF	5% Crit.	p/p_{max}	root	β_{Y_1}	Conclusion
Δw_t	-2.850	-2.89	6/6	0.40204	$\Delta w_t \sim I(1) \rightarrow w_t \sim I(2)$	
Δp_t	-3.605	-2.89	5/6	0.69150	$\Delta p_t \sim I(0) \rightarrow p_t \sim I(0)$ or $p_t \sim I(1)$	
p_t	-1.506	-3.45	5/6	0.98909	$p_t \sim I(1)$	
Δu_t	-2.616	-2.89	6/6*	0.82084	$\Delta u_t \sim I(1) \rightarrow u_t \sim I(2)$	
Δu_t	-3.152	-2.89	5/5*	0.79507	$\Delta u_t \sim I(0) \rightarrow u_t \sim I(0)$ or $u_t \sim I(1)$	
u_t	-2.477	-3.45	5/5	0.98909	$u_t \sim I(1)$	
Δy_t	-4.837	-2.89	2/5	0.31684	$\Delta y_t \sim I(0) \rightarrow y_t \sim I(0)$ or $y_t \sim I(1)$	
y_t	-2.405	-3.45	3/5	0.91345	$y_t \sim I(1)$	
Δl_t	-3.497	-2.89	4/5	0.67791	$\Delta l_t \sim I(0) \rightarrow l_t \sim I(0)$ or $l_t \sim I(1)$	
l_t	-3.786	-3.45	4/5	0.93243	$l_t \sim I(0)$	
l_t	-3.683	-3.45	4/7	0.93317	$l_t \sim I(0)$	
Δp_t^p	-3.761	-2.89	2/5	0.66978	$\Delta p_t^p \sim I(0) \rightarrow l_t \sim I(0)$ or $p_t^p \sim I(1)$	
p_t^p	-2.270	-3.45	3/5	0.97950	$p_t^p \sim I(1)$	

Table 2. The unit root test results are based on the ADF test, where the number of selected lags is based on the AIC information criteria.

Derived variable	t-ADF	5% Crit.	p/p_{max}	root	β_{Y_1}	Conclusion
$\Delta Y, \Delta(w_t - p_t)$	-4.300	-2.89	3/5	-0.1296	$\Delta Y \sim I(0), Y \sim I(0/1)$	
$Y, (w_t - p_t)$	-2.645	-3.45	4/5	0.86312	$Y \sim I(1)$	
$\Delta X_2, \Delta(y_t - l_t - p_t)$	-5.030	-2.89	4/5	-0.1239	$\Delta X_2 \sim I(0), Y \sim I(0/1)$	
$X_2, (y_t - l_t - p_t)$	-2.263	-3.45	3/5	0.93547	$X_2 \sim I(1)$	
$\Delta X_3, \Delta(p_t - p_t^p)$	-4.872	-2.89	4/5	-0.1159	$\Delta Y \sim I(0), X_3 \sim I(0/1)$	
$X_3, (p_t - p_t^p)$	-1.802	-3.45	3/5	0.92889	$X_3 \sim I(1)$	

$$(w_t - p_t) = \beta_0 + \beta_1 \cdot u_t + \beta_2 \cdot (y_t - l_t - p_t) + \beta_3 \cdot (p_t^p - p_t) + \beta_4 \Delta p_t, \quad (5)$$

where all data is in natural logarithms. $(w_t - p_t)$ is the target real wage [14]. u_t is unemployment rate, $(y_t - l_t - p_t)$ is measure of productivity, $(p_t^p - p_t)$ is tax wedge, and Δp_t is change of consumer prices. The tax wedge is the deviation from equilibrium price/quantity as a result of a taxation, which results in consumers paying more, and suppliers receiving less. Politically-determined factors such as

the tax wedge is commonly used to explain wage formation [1]. Following from the *Law of Supply and Demand*, as the price for consumers increases, and the price suppliers receive decreases, the quantity each wishes to trade will decrease.

Unit root tests prove that expected MLR modelling results are not perfect. u_t (X_1 , see Eq. 5) is on the border of $I(2)$ and $I(1)$ process. And Δp_t is $I(0)$ (X_4 , see Eq. 5) process as it is shown already in Table 1. Despite ADF test result, a difference of the consumer price index is used in the model, because p_t has a clear trend, see Fig. 1. Other roots of X_i and Y are $I(1)$ processes, see Table 2. In practice this is common, and therefore it can be useful to apply Neural Networks to financial models, because commonly used MLR model can give spurious regression.

3.2 Modelling

First modelling phase is the selection of lags. It is based on analysis of the lag structure, because it is important to avoid a complicated model. In analysis of the lag structure several F-test results were calculated. With higher than five lags, the F-test results were not significant. Also, based on unit root test results, see Tables 1 and 2, it can assume that five lags are enough.

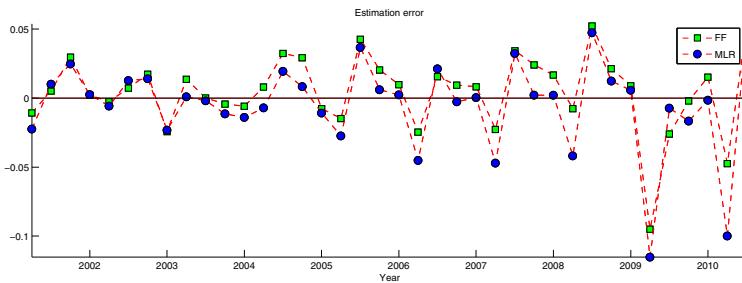


Fig. 2. Model error analysis. Estimation error of MLR and one output of FF model with one hidden neuron. The effect of last economic crisis can be seen in the end of time series.

The target real wage ($w_t - p_t$) was modelled by MLR. First interpretative variable is the target variable with lags $L^1 \dots L^5$. Other interpretative variables are in the model with lags $L^0 \dots L^5$ expect constant and seasonal dummies. The MLR estimator is optimal in the class of linear unbiased estimators when the errors are homoscedastic and serially uncorrelated. In our analysis the error term is heteroscedastic, see Fig. 2.

FF net was tested with one, two, three and four hidden neurons and the same interpretative variables. The best modelling results were achieved by FF with one hidden neuron with a tansig transfer function. The most effective training method was the Levenberg-Marquardt back-propagation method.

3.3 Results

FF models were tested with the different number of hidden neurons and MSE limits, see Table 3. The goodness of fit, R^2 value is for training set when MSE limit is used to break FF model training. R^2 value is for validation set when the number of validation checks is used. Q^2 for test set, of a statistical model describes how well it fits in a set of observations. Better fit was achieved to both sets with FF model. All models had problems to model time 2009 Q1. It is supposed that reason for that is the past financial crisis.

Table 3. Comparison of MLR and FF models. Training was limited by the MSE limit value or the maximum number of validation checks (VC). Test results varies too much, if a number of epochs is small or large and number of hidden neurons (N) is large. (* = not enough training), (**) = Over training)

Model	N	MSE limit	VC limit	epochs	R^2	Q^2
MLR					0.9403	0.8912
FF*	1	0.0001	-	9	0.9488	0.9309
FF	1	0.00005	-	46	0.9532	0.9313
FF	1	0.00001	-	103	0.9532	0.9312
FF	1	-	20	1000	0.9990	0.9604
FF*	2	0.0001	-	7	0.9603	0.5437
FF	2	0.00005	-	22	0.9729	0.6513
FF**	2	0.00001	-	799	0.9920	-1.3856
FF	2	-	20	51	0.9968	0.3834
FF	3	0.00005	-	34	0.9841	0.6093
FF	3	-	20	25	0.9986	0.5304
FF**	4	0	-	40	1	0.7612
FF	4	-	20	23	0.9164	0.6906

FF models are overtrained when the number of hidden neurons and epochs are large. The effect of overtraining can be seen as large variance in the goodness of fit, see Table 3. Therefore only one hidden neuron is suggested in order to use in the neural modelling of wage equation, see the bold Q^2 values on Table 3.

4 Conclusions

A hypothetical wage equation was created and solved by two different methods. Real data from United Kingdom was used. Data was collected from Eurostat, Source OECD databases and Office for National Statistics UK. In practice, the

wage model has a great importance for the overall features of large-scale macro models and for example for the different policy actions. Seldom all variables are $I(1)$, as the results of augmented Dickey-Fuller tests proved. In this research it was proved that simple FF model with one hidden neuron has better fit for training and in prior, to the testing set, than the MLR model.

The results indicate that the system is probably non-linear, even it was paid effort to the model creation. Some other non-linear modelling methods than FF should be tested in the future research.

References

1. Friberg, K.: Why do central banks produce forecasts of wage development? *Economic Review*, 52 (2009)
2. Eurostat, <http://epp.eurostat.ec.europa.eu/>
3. Source OECD, <http://www.oecd-ilibrary.org/statistics/>
4. Office for national statistics, <http://www.statistics.gov.uk/>
5. Haykin, S.: Neural networks: a comprehensive foundation. Prentice Hall PTR, Upper Saddle River (1994)
6. Tang, Z., Fishwick, P.: Feedforward neural nets as models for time series forecasting. *ORSA Journal on Computing* 5, 374–374 (1993)
7. Talonen, J., Sirola, M., Augilius, E.: Modelling power output at nuclear power plant by neural networks. In: Diamantaras, K., Duch, W., Iliadis, L.S. (eds.) *ICANN 2010*. LNCS, vol. 6352, pp. 46–49. Springer, Heidelberg (2010)
8. Christopoulos, D., Tsionas, E.: Financial development and economic growth: evidence from panel unit root and cointegration tests. *Journal of Development Economics* 73(1), 55–74 (2004)
9. Dickey, D., Fuller, W.: Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American statistical association*, 427–431 (1979)
10. Cheung, Y., Lai, K.: Lag order and critical values of the augmented Dickey-Fuller test. *Journal of Business & Economic Statistics* 13(3), 277–280 (1995)
11. Sakamoto, Y., Kitagawa, G.: Akaike information criterion statistics (1987)
12. Hyotyniemi, H.: Multivariate regression - techniques and tools. Tech. rep (2001)
13. Kim, S., Wong, K.: Effects of news shock on inbound tourist demand volatility in Korea. *Journal of Travel Research* 44(4), 457 (2006)
14. Hecq, A., Mahy, B.: Testing for the price-and wage-setting model in Belgium using multivariate cointegration tests. *Labour* 11(1), 177–199 (1997)

On the Designing of Spikes Band-Pass Filters for FPGA

M. Domínguez-Morales, A. Jimenez-Fernandez, E. Cerezuela-Escudero,
R. Paz-Vicente, A. Linares-Barranco, and G. Jimenez^{*}

Departamento de Arquitectura y Tecnología de Computadores,
ETS Ingeniería Informática - Universidad de Sevilla,
Av. Reina Mercedes s/n, 41012-Sevilla, Spain
{mdominguez, ajimenez, ecerezuela, rpaz,
alinares, gaji}@atc.us.es

Abstract. In this paper we present two implementations of spike-based band-pass filters, which are able to reject out-of-band frequency components in the spike domain. First one is based on the use of previously designed spike-based low-pass filters. With this architecture the quality factor, Q , is lower than 0.5. The second implementation is inspired in the analog multi-feedback filters (MFB) topology, it provides a higher than 1 Q factor, and ideally tends to infinite. These filters have been written in VHDL, and synthesized for FPGA. Two spike-based band-pass filters presented take advantages of the spike rate coded representation to perform a massively parallel processing without complex hardware units, like floating point arithmetic units, or a large memory. These low requirements of hardware allow the integration of a high number of filters inside a FPGA, allowing to process several spike coded signals fully in parallel.

Keywords: Spike processing, neuromorphic engineering, band-pass filter.

1 Introduction

Neuromorphic engineers work in the study, design and development of neuro-inspired systems, like aVLSI chips for sensors [1][2], neuro-inspired processing, filtering or learning [3][4][5][6], neuro-inspired control pattern generators (CPG), neuro-inspired robotics [7][8][15] and so on. Spiking systems are neural models that mimic the neuron layers of the brain for processing purposes. Signals in the spike-domain are composed of short pulses in time, called spikes [9]. Information is carried by the spike frequency or rate [11], following a Pulse Frequency Modulation (PFM) scheme, and from other point of view, in the inter-spike-time (ISI) [6]. Spike rate coded information can be processed in a continuous way, without codifying information into discrete samples. Spike processing hardware can be easily replicated performing a massively parallel information processing [12].

In this context, previous work presented a set of building blocks, to process spike coded signals in real time [17], mimicking elementary analogical components to

^{*} This work has been supported by the Spanish project VULCANO (TEC2009-10639-C04-02).

design spike filters, equivalent to classical analogical filters but filtering spikes in time (Spike Signal Processing, SSP). In [17] it was thoroughly explained how to design and tune spike-based low-pass (SLPF) and high-pass filters (SHPF). This paper presents in detail how we can design spike band-pass filters (SBPF), using the elements presented in [17]. So a short description and operation theory of previous elements for SSP is presented in section 2. In section 3, a low quality, Q, factor SBPF is designed using simply SLPF. In section 4, inspired in the topology of multi-feedback (MFB) band-pass filters, we design and analyze a high Q factor SBPF, were Q can be ideally infinite. Both filters topologies have been simulated with the help of MATLAB Simulink and the addition Xilinx SystemGenerator, which allows integrating, among many other things, VHDL files with Simulink common elements.

2 Building Blocks for SSP

In this section we will make a short introduction to basic building blocks and full gain adjustable SLPF [17], because this information is necessary to understand SBPF.

2.1 Spike Synthetic Generator (SSG)

This element generates a spikes sequence from a discrete digital number, and will generate the stimulation of spike-based filters simulations. We use in particular the Reverse Bit-wise Synthetic Spike Generator (RB-SSG) found in [14]. RB-SSG is based in digital counters, and it does not use probabilistic techniques, providing deterministic and well characterized spike rates or ISI. RB-SSG generates a spike stream from an N-bits signed discrete number, being the output spike rate proportional to the input value, X, as (1) denotes. RB-SSG gain, k_{RB-SSG} can be calculated according to (2).

$$RB-SSG_{SpikeRate} = k_{RB-SSG} * X \quad (1)$$

$$k_{RB-SSG} = \frac{F_{CLK}}{2^{N-1}(genFD + 1)} \quad (2)$$

F_{CLK} is the FPGA clock frequency, N input signal number of bits, and genFD is a clock divider used to obtain accurate values of k_{RB-SSG} . RB-SSG is able to generate positive and negative spikes, so two 1-bit signals are necessary to propagate generated spikes, positive and negative by different channels.

2.2 Spikes Integrate and Generate (SI&G)

SI&G performs the temporal integration operation, which means to provide a stream of spikes, with a spike rate proportional to the integration of input spikes. Fig. 1 shows internal SI&G blocks, where input spikes enter the left block and output integrated spikes are generated by the right block. SI&G has a digital counter connected to a RB-SSG. In order to integrate spikes we can use an N-bit digital up/down counter (whit saturation), where a positive spike increases by one counter

value, and a negative spike decreases in the same quantity. Spike counter output provides a digital N-bit number; therefore it is necessary to convert spikes integration to a spike stream. For this task a RB-SSG is used.

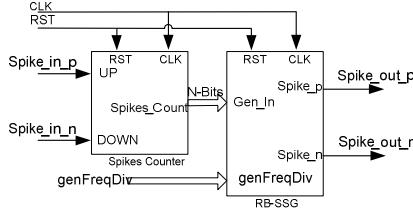


Fig. 1. SI&G internal blocks: Counter and RB-SSG

The RB-SSG included inside the SI&G will provide a spike stream, with a spike rate proportional to spikes counter, or integration. SI&G operation is equivalent to a continuous integration, where SI&G gain, $k_{SI\&G}$ is equivalent to k_{RB-SSG} and can be calculated using (2). Making an analogy to continuous systems, we can calculate equivalent SI&G transfer function applying Laplace Transform, there is a result:

$$SI\&G(s) = \frac{F_{SI\&G}(s)}{F_{inputSpikes}(s)} = \frac{k_{SI\&G}}{s} = \frac{F_{CLK}}{2^{N-1}(genFD + 1)} * \frac{1}{s} \quad (3)$$

2.3 Spikes Hold and Fire (SH&F)

In this section we present an element designed to subtract signed spikes, that have been designed and used successfully for spike filter designs [17] and spike-based closed-loop robotic controllers [15]. Subtracting a spike-based input signal (f_U) to another (f_Y), will yield a new spike signal whose spike rate ($f_{SH\&F}$) will be the difference between both input spike rates (4):

$$f_{SH\&F} = f_U - f_Y \quad (4)$$

The procedure of the SH&F is to hold the incoming spikes a fixed period of time while monitoring the input evolution to decide output spikes: holding, canceling, or firing spikes according to input spike ports and signs.

2.4 Spike Frequency Divider (SFD)

This block will divide the spike rate of an input signal by a constant. SFD have implemented inspired in the way that RB-SSG works. Fig. 3 shows SFD internal components. In SFD *spikesDiv* behaves like the constant to divide. Output buffers only enable output spikes according to an N-bit divider, *spikesDiv*, homogenously in time. SFD transfer function can be calculated using (6), where N represents the SFD number of bits. SFD transfer function is equivalent to a gain block, k_{SFD} , with a value in the range of [0, 1], with 2^N possible steps. SFD accuracy can also be adjusted with N (N=16 bits in Fig.5) getting an accuracy of 2^{-N} .

$$k_{SFD} = \frac{F_{outSpikes}}{F_{inputSpikes}} = \frac{spikesDiv}{2^N} \quad (5)$$

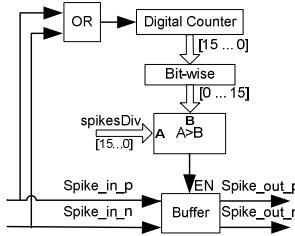


Fig. 3. SFD internal components

2.4 Spike Low-Pass Filters (SLPF)

A SLPF will attenuate high frequency components of spikes rate signals. In order to build a SLPF, a SI&G has been feedbacked using a SH&F and two SFD, as it is shown in Fig. 4. The idea is to integrate input spikes, but subtracting output integrated spikes from the integrator, with the help of a SH&F. Two SFD have been included in order to provide an adjustable gain. Taking into account SLPF components topology and transfer functions shown in (3), (4) and (5), SLPF transfer function can be calculated as shown in (6):

$$F_{SLPF}(s) = \frac{F_{outSpikes}(s)}{F_{inputSpikes}(s)} = \frac{k_{SFDOout} * SI\&G(s)}{1 + k_{SFDFB} * SI\&G(s)} = \frac{k_{SFDOout} * k_{I\&G}}{s + k_{SFDFB} * k_{I\&G}} \quad (6)$$

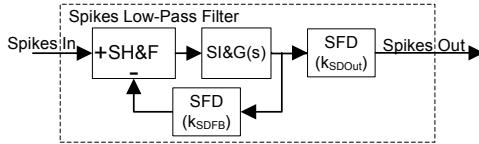


Fig. 4. SLPF blocks topology

SLPF is equivalent to a first order low-pass filter, with adjustable cut-off frequency ($\omega_{cut-off}$ in rads/sec) and gain (k_{SLPF}):

$$\omega_{cut-off} = k_{SFDFB} * k_{I\&G}, \quad k_{SLPF} = \frac{k_{SFDOout}}{k_{SFDFB}} \quad (7) \quad (8)$$

3 Low Q-Factor Spikes Band-Pass Filter (LQ-SBPF)

In order to design a SBPF, our first approximation was using SLPF. The idea is to use two SLPF, with different cut-off frequencies, subtracting to the highest cut-off

frequency SLFP output spikes, the low cut-off frequency SLFP spikes, as it is shown in Fig. 5. LQ-SPBF transfer function can be calculated from SLFPs transfer functions:

$$F_{SBPF}(s) = F_{SLPF_{HF}}(s) - F_{SLPF_{LF}}(s) = \frac{k_{spikesDivHF} * k_{BWSpikesGenHF}}{s + k_{spikesDivFBHF} * k_{BWSpikesGenHF}} - \frac{k_{spikesDivLF} * k_{BWSpikesGenLF}}{s + k_{spikesDivFBLF} * k_{BWSpikesGenLF}} \quad (9)$$

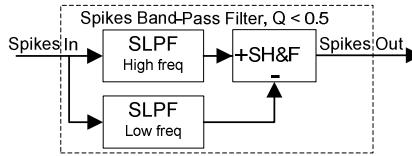


Fig. 5. LQ-SBPF subtracting two SLFP with different cut-off frequencies

If both SLFP gain are equal to 1, $k_{spikesDivHF} = k_{spikesFBHF} = k_{spikesDivLF} = k_{spikesFBLF}$, (9) is simplified, can be compared with ideal second order continuous band-pass filter, in (10). So adjusting SLFP cut-off frequencies, we can design LQ-SBPF with a desired mid band pass frequency ω_{mid} (11), Q factor (12) and SBPF gain (13). Since both SLFP poles are real, equivalent SBPF poles are also real, so $Q \leq 0.5$.

$$F_{Ideal\ BPF}(s) = k_{BPF} \frac{s * \omega_{mid}/Q}{s^2 + s * \omega_{mid}/Q + \omega_{mid}^2} \quad (10)$$

$$\omega_{mid} = \sqrt{\omega_{HF} * \omega_{LF}}, \quad Q = \frac{\sqrt{\omega_{HF} * \omega_{LF}}}{\omega_{HF} + \omega_{LF}}, \quad k_{BPF} = \frac{\omega_{HF} - \omega_{LF}}{\omega_{HF} + \omega_{LF}} \quad (11) \quad (12) \quad (13)$$

We have simulated LQ-SBPF using a simulation framework described in [10]. First simulation of LQ-SBPF can be found in Fig. 6. We have set the right parameters to obtain a LQ-SBPF with a mid frequency of 3kHz, and a Q factor of 0.4. We can see the input spike rate frequency, in blue, composed of 3 tones, 300Hz, 3kHz, and 30kHz with different amplitudes. In red, the response of an ideal BPF with same characteristics, and in black, the reconstruction of the LQ-SBPF output spike frequency. Fig. 6 evidences how the low and high frequency tones are removed, remaining only the spikes of the 3kHz tones. For analyzing the LQ-SBPF frequency response, we have designed a set of LQ-SBPFs and simulated each of them for different tone frequencies. Making a frequency sweep, and annotating LQ-SBPF output spike rate for each tone. Using this we can calculate a Bode diagram. Fig. 7 shows Bode diagram for LQ-SBPF together with their ideal response, in discontinuous traces. Simulated filters have two mid-frequencies, 500Hz and 5kHz, and different Q factors, from 0.3 to 0.5. LQ-SBPF Bode diagram denotes that the filters have a correct behavior according to the theoretical model, rejecting out of band components with a slope of 20dB/decade.

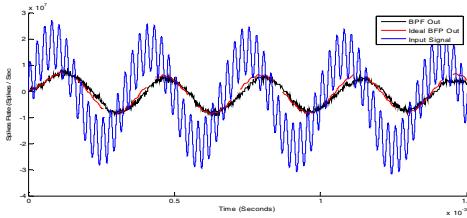


Fig. 6. LQ-SBPF 3-tone temporal simulation

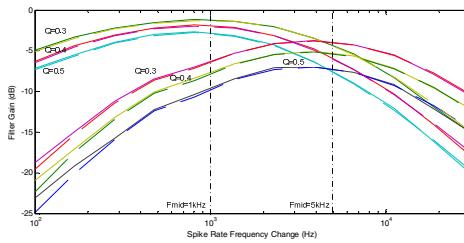


Fig. 7. Diverse LQ-SBF bode diagram

4 High Q-Factor Spike Band-Pass Filter (HQ-BPF)

One common topology used to build high Q BPF, is the multi-feedback (MFB) topology. Applying the principles of MFB filters, the HQ-BPF topology is shown in Fig. 8. This filter is basically a SLPF, with the difference that a second SI&G has been added in the feedback loop. Spikes coming from this SI&G and the SFD, also present in SLPF, are added using an SH&F that implements the multi-feedback.

We can calculate HQ-BPF equivalent transfer function applying basic systems theory. First of all, the transfer function of the elements placed at the feedback can be:

$$FB(s) = k_{SFD} + SI\&G(s)_{FB} = \frac{k_{SFD} * s + k_{SI\&G_FB}}{s} \quad (14)$$

Using (14) we are able to calculate HQ-BPF equivalent transfer function:

$$BPF(s)_{HQ} = \frac{I\&G(s)}{1 + FB(s) * I\&G(s)} = \frac{k_{SI\&G} * s}{s^2 + k_{SI\&G} * k_{SFD} * s + k_{SI\&G} * k_{SI\&G_FB}} \quad (15)$$

If we assume that both SI&G have the same gain, $k_{SI\&G} = k_{SI\&G_FB}$, we can extract filter features from (15), being the mid-frequency equivalent to $k_{SI\&G}$, in (16) and the Q factor, the inverse of SFD gain(k_{SFD}), in (17). As theoretical equations denote, HQ-SBPF is tunable using only two parameters: SI&G gain and SFD gain.

$$\omega_{mid} = \sqrt{k_{SI\&G}^2} = k_{SI\&G} \quad Q = \frac{k_{SI\&G}}{k_{SI\&G} * k_{SFD}} = \frac{1}{k_{SFD}} \quad (16) \quad (17)$$

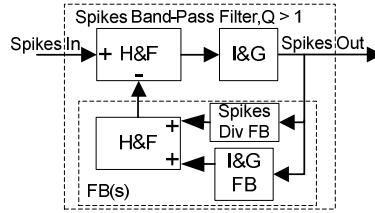


Fig. 8. Spikes-based building blocks topology for HQ-SBPF

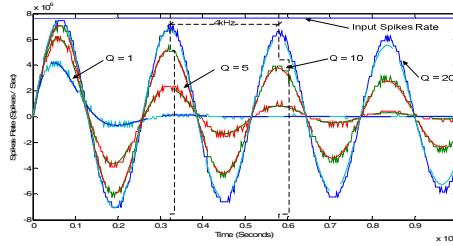


Fig. 9. HQ-SBPF with diverse Q factors, step response

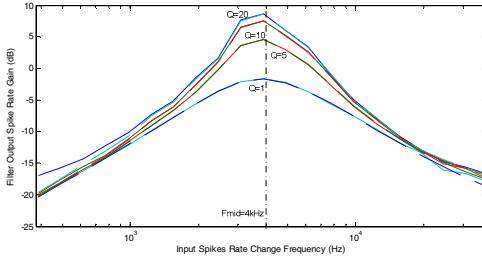


Fig. 10. Diverse Q factor HQ-SBF bode diagram

Similar simulations presented with LQ-BPF have been made for HQ-BPF. Fig. 9 shows step response of several HQ-BPF with equal mid-frequencies, at 4kHz, but different Q factors. When a step is present, only the frecuencial component of the mid-frequency can go through the HQ-BPF. However, this filter presents two complex poles and a sub-damped response, introducing an oscillation to output frequency. Frecuencial simulations have been made in to determine the Bode diagram of HQ-SBPF, which can be found in Fig. 10. For this figure, we have fixed the HQ-SBPF mid-frequency at 4 kHz, and modified the Q factor. These filters have resonance frequencies placed at $k_{SI\&G}$ (rad/sec), and the Q factor can be controlled using k_{SFD} . This figure also includes ideal BPF responses, these are very accurate with HQ-SBPF output, as predicted from equations (15), (16) and (17).

5 Conclusions

This paper has presented two new elements for Spike Signal Processing (SSP), previously existent elements implement basic arithmetic operations and low/high-pass spike-based filters. Based on these basic elements and filters we have designed two spike band-pass filters, one with different ranges of Q factors. SBPF are being used nowadays in our labs for many practical applications, for example a Synthetic AER Cochlea that uses LQ-SBPF, spike-based frequency-domain modulation for transmitting a lot of spike-coded information, as it is made in radio communications.

References

1. Lichtsteiner, P., et al.: A 128×128 120dB 15 us Asynchronous Temporal Contrast Vision Sensor. *IEEE Journal on Solid-State Circuits* 43(2) (2008)
2. Chan, V., et al.: AER EAR: A Matched Silicon Cochlea Pair With Address Event Representation Interface. *IEEE TCAS I* 54(1) (2007)
3. Serrano-Gotarredona, R., et al.: On Real-Time AER 2-D Convolutions Hardware for Neuromorphic Spike-Based Cortical Processing. *IEEE TNN* 19(7) (2008)
4. Oster, M., et al.: Quantifying Input and Output Spike Statistics of a Winner-Take-All Network in a Vision System. In: *IEEE International Symposium on Circuits and Systems, ISCAS* (2007)
5. Hafliger, P.: Adaptive WTA with an Analog VLSI Neuromorphic Learning Chip. *IEEE Transactions on Neural Networks* 18(2) (2007)
6. Indiveri, G., et al.: A VLSI Array of Low-Power Spiking Neurons and Bistables Synapses with Spike-Timing Dependant Plasticity. *IEEE Trans. on Neural Networks* 17(1) (2006)
7. Gomez-Rodriguez, F., et al.: AER Auditory Filtering and CPG for Robot Control. In: *IEEE International Symposium on Circuits and Systems, ISCAS* 2007 (2007)
8. Linares-Barranco, A., et al.: Using FPGA for visuo-motor control with a silicon retina and a humanoid robot. In: *IEEE International Symposium on Circuits and Systems, ISCAS* 2007 (2007)
9. Shepherd, G.: *The Synaptic Organization of the Brain*. Oxford University Press, Oxford (1990)
10. Jimenez-Fernandez, A., et al.: Simulating Building Blocks for Spikes Signals Processing. In: *International WorkShop in Artificial Neural Networks , IWANN* 2011 (2011)
11. Mahowald, M.: VLSI Analogs of Neuronal Visual Processing: A Synthesis of Form and Function. PhD. Thesis, California Institute of Technology Pasadena, California (1992)
12. Serrano-Gotarredona, R., et al.: CAVIAR: A 45k-neuron, 5M-synapse AER Hardware Sensory-Processing-Learning-Actuating System for High-Speed Visual Object Recognition and Tracking. *IEEE Trans. on Neural Networks* 20(9) (2009)
13. Gomez-Rodriguez, F., Paz, R., Miro, L., Linares-Barranco, A., Jimenez, G., Civit, A.: Two hardware implementations of the exhaustive synthetic AER generation method. In: Cabestany, J., Prieto, A.G., Sandoval, F. (eds.) *IWANN 2005. LNCS*, vol. 3512, pp. 534–540. Springer, Heidelberg (2005)
14. Paz-Vicente, R., et al.: Synthetic retina for AER systems development. In: *International Conference on Computer Systems and Applications, AICCSA* 2009 (2009)
15. Jimenez-Fernandez, A., et al.: AER-based robotic closed-loop control system. In: *IEEE International Symposium on Circuits and Systems, ISCAS* 2008 (2008)
16. Jimenez-Fernandez, A., et al.: AER and dynamic systems co-simulation over Simulink with Xilinx System Generator. In: *IEEE Int.Symp. on Circuits and Systems, ISCAS* 2008 (2008)
17. Jimenez-Fernandez, A., et al.: Building Blocks for Spike-based Signal Processing. In: *IEEE International Joint Conference on Neural Networks, IJCNN* 2010 (2010)

An Information Geometrical View of Stationary Subspace Analysis

Motoaki Kawanabe^{1,2}, Wojciech Samek^{1,2,*},
Paul von Bünau¹, and Frank C. Meinecke¹

¹ Fraunhofer Institute FIRST, Kekuléstr. 7, 12489 Berlin, Germany

² Berlin Institute of Technology, Franklinstr. 28/29, 10587 Berlin, Germany
motoaki.kawanabe@first.fraunhofer.de, wojwoj@mail.tu-berlin.de,
buenau@cs.tu-berlin.de, frank.meinecke@tu-berlin.de

Abstract. Stationary Subspace Analysis (SSA) [3] is an unsupervised learning method that finds subspaces in which data distributions stay invariant over time. It has been shown to be very useful for studying non-stationarities in various applications [5,10,4,9]. In this paper, we present the first SSA algorithm based on a full generative model of the data. This new derivation relates SSA to previous work on finding interesting subspaces from high-dimensional data in a similar way as the three easy routes to independent component analysis [6], and provides an information geometric view.

Keywords: stationary subspace analysis, generative model, maximum likelihood estimation, Kullback-Leibler divergence, information geometry.

1 Introduction

Finding subspaces which contain interesting structures in high-dimensional data is an important preprocessing step for efficient information processing. A classical example is Principal Component Analysis (PCA) that extracts low-dimensional subspaces which capture as much variance of the observations as possible. In contrast, more recent approaches consider other criteria than maximum variance. For instance, non-Gaussian component analysis (NGCA) [2,7], a general semi-parametric framework including projection pursuit [8], extracts *non-Gaussian* structures, e.g. subspaces which reveal clusters or heavy tailed distributions. Another recent direction is colored subspace analysis [13] which searches for linear projections having *temporal correlations*. What all these methods have in common is that i.i.d. (white) Gaussian random variables are considered as noise, and each method seeks to maximize a specific deviation from it to uncover an informative subspace. This fact reminded us of the thought-provoking paper [6] showing three easy routes to Independent Component Analysis (ICA). Independent components can be extracted either by using non-Gaussianity, or

* née Wojcikiewicz.

by non-whiteness of spectrum, or by non-stationarity [11] of the variance. The goal of this paper is to discuss projection methods corresponding to the third route, i.e. based on finding *non-stationary* structures.

Recently, Bünau et al. [3] proposed a novel technique called Stationary Subspace Analysis (SSA) which finds the low-dimensional projections having stationary distributions from high-dimensional observations. This is an instrumental analysis whenever the observed data is conceivably generated as a mixture of underlying stationary and non-stationary components: a single non-stationary source mixed into the observed signals can make the whole data appear non-stationary; and non-stationary sources with low power can remain hidden among strong stationary components. Uncovering these two groups of sources is therefore a key step towards understanding the data. Stationary Subspace Analysis has been applied successfully to biomedical signal processing [5], Computer Vision [10], high-dimensional change point detection [4] and domain adaptation problems [9].

However, the SSA algorithm [3] is based on finding the stationary sources and does not model the non-stationary components, i.e. unlike [11], the SSA algorithm is not derived from any generative model of the observed data. In this paper, we assume a linear mixing model with uncorrelated stationary and non-stationary sources and compute the maximum likelihood estimators of the projections onto both subspaces. That is, we also explicitly model the non-stationary part. The objective function turns out to be a combination of the original SSA objective and a term which penalizes cross-covariances between the stationary and non-stationary sources.

The remainder of this paper is organized as follows. After explaining our model assumptions in Section 2, we derive the objective function of the maximum likelihood approach in Section 3 and provide a geometrical illustration. Then, we present the results of the numerical simulations in Section 4.

2 Block Gaussian Model for SSA

In the SSA model [3], we assume that the system of interest consists of d stationary source signals $\mathbf{s}^s(t) = [s_1(t), s_2(t), \dots, s_d(t)]^\top$ (called **s-sources**) and $D-d$ non-stationary source signals $\mathbf{s}^n(t) = [s_{d+1}(t), s_{d+2}(t), \dots, s_D(t)]^\top$ (also **n-sources**) where the observed signals $x(t)$ are a linear superposition of the sources,

$$\mathbf{x}(t) = A \mathbf{s}(t) = [A^s \ A^n] \begin{bmatrix} \mathbf{s}^s(t) \\ \mathbf{s}^n(t) \end{bmatrix}, \quad (1)$$

and A is an invertible matrix. Note that we do *not* assume that the sources $\mathbf{s}(t)$ are independent. We refer to the spaces spanned by A^s and A^n as the **s**- and **n**-space respectively. The goal is to factorize the observed signals $x(t)$ according to Eq. (1), i.e. to find a linear transformation \hat{A}^{-1} that separates the **s**-sources from the **n**-sources. Given this model, the **s**-sources and the **n**-space are uniquely identifiable whereas the **n**-sources and the **s**-space are not (see [3] for details).

In order to invert the mixing of stationary and non-stationary sources, we divide the time series $x(t)$ of length T into L epochs defined by the index sets $\mathcal{T}_1, \dots, \mathcal{T}_L$. Even though the SSA model includes both the stationary and the non-stationary sources, the SSA algorithm [3] optimizes only the stationarity of the estimated stationary sources and does not optimize the non-stationary source estimates: the approach is to find the projection such that the difference between the mean and covariance in each epoch is identical to the average mean and covariance over all epochs.

In this paper, we derive a maximum likelihood estimator for the SSA model under the following additional assumptions.

- The sources \mathbf{s} are independent in time.
- The \mathbf{s} -sources \mathbf{s}^s are drawn from a d -dimensional Gaussian $\mathcal{N}(\boldsymbol{\mu}^s, \Sigma^s)$, which is constant in time
- In each epoch ℓ , the \mathbf{n} -sources \mathbf{s}^n are drawn from a $(D - d)$ -dimensional Gaussian $\mathcal{N}(\boldsymbol{\mu}_\ell^n, \Sigma_\ell^n)$, i.e. their distribution varies over epochs
- The \mathbf{s} - and \mathbf{n} -sources are group-wise uncorrelated.

Thus the true distribution of the observed signals $\mathbf{x}(t)$ in epoch ℓ is $\mathcal{N}(\mathbf{m}_\ell, R_\ell)$, where $\mathbf{m}_\ell = A\boldsymbol{\mu}_\ell$, $R_\ell = A\Sigma_\ell A^\top$ and

$$\boldsymbol{\mu}_\ell = \begin{bmatrix} \boldsymbol{\mu}^s \\ \boldsymbol{\mu}_\ell^n \end{bmatrix}, \quad \Sigma_\ell = \begin{bmatrix} \Sigma^s & 0 \\ 0 & \Sigma_\ell^n \end{bmatrix}. \quad (2)$$

The unknown parameters of the model are the mixing matrix A , the mean and covariance of the \mathbf{s} -sources $(\boldsymbol{\mu}^s, \Sigma^s)$ and those of the \mathbf{n} -sources $\{(\boldsymbol{\mu}_\ell^n, \Sigma_\ell^n)\}_{\ell=1}^L$.

3 Maximum Likelihood and Its Information Geometric Interpretation

In this section, we derive the objective function of the maximum likelihood SSA (MLSSA). Under the block Gaussian model, the negative log likelihood to be minimized becomes

$$\begin{aligned} \mathcal{L}_{\text{ML}} &= - \sum_{\ell=1}^L \sum_{t \in \mathcal{T}_\ell} \log p(\mathbf{x}(t) | A, \boldsymbol{\mu}^s, \boldsymbol{\mu}_\ell^n, \Sigma^s, \Sigma_\ell^n) \\ &= T \log |\det A| + \frac{TD}{2} \log 2\pi + \frac{1}{2} \sum_{\ell=1}^L \sum_{t \in \mathcal{T}_\ell} \left[\log \det \Sigma_\ell \right. \\ &\quad \left. + \text{Tr} \left\{ \Sigma_\ell^{-1} A^{-1} (\mathbf{x}(t) - \mathbf{m}_\ell) (\mathbf{x}(t) - \mathbf{m}_\ell)^\top A^{-\top} \right\} \right]. \end{aligned} \quad (3)$$

It is known that the maximum likelihood estimation can be regarded as the minimum Kullback-Leibler-divergence (KLD) projection [1]. Let

$$\bar{\mathbf{x}}_\ell = \frac{1}{|\mathcal{T}_\ell|} \sum_{t \in \mathcal{T}_\ell} \mathbf{x}(t), \quad \hat{R}_\ell = \frac{1}{|\mathcal{T}_\ell|} \sum_{t \in \mathcal{T}_\ell} (\mathbf{x}(t) - \bar{\mathbf{x}}_\ell) (\mathbf{x}(t) - \bar{\mathbf{x}}_\ell)^\top \quad (4)$$

be the sample mean and covariance of the ℓ -th chunk, where $|\mathcal{T}_\ell|$ denotes its length. Indeed, the likelihood \mathcal{L}_{ML} can be expressed as

$$\mathcal{L}_{\text{ML}} = \sum_{\ell=1}^L |\mathcal{T}_\ell| D_{\text{KL}} \left[\mathcal{N}(\bar{\mathbf{x}}_\ell, \hat{R}_\ell) \parallel \mathcal{N}(\mathbf{m}_\ell, R_\ell) \right] + \text{const.} \quad (5)$$

In the following, we derive the estimators of the moment parameters explicitly as functions of the mixing matrix A which leads to an objective function for estimating A . Since the KL-divergence is invariant under linear transformations, the divergence between the two distributions of the observation \mathbf{x} is equal to that between the corresponding distributions of the sources $\mathbf{s} = A^{-1}\mathbf{x}$, i.e.

$$\begin{aligned} D_{\text{KL}} \left[\mathcal{N}(\bar{\mathbf{x}}_\ell, \hat{R}_\ell) \parallel \mathcal{N}(\mathbf{m}_\ell, R_\ell) \right] \\ = D_{\text{KL}} \left[\mathcal{N}(A^{-1}\bar{\mathbf{x}}_\ell, A^{-1}\hat{R}_\ell A^{-\top}) \parallel \mathcal{N}(\boldsymbol{\mu}_\ell, \Sigma_\ell) \right]. \end{aligned} \quad (6)$$

The central idea of this paper is to regard the KL-divergence as a distance measure in the space of Gaussian probability distributions, which leads to an information geometric viewpoint of the SSA objective. The divergence in Equation (6) is the distance between the empirical distribution of the demixed signals and the true underlying sources. According to our assumption the true model lies on a manifold \mathcal{M} defined by Equation (2), i.e.

$$\mathcal{M} := \left\{ N(\boldsymbol{\mu}, \Sigma) \mid \Sigma = \begin{bmatrix} \Sigma^s & 0 \\ 0 & \Sigma^n \end{bmatrix} \right\}. \quad (7)$$

Therefore it is convenient to split the divergence (6) into the following two parts,

1. an orthogonal projection D_1 onto the manifold \mathcal{M} ,

$$D_1 = D_{\text{KL}} \left[\mathcal{N}(A^{-1}\bar{\mathbf{x}}_\ell, A^{-1}\hat{R}_\ell A^{-\top}) \parallel \mathcal{N}(\tilde{\boldsymbol{\mu}}_\ell, \tilde{\Sigma}_\ell) \right],$$

2. and a component D_2 in the manifold,

$$D_2 = D_{\text{KL}} \left[\mathcal{N}(\tilde{\boldsymbol{\mu}}_\ell, \tilde{\Sigma}_\ell) \parallel \mathcal{N}(\boldsymbol{\mu}_\ell, \Sigma_\ell) \right].$$

This decomposition is illustrated in Figure 1. It is also known as the generalized pythagorean theorem in information geometry [1]. The orthogonal projection onto the manifold \mathcal{M} is given by

$$\tilde{\boldsymbol{\mu}}_\ell = \begin{bmatrix} \tilde{\boldsymbol{\mu}}_\ell^s \\ \tilde{\boldsymbol{\mu}}_\ell^n \end{bmatrix} = \begin{bmatrix} (A^{-1})^s \bar{\mathbf{x}}_\ell \\ (A^{-1})^n \bar{\mathbf{x}}_\ell \end{bmatrix} = A^{-1}\bar{\mathbf{x}}_\ell, \quad (8)$$

$$\tilde{\Sigma}_\ell = \begin{bmatrix} \tilde{\Sigma}_\ell^s & 0 \\ 0 & \tilde{\Sigma}_\ell^n \end{bmatrix} = \begin{bmatrix} (A^{-1})^s \hat{R}_\ell \left\{ (A^{-1})^s \right\}^\top & 0 \\ 0 & (A^{-1})^n \hat{R}_\ell \left\{ (A^{-1})^n \right\}^\top \end{bmatrix}, \quad (9)$$

where $(A^{-1})^s$ and $(A^{-1})^n$ are the projection matrices to the stationary and non-stationary sources respectively. The projection onto the manifold \mathcal{M} does

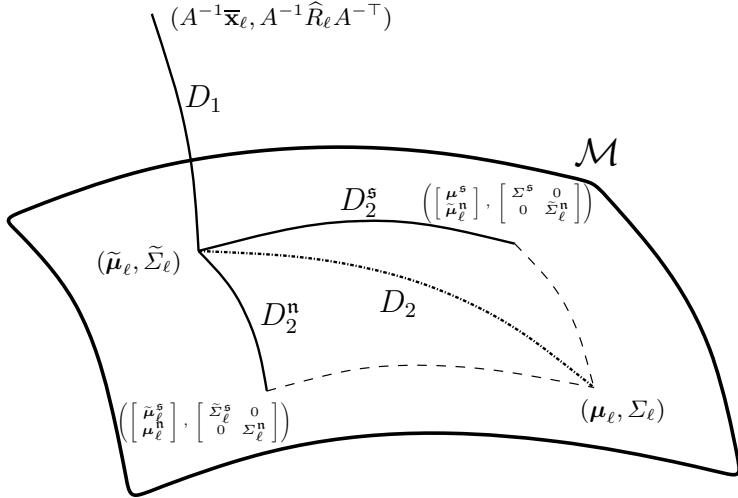


Fig. 1. A geometrical view of the divergence decomposition. The divergence D_1 corresponds to the orthogonal projection onto the manifold of models and D_2 is the divergence on the manifold, which can be further decomposed according to the block diagonal structure of the source covariance matrix.

not affect the mean, it merely sets the off diagonal blocks of the covariance matrix to zero. Conversely, on the manifold \mathcal{M} , only the diagonal blocks vary, thus the above projection is orthogonal w.r.t. the Fisher information matrix. For more details, see [1]. Since on \mathcal{M} the stationary and non-stationary sources are independent by definition, we can further decompose the distance D_2 from $\mathcal{N}(\tilde{\mu}_\ell, \tilde{\Sigma}_\ell)$ to the true distribution $\mathcal{N}(\mu_\ell, \Sigma_\ell)$ into two independent parts,

$$\begin{aligned} D_2 &= D_2^s + D_2^n \\ &= D_{\text{KL}} \left[\mathcal{N}(\tilde{\mu}_\ell^s, \tilde{\Sigma}_\ell^s) \parallel \mathcal{N}(\mu^s, \Sigma^s) \right] + D_{\text{KL}} \left[\mathcal{N}(\tilde{\mu}_\ell^n, \tilde{\Sigma}_\ell^n) \parallel \mathcal{N}(\mu^n, \Sigma^n) \right]. \end{aligned}$$

This decomposition leads to the following expression for the likelihood,

$$\begin{aligned} \mathcal{L}_{\text{ML}} &= \sum_{\ell=1}^L |\mathcal{T}_\ell| \left\{ D_{\text{KL}} \left[\mathcal{N}(A^{-1}\bar{x}_\ell, A^{-1}\hat{R}_\ell A^{-\top}) \parallel \mathcal{N}(\tilde{\mu}_\ell, \tilde{\Sigma}_\ell) \right] \right. \\ &\quad + D_{\text{KL}} \left[\mathcal{N}(\tilde{\mu}_\ell^s, \tilde{\Sigma}_\ell^s) \parallel \mathcal{N}(\mu^s, \Sigma^s) \right] \\ &\quad \left. + D_{\text{KL}} \left[\mathcal{N}(\tilde{\mu}_\ell^n, \tilde{\Sigma}_\ell^n) \parallel \mathcal{N}(\mu^n, \Sigma^n) \right] \right\}. \end{aligned} \quad (10)$$

The moment parameters (μ^s, Σ^s) of the s -sources appear only in the second term, while those of the n -sources $\{(\mu_\ell^n, \Sigma_\ell^n)\}_{\ell=1}^L$ are included only in the third term. Therefore, for fixed A , the moment estimators (denoted with hats) minimizing the likelihood \mathcal{L}_{ML} can be calculated from $\{(\tilde{\mu}_\ell, \tilde{\Sigma}_\ell)\}_{\ell=1}^L$:

- for the \mathbf{n} -sources, $(\widehat{\boldsymbol{\mu}}_\ell^\mathbf{n}, \widehat{\Sigma}_\ell^\mathbf{n}) = (\widetilde{\boldsymbol{\mu}}_\ell^\mathbf{n}, \widetilde{\Sigma}_\ell^\mathbf{n})$, which means that $D_2^\mathbf{n} = 0$,
- the mean $\widehat{\boldsymbol{\mu}}^\mathbf{s}$ of the \mathbf{s} -sources is the weighted average $\sum_{\ell=1}^L \frac{|\mathcal{T}_\ell|}{T} \widetilde{\boldsymbol{\mu}}_\ell^\mathbf{s}$ or equivalently $(A^{-1})^\mathbf{s} \overline{\mathbf{x}}$, where $\overline{\mathbf{x}} = \frac{1}{T} \sum_{t=1}^T \mathbf{x}(t)$ is the global mean,
- the covariance $\Sigma^\mathbf{s}$ of the \mathbf{s} -sources is the projection of the global covariance $\widehat{R} = \frac{1}{T} \sum_{t=1}^T (\mathbf{x}(t) - \overline{\mathbf{x}})(\mathbf{x}(t) - \overline{\mathbf{x}})^\top$ onto the \mathbf{s} -space, i.e. $(A^{-1})^\mathbf{s} \widehat{R} \left\{ (A^{-1})^\mathbf{s} \right\}^\top$

By substituting these moment estimators, we obtain an objective function to determine the mixing matrix A ,

$$\begin{aligned} \mathcal{L}_{\text{ML}} = & \sum_{\ell=1}^L |\mathcal{T}_\ell| \left\{ D_{\text{KL}} \left[\mathcal{N}(A^{-1} \overline{\mathbf{x}}_\ell, A^{-1} \widehat{R}_\ell A^{-\top}) \parallel \mathcal{N}(\widetilde{\boldsymbol{\mu}}_\ell, \widetilde{\Sigma}_\ell) \right] \right. \\ & \left. + D_{\text{KL}} \left[\mathcal{N}(\widetilde{\boldsymbol{\mu}}_\ell^\mathbf{s}, \widetilde{\Sigma}_\ell^\mathbf{s}) \parallel \mathcal{N}(\widehat{\boldsymbol{\mu}}^\mathbf{s}, \widehat{\Sigma}^\mathbf{s}) \right] \right\}. \end{aligned} \quad (11)$$

Moreover, since the solution is undetermined up to scaling, sign and linear transformations within the \mathbf{s} - and \mathbf{n} -space, we set the estimated \mathbf{s} -sources to zero mean and unit variance by centering and whitening of the data, i.e. we write the estimated demixing matrix as $A^{-1} = B\widehat{W}$ where $\widehat{W} = \widehat{R}^{-1/2}$ is a whitening matrix and B is an orthogonal matrix. Let $\overline{\mathbf{y}}_\ell = \widehat{W} \overline{\mathbf{x}}_\ell$ and $\widehat{V}_\ell = \widehat{W} \widehat{R}_\ell \widehat{W}^\top$ be the mean and covariance of the centerized and whitened data $\mathbf{y}(t) = \widehat{W} \mathbf{x}(t)$ in the ℓ -th epoch. Then, the objective function becomes

$$\begin{aligned} \mathcal{L}_{\text{ML}} = & \sum_{\ell=1}^L |\mathcal{T}_\ell| \left\{ D_{\text{KL}} \left[\mathcal{N}(B\overline{\mathbf{y}}_\ell, B\widehat{V}_\ell B^\top) \parallel \mathcal{N}(B\overline{\mathbf{y}}_\ell, \text{b-diag}[B^\mathbf{s} \widehat{V}_\ell (B^\mathbf{s})^\top, B^\mathbf{n} \widehat{V}_\ell (B^\mathbf{n})^\top]) \right] \right. \\ & \left. + D_{\text{KL}} \left[\mathcal{N}(B^\mathbf{s} \overline{\mathbf{y}}_\ell, B^\mathbf{s} \widehat{V}_\ell (B^\mathbf{s})^\top) \parallel \mathcal{N}(\mathbf{0}, I_d) \right] \right\}, \end{aligned}$$

where $B^\top = [(B^\mathbf{s})^\top, (B^\mathbf{n})^\top]$ and ‘‘b-diag’’ denotes the block diagonal matrix with given sub-matrices. The second term coincides with the original SSA objective. The first term comes from the orthogonality assumption and can be regarded as a joint block diagonalization criterion. The implication of this joint objective is illustrated in Section 4. The objective function (??) can be further simplified as

$$\mathcal{L}_{\text{ML}} = \sum_{\ell=1}^L \frac{|\mathcal{T}_\ell|}{2} \left\{ \log \det(\widetilde{\Sigma}_\ell^\mathbf{n}) - \log \det(\overline{\Sigma}_\ell) + (\widetilde{\boldsymbol{\mu}}_\ell^\mathbf{s})^\top \widetilde{\boldsymbol{\mu}}_\ell^\mathbf{s} \right\} + \text{const.} \quad (12)$$

where $\widetilde{\boldsymbol{\mu}}^\mathbf{s} = B^\mathbf{s} \overline{\mathbf{y}}_\ell$, $\widetilde{\Sigma}_\ell^\mathbf{s} = B^\mathbf{s} \widehat{V}_\ell (B^\mathbf{s})^\top$ and $\overline{\Sigma}_\ell := B\widehat{V}_\ell B^\top = A^{-1} \widehat{R}_\ell A^{-\top}$. As in the SSA algorithm [3], we optimize the objective function (??) using a gradient descent algorithm taking into account the Lie group structure of the set of orthogonal matrices [12].

4 Numerical Example

We compare the performance of our novel method with the original SSA algorithm [3]. In particular, we study the case where the epoch covariance matrices are exactly block diagonalizable as this matches the assumptions of our method.

In Figure 2, we compare MLSSA and SSA over varying numbers of epochs using 5 stationary sources and a 20-dimensional input space, i.e. $d = 5$ and $D = 20$. For each epoch we randomly sample a block diagonalizable covariance matrix and a mean vector and mix them with a random, but well-conditioned matrix A . Note that no estimation error is introduced as we sample both moments directly. As performance measure we use the median angle to the true n -subspace over 100 trials and represent the 25% and 75% quantiles by error bars. For each trial, in order to avoid local minima, we restart the optimization procedure five times and select the solution with the lowest objective function value.

From the results we see that our method yields a much lower error than the original approach, especially when a small number of epochs is used. For instance, when using MLSSA (red line) 10 epochs are sufficient to achieve negligible errors (around 0.01 degree) whereas for the original SSA algorithm (blue line) more than 30 epochs are required. Although it seems that thanks to the first term in Eq. (??), i.e. joint block diagonalization, MLSSA allows a much faster convergence than the original method, more experiments will be needed to obtain a full picture.

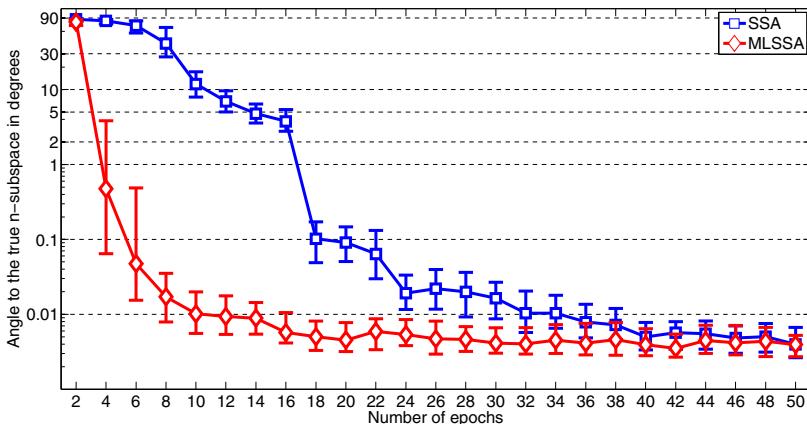


Fig. 2. Performance Comparison between the proposed method (MLSSA) and the original SSA for a varying number of epochs. The red curve shows the performance, measured as the median angle to the true n -subspace over 100 trials, of our method whereas the blue curve stands for the results obtained with SSA. The 25% and 75% quantiles are represented by the error bars. Note that we use randomly sampled block diagonalizable covariance matrices in our experiments and set the dimensionality of the stationary subspace to 5 (out of 20 dimensions). Our proposed method significantly outperforms SSA in this example, especially when less than 30 epochs are used. MLSSA obviously exploits (by the first term in Eq. (??)) the block diagonalizable structure of the covariance matrices and thus reconstructs the true stationary subspace much faster (i.e. with less epochs) than SSA.

5 Conclusions

In this paper, we developed the first generative version of SSA. The generative model is a block Gaussian model and the objective function of the maximum likelihood approach turns out to be a combination of the original SSA and a joint block diagonalization term. This new derivation not only helps the theoretical understanding of the procedure in the information geometrical framework, but the algorithm also yields competitive results. Moreover, the likelihood formulation allows future extension towards model selection methods and for incorporating prior knowledge.

Acknowledgement. The authors acknowledge K.-R. Müller, F. J. Király and D. Blythe for valuable discussions.

References

1. Amari, S.: Differential-geometrical Methods in Statistics. Lecture Notes in Statistics. Springer, Berlin (1985)
2. Blanchard, G., Sugiyama, M., Kawanabe, M., Spokoiny, V., Müller, K.: In search of non-Gaussian components of a high-dimensional distribution. *Journal of Machine Learning Research* 7, 247–282 (2006)
3. Bünau, P.v., Meinecke, F.C., Király, F., Müller, K.-R.: Finding stationary subspaces in multivariate time series. *Physical Review Letters* 103, 214101 (2009)
4. Bünau, P.v., Meinecke, F.C., Müller, J.S., Lemm, S., Müller, K.-R.: Boosting High-Dimensional Change Point Detection with Stationary Subspace Analysis. In: *Workshop on Temporal Segmentation at NIPS* (2009)
5. von Bünau, P., Meinecke, F.C., Scholler, S., Müller, K.R.: Finding stationary brain sources in EEG data. In: *Proceedings of the 32nd Annual Conference of the IEEE EMBS*, pp. 2810–2813 (2010)
6. Cardoso, J.F.: The three easy routes to independent component analysis; contrasts and geometry. In: *Proc. ICA 2001*, pp. 1–6 (2001)
7. Diederichs, E., Juditsky, A., Spokoiny, V., Schtte, C.: Sparse non-gaussian component analysis. *IEEE Trans. Inform. Theory* 56, 3033–3047 (2010)
8. Friedman, J.H., Tukey, J.W.: A projection pursuit algorithm for exploratory data analysis. *IEEE Trans. Computers* 23, 881–890 (1974)
9. Hara, S., Kawahara, Y., Washio, T., von Bünau, P.: Stationary subspace analysis as a generalized eigenvalue problem. In: Wong, K.W., Mendis, B.S.U., Bouzerdoum, A. (eds.) *ICONIP 2010, Part I. LNCS*, vol. 6443, pp. 422–429. Springer, Heidelberg (2010)
10. Meinecke, F., von Bünau, P., Kawanabe, M., Müller, K.R.: Learning invariances with stationary subspace analysis. In: *IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*, 2009 . pp. 87 –92 (2009)
11. Pham, D.T., Cardoso, J.F.: Blind separation of instantaneous mixtures of non stationary sources. In: *Proc. ICA 2000*, Helsinki, Finland, pp. 187–192 (2000)
12. Plumley, M.D.: Geometrical methods for non-negative ica: Manifolds, lie groups and toral subalgebras. *Neurocomputing* 67(161-197) (2005)
13. Theis, F.: Colored subspace analysis: Dimension reduction based on a signal's autocorrelation structure. *IEEE Trans. Circuits & Systems I* 57(7), 1463–1474 (2010)

Forecasting Road Condition after Maintenance Works by Linear Methods and Radial Basis Function Networks

Konsta Sirvio and Jaakko Hollmén

Department of Information and Computer Science, Aalto University
School of Science, PO Box 15400, FI-00076 Aalto, Finland
konsta.sirvio@sirway.fi, jaakko.hollmen@aalto.fi

Abstract. Forecasting road condition after maintenance can help in better road maintenance planning. As road administrations annually collect and store road-related data, data-driven methods can be used in determining forecasting models that result in improved accuracy. In this paper, we compare the prediction models identified by experts and currently used in road administration with simple data-driven prediction models, and parsimonious models based on a input selection algorithm. Furthermore, non-linear prediction using radial basis function networks is performed. We estimate and validate the prediction models with a database containing data of over two million road segments.

Keywords: Forecasting, variable selection, road maintenance planning, road condition, rutting, International Roughness Index, Radial Basis Function networks.

1 Introduction

The demand for funds for road maintenance in Finland is higher than the current allocation, which has remained rather constant over the years, although the traffic has increased [4]. As the result, the average road condition has deteriorated. One way to increase the efficient allocation of road maintenance funds is through improved maintenance planning utilising more accurate forecasting methods, which would use the collected data of the road network and machine learning methods [10].

When network level maintenance plan is created for periodic maintenance works, it is essential to estimate road deterioration in terms of collected condition variables. When defining multi-year maintenance plans, it is also important to know the start condition of the road after previous maintenance work [7].

Currently, it is a common practice to rely on linear models based on one explanatory variable in forecasting road condition both after maintenance and time-series of the condition when maintenance works have not been performed. It is also common not to make full use of the data characterising roads [9]. The purpose of this study is to test various forecasting methods on International

Roughness Index (IRI) and rutting after maintenance works. Previous research has showed that complex methods do not necessarily result in better results [9]. Therefore, relatively simple methods are selected with an intention to improve results obtained by the methods in current use.

The rest of the paper is organised as follows: Section 2 contains the description of the database containing road condition measurements. In Sect. 3, we formulate the problem of road condition forecasting. In Sect. 4, we describe the data-driven methods to identify the models used and in Sect. 5 we present the experimental results. The paper is summarised in Sect. 6.

2 Description of the Road Data

Most of periodic maintenance works rely on knowledge on the road surface condition. Automatised, machine-based data collection currently covers IRI and rutting while condition variables such as potholes, ravelling, edge break and cracking is still often based on visual inspection [11]. IRI describes the road user experience of road condition and it is measured by a string in the survey equipment moving vertically due to road roughness. The variable is measured by how many millimetres the string moves on 1 metre driven on the road. Rutting is formed along the road usually on the locations, where road tyres mostly touch the road surface, and it is measured by millimetres.

Road Administration in Finnish Transport Agency (FTA) collects annually road condition data. The collection interval on the same road segment varies according to the importance of the road and only the most important segments goes through data collection procedures at least once a year. The spatial distance of measurement points vary depending on the variable, but surface condition data stored in the road condition database is averaged to 100 metre long road segments [8]. Other characteristics of the roads such as maintenance works are stored in a separate road database.

In this study, the data from the two databases between years 2003 and 2010 are taken for whole Finland and combined so that there are static road characteristics, road condition variables before maintenance works, maintenance work descriptors and values of condition variables after road maintenance. There are two variables to forecast, IRI and rutting that are regarded as the most important in terms of surface condition. The number of explanatory variables is 54 including IRI and rutting measured prior to the maintenance work. The variables can be grouped logically to 1) maintenance treatment and impact, 2) road location and classes, 3) survey and maintenance time, 4) traffic, 5) road geometry, 6) pavement and material, and 7) road condition variables.

Total amount of road surveys were 2 529 747 in the road condition database and 554 640 rows in the road database spanning almost 80 000 kilometres of road network. When these were combined together to include one previous condition survey, road maintenance and consequent next survey, the total data set contains 186 217 data rows.

In data preprocessing unlikely rutting and IRI values were discarded by rounding high values to 50 mm/m and 20 mm/m, respectively, as an error either during

the survey or data entry has been most likely reason. Larger values present less than 0.05% of the data set.

3 Formulation of the Forecasting Problem

Road condition surveys have been conducted both prior and after road maintenance, but the duration between survey and maintenance is variable. The idea is to forecast the condition survey values of IRI and rutting at time t_{i+1} . With the explanatory variables that includes the temporal information of t_{i+1} . The forecasting strategy applied on this paper takes the variable delays between condition surveys and maintenance works as explanatory variables.

Mathematically, the problem can be formulated in a linear form according to Eq. (1). The road condition variable y at time $t+1$ can be explained by the previous value of the condition variable, y_t , explanatory variables x_i weighted by regression coefficients α and β_i as well as by the error term ϵ_{t+1} . The explanatory variables are either independent of time, measured prior to time t or defining the measurement time. The error terms are assumed to follow a normal distribution with zero mean and finite variance.

$$y_{t+1} = \sum_{i=1}^n \beta_i x_i + \alpha y_t + \epsilon_{t+1} \quad (1)$$

4 Forecasting Methods

4.1 Current Methods in Finnish Transport Agency

Currently, the Finnish Transport Agency utilises three forecasting methods for IRI depending on the road surface class. Forecasting model for asphalt pavements is in Eq. (2), for soft asphalt pavements in Eq. (3) and for surface treated gravel roads in Eq. (4) [2]. In all the models ADT stands for annual daily traffic volume including all motorised vehicle categories.

$$IRI = 1.95 - 0.16 \log_{10}(ADT) \quad (2)$$

$$IRI = 2.10 - 0.20 \log_{10}(ADT) \quad (3)$$

$$IRI = 3.20 - 0.20 \log_{10}(ADT) \quad (4)$$

Similarly for the rutting, there is a separate model for asphalt pavement roads in Eq. (5) and the model for both soft asphalt pavements and surface treated gravel roads in Eq. (6) [2].

$$RUTTING = 4.80 - 0.70 \log_{10}(ADT) \quad (5)$$

$$RUTTING = 4.70 - 0.65 \log_{10}(ADT) \quad (6)$$

The models are expert-based using 200-400 kilometres of road data [2].

4.2 Sequential Input Selection Algorithm (SISAL)

Forecasting models can have several explanatory variables. In the basic form of ordinary least squares estimates, all explanatory variables are selected and the Mean Square Error (MSE) between the forecast estimate, \hat{y}_{t+1} and the actual value, y_{t+1} is minimised resulting at the required regression coefficient values. The error measure is presented by $MSE = \frac{1}{N} \sum_{t=1}^N (y_{t+1} - \hat{y}_{t+1})^2$ [12].

For the sake of understanding, visualisation, prediction and computing time it is useful to select only the most relevant variables having the greatest explanatory power. In SISAL methods [12], Ordinary Least Squares (OLS) estimates of the parameters β_i are calculated. The sampling distributions of the parameters and standard deviation of the training MSEs are estimated using M times k - fold cross validation. The median parameter values m_{β_i} are calculated from Mk estimates of β_i^j serving as the location parameters for the distribution. The distribution widths can be estimated by evaluating the differences $\hat{\beta}_i^{high} - \hat{\beta}_i^{low}$, where $\hat{\beta}_i^{high}$ and $\hat{\beta}_i^{low}$ are Mk(1-q)th and Mkqth values in the ordered list of Mk estimates of β_i . The constant q is selected to be $q = 0.165$ [12].

The least significant variable is deleted from the list of input variables by using the ratio m/d being an approximation of signal-to-noise ratio. The lower the value the less significant the variable is considered. The list of all input variables is gone through excluding input variables one by one and the final model is selected according two criteria: the sparsity and prediction accuracy. An ideal model includes only part of the input variables without compromising the prediction accuracy [12].

Model selection is based on minimum validation error and penalisation of model complexity. An ideal model is as sparse as possible with only a few explanatory variables with a minimum validation error. In the selected model the validation error is less than the sum of the minimum validation error and uncertainty measured by standard deviation of the corresponding training error.

4.3 Radial Basis Function (RBF) Networks

Instead of pure linearity, radial basis functions introduce a non-linear element for forecasting. Values of radial basis functions depend on the distance from the centre point, c and therefore any function satisfying the condition $\phi(\mathbf{x}, c) = \phi(\|\mathbf{x} - c\|)$ is a RBF, where \mathbf{x} is an n-dimensional vector. Sums of radial basis functions are simple form of neural networks that can be used for function approximations. Such radial basis networks follow the form of Eq. (7).

$$y(\mathbf{x}) = \sum_{i=1}^N w_i \phi(\|\mathbf{x} - c_i\|), \quad (7)$$

where each radial basis function is weighted by w_i . The approximator is therefore a combination of linear and nonlinear elements of N input variables. A Gaussian radial basis function $y(\mathbf{x}) = \sum_{i=1}^N w_i \exp\left\{-\frac{\|\mathbf{x} - c_i\|^2}{\sigma^2}\right\}$ is commonly used with a predefined variance σ^2 . This is applied in this study as well.

5 Experiments and Results

The data set was divided into training and validation sets. New models were created using the training data and forecasting was tested with the validation data set. Mean value is the baseline method that is the mean value for IRI and rutting of the whole data set before maintenance works. Forecasting results for models with 20 input variables in IRI models and 33 variables in rutting models are also presented after selection of the best explanatory variables in the linear model based on Fig. 1. Validation error of the selected sparse model should not exceed the sum of the minimum validation error and standard deviation of the corresponding training error.

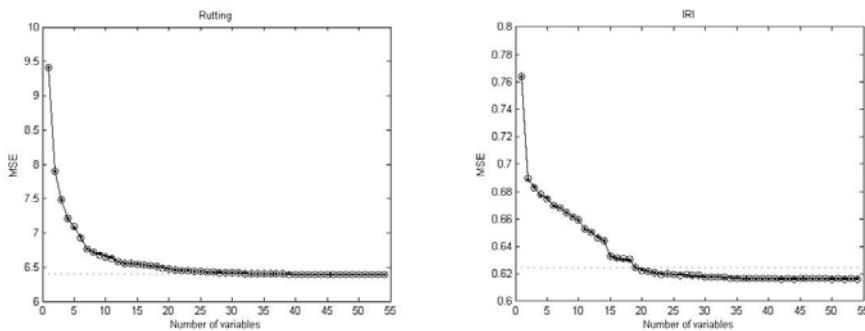


Fig. 1. Input selection procedure for linear rutting and IRI models. Training error is presented by dotted line, validation error by circled line and the vertical dots represents the sum of minimum validation and training errors.

Table 1 summarises forecasting errors of IRI and rutting for the selected methods. FTA methods are the methods currently used by Finnish Transport Agency. All the presented methods were used depending on the forecast variable and pavement type. OLS stands for ordinary least squares estimation. The methods having less than the maximum amount of values forecast use the rest of the data for training. As standard models are deterministic when the whole data set is used for testing there are no variations and they are not applicable marked by NA.

In case of the OLS methods 10-fold cross-validation was used having 9/10 of the data for training and running the algorithm 5 times. The average mean square error is the average squared error of the 50 runs. The minimum, maximum and standard deviation of the squared error is also reported. Standard deviation is calculated out of the average mean square errors of cross-validation rounds. Similar cross-validation was also used in RBF, but 1/30 of the data was used for training. Training data set is smaller than in the linear model due to

Table 1. Errors of forecast IRI values on the left and rutting values on the right

Method	Number of forecast values	IRI				Rutting			
		MSE	Min MSE	Max MSE	$\sigma(MSE)$	MSE	Min MSE	Max MSE	$\sigma(MSE)$
Mean value	186217	1.192	NA	NA	NA	42.060	NA	NA	NA
FTA methods	186217	0.750	NA	NA	NA	14.225	NA	NA	NA
OLS, 54 vars	18620	0.617	0.451	0.815	0.066	6.395	6.103	6.740	0.119
OLS, 20 (33) vars	18620	0.623	0.464	0.790	0.067	6.408	6.131	6.740	0.119
RBF, 54 vars	180010	0.297	0.274	0.382	0.019	4.327	4.201	4.460	0.063
RBF, 20 (33) vars	180010	0.372	0.314	0.645	0.051	9.640	9.543	9.770	0.059

Table 2. Ranking order of explanatory variables for IRI (and rutting in parentheses)

1-Length(27), 2-Right IRI prior maint.(33), 3-Maint. method(4), 4-Road(26), 5-Survey season after maint.(6), 6-Road region(15), 7-IRI prior maint.(33), 8-Surface material in maint.(31), 9-Maint. month(5), 10-Survey month prior maint.(11), 11-Maint. impact on pavement depth(8), 12-Speed limit(10), 13-Maint. impact on bearing capacity(12), 14-Traffic direction(47), 15-Survey month after maint.(43), 16-Right rutting prior maint.(36), 17-Survey day prior maint.(13), 18-Pavement width(28), 19-Left water rutting prior maint.(37), 20-Left rutting prior maint.(39), 21-Average daily heavy traffic(35), 22-Lane(53), 23-Survey day after maint.(52), 24-Road section of the segment start(16), 25-Roadclass(18), 26-Days between survey and maint.(22), 27-Maint. impact on surface defects(42), 28-Maint. impact on roughness(2), 29-Maint. day(34), 30-Rutting prior maint.(3), 31-Road width(19), 32-Survey direction prior maint.(46), 33-Pavement type(38), 34-Municipality(24), 35-Right water rutting prior maint.(48), 36-Start chainage from the road start(30), 37-Maint. impact on rutting(20), 38-Survey season prior maint.(50), 39-Road section of the segment end(54), 40-Carriageway width(17), 41-Average ridge prior maint.(21), 42-Average daily traffic of a lane(25), 43-Average Daily Traffic(51), 44-Days between maint. and survey(7), 45-Survey direction after maint.(40), 46-Required mass kg/m ² in maint.(44), 47-Carriageway(23), 48-End chainage from the road start(32), 49-Maint. class(14), 50-Maximum gravel size(1), 51-Survey year prior maint.(49), 52-Maint. year(29), 53-Survey year after maint.(45), 54-Pavement class(9)
--

computational reasons resulting thus in different number of forecast values compared to the OLS. Selection of the training set size was practical, having as large a set as it was computationally feasible. Standard deviation of the function radius was selected to be 6 after various testings. The target error during training

i.e. stopping criterion was 0.2. In case of rutting the target training error was set to 0.3. The mean square errors are higher than with IRI partly since the values go up to 50 while with IRI they go up to 20.

Ranking of explanatory variables was calculated by repeating 10 times a setup of 20 runs of 10-fold cross-validation and applying SISAL to the cross-validation data for the remaining variables to be ranked. It was noticed that there are variation in importance of some variables in case of rutting, but not IRI. The ranking order is shown in Table 2.

The models currently in use are based on traffic volumes as well as pavement types. Traffic seems to be on the position of 51 and 43 in importance in rutting and IRI respectively, while pavement type is on the position of 38 for rutting and 33 for IRI. The three most explaining variables are maximum gravel size, estimated maintenance impact on roughness and rutting before maintenance in case of rutting and road segment length, right IRI before maintenance and maintenance method in case of IRI.

6 Summary and Conclusions

In this paper linear and non-linear forecasting methods with explanatory variable selection was applied to road condition. The results show that it is justifiable to replace currently used linear expert models based on only one variable with forecasting methods taking more variables and information into consideration. Better linear methods could improve by IRI forecast results by almost 20% and rutting forecasts by 55%. When non-linear methods based on radial basis functions are used is the average forecast improvement in IRI around 0.32 mm/m (52% better) and in rutting approximately 2.07 mm (32% better). These improvements are based on the average results obtained by cross-validation. RBF with reduced number of variables performed much better with IRI than rutting. It seems that nonlinear mutual effects on rutting are not preserved when linear selection procedure of explanatory variables is applied for RBF.

The impacts of improved forecasting accuracy are manifold. If long-term strategic analysis of required maintenance funding under various road condition scenarios is considered the inaccuracy is accumulated in simulations and therefore wrong funding requirements may be selected. In a shorter-term maintenance programming misallocated maintenance decisions may take place as the whole network is not surveyed every year and therefore estimates are required. Reliable forecasting methods could also reduce the need of frequent survey data collection. The reported improvement of IRI in forecasting would mean approximately 93-150 millions of euros of difference in estimated fuel costs (with average fuel price 1.5 euros a litre) for road users as it is estimated that an increase of 1 mm/m in IRI would mean increase in fuel consumption by 0.5% for heavy vehicles and 0.8% for passenger cars [3] and as there are around 35 billion road-kilometres driven annually consuming around 3900 millions of litres of fuel [5].

The future research will concentrate on improved forecasting models on road deterioration without road maintenance. The forecasting models in various

scenarios are combined into a simulation model that will estimate long-term funding and road condition for the Finnish road network administered by the FTA.

Acknowledgments

We would like to express our gratitude to Juho Meriläinen, Vesa Männistö, Tuomas Toivonen, Anne Leppänen from Finnish Transport Agency, Markku Knuuti from Roadscanners Ltd., Esko Sirvio from Sirway Ltd. and Petri Jusi from FinnOC Ltd. for providing data and information on current practices.

References

1. Bishop, C.M.: Neural Networks for Pattern Recognition, Oxford University Press (1996)
2. Finnish Road Administration: PMSPro:n kuntoennustemallit 2004. Tiehallinnon selvityksiä 9/2005. Finnish Road Administration (2005) (In Finnish)
3. Finnish Road Administration: Tien päälyysteen epätasaisuuden vaikutus ajoneuvojen vierintävastukseen ja ajoneuvokustannuksiin. Tiehallinnon selvityksiä 27/2005. Finnish Road Administration (2005) (In Finnish)
4. Finnish Transport Agency: Finnish Road Statistics 2009. Statistics from the Finnish Transport Agency 2/2010. Finnish Transport Agency (2010)
5. Finnish Transport Agency: Päälysteiden pintakarkeuden vaikutus tienkäyttäjiin ja tienviitoon. Liikenneviraston tutkimuksia ja selityksiä 1/2010. Finnish Transport Agency (2010) (In Finnish)
6. Guyon, I., Elisseeff, A.: An Introduction to Variable and Feature Selection. Journal of Machine Learning Research, volume 3, pp. 1157–1182 (2003)
7. Kerali, H., Odoki, J.B., Stannard, E.: Volume one. Overview of HDM-4. The Highway Development and Management Series. The World Road Association (2006)
8. Ruotoistenmäki, A.: Kuntotiedon käyttö tie- ja katuverkon ylläpidon päätöksenteossa. Tiehallinnon selvityksiä 7/2005. Finnish Road Administration (2005)
9. Sirvio, K., Hollmén, J.: Spatio-temporal road condition forecasting with Markov chains and artificial neural networks. In Emilio Corchado, Ajith Abraham, and Witold Pedrycz, editors, Third International Workshop in Hybrid Artificial Intelligent Systems (HAIS'08), volume 5271 of Lecture Notes in Artificial Intelligence, pp. 204–211. Springer-Verlag (2008)
10. Sirvio, K., Hollmén, J.: Multi-year network level road maintenance programming by genetic algorithms and variable neighbourhood search. In Proceedings of the 13th International IEEE Conference on Intelligent Transportation Systems, pages 581–586, (2010)
11. Sirvio, K., Huda, K.: Implementation of the Road Assets Management System in Sindh Province of Pakistan, G1-01014. In Viegas, José Manuel; Macário, Rosário, editors. General Proceedings of the 12th World Conference on Transport Research Society (2010)
12. Tikka, J., Hollmén, J.: Sequential input selection algorithm for long-term prediction of time series. Neurocomputing 71, 2604–2615 (2008)

Multistart Strategy Using Delta Test for Variable Selection

Dušan Sovilj

Aalto University School of Science
Konemiehentie 2, Espoo, Finland
`dusans@cis.hut.fi`

Abstract. Proper selection of variables is necessary when dealing with large number of input dimensions in regression problems. In the paper, we investigate the behaviour of landscape that is formed when using Delta test as the optimization criterion. We show that simple and greedy Forward-backward selection procedure with multiple restarts gives optimal results for data sets with large number of samples. An improvement to multistart Forward-backward selection is presented that uses information from previous iterations in the form of long-term memory.

Keywords: Delta test, noise variance, long-term memory, restart strategy, variable selection, forward-backward selection.

1 Introduction

The number of features or attributes in newly available data sets grows rapidly mostly due to easier data acquisition, storage and retrieval. When the problem is regression, i.e. predicting the real value for a fresh sample, most machine learning methods use all available features which can degrades the predictions [1]. Therefore, proper selection of variables is needed before training the model. Benefits of variable selection are twofold: increasing prediction accuracy and interpretability.

One of the criteria used for variable selection is the Delta test, a noise variance estimator as proposed in [2]. In order to select an optimal set of variables, one should examine an exponential number of possibilities which depends on the dimensionality of the data set. In the case of Delta test, certain variables subsets may be ignored due to the nature of the criterion as explained in later sections.

Forward-backward selection [3] is a simple and widely used procedure for variable selection. The idea is to either include or exclude a single variable at a time and compute the criterion for the obtained subset of variables and repeat the process until the criterion does not improve. This procedure is sufficient when using Delta test and when there are enough samples, but it requires several restarts from random subsets to reach a satisfying solution. When restarting is involved a lot of information about the search process can be used to influence the later stages. This information is reused in some search algorithms, such as

Tabu search [4] and variants of Greedy randomized adaptive search procedures (GRASP) [5][6].

The paper is organized as follows. Section 2 explains the Delta test criterion. In Section 3 Forward-backward selection is briefly mentioned. Section 4 explains the idea of multistart strategies and a specific implementation for Delta test optimization. The results of experiments are given in Section 5 and finally in Section 6 the concluding remarks are presented.

2 Delta Test

Delta test (DT) is a non-parametric noise variance estimator based on a nearest neighbour principle. The estimator is used when a functional dependence is assumed between inputs \mathbf{x}_i and output y_i with additive noise term, i.e. $y_i = g(\mathbf{x}_i) + \epsilon_i$, given finite number of samples $(\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, $i = 1, \dots, M$. The function g is assumed to be smooth, and the noise terms ϵ_i are i.i.d. with zero mean. The Delta test estimates the variance with the following formula:

$$\text{Var}(\epsilon) \approx \frac{1}{2M} \sum_{i=1}^M (y_i - y_{NN(i)})^2, \quad (1)$$

where $NN(i)$ defines the nearest neighbour of sample \mathbf{x}_i in the input space. In [7] it is shown that the estimator converges to the true value of the noise variance in the limit $M \rightarrow \infty$.

As shown in [2], DT requires many samples in order to find correct subset of variables in a noisy setting. On the other hand, when presented with an adequate sample pool, it is able to distinguish between important and noisy variables in most subsets (Section 5.1). In this case, Forward-backward selection is a potent procedure to look for the optimal selection.

The case with an inadequate number of samples is an ill-posed situation. Although DT overestimates the noise variance in this case [7], the selection with the optimal DT selection may also include noisy variables. Therefore, the search should focus on subsets that contain small number of variables to more precisely estimate the variance, while the subsets with more variables can be completely ignored.

The aim in variable selection is to find a subset of variables that minimizes DT estimate. Previous work in this domain [3] is oriented on comparing different search algorithms on data sets with small number of samples and with a time constraint. The main focus in this paper is shape of Delta test landscape when we have enough samples.

3 Forward-Backward Selection

Forward-backward selection (FBS) is a simple procedure for variable selection involving any criterion in machine learning. For DT optimization, the procedure starts from any solution s , i.e. any non-empty subset of variables. Then

it evaluates all *neighbours* $N(s)$ (solutions with either 1 more or 1 less variable than in s) and picks the subset with the smallest DT value from $N(s)$ as the *new s*. These two phases (evaluation and selection) are repeated until no further improvement is possible.

4 Multistart/Restart Strategies

Restarting phase to improve results has been of interest in different search algorithm domains [5][10][11]. Our focus is on simple construction of a solution, from which FBS converges to a local minima. Such approach uses *long-term memory*, a structure that originates from Tabu search [4]. The idea in these search procedures is to construct a high quality solution from which greedy algorithm requires less steps than from a random starting position.

Long-term memory structures keep track of certain aspects of solutions encountered during various stages of the search. Solutions that are good in terms of criterion (also called objective function) are called elite solutions, and most of long-term information relates to elite solutions. Information gathered usually involves: frequency of a variable residing within elite solutions; and the impact of a variable change on objective function value. The former is called consistency and the latter strong determination [8]. This information is used in the first phase of the approach – the construction phase used to build the starting position. Each piece of information contributes to the *energy* of a variable $E(i)$, $i = 1, \dots, d$ with d being the dimensionality of the data set. Construction is done by adding single variable at a time, usually in a probabilistic setting. The probabilities are obtained from energies in the usual way, i.e. $p(i) = E(i) / \sum_{i=1}^d E(i)$. After the construction phase, greedy algorithm is used to find the local optimum. Construction phase and convergence constitute one iteration of the approach.

4.1 Multistart Strategy for Delta Test

For the experiments, both consistency and strong determination are used to guide the construction phase towards promising starting solutions. After obtaining the starting position, the FBS is used for descent phase.

For the rest of the paper we use the following notation: s is the solution or subset of variables; $f(s)$ is DT estimate using the solution s ; f_{\min} the smallest estimate found during search; $C(i)$ is consistency of a variable i ; $S(i)$ is strong determination of variable i ; S_v is the number of variables to add in construction phase; s_e an elite solution; S_e a set of elite solutions (elite memory); and s^{i-} and s^{i+} indicate that i -th variable is excluded and included in solution s respectively.

Energy function is defined as

$$E(i) = \lambda S(i) + C(i), \quad i = 1, \dots, d, \quad (2)$$

where λ controls the trade-off between the two terms and for the experiments we set λ to give equal weight to both terms.

Consistency for a variable i is computed as

$$C(i) = \sum_{s_e^{i+}} \frac{f_{min}}{f(s_e)}, \quad (3)$$

i.e. the sum of DT ratios between best solutions found during the search and elite solutions that have i -th variable included. Consistency tells how frequent is a variable in elite solutions with higher values indicating that a variable is more important.

Strong determination is computed as

$$S(i) = \frac{1}{K} \sum_{j=1}^K \frac{f(s_j^{i-})}{f(s_j^{i+})}, \quad (4)$$

where the fraction denotes the change in objective function for each solution s_j when i -th variables is included, and K the number of such fractions. The higher the value of the fraction in the sum indicates that variable i should be included in the current solution keeping all other variables intact. During the descent phase with FBS, each variable is flipped to check the neighbouring solutions for improvement. These flips also enable us to compute the strong determination for all variables since both $f(s_j^{i-})$ and $f(s_j^{i+})$ are available.

We take only the last 3 changes of descent phases where new local minima are discovered, since those are more important ones that contribute to the f around local minimum. Also, the first iterations have longer descents than later ones as better solutions are generated in later iterations, and we want to treat all iterations equally. The idea of strong determination is to find variables which are good no matter what area of optimization landscape the search algorithm is exploring.

Since $C(i)$ depends on the number of elite solutions, it is at most $|S_e|$. To make both terms equal in Equation (2), λ is set to $|S_e|$ and all values of $S(i)$ are divided by the maximum among $S(i)$. Thus, both terms have roughly the same magnitude when computing energy.

Number of variables. Construction phase selects S_v variables based on the following procedure. For S_v passes a variable is selected probabilistically, but only certain number of best variables (not included in the partial solution) are kept for the selection step. Thus, we first take $P\%$ of the largest $E(i)$ and then select one variable based on $p(i)$. The energy function usually includes the value of objective function $f(s^{i+})$ as an extra term, but in the case of Delta test it is not needed as consistency and strong determination are sufficient to guide the construction toward good solutions making the building phase much quicker.

In order to properly explore the solution space, a diversification strategy is needed. One of the ways to achieve this is by changing the P value. With smaller values, the focus is only on good variables and the generated solutions to not differ too much. On the other hand, with $P \approx 1$ the generated solutions are too

diverse and gathered information is not properly exploited. In the experiments, we set P to a constant 0.5 value during complete search process. It is a good value for tested data sets without hindering the exploration. More refined strategies change P based on diversity of generated solutions with some kind of measure [9]. As mentioned, a constant value of 0.5 makes a good compromise that does not involve any complex strategies with additional parameters.

The correct number of variables before the search is unknown, but since DT estimates noise variance more precisely with less variables, the desired goal is to obtain solution s with smallest number of variables and still keep estimate $f(s)$ minimized. Therefore, parameter S_v should be initiated to a small value and adjusted as the search progresses. In each iteration, S_v is made equal to the number of variables in the best solution found during search. The idea is to have at most variables as in the best solution, and still focus on minimizing f . Of course, more complex approaches are possible.

Diversity of solutions. The consistency value depends on elite solutions s_e in memory. To be able to produce diverse starting solutions, the elite ones have to be diverse enough themselves. Therefore, we define solution s an elite solution if:

1. $f(s) < f_{\min}$ i.e. it has the smallest DT estimate
2. $f(s) < f(s_{e_k})$ for some s_{e_k} and s is sufficiently diverse from better solutions $\{s_{e_j} | f(s_{e_j}) < f(s), j = 1, \dots, k-1\}$

The diversity is taken as the percent of different values of variables, but only on those positions where the variables are set to 1. If $D_{s,t} = \{i | s^{i+} \vee t^{i+}, i = 1, \dots, d\}$ defines a set of variables which have value 1 in s or t , then solutions s and t are diverse if $\sum_{i \in D_{s,t}} s^i \neq t^i \geq \max(2, |D_{s,t}|/4)$, i.e. solutions s and t have to *disagree* on at least quarter of selected variables combined together. If the solution s enters the elite memory, then all similar solutions t with $f(t) > f(s)$ are removed from elite memory ensuring diversity over entire memory range.

5 Experiments

5.1 Synthetic Data

In order to see how good FBS is, consider a simple artificial data set as a function of three variables $f(x_1, x_2, x_3) = \cos(2\pi x_1)\cos(4\pi x_2)\exp(x_2)\exp(2x_3) + \epsilon$ with signal-to-noise ratio close to 1 (as used in [2]) and with increasing number of completely irrelevant ones. Figure 1 shows the influence of the number of samples and dimensions on the number of local minima and the optimization landscape. More samples makes the valley around global minima more steeper and “wider” enabling FBS to reach optimal solution from most starting points. When the number of samples is low, the global minima does not necessary correspond to correct selection of variables. Nevertheless, even in such a scenario, using FBS provides that global solution from certain number of starting positions which

decreases with increasing number of dimensions. The number of samples clearly separates the problem into two categories, and thus different algorithms are needed depending on the situation.

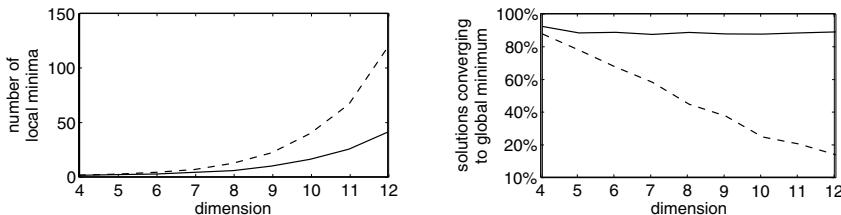


Fig. 1. Average number of local minima (left) and average percentage of all solutions from which FBS converges to global minimum (right). Results are averages over 100 generated data sets with $M=256$ samples (dashed line) and $M=8192$ (solid line).

With high number of samples DT more easily identifies noisy variables from important ones in almost complete optimization landscape. To make sure the global minimum is reached with FBS, several starting positions are needed, but not too many since from most solutions FBS converges to the global one. The next subsection shows results when evaluation of all solutions is not feasible due to the high number of variables.

5.2 Real-World Data

We denote with M_{rnd} the restart strategy with FBS from a random point, and with M_{con} the proposed strategy with construction phase. The strategies are tested on two data sets. One is modified Anthrokids¹ data set with removed missing values, containing 1019 samples in 53 dimensions. The other data set is formed from well known Santa Fe Competition Data – Series A², but with training and test parts combined. The data set is formed by having a regressor of size 36 producing 10057 samples.

Figure 2 shows the convergence of both strategies for two data sets. The results are averages over 10 runs for both strategies, and the parameters for Santa Fe are ($S_v = 5, |S_e| = 10$) and for Anthrokids ($S_v = 10, |S_e| = 20$). Construction phase in the first couple of iterations should not heavily favor any variables until enough solutions have been evaluated. Thus, strong determination is slightly altered by adding constant value of 10 to the sum in Equation (4) for the first 5 iterations, after which it is dropped.

From the figure we see that the final value of DT estimate is almost the same given more iterations, but the proposed M_{con} strategy has the advantage of converging to those values faster. For Santa Fe 9 variables are selected giving DT estimate of 7.0107, while for Anthrokids we have 15 variables and estimate

¹ <http://research.ics.tkk.fi/eiml/datasets/Anthrokids.zip>

² <http://www-psych.stanford.edu/~andreas/Time-Series/SantaFe.html>

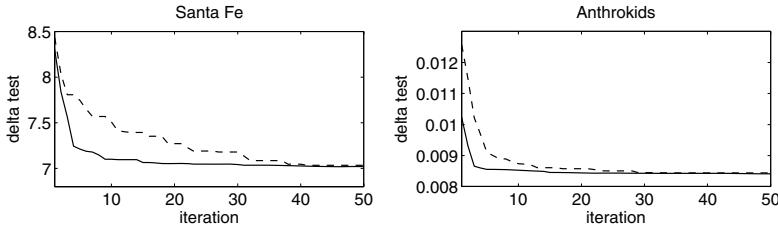


Fig. 2. Convergence of DT as a function of iterations for Santa Fe (left) and Anthrokids (right). Dashed line represents M_{rnd} strategy and solid line M_{con} strategy.

of 0.0085 (normalized output). Given more iterations for Anthrokids (greater than 50), even better solution is possible: 0.0083 with 12 variables.

Figure 3 shows the number of steps per iteration for the same set of experiments. Proposed M_{con} strategy clearly enables better starting position by including long-term memory information. Therefore, a lot less changes are required to converge to a local minima and the total number of DT evaluations is decreased.

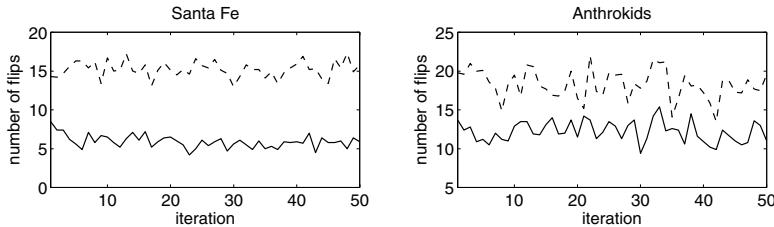


Fig. 3. Average number of steps for FBS per iteration for Santa Fe (left) and Anthrokids (right). Dashed line is M_{rnd} and solid line M_{con} strategy.

One downside of the approach is reliance on FBS which requires examining *all* d neighbours at each step. In situations where d is large, making a single flip can be computationally demanding, as well as one iteration. In this situation, one approach is to fix variables with higher energies before construction phase, or those that are included in all elite solutions.

6 Conclusions

We proposed a multistart strategy for Delta test optimization for variable selection. Due to the nature of landscape that is formed by DT and given the adequate number of samples, the estimator is able to distinguish between noisy and useful variables in most of the optimization landscape. The simple Forward-backward selection procedure is able to reach global minimum given couple of

starting points. To speed up convergence, long-term memory information in form of consistency and strong determination is used. In the experiments, this information showed to be good both in terms of faster convergence and generating solution from which couple of changes are needed to reach local minima.

For further work, a lot more data sets must be tested with large number of samples and without information about relevancy of variables. Proposed strategy includes a lot of parameters, but their meanings should be more intuitive compared to other optimization algorithms and constitute a trade-off between speed of convergence and quality of solutions. For much higher dimensional data sets with over 100 variables, the strategy should be able to provide good results in few iterations.

References

1. Verleysen, M., François, D.: The curse of dimensionality in data mining and time series prediction. In: Cabestany, J., Prieto, A.G., Sandoval, F. (eds.) IWANN 2005. LNCS, vol. 3512, pp. 758–770. Springer, Heidelberg (2005)
2. Eirola, E., Liitiäinen, E., Lendasse, A., Corona, F., Verleysen, M.: Using the Delta Test for Variable Selection. In: European Symposium on Artificial Neural Networks 2008, pp. 25–30 (2008)
3. Guillén, A., Sovilj, D., Mateo, F., Rojas, I., Lendasse, A.: Minimizing the Delta Test for Variable Selection in Regression Problems. International Journal of High Performance Systems Architecture 1, 269–281 (2008)
4. Glover, F., Laguna, F.: Tabu Search. Kluwer Academic Publishers, Norwell (1997)
5. Fernandes, E.R., Ribeiro, C.C.: Using an adaptive memory strategy to improve a multistart heuristic for sequencing by hybridization. In: Nikoletseas, S.E. (ed.) WEA 2005. LNCS, vol. 3503, pp. 4–15. Springer, Heidelberg (2005)
6. Fleurent, C., Glover, F.: Improved Constructive Multistart Strategies for the Quadratic Assignment Problem Using Adaptive Memory. INFORMS J. on Computing 11, 195–197 (1999)
7. Liitiäinen, E., Corona, F., Lendasse, A.: Nearest neighbor distributions and noise variance estimation. In: European Symposium on Artificial Neural Networks 2007, pp. 67–72 (2007)
8. Resende, M.G.C.: Greedy randomized adaptive search procedures (grasp). Technical report, AT&T Labs Research (1998)
9. Morrison, R.W., De Jong, K.A.: Measurement of population diversity. In: Collet, P., Fonlupt, C., Hao, J.-K., Lutton, E., Schoenauer, M. (eds.) EA 2001. LNCS, vol. 2310, pp. 31–41. Springer, Heidelberg (2002)
10. Jansen, T.: On the analysis of dynamic restart strategies for evolutionary algorithms. In: Guervós, J.J.M., Adamidis, P.A., Beyer, H.-G., Fernández-Villacañas, J.-L., Schwefel, H.-P. (eds.) PPSN 2002. LNCS, vol. 2439, pp. 33–43. Springer, Heidelberg (2002)
11. James, T., Rego, C., Glover, F.: Multistart Tabu Search and Diversification Strategies for the Quadratic Assignment Problem. IEEE Transactions on Systems, Man and Cybernetics 39, 579–596 (2009)

Speech Recognition Based on the Processing Solutions of Auditory Cortex

Patrick J.C. May and Hannu Tiitinen

Mind and Brain Laboratory (MBL),
Department of Biomedical Engineering and Computational Science (BECS),
Aalto University, P.O. Box 12200, FI-00076 AALTO, Finland
patrick.may@tkk.fi

Abstract. Speech recognition in the human brain depends on spectral analysis coupled with temporal integration of auditory information. In primates, these processes are mirrored as selective responsiveness of neurons to species-specific vocalizations. Here, we used computational modeling of cortical neural networks to investigate how they achieve selectivity to speech stimuli. Stimulus material comprised multiple pseudowords. We found that synaptic depression was crucial for the emergence of neurons sensitive to the temporal structure of the stimuli. Further, the subdivision of the network into several parallel processing streams was needed for stimulus selectivity to occur. In general, stimulus selectivity and temporal integration seems to be supported by networks with high values of small-world connectivity. The current results might serve as a preliminary pointer for developing speech recognition solutions based on the neuroanatomy and -physiology of auditory cortex.

Keywords: Adaptation, auditory cortex, computational model, automatic speech recognition, stimulus selectivity, synaptic depression, temporal integration.

1 Introduction

Recent advances in automatic speech recognition (ASR) seem to have become incremental and humans seem set to remain far superior to machines in recognizing speech. Thus, it has been suggested that ASR research might be boosted by implementing models developed in the research into speech recognition in humans [1]. In this spirit of cross-fertilization, we present results from our ongoing efforts to model the processing of speech and other complex auditory stimuli in the brain.

Speech has a special relevance for behavior and survival and is one of the most complex kinds of sounds in the auditory environment. Speech comprises uniquely ordered harmonic components, band-pass noise, and silent periods. The brain must therefore rely not only on spectral analysis, but also on temporal binding of spectral information over periods spanning these vocalizations. It seems that a moment-to-moment spectral representation is achieved in the initial part of the auditory pathway, with the cochlea providing a filter bank which feeds information into multiple tonotopic streams extending to the cortex [2]. Spectral analysis would therefore seem

to be a redundant task for auditory cortex, and it has been suggested that its function is to perform temporal binding [3]. However, despite a wealth of *in vivo* single-cell results and non-invasive, mass-action observations, it is still unclear *how* the cortex is able to represent temporally structured complex sounds such as speech.

Results from the human brain indicate the presence of areas in the anterior temporal cortex which are selectively activated by speech [4] and its acoustic-phonetic content [5]. For an intracortical window into the possible neural mechanisms of these areas, we must content ourselves with the results from primates [6,7]: Subcortical information originating from the cochlea arrives via the inferior colliculus and the thalamus to at least three core areas of auditory cortex (including primary auditory cortex) in the lateral sulcus. From here, activation spreads - sequentially - first to surrounding belt areas and then to lateral parabelt areas along the superior temporal gyrus. In this serial scheme, areas are mostly connected to neighboring ones so that multiple core-belt-parabelt streams are formed, with an overall rostral-caudal subdivision being evident. Thus, the auditory cortex of both humans and non-human primates is characterized by multiple parallel streams of information flow with a distinctly serial structure (which is not present in, e.g., visual cortex). Single-cell measurements in the rhesus monkey have revealed that cells in belt and parabelt areas respond selectively to monkey calls [8]. These responses are characterized by a sensitivity to the temporal structure of the stimuli [9]. A preponderance of cells selective to call identity has been found in a pathway that extends from the lateral belt areas anterior to the core, and extends to anterior prefrontal cortex [10]. This kind of selectivity might, hypothetically, tie in with non-invasive results from the human brain on selectivity to the acoustic-phonetic content of speech [5].

However, the neural mechanisms of the temporal binding required by selectivity to vocalizations are currently unknown. Delay lines have been suggested as a solution, whereby a vocalization-selective cell is activated by the concurrent arrivals of delayed and on-time representations of the respective initial and later portions of the vocalization [5]. Given the time span of human and monkey vocalizations, this explanation would require cortical activation delay lines of (at least) several hundred milliseconds. As this is physiologically somewhat implausible (e.g., in view of the 10-ms delay between cochlea and cortex activations), it may be prudent to investigate other possibilities. We recently suggested that the mechanism of temporal binding in auditory cortex is provided by stimulus-specific adaptation, expressed through activity-dependent modifications of synaptic strengths on the single-cell level and modifications of the N1 response measured in the EEG and MEG [11]. In terms of artificial neural networks, this model suggests that the activation of the network modifies the network weights in a local manner and with a slow decay constant. Consequently, the structure of the network evolves with the stimulation and, thus, the input-output transformation becomes dependent on the set of past input patterns and their temporal order.

Here, we explore how the neural mechanisms of speech recognition might be explained by synaptic depression and the structure of auditory cortex. In simulations of the auditory cortex, we varied the decay time of synaptic depression as well as the structure of the model in an attempt to capture the features which are salient for the emergence of single-cell selectivity to species-specific vocalizations as previously reported in the animal-model literature. As stimulus material, the current simulations used pseudowords comprising random combinations of consonant-vowel pairs.

2 Methods

2.1 Model Dynamics and Structure

Dynamics. We simulated auditory cortex with a model comprising $N = 208$ "microcolumns", containing a population each of excitatory (pyramidal) cells and inhibitory interneurons. The dynamics of the model was determined by the Wilson and Cowan firing rate model [12]. Firing rate g depended on the membrane potential u through a non-linear monotonically increasing function $g(u) = \tanh(2/3)(u-\theta)$, $u > \theta$, $g(u) = 0$ otherwise, where $\theta = 0.1$ is the threshold for firing. With the membrane potentials of the excitatory and inhibitory cells described by the vectors $\mathbf{u} = [u_1 \dots u_N]$ and $\mathbf{v} = [v_1 \dots v_N]$, respectively, the dynamic equations describing neural interactions are

$$\begin{cases} \tau_m \dot{\mathbf{u}}(t) = -\mathbf{u}(t) + W_{ee} \cdot g[\mathbf{u}(t)] - W_{ei} \cdot g[\mathbf{v}(t)] + \mathbf{I}_{aff}(t) \\ \tau_m \dot{\mathbf{v}}(t) = -\mathbf{v}(t) + W_{ie} \cdot g[\mathbf{u}(t)] \end{cases}, \quad (1)$$

where $\tau_m = 30$ ms is the membrane time constant, $W_{ee} > 0$ is the matrix of synaptic weights connecting the pyramidal populations to each other, $W_{ie} > 0$ and $W_{ei} > 0$ mediate the inhibitory effects of the interneurons, and \mathbf{I}_{aff} is the vector describing afferent input arriving to cortex from the auditory pathway. Synaptic depression was assumed to affect the interactions between the pyramidal cells and was realized by modifying W_{ee} by a time-dependent depression term $a(t)$ so that the effective synaptic weight between columns i and j is $a_{ij}(t)w_{ij}$ and depends on the presynaptic activity through

$$\dot{a}_{ij}(t) = \frac{1-a_{ij}(t)}{\tau_a} - k w_{ij}(t) g[u_j(t)], \quad (2)$$

where $\tau_a = 0.8$ s is the decay time constant and $k = 20$ is a constant [13].

Structure. Synaptic weights were strongest within a column, with intra-column recurrent excitation mediated through the self-connection weight $w_{jj} = 6$ (i.e., diagonal values of W_{ee}). Inhibition was assumed to be local, so that W_{ei} had only diagonal values of magnitude 10. Longer range connections were made by the pyramidal cells. For determining inter-column connectivity, the columns were divided into 13 cortical areas, each containing $N_A = 16$ columns. Within each area, the probability of a directed connection between two columns was $p_0 = 0.75$. This connection could either be excitatory or inhibitory (i.e., targeting the pyramidal or interneuron population, respectively) with the probability of inhibitory connections being $p = 0.4$. All column-to-column excitatory and inhibitory connections had a magnitude of $100/N \approx 0.48$ and 2.0, respectively. Connection probability had a Gaussian drop-off from the diagonal with a standard deviation of $\sigma = 0.6 p_0 N_A$. Parameter values were chosen so that columns in core areas responded to preferred pure tone stimulation with a transient response followed by sustained activity [14].

Inter-area connectivity was modeled on results from primates [7]. Afferent tonotopically organized input targeted three "core" areas, which were interconnected with eight "belt" areas which, in turn, were connected with two "parabelt" areas (Fig. 1). This resulted in multiple parallel streams of connections, with a roughly

"rostral" and "caudal" subdivision. With no direct connections between the core and parabelt, the model had a serial structure. Connections between areas were excitatory and topographic, with most connections occurring near the diagonal on the relevant subdivision of W_{ee} (Gaussian drop-off, $\sigma = 0.6 \text{ p}N_A$). Strong and weak connectivity as indicated by the results of [7] was equivalent to connection probabilities $p_1 = 0.1$ and $p_2 = 0.05$, respectively.

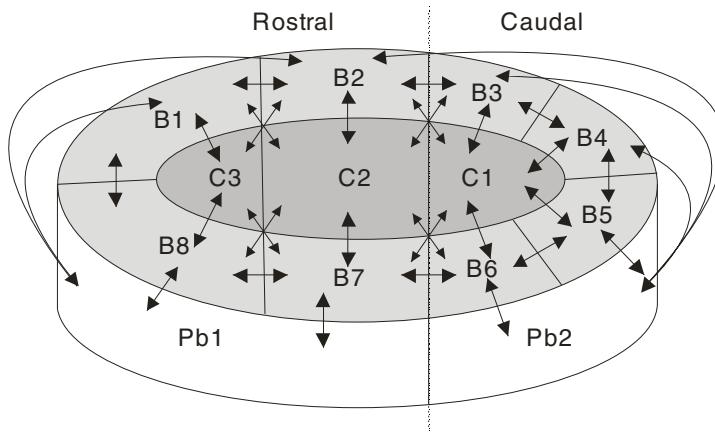


Fig. 1. The network consisted of core (C), belt (B), and parabelt (Pb) areas with strong (large) and weak (small) interconnections. Pb is targeted by a rostral and a caudal stream

2.2 Stimuli

We used 12 American-English consonant-vowel (CV) combinations of voiced stop consonants /b/, /d/, and /g/ and vowels /a/, /ae/, /i/, and /u/ [15]. These were randomly combined into eight CVCV pseudowords with an average duration of 450 ms. The stimuli were normalized with respect to their root-mean-square values. To model the subcortically carried out spectral analysis, the stimuli were transformed into spectrograms with 16 frequency channels (spanning frequencies 20 -13600 Hz) and 1-ms time resolution. This crude approximation provided a tonotopic representation of the stimuli capturing the time-evolution of their spectral content. To study the sensitivity of the model to the temporal structure of stimulation, the stimuli were also presented as time-reversed versions. Also, the pseudowords were divided into their constituent CV stimuli and these were presented separately to the model. Thus, the total stimulus set comprised 32 vocalizations. All stimuli had a linear onset and offset ramp of 5 ms. The stimulus spectrograms were normalized to unity and presented to the three core areas, with each frequency channel targeting one column per core area.

2.3 Analysis

To study the selectivity of the model to the speech stimuli, we calculated the Preference Index (PI) for each column on the basis of the firing rate of the pyramidal cell population. The PI, commonly utilized in primate studies to measure selectivity to

species-specific vocalizations, is derived by first identifying the preferred stimulus (PS), that is, the stimulus that elicits the maximal response. The PI is then defined as the number of stimuli to which a neuron yielded a response $>50\%$ of the maximal response. An index value of 1-3 indicates "strong" preference, that is, the neuron responds selectively to only a few of the stimuli.

A neuron is sensitive to the temporal structure of the stimulus if it elicits a large response only when the spectral content of the stimulus is delivered in a specific order. We used the time-reversed pseudowords and the constituent CV syllables to measure whether the model exhibited this so called temporal combination sensitivity. A column was considered temporally sensitive if the magnitude of the response to its preferred stimulus was more than double that to the reversed version of the PS and to the individual CV elements of the PS presented in isolation.

To study how selectivity to the vocalizations emerges out of the model, we manipulated the adaptation decay time constant in five logarithmic steps in the 60-960 ms range. Model structure was also varied by using five modified versions in which (1) the topographic connectivity between areas was transformed into random connectivity while keeping the number of connections constant, (2) the parabelt-belt and belt-core feedback connections were removed, (3) the serial structure was removed by introducing connections between the core and parabelt areas, (4) the parallel structure was removed by adding long-range connections between rostral and caudal areas, and (5) the area structure was completely removed by turning the model into a random network. The structures of these model variations were quantified by using the scaled small-world index σ [16]. Small-world networks inhabit the continuum between regular architectures and random networks. Unlike regular networks, they have short average node-to-node path lengths λ and, unlike random networks, they have a high clustering coefficient γ which expresses the probability that connection neighbors of a node are also connected to each other. The index σ is the ratio γ/λ with values >1 indicating a small-world structure (both γ and λ are scaled relative to equivalent values in random networks). Because of the probabilistic nature of synaptic weight distributions, each variation of the model was realized 30 times, and measurements were averaged over this set.

3 Results

With the default parameter values, the model exhibited both selectivity to the pseudowords and temporal combination sensitivity. The PI distribution had a peak at the value PI = 1 in all regions of the model, and thus resembled neurophysiological results from primates [8,10]. Strong selectivity was evident in all areas, with an average of 40% of belt and parabelt columns having a PI in the 1-3 range. Temporally sensitive columns were also present in all regions. However, they were relatively rare in the core areas (4%) but more frequently found in belt (17%) and parabelt (17%).

Manipulating the adaptation decay constant τ_a had little effect on the PI (Fig. 2, *top*). In contrast, the temporal sensitivity decreased monotonically to zero as τ_a was decreased from 960 ms to 60 ms. This effect was evident in all regions.

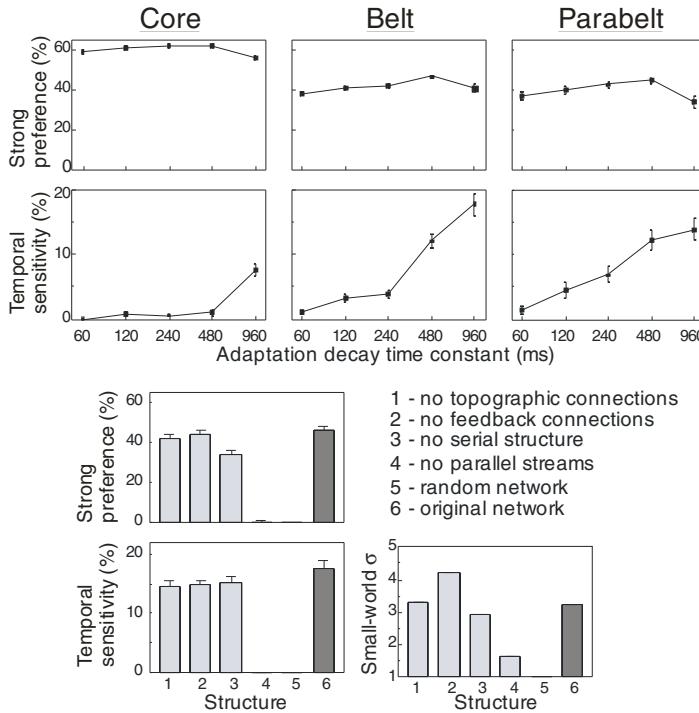


Fig. 2. Adaptation and network structure modify selectivity to speech signals. The number of columns displaying strong preference to certain pseudowords ($PI \leq 3$) is unaffected by the adaptation decay constant. In contrast, the number of temporally sensitive columns depends strongly on the constant (*top*). In such a column, the preferred pseudoword elicits a stronger response than its time-reversed version or the constituent syllables presented in isolation. In belt and parabelt, both strong preference and temporal sensitivity depends on network structure (*bottom*). The presence of columns selective to pseudowords is associated with a high small-world index.

The performance of the model in the face of variations in structure was robust in cases where inter-area topographic connections were randomized, feedback connections were abolished, or the serial core-belt-parabelt structure removed (cases 1-3; Fig. 2, *bottom*). However, in all regions, PI decreased to zero and temporal sensitivity was abolished as a result of added long-term connections which removed the rostral-caudal subdivision in the parallel structure (case 4). Performance was equally poor in the completely randomized network (case 5). Interestingly, the small-world index correlated with performance, with high selectivity to pseudowords and their temporal structure being associated with high values of σ (>3 ; Fig. 2, *bottom*).

4 Discussion

In the current study, we explored how the dynamics and structure of auditory cortex contributes to its ability to represent speech sounds characterized by a complex,

time-evolving structure. First, we found that shortening the decay time constant of activity-dependent synaptic depression (adaptation) abolishes the temporal sensitivity found in the model. Thus, adaptation with a long decay constant, matching the time span of the stimuli, seems to provide the central mechanism through which cells in auditory cortex become sensitive to the temporal structure of speech. Second, removing the rostral-caudal subdivision of activation decreased the number of columns displaying temporal combination sensitivity. Therefore, this sensitivity benefits from the presence of multiple processing streams. More generally, it seems to benefit from a small-world structure of the network. A randomized version of the model (with $\sigma = 1$) resulted in the abolishing of temporal sensitivity, while other equally drastic modulations of the structure (i.e., removal of serial activation, feedback connections, or topographic connectivity), retaining high values of σ , had little effect on temporal sensitivity.

We emphasize that the current selectivity to vocalization stimuli was not the result of training, that is, long-term adaptation to the statistical structure of the stimulus environment. Rather, selectivity was "instantaneously" available, the presence of synaptic decay coupled with an idiosyncratic structure being sufficient conditions for it to emerge. This implies that within-category selectivity in auditory cortex (e.g., in terms of individual speech sounds or monkey calls) does not yet signify category-specific processing (e.g., preference to speech sounds). For example, strong selectivity and temporal sensitivity seems to be evident with arbitrary sets of stimuli, as is indicated by further simulations of the current model using monkey vocalizations and abstract inputs. Our results therefore predict that auditory cortex provides an all-purpose machine for analyzing and segregating temporal structure, irrespective of stimulus category. In this context, it is an interesting question (albeit beyond the scope of the current study) how long-term exposure to a particular stimulus environment (e.g., native language) modifies the temporal sensitivity of auditory cortex. It is also possible that the readout from this analysis (e.g., by association areas bordering the parabelt, or by prefrontal cortex further down the chain) occurs in a category/speech-specific manner, as indicated by recent non-invasive results [4].

The current results could be useful for ASR development. Our simulations replicate the results from the primate brain whereby cells in the anterior auditory and prefrontal cortex respond selectively to vocalizations. Given that most neurons prefer more than one stimulus (both experimentally and in simulations), it is unlikely that speech sounds are represented via "grandmother cells" in a place code. It is more likely that a population code is used, and we speculate that this might exhibit the robustness necessary for speech recognition to occur in noisy environments - one of the primary and most difficult challenges in current ASR research. From the perspective of artificial neural networks, this population code might be separable through linear methods, making the current model effectively the reservoir in an echo state network [17]. Interestingly, paralleling the current results on the benefits of small-worldness, the predictive performance of such networks is enhanced by the reservoir having a small-world structure [18].

Acknowledgments. This study was supported by the Academy of Finland (Research Programme LASTU, projects 135003 & 135009).

References

1. Scharenborg, O.: Reaching Over the Gap: A Review of Efforts to Link Human and Automatic Speech Recognition Research. *Speech Comm.* 49, 336–347 (2007)
2. Young, E.D.: Neural Representation of Spectral and Temporal Information in Speech. *Phil. Trans. R. Soc. B* 363, 923–945 (2008)
3. Nelken, I.: Processing of Complex Stimuli and Natural Scenes in the Auditory Cortex. *Current Opin. Neurobiol.* 14, 474–480 (2004)
4. Binder, J.R., Frost, J.A., Hammeke, T.A., Bellgowan, P.S., Springer, J.A., Kaufman, J.N., Possing, E.T.: Human Temporal Lobe Activation by Speech and Nonspeech Sounds. *Cereb. Cortex* 10, 512–528 (2000)
5. Leaver, A.M., Rauschecker, J.P.: Cortical Representation of Natural Complex Sounds: Effects of Acoustic Features and Auditory Object Category. *J. Neurosci.* 30, 7604–7612 (2010)
6. Kaas, J.H., Hackett, T.A.: Subdivisions of Auditory Cortex and Processing Streams in Primates. *Proc. Natl Acad. Sci. USA* 97, 11793–11799 (2000)
7. Hackett, T.A., Stepniewska, I., Kaas, J.H.: Subdivisions of Auditory Cortex and Ipsilateral Cortical Connections of the Parabelt Auditory Cortex in Macaque Monkeys. *J. Comp. Neurol.* 394, 475–495 (1998)
8. Rauschecker, J.P., Tian, B., Hauser, M.: Processing of Complex Sounds in the Macaque Nonprimary Auditory Cortex. *Science* 268, 111–114 (1995)
9. Rauschecker, J.P.: Processing of Complex Sounds in the Auditory Cortex of Cat, Monkey, and Man. *Acta Otolaryngol. Suppl.* 532, 34–38 (1997)
10. Tian, B., Reser, D., Durham, A., Kustov, A., Rauschecker, J.P.: Functional Specialization in Rhesus Monkey Auditory Cortex. *Science* 292, 290–293 (2001)
11. May, P.J.C., Tiitinen, H.: Mismatch Negativity (MMN), the Deviance-Elicited Auditory Deflection, Explained. *Psychophysiol.* 47, 66–122 (2010)
12. Wilson, H.R., Cowan, J.D.: Excitatory and Inhibitory Interactions in Localized Populations of Model Neurons. *Biophys. J.* 12, 1–24 (1972)
13. Loebel, A., Nelken, I., Tsodyks, M.: Processing of Sounds by Population Spikes in a Model of Primary Auditory Cortex. *Front. Neurosci.* 1, 197–209 (2007)
14. Wang, X., Lu, T., Snider, R.K., Liang, L.: Sustained Firing in Auditory Cortex Evoked by Preferred Stimuli. *Nature* 435, 341–346 (2005)
15. Stephens, J.D.W., Holt, L.L.: A Standard Set of American-English Voiced Stop-Consonant Stimuli from Morphed Natural Speech (submitted),
<http://www.u.arizona.edu/~alotto/ACNS/StimuLibrary.htm>
16. Rubinov, M., Sporns, O.: Complex Network Measures of Brain Connectivity: Uses and Interpretations. *NeuroImage* 52, 1959–1969 (2010),
<http://www.brain-connectivity-toolbox.net>
17. Jaeger, H., Haas, H.: Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication. *Science* 304, 78–80 (2004)
18. Deng, Z., Zhang, Y.: Collective Behavior of a Small-World Recurrent Neural System with Scale-Free Distribution. *IEEE Trans. Neural Netw.* 18, 1364–1375 (2007)

A Geometric Bio-inspired Model for Recognition of Low-Level Structures

E. Ulises Moya-Sánchez and Eduardo Vázquez-Santacruz*

Department of Electrical Engineering and Computer Science,
CINVESTAV-IPN, Unidad Guadalajara, Av. del Bosque 1145, Colonia El Bajío,
Zapopan, Jalisco 45015, Mexico
{emoya, evazquez}@gd1.cinvestav.mx

Abstract. A new bio-inspired model is proposed in this paper. This model mimetizes the simple cells of the mammalian visual processing system in order to recognize low-level geometric structures such as oriented lines, edges and other constructed with these. It takes advantage of geometric algebra in order to represent structures and symmetric operators by estimating the relation between geometric entities and encoding it. This geometric model uses symmetric relations in which exist a invariance under some transformation according to biological models. It is based on a Quaternionic Atomic Function and its phase information to detect oriented lines, edges and geometric structures defined by lines. Also, it uses a geometric neural network to encode the transformation between lines and then for classifying of geometric structures.

Keywords: atomic function, quaternion, geometric neural networks.

1 Introduction

The human vision system has been studied in different ways such as light perception, motion and pattern recognition [1,2]. Lines and edges are important low-level information for biological systems, this information can be described in terms of symmetric relations allowing a compact data representation (a model) [2]. An architecture is needed to estimate a transformation between geometric entities. There are many bio-inspired models considering feature extraction without geometric representation [12,11]. Nevertheless, few investigations consider geometric aspects. This can be done using the geometric algebra framework. This new model mimetizes the behaviour of the orientation-selective simple cells of the primary visual system to extract geometric data and to encode the relation between these. It is strongly influenced by what is known about the visual system of human and animals. Our model is based on a Quaternionic Atomic Function (QAF) [3] and Geometric RBF Networks (GRBF-Ns) [4] in order to define the description of geometric patterns and classify them. The main contribution is the representation of a model which mimetizes the simple cells of the mammalian

* The authors thank CINVESTAV and CONACYT for supporting this work.

visual processing system using geometric algebra. It allows to encode geometric structures as complex as needed and it uses the GRBF-N [4] to estimate rotation transformation between geometric entities, then it can classify between different structures given that GRBF-N weights are *rotors*. The model can separate a quaternion into a magnitude (amplitude) and phases defined as complex numbers which are invariant to the brightness. Also there is a geometric invariance in terms of symmetric rotations in 3D space in the neural network. The model uses a special kind of filter, the *AF up*(x), instead of Gauss with a frequency shift (Gabor filters). Gabor filters have been used traditionally to mimetize the behavior of the simple cells of the primate visual system [2,1,12]. The main difference between Gabor and the *AF up*(x) is that *up*(x) has a compact support (good locality characteristic), and it is easy to derivate in any order. These two aspects take relevance for a future wavelet approach based on [3,6].

2 Atomic Functions

The *AFs* were introduced in 70's by V.L and V.A. Rvachev. They provide a new tool to new research areas in signal processing, pattern recognition, numerical methods, etc. [10]. The *AFs* are compactly supported infinitely differentiable solutions of differential equations (See Eq. 1) with a shifted argument [6] *i.e.*

$$Lf(x) = \lambda \sum_{k=1}^M c(k) f(ax - b(k)), |a| > 1, b, c \in \mathbb{R} \quad (1)$$

where $L = \frac{d^n}{dx^n} + a_1 \frac{d^{n-1}}{dx^{n-1}} + \dots + a_n$ is a linear differential operator with constant coefficients. In the *AF* class, *up*(x) is the simplest and at the same time, the most useful primitive function to generate other kinds of *AFs* [6]. The *AF up*(x) satisfies the Eq. 1 as follows $dup(x) = 2 (up(2x + 1) - up(2x - 1))$. The function *up*(x) has the following representation in terms of the Fourier transform:

$$up(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{iux} \prod_{k=1}^{\infty} \frac{\sin(u2^{-k})}{u2^{-k}} du. \quad (2)$$

The function *up*(x) is infinitely differentiable, *up*(0) = 1, it is even *up*(- x) = *up*(x), it is analytical in all the range, also it can represent any polynomial by means of its translations [6]. Atomic windows were compared with classic ones [6], for instance: the equivalent noise bandwidth, the 50% overlapping region correlation, the maximum conversion losses (in decibels), the asymptotic decay rate of the side lobes, among others. Atomic windows exceed classic ones in terms of the asymptotic decay rate [6]. Figure 1 shows the *up*(x), its Fourier Transform and its first derivate (*dup*(x)) (odd function). We use the *up*(x) because, its versatile, easy to derivate (only a shift) and compact in space domain. Also, $d^n up(x)$ can be used in a future work to mimetize the V1 response to the oriented movement in an analytic and natural way. The *AF* can detect lines and edges in a selective orientation using quaternion and phase information [2].

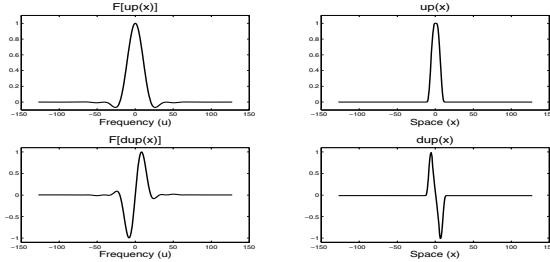


Fig. 1. Fourier Transforms: $F(up(x))$, $F(dup(x))$. Atomic functions: $up(x)$, $dup(x)$

3 Geometric Algebra

Geometric algebra allows to describe in a compact way to the lines, planes, and other geometric entities, as well as the representation of rotations, translations and other transformations [5]. This framework is used in order to update the weights of a GRBF-N and also to define the *QAF*. Here, the Clifford product is defined as the sum of the *inner product* and the *wedge product* of two vectors: $\mathbf{ab} = \mathbf{a} \cdot \mathbf{b} + \mathbf{a} \wedge \mathbf{b}$. A geometric algebra is a linear space of dimension 2^n , $n = p + q + r$, and $(p, q, r, \in N)$. The notation $\mathcal{G}_{p,q,r}$ represents the p vector basis that squares to +1, the q basis that squares to -1 and the r vector basis that squares to 0. The 3D space ($\mathcal{G}_{3,0,0}$) ($p=3$) has the following blade basis:

$$\left\{ \underbrace{1}_{\text{scalar}}, \underbrace{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3}_{\text{vectors}}, \underbrace{\mathbf{e}_{23}, \mathbf{e}_{31}, \mathbf{e}_{12}}_{\text{bivectors}}, \underbrace{\mathbf{e}_{123}}_{\text{trivectors}} \equiv I \right\} \quad (3)$$

In Eq. (3) the scalar is a blade¹ of zero degree, vectors are of first degree, bivectors are blades of second degree and *pseudoscalar* I is a blade of third degree. I is used to convert a geometric entity from its standard form to its dual form and vice versa.

3.1 Rotors in \mathcal{G}_3

The rotors are used to rotate geometric objects. A rotor in \mathcal{G}_3 satisfies the relation $\mathbf{R}\tilde{\mathbf{R}} = 1$, where $\mathbf{R} = u_0 + u_1\mathbf{e}_{23} + u_2\mathbf{e}_{31} + u_3\mathbf{e}_{12}$, $\tilde{\mathbf{R}} = u_0 - u_1\mathbf{e}_{23} - u_2\mathbf{e}_{31} - u_3\mathbf{e}_{12}$. The rotor can be expressed in exponential form as follows:

$$\mathbf{R} = \exp\left(-\frac{\theta}{2}\underline{n}\right) = \cos\left(\frac{\theta}{2}\right) - \sin\left(\frac{\theta}{2}\right)\underline{n} \quad (4)$$

where \underline{n} is a unit bivector that represents the rotation plane and $\theta \in \mathcal{R}$ represents the rotation angle. Each line orientation can be represented using a bivector and it is given by \mathbf{n} . Bivectors for the line direction are computed using two bivector points \mathbf{x}_1 and \mathbf{x}_2 (\mathbf{p}, \mathbf{q} in Fig. 2(a)), lying on the line, as follows:

$$\begin{aligned} \mathbf{n} &= (x_2 - x_1) = (x_{21} - x_{11})\mathbf{e}_2\mathbf{e}_3 + (x_{22} - x_{12})\mathbf{e}_3\mathbf{e}_1 + (x_{23} - x_{13})\mathbf{e}_1\mathbf{e}_2 \\ &= L_{n1}\mathbf{e}_2\mathbf{e}_3 + L_{n2}\mathbf{e}_3\mathbf{e}_1 + L_{n3}\mathbf{e}_1\mathbf{e}_2 \end{aligned} \quad (5)$$

Rotations of line orientations in $\mathcal{G}_{3,0,0}$ are given by $\mathbf{n}' = \mathbf{R}\mathbf{n}\tilde{\mathbf{R}}$

¹ The elements that limit a geometric algebra are called blades.

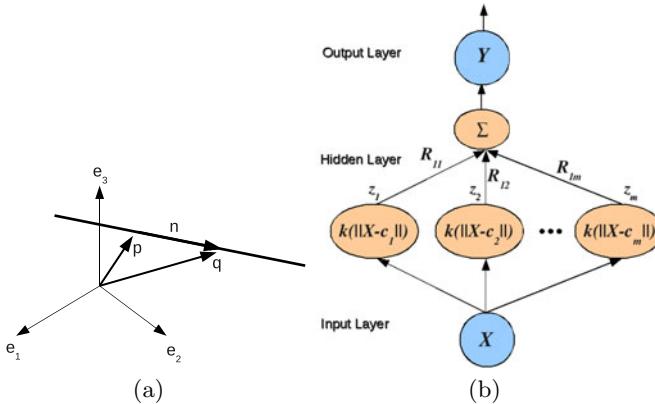


Fig. 2. (a) Line orientation in 3D. (b) Geometric RBF Network. See equation 9.

3.2 Quaternion Algebra

In terms of geometric algebra, the quaternion algebra \mathcal{H} is the basis of an even sub algebra [5]. Specifically the sub algebra \mathcal{G}_3^+ is isomorphic to quaternion algebra which is an associative non-commutative four-dimensional algebra, it consists of one real scalar and three imaginary elements [5,7].

$$q = a + bi + cj + dk \quad a, b, c, d \in \mathbb{R} \quad (6)$$

The units i, j obey the relations $i^2 = j^2 = -1, ij = -k$. Quaternion algebra is geometrically inspired [7] and the imaginary components can be described in terms of the basis of \mathbb{R}^3 space, $i \rightarrow e_2e_3, j \rightarrow e_1e_2, k \rightarrow e_3e_1$ where the $\{e_1, e_2, e_3\}$ are the basis of \mathbb{R}^3 . Another important property of the \mathcal{H} is the phase concept. A polar representation of q is $(|q|, \phi, \theta, \psi)$, where $|q|$ is the magnitude and the angles (ϕ, θ, ψ) represent the three quaternion phases[7] $q = |q|e^{i\phi}e^{k\psi}e^{j\theta}$. In image processing, the phase component has the main part of image information [9], it is useful for encoding lines and edges and are invariant to brightness changes [8], [2].

3.3 Quaternion Atomic Function Qup

The $up(x)$ function is easily extendable in two dimensions with the tensor product. Since a $2D$ signal can be split into an even (e) and odd (o) parts [7], we can separate the four components of $up(x, y)$ and represent it as a quaternion as follows [7,3]:

$$Qup(x, y) = Qup_{ee} + iQup_{oe} + jQup_{eo} + kQup_{oo} \quad (7)$$

Figure 3 shows a $Qup(x, y)$ in a space domain with its four components: real part Qup_{ee} and the imaginary parts $Qup_{eo}, Qup_{oe}, Qup_{oo}$, we can see even and odd symmetries.

4 Geometric Radial Basis Function Networks

Define the neural network as $g(P, X)$ where P is the set of its parameters and X is its input (figure 2(b)). This network uses hypercomplex valued neurons. By adaptive



Fig. 3. From left to right. Qup_{ee} , Qup_{oe} , Qup_{eo} and Qup_{oo} .

learning, P is tuned so that the training data fits the network model as well as possible. This network uses Gaussian basis functions (Eq. 8) to define the closeness of X to the centers $\hat{\mathbf{c}}$, also it uses rotors ($\hat{\mathbf{R}}$) as weights of output layer which are combined linearly to define the output Y of the network.

$$z_j = k \left(\frac{\|X - c_j\|}{\sigma_j^2} \right) \quad (8)$$

In Eq. 8, $k(\cdot)$ is a strictly positive radially symmetric function (kernel) with a unique maximum at its ‘centre’ c_j and which drops off rapidly to zero away from the centre, and has an appreciable value only when the distance $\|X - c_j\|$ is smaller than σ_j . The parameter σ_i is defined for each Gaussian unit to equal the maximum distance between centers. $X, Y, \hat{\mathbf{R}}, \hat{\mathbf{c}}$ are in \mathcal{G}_3 and are normalized. These parameters and σ_i values of each center define P [4]. Training set is a labeled pair X_i, Y_i that represents associations of a given mapping. The network updates \hat{R} considering to rotate the center c_i , this is done using the associated R_i (weight) and the respective radial basis function output [4]. This GRBF-NN works iteratively as follows [4]:

- 2) Iterate until no noticeable changes are observed in the centers c_k
 - a) *Sampling.* Draw a sample input \mathbf{x} coded by a multivector from the input space I . \mathbf{x} is input into the algorithm at iteration n
 - b) *Similarity definition.* $k(x)$ denote the index of the best-matching center for input \mathbf{x} . Find $k(x)$ at iteration n by using the minimum-distance Euclidean criterion or directance: $k(x) = \operatorname{argmin}_k \|x(n) - c_k(n)\|$, $k = 1, 2, \dots, m$,
- $c_k(n)$ is the center of the k th radial function at n th iteration. The dimension of the k -vector x is the same as that of the k -vector c_k and in this case is 3.
- c) *Updating.* Adjust the centers of the radial basis functions:

$$c_k(n+1) = \begin{cases} c_k(n) + \rho[x(n) - c_k(n)], & \text{if } k = k(x), \\ c_k(n), & \text{otherwise} \end{cases}$$

ρ is a *learning-rate parameter* in the range $0 \leq \rho < 1$

- d) *Error.* $e(n) = Y(n) - \sum_{i=1}^m R_i(n) (z_i(n) c_i(n)) \widetilde{R}_i(n)$
- e) *Rotor vector updating.* $R_i(n+1) = R_i(n) + \eta z_i(n) e(n)$
- f) *Continuation.* Increment n by 1

where $z_i(n) \in \mathbb{R}$ and $R_i(n), c_i(n), X_i(n), Y_i(n) \in \mathcal{G}_3$. As we can observe in the training algorithm step d), the GRBF-N uses the Clifford product which allows to do computations of training in the geometric algebra framework in order to develop the full model using a geometric framework. Note that the network is defined by,

$$g(P, X, n) = \sum_{i=1}^M R_i (z_i(n) c_i(n)) \widetilde{R}_i \quad (9)$$

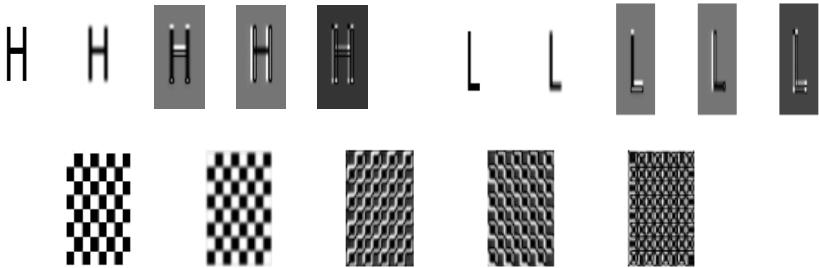


Fig. 4. Filtered images of H, L and chessboard. From left to right: image, real part, i-part, j-part.

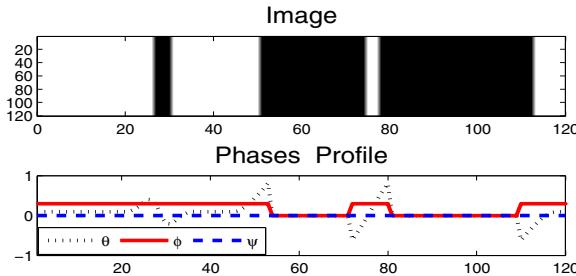


Fig. 5. From top to bottom: line and edge image, its phases profile

5 Experimental Analysis

We test our model with three images, we only work with oriented lines extracted with the *Qup* filter, however we show the capability of the model to extract lines or edges and to encode the geometric structure constructed by lines. Figure 4 shows the convolution of *Qup* filter on letters *L*, *H* and chessboard (*Qup* which was rotated 45°) images. We see a selective detection of lines in horizontal and vertical orientation, particularly in *i* and *j*. Fig. 5 shows the lines and edges image and quaternion phases profiles, the phase ϕ only responses to edges and white lines while θ responses with black and white lines considering each edge. The GRBF-N encodes the rotation transformation between lines and then it can work like a classifier of our basic geometric structures. For now, GRBF-N only allows one input and one output. If the geometric structure is more complex, then it is possible to use a set of filters and networks to detect the full structure. Input pairs (for training and testing) of lines were extracted from the filtered images using the *AF*. Different filters give different oriented lines (see Figs. 4, 5). Figure 6 (below) shows the evolution of the network output until finally encode the true transformation between two orthogonal lines from letter *L*. It can be seen that in first iterations, *i*, *j*, *k* are changing in a bigger range than in last ones. At the end, *i*, *j*, *k* change in a very small range. This shows that rotors converge

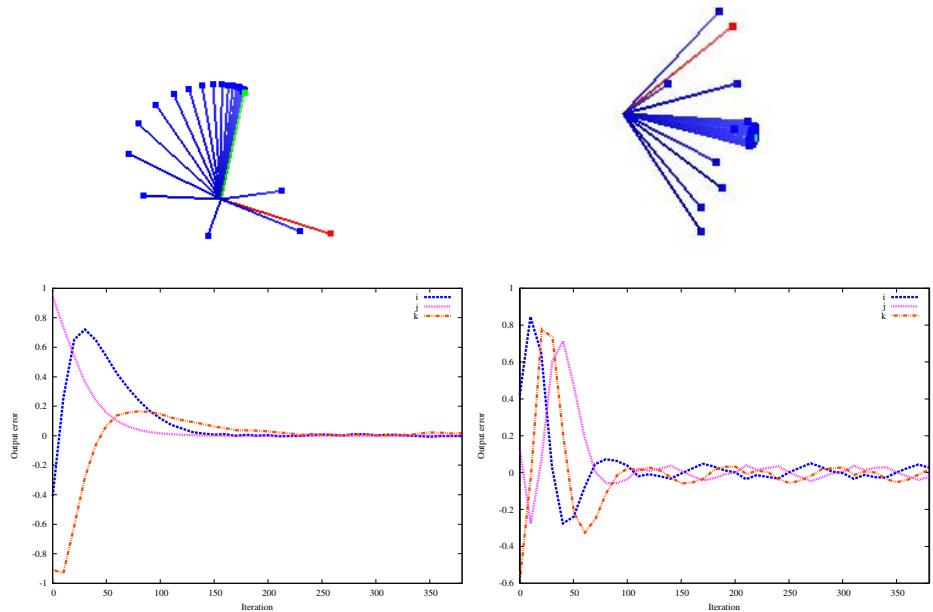


Fig. 6. Estimation of transformation between lines during learning process of networks 2 (left), 4 (right). Output evolution (above, in blue lines), Output error (below).

Table 1. Parameters related to the geometric networks

Network	Inputs	Centroids	Outputs	Geometric structure	η	ρ	Testing error
1	1	6	1	Right angle H	0.14	0.060	0.0213211 ± 0.00821
2	1	6	1	Right angle L	0.74	0.060	0.0463731 ± 0.00424
3	1	6	1	Right angle Chessboard	0.74	0.060	0.0297221 ± 0.01327
4	1	6	1	45 degrees Chessboard	0.15	0.055	0.0351754 ± 0.00534

to the expected value. Another GRBF-N shows a similar result using orthogonal lines extracted from letter *H* image. We have used 4 networks for experiments, their parameters are showed in Table 5. Networks 1, 2, 3 were used to find the transformation using data from *H*, *L* and chessboard images, respectively. Network 4 used non orthogonal lines from chessboard image. Figure 6 (above) shows that at first time, the network completely unknown the transformation (blue lines) between the input orientation (red line) and output² orientation (green line, which is considered as the target orientation). However, while more pairs of orientations are fed to the network, it can approximate to the real rotation between the geometric entities. Considering noise in image processing, this grade of accuracy is so useful to identify this kind of geometric patterns.

Once a network is trained to detect a specific structure, it is tested using similar structures (with orthogonal lines in the case of network trained using orthogonal lines and with non orthogonal lines in case of network 4), we get the

² Network output is expressed as $X = xe_1e_2 + ye_2e_3 + ze_3e_1$

testing errors showed in Table 5 which indicates that it works well. If the network has a big error, bigger than 0.15 (defined by experimental manner), then the tested structure is not recognized by it. Error is defined using Euclidean distance between target orientation and GRBF-N output orientation. It is computed as the training algorithm step **d**) indicates.

6 Conclusions

We have designed a new geometric bio inspired model which mimetizes the simple cells of the mammalian visual processing system in order to recognize low-level geometric structures. In this case, geometric structures are simple and defined by lines. GRBF-Ns encode each structure using geometric data extracted from images. *Qup* lets detect lines or edges in a specific orientation. The main advantage of this model is that filter, networks, and entities are defined in geometric algebra using hypercomplex numbers. The GRBF-N uses rotors which have advantages over other hypercomplex valued transformations given that rotors can be defined for any dimension and can also rotate blades of any grade. We are developing the MIMO (Multiple Input Multiple Output) GRBF-N which will be able to differentiate between different inputs. We will improve our model to detect and encode a more complex structure using wavelets (multiscale), also we consider to work with a oriented detection in motion using *dup*(x).

References

1. Bigun, J.: *Vision with Direction*. Springer, Heidelberg (2006)
2. Granlund, K.: *Signal Processing for Computer Vision*. Kluwer Academic Publishers, Dordrecht (2006)
3. Moya-Sánchez, E.U., Bayro-Corrochano, E.: Quaternion atomic function wavelet for applications in image processing. In: Bloch, I., Cesar Jr., R.M. (eds.) CIARP 2010. LNCS, vol. 6419, pp. 346–353. Springer, Heidelberg (2010)
4. Bayro-Corrochano, E., Vázquez-Santacruz, E.: A Geometric Radial Basis Function Network for Robot. IJCNN (2010)
5. Bayro-Corrochano, E.: *Geometric Computing for Perception Action Systems: Concepts, Algorithms, and Scientific Applications*. Springer, Heidelberg (2001)
6. Kravchenko, V.F., Perez-Meana, H.M., Ponomaryov, V.I.: *Adaptive digital processing of multidimensional signals with applications*. Moscow Fizmatlit (2009)
7. Bülow, T.: *Hypercomplex Spectral Signal Representations for the Processing and Analysis of Images*. Christian-Albert, Kiel University, Ph.D Thesis (1999)
8. Jähne, B.: *Digital Image Processing*. Springer, Germany (2002)
9. Kovesi, P.: Invariant measures of images features from phase information. University of Western Australia, PhD Thesis (1996)
10. Guyaev, Y.V., Kravchenko, V.F.: A New class of WA-Systems of Kravchenko-Rvachev Functions. Doklady Maths 75(2), 325–332 (2007)
11. Petrov, N.: Biologically motivated computationally intensive approaches to image pattern recognition. Future Generation Computer Systems 11, 451–465 (1995)
12. Castellanos-Sánchez, C.: A bio-inspired connectionist approach for motion description through sequences of images. In: de Sá, J.M., Alexandre, L.A., Duch, W., Mandic, D.P. (eds.) ICANN 2007. LNCS, vol. 4669, pp. 573–582. Springer, Heidelberg (2007)

View-Tuned Approximate Partial Matching Kernel from Hierarchical Growing Neural Gases

Marco Kortkamp¹ and Sven Wachsmuth^{1,2}

¹ Applied Informatics, Faculty of Technology

² Central Lab Facilities, CITEC

Universitätsstraße 25, 33615 Bielefeld, Germany

`{mkortkam, swachsmu}@TechFak.Uni-Bielefeld.DE`

Abstract. The problem of comparing images or image regions can be considered as the problem of matching unordered sets of high dimensional visual features. We show that an hierarchical Growing Neural Gas (GNG) can robustly be used to approximate the optimal partial matching cost between vector sets. Further, we extend the unordered set matching, such that the matching of local features pays attention to the structure of the object and the relative positions of the parts. This view-tuning is also realized with hierarchical GNGs and yields an efficient Mercer Kernel.

Keywords: kernel, partial set matching, view-tuned, hierarchical, GNG.

1 Introduction

Computing the similarity between images is a very difficult problem that is involved in many real world tasks, e.g. object recognition or image search. Today, the comparison of images is often based on local visual features (e.g. SURF [1]). Thereby, an image is represented by a set of d -dimensional real descriptors, where each vector robustly represents the distribution of gradients in a local image patch. Hence, the problem of comparing two images becomes the problem of comparing two vector sets of unequal cardinality. The optimal partial matching (detailed in Sec. 3) aligns the elements of the smaller set to the elements of the other set, such that the pairwise distances are minimized. See Fig. 1 (a) for the basic setup. The matching correspondence between local features is often a good similarity measure between the sets. However, because the feature sets are unordered, the sets similarity can be high despite a dissimilar composition of the parts, as seen in Fig. 1 (a) when comparing the cup and the abstract image.

In this paper, we discuss how a hierarchical Growing Neural Gas, namely the lbTreeGNG [2] can be used as an efficient Mercer kernel that approximates the optimal partial matching cost (Sec. 3). Further, we extend the unordered set matching idea and introduce a kernel that matches sets of features in a more structured way (Sec. 4). To that end, an additional lbTreeGNG is trained in the 2D image plane to build a view-tuned matching kernel that takes the relative positions of the features into account. The idea is illustrated in Fig. 1 (b). In doing so, the presented approach combines ideas from part-based models as well as Bag of Words (BOW) based methods, as detailed the next section.

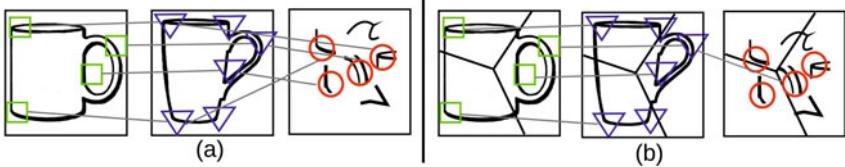


Fig. 1. (a) We see three different images: two cups and one “abstract” image. The cup in the middle is compared to the other two images. The green squares, blue triangles and red circles depict some local image regions for which visual features are computed. The gray lines sketch the alignment of the feature sets with the minimal cost. (b) The basic idea of the work is to learn a (hierarchical) decomposition of the image plane (indicated by black lines) and to match features only locally in the Voronoi cells. In doing so, the cup and the abstract image get more dissimilar, because only one of the global minimal alignments is preserved when taking the positions of the local features into account.

2 Related Work

There is a wide array of methods, that build on bag-of-words (BOW) representations, where images or regions are described by unordered collections of feature descriptors (e.g. see [3] for a detailed study). Further, there are some approaches that use BOW representations driven by a (hierarchical) spatial structure, e.g. [4], [5] and [6]. In [4], the authors propose a spatial pyramid matching (i.e. a hierarchical grid in 2D) to associate BOW sets with a loose notion of 2D spatial location. They demonstrate a significant gain, compared to kernels matching the BOW histograms globally. Also [5] uses a spatial pyramid match kernel for the representation and matching of shapes. The work of [6] divides an image into a regular grid and represents local BOW sets as sequences employed in a mismatch string kernel to measure similarity. Our approach goes beyond those static grid approaches and learns view-tuned spatial pyramids based on canonical object views. This also moves the notion of spatial pyramid more towards the idea of part-based models, where an object is described by parts and their geometrical relationship. Examples of part based models are pictorial structures [7] or the probabilistic graphical model of [8]. Moreover, there are approaches that try to learn distance functions from training data directly for application in nearest neighbor approaches (e.g. [9]).

3 Approximate Partial Matching with lbTreeGNGs

Let $X = \{x_1, \dots, x_m\}$ and $Y = \{y_1, \dots, y_n\}$ denote two sets of local visual features extracted from two different images or image regions. In this connection, let be $m \leq n$ and all $x_i, y_j \in \mathcal{F} = \mathbb{R}^d$. A *partial matching* is defined as $\mathcal{M}(X, Y; \pi) = \{(x_1, y_{\pi_1}), \dots, (x_m, y_{\pi_m})\}$, where $\pi: \{1, \dots, m\} \rightarrow \{1, \dots, n\}$ is an injective mapping of the indexes. The *cost* of a partial matching is defined as $\mathcal{C}(\mathcal{M}(X, Y; \pi)) = \sum_{x_i \in X} \|x_i - y_{\pi_i}\|_2$. An *optimal partial match* π^* is given by the assignment with the minimal matching cost, i.e. $\pi^* = \operatorname{argmin}_{\pi} \mathcal{C}(\mathcal{M}(X, Y; \pi))$.

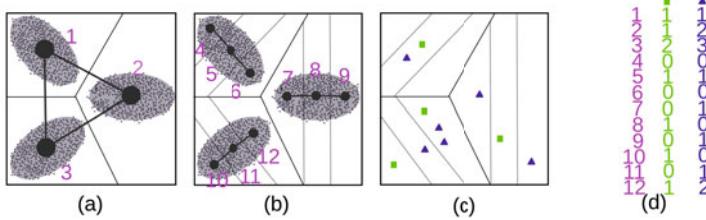


Fig. 2. Fig. (a) and (b) sketch $L = 2$ levels of a hierarchical codebook tree (lbTreeGNG) in a feature space \mathcal{F} with a branching factor $k = 3$. The light gray ellipses show the input distributions, while the dark gray circles show the learned reference vectors of the nodes. The dark gray lines represent the topological edges, which reflect the neighborhood of the nodes as learned from the input distribution. Further, the hierarchical Voronoi tessellation of the feature space is drawn as thin black solid and dotted lines. The magenta numbers are unique bin indexes that are associated with each node. Image (c) shows two feature sets X and Y in \mathcal{F} , drawn as green squares and blue triangles. Further, (d) shows the encoding of the feature sets as histograms over the tree bins.

The cost of the optimal partial matching can be used for measuring the similarity between two sets and it defines an alignment of the sets elements. The computation of the optimal partial matching is an instance of the assignment problem and can be solved optimally with a modified version of the Hungarian method [10] in $O(n^3)$.

Given a hierarchical codebook in a feature space \mathcal{F} . Fig. 2 (a,b) illustrates such a codebook with $L = 2$ levels and a tree branching factor of $k = 3$. A codebook tree of depth L and breadth k has a total number of $N_{L,k} = \sum_{i=0}^{L-1} k^i$ nodes. To each of such nodes n , we assign a unique index $j(n) \in \{1, \dots, N_{L,k}\}$ (cf. Fig. 2). When we encode a feature $x \in X$ by the tree, we pass it down the tree. This means, we start at level 0, find the node with the nearest reference vector and go to the nodes successor. Here, again the nearest node is found and the input vector moves to its successor. This is repeated until a leaf node is reached. Thus each input vector moves along a path of nearest nodes from level 0 to level $L - 1$. A set X is represented by an $N_{L,k}$ -dimensional histogram $H(X) = [h_1^X, \dots, h_{N_{L,k}}^X]$. The nodes n of the tree and the histogram bins are associated by the node index j as mentioned before, i.e. the count of a node n with the index $j(n)$ is h_j . To encode the feature set, each vector of $x \in X$ is passed down the codebook tree and the bin count of every node it passes is increased by 1. Examples for encoded sets can be seen in Fig. 2 (c,d).

In this paper, we use a limited branching Tree Growing Neural Gas (lbTreeGNG) [2] with a branching factor k and a depth of L to learn hierarchical codebooks. The lbTreeGNG is an efficient hierarchical method for online vector quantization, that keeps track of the topological structure of the input space and avoids overfitting. Fig. 2 (a,b) sketch a lbTreeGNG learned on the elliptic input distributions. Based on the histogram encoded feature sets $H(X)$ and $H(Y)$, we define the *vocabulary-guided pyramid match kernel* as follows:

$$\mathcal{K}(H(X), H(Y)) = \sum_{j=1}^{N_{L,k}} w_j \min(h_j^X, h_j^Y) \quad (1)$$

Thereby, w_j is a node dependent constant weighting factor with $w_j > 0$ for all $j \in \{1, \dots, N_{L,k}\}$. Similar to [11], the weights are set to $\frac{1}{\delta_j}$, where δ_j is learned for each node of the lbTreeGNG as the average distance of samples in the bin j . Equation (1) is a Mercer kernel, because the minimum intersection is a Mercer kernel [12] and kernels are closed under summation and scaling with a positive constant [13]. Note, that we use a simplified¹ version of the original matching in [11].

Experiments. In our experiments we investigate two things: First, we relate the performance of our simplified kernel $\mathcal{K}(\cdot, \cdot)$ to the performance of the originally proposed matching $\mathcal{C}(\cdot, \cdot)$. Second, using the standard parameters of the lbTreeGNG as stated in [2] we present how strong the matching performance of $\mathcal{K}(\cdot, \cdot)$ depends on the choice of the residual lbTreeGNG parameters m , b and the training set. The experiments are done on real image data from the ETH-80 database [14] in similar fashion to [11]. For each of 100 images, we compute 256 128-dimensional SURF descriptors on a grid. Then the resulting 100 descriptor sets are compared, which produces 10,000 matching costs. As a baseline, the cost of the optimal assignment π^* is computed (using GLPK²). As in [11], the quality of an approximation is assessed by the Spearman Rho, when comparing the optimal ranks to the approximative ranks. We compare the Spearman Rho of our $\mathcal{K}(\cdot, \cdot)$ to the original $\mathcal{C}(\cdot, \cdot)$ for different parameters $m \in \{m_1 = .15, m_2 = .015, m_3 = .0015\}$, $b \in \{3, 5, 10\}$, and $S \in \{101, 256\}$. Here, S denotes the training set, which is used to train the lbTreeGNG and 101 refers to the CalTech 101 database [15] (9144 images) and 256 refers to the CalTech 256 database [16] (30607 images).

¹ This is because in our view the argument made in [11] for $\mathcal{C}(\Psi(X), \Psi(Y))$ being a kernel does not hold. There, the authors define the vocabulary-guided pyramid match as $\mathcal{C}(\Psi(X), \Psi(Y)) = \sum_{i=0}^{L-1} \sum_{j=1}^{k^i} w_{ij} [\min(n_{ij}(X), n_{ij}(Y)) - \sum_{h=1}^k \min(c_h(n_{ij}(X)), c_h(n_{ij}(Y)))]$. Then this equation is re-written as $\mathcal{C}(\Psi(X), \Psi(Y)) = \sum_{i=0}^{L-1} \sum_{j=0}^{k^i} (w_{ij} - p_{ij}) \min(n_{ij}(X), n_{ij}(Y))$. We can infer that $p_{ij} = \frac{w_{ij} \sum_{h=1}^k \min(c_h(n_{ij}(X)), c_h(n_{ij}(Y)))}{\min(n_{ij}(X), n_{ij}(Y))}$. The authors now argue that $\mathcal{C}(\Psi(X), \Psi(Y))$ is a kernel for $w_{ij} \geq p_{ij}$, because the minimum intersection is a Mercer kernel [12] and kernels are closed under summation and scaling with a positive constant [13]. However, $(w_{ij} - p_{ij})$ can not be treated as a constant, because we can see above that the term p_{ij} does depend on the input of the kernel and thus can not be constant, i.e. $p_{ij} = p_{ij}(\Psi(X), \Psi(Y))$. As we did not find a correction for the proof, we dropped the subtraction of the second part in $\mathcal{C}(\Psi(X), \Psi(Y))$, which causes the problem. In fact, the subtraction only cleans the histogram counts by removing matches that has already been found in deeper levels of the tree.

² <http://www.gnu.org/software/glpk/>

Results. The results are summarized in Fig. 3. As it can be seen, the introduced simplification is not problematic, because the approximation performance of $\mathcal{K}(\cdot, \cdot)$ and $\mathcal{C}(\cdot, \cdot)$ is nearly identical. In addition, the results suggest that the performance is very robust to the concrete choice of the parameters, because there are no large differences w.r.t. the Spearman Rho. The fact that the Spearman Rho in our experiments is approx. .05 smaller than in [11] (with global weights) may either be a consequence from using SURF instead of SIFT, or from not training the codebook on the test images.

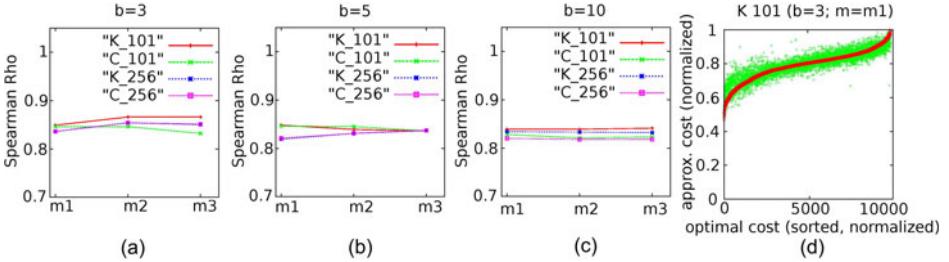


Fig. 3. (a-c) Approximation quality of the original $\mathcal{C}(\cdot, \cdot)$ [11] and our the simplified version $\mathcal{K}(\cdot, \cdot)$ w.r.t. the optimal partial matching under different parameter settings (details see text). (d) Optimal costs versus approximated costs.

4 View-Tuned Approximate Partial Matching

In this section, we introduce the view-tuned vocabulary-guided pyramid match kernel. The kernel employs two lbTreeGNGs: one class independent GNG in the d -dimensional feature space \mathcal{F} (as explained in Sec. 3) and one class specific GNG in the 2D image plane. To learn the 2D lbTreeGNG, we sample points from the unit image plane $\xi \in [0, 1]^2$ according to a distribution $P(\xi)$. Here, $P(\xi)$ should have high values for locations where relevant object parts occur, with respect to a specific feature type and view-point. Fig. 4 column (a) shows two examples on the ETH-80 database, where we derived the $P(\xi)$ distribution from normalized averaged gradient magnitude images for similar views of cars and horses. Column (b) illustrates points sampled according to those distributions, where the coordinates are used as 2D input vectors to train the 2D lbTreeGNG. Further, column (c) and (d) show the first and the second level of the trained lbTreeGNGs. The thin circles indicate the average point distance estimation δ_j , which are used in the weighting (compare (1) and (2)). Subsequently, column (e) presents the hierachal Voronoi tessellation induced by the 2D codebook trees. Finally, Fig. (f,1) and (f,2) show 256 keypoint locations that are also sampled according to $P(\xi)$ and which are associated to the class specific 2D lbTreeGNGs. When comparing images for a specific class, e.g. cars, the associated keypoint set is used. Thereby, at each keypoint location a vector is computed that represents the distribution of gradients in a local image patch around the keypoint. Note,

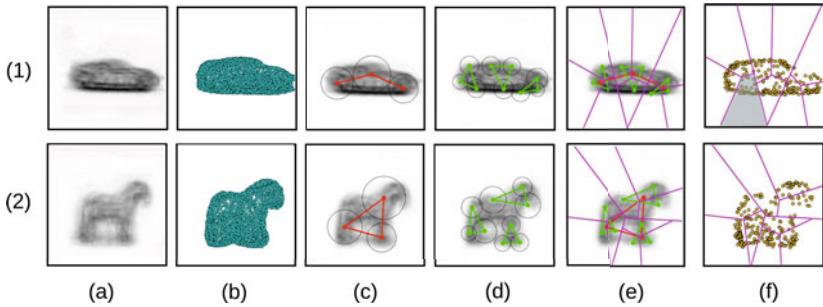


Fig. 4. Two examples for view-tuned vocabulary guided pyramid match kernels (explanation given in the text)

that before in the last section, that keypoint locations have just been defined to be on a static regular 16×16 grid. By incorporating $P(\xi)$ it is possible to get a better tuning of the keypoint locations to a canonical view of a specific object. When comparing the two descriptor sets, the same learned 2D tessellation is employed as follows to pay to attention the relative locations of the local features.

Given a trained 2D lbTreeGNG associated to a class ω , and with a depth R and a branching factor b . Let \mathcal{V}_l^ω denote the hierarchical Voronoi cell belonging to the node with index l of that lbTreeGNG. Further, let $f(\mathcal{V}_l^\omega, I)$ denote the set of descriptors on an image I , which keypoints are located inside the hierarchical Voronoi cell of node l (e.g. features computed in the gray region in Fig. 4 (f,1)).

Let I and J be two images or image regions that should be compared. We define the *view-tuned vocabulary-guided pyramid match cost* as

$$\mathcal{T}^\omega(I, J) = \sum_{l=0}^{N_{R,b}} \tilde{w}_l \mathcal{K}(H(f(\mathcal{V}_l^\omega, I)), H(f(\mathcal{V}_l^\omega, J))). \quad (2)$$

Thereby, \tilde{w}_l is a node dependent constant weighting factor with $\tilde{w}_l > 0$ for all $l \in \{1, \dots, N_{R,b}\}$. As before the weights are set to $\frac{1}{\delta_l}$, where δ_l is learned for each node of the lbTreeGNG as the average distance of samples in the bin j . Since (1) is a Mercer kernel, we can obviously use the same argument made for (1) to proof that (2) is a kernel as well.

The proposed kernel $\mathcal{T}^\omega(., .)$ implements a tuned fine to coarse matching of local image parts for similar views of instances of a class ω . The approach combines ideas from part-based models as well as BOW-based models: the kernel locally matches BOW representations, while paying attention to the structure of the object and the relative positions of the BOW-based parts. Due to the hierachal form of the matching in 2D, the proposed method is intended to be robust to small deformations and rotations. In addition, the minimum intersection is intended to be robust to clutter and occlusion, to a certain extend. Due to the fact that the presented matching is a Mercer kernel, it is directly suitable for kernel-based machine learning techniques, like the Support Vector Machines (e.g. [13]). Although, a strategy for ensembling is needed when combining

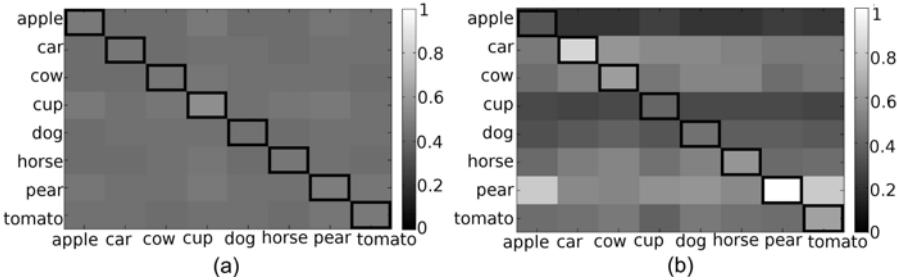


Fig. 5. Results for the second experiment. A view-tuned kernel enhances the in-class and between-class similarity, i.e. the difference between a diagonal element and the other elements in the same row.

several view-tuned classifiers. Beyond, our matching kernel is independent from the articular type of the local feature, i.e. color-, appearance- or shape-based local descriptors could be used.

The space complexities are as follows: $O(b^R)$ for a pyramid structure in $2D$; $O(k^L)$ for a pyramid structure in \mathcal{F} and $O(nL)$ for a sparse histogram $H(.)$. The time complexities w.r.t. distance computations are as follows: $O(2nL)$ for the histograms $H(.)$ in $\mathcal{K}(.,.)$ and $O(R2nL)$ for a complete $\mathcal{T}^\omega(.,.)$ kernel. Thus, the time complexity of the proposed kernel is still linear w.r.t. n and allows efficient matching of images or regions.

Experiment. In this experiment, we want to study how the similarity between image pairs is altered, when using the view-tuned kernel instead of $K(.,.)$ as the similarity. We measure this in terms of the average in-class and between-class similarity using the side views (090 images) of objects in the ETH-80 database. For instance, the average in-class similarity for apple is given by the average of all pairwise apple image similarities. Further, the between-class similarity is given by the average similarity between all pairs of images from two different classes, e.g. apple and car. The results for the vocabulary-guided pyramid match kernel $\mathcal{K}(.,.)$ are plotted in 5 (a). In addition, the results for the view-tuned vocabulary-guided pyramid match kernel $\mathcal{T}^\omega(.,.)$ are plotted in 5 (b). Note, that the result matrix of $\mathcal{T}^\omega(.,.)$ is not symmetric, because the classes of the rows refer to the class ω of 2D lbTreeGNG. The results have been normalized to $[0, 1]$ to be comparable.

Results. As visible in Fig. 5, the difference between the a diagonal element and the elements of the same row is higher in (b) than in (a). This means, that elements of a class get similar when using the view-tuned kernel. This is what we intended by incorporating additional knowledge into the matching. In numbers, the in-class similarity is increased by 14%. On the other hand, the global between-class similarity (the off-diagonal elements) is only slightly decreased by 3%. One explanation for this could be, that some classes still have a quite similar shape, for instance cow, dog and horse. However, the results

also suggest that a combination of several view-tuned kernels for each class may further enhance the reported effects (e.g. car side view and car front view).

5 Conclusion and Future Work

We introduced a simplified version of the vocabulary-guided pyramid match and showed that a hierarchical Growing Neural Gas can robustly be used to approximate the optimal partial matching cost between high dimensional feature sets. Further, we presented an extension for matching unordered feature sets alongside a learned 2D structure. Thereby, another hierarchical Growing Neural Gas was employed in order to pay attention to the structure of the object and the relative positions of the parts. The matching is an efficient Mercer Kernel and increases the in-class similarity while decreasing the between-class similarity of images. One of the next steps is to evaluate the proposed view-tuned matching kernel in kernel-based classification tasks with different types of features.

References

1. Bay, H., Tuytelaars, T., Van Gool, L.: SURF: Speeded up robust features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) *ECCV 2006*. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006)
2. Kortkamp, M., Wachsmuth, S.: Continuous visual codebooks with a limited branching tree growing neural gas. In: Diamantaras, K., Duch, W., Iliadis, L.S. (eds.) *ICANN 2010*. LNCS, vol. 6354, pp. 188–197. Springer, Heidelberg (2010)
3. Zhang, J., Marszalek, M., Lazebnik, S., Schmid, C.: Local features and kernels for classification of texture and object categories: A comprehensive study (2006)
4. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: *CVPR* (2006)
5. Bosch, A., Zisserman, A., Munoz, X.: Representing shape with a spatial pyramid kernel (2007)
6. Lu, Z., Ip, H.: Image categorization with spatial mismatch kernels. In: *CVPR* (2009)
7. Felzenszwalb, P., Huttenlocher, D.: Pictorial structures for object recognition. *International Journal of Computer Vision* 61(1), 55–79 (2005)
8. Fergus, R., Perona, P., Zisserman, A.: A sparse object category model for efficient learning and exhaustive recognition. In: *CVPR* (2005)
9. Ramanan, D., Baker, S.: Local distance functions: A taxonomy, new algorithms, and an evaluations. In: *ICCV* (2009)
10. Kuhn, H.W.: The hungarian method for the assignment problem. *Naval Research Logistics Quarterly* 2(1-2), 83–97 (1955)
11. Grauman, K., Darrell, T.: Approximate correspondences in high dimensions. In: *NIPS* (2006)
12. Odene, F., Barla, A., Verri, A.: Building kernels from binary strings for image matching. *IEEE Transactions on Image Processing* 14(2), 169–180 (2005)
13. Shawe-Taylor, J., Cristianini, N.: *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge (2004)

14. Leibe, B., Schiele, B.: Analyzing appearance and contour based methods for object categorization. In: ICCV (2003)
15. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In: CVPR (2004)
16. Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology (2007)

ANGE – Automatic Neural Generator

Leonardo Reis, Luis Aguiar, Darío Baptista, and Fernando Morgado Dias

Centro de Competências de Ciências Exactas e da Engenharia, Universidade da Madeira –
Campus penteada, 9000-039 Funchal, Madeira, Portugal

Abstract. Artificial Neural Networks have a wide application in terms of research areas but they have never really lived up to the promise they seemed to be in the beginning of the 80s. One of the reasons for this is the lack of hardware for their implementation in a straightforward and simple way. This paper presents a tool to respond to this need: An Automatic Neural Generator. The generator allows a user to specify the number of bits used in each part of the neural network and programs the selected FPGA with the network. To measure the accuracy of the implementation an automatically built neural network was inserted in a control loop and compared with Matlab.

Keywords: Artificial Neural Networks, Feedforward Neural Networks, System Generator, Matlab, Xilinx, Simulink, Integrated Software Environment.

1 Introduction

Artificial Neural Networks (ANN) are structures composed of neurons, connected in networks with massive parallelism that can greatly benefit from hardware implementation. To benefit from this parallelism a hardware implementation is needed but the number of implementations present in the literature and their capacity for being generic is very low. This situation can be changed if a simple and fast alternative for implementing ANNs in hardware is supplied[1].

A few attempts have been made in building such solutions [2-4] but the proposed implementations are still very simple. In [2] we can find very simple blocks with low resolution and oversimplified activation functions. In [3], though using only Heaviside functions, we find a generic Simulink block for a neuron that can be translated by System Generator. The most promising proposal found in the literature is in [4]. In this paper an IP Core is proposed for building synthesizable VHDL code for ANN but it only uses a simplified fuzzy prepared activation function that reduces the precision.

More work has been done regarding the manual implementation of ANN. The difficulties for these implementations are well known: the non-linearity of the hyperbolic tangent; the number of bits necessary to obtain high precision; the choice of using floating or fixed notation and the resources needed to implement a true parallel solution.

The implementation of the hyperbolic tangent as activation function received a large share of the attention in this area. The best solutions can be found in [5], where the maximum error is 5×10^{-8} in a control loop; in [6] the solution is based in a Taylor

series which achieved an error of 0,51%; in [7] a piecewise linear implementation is proposed which obtained 0.0254 of “standard deviation with respect to the analytic form”, in [8] a set of five polynomial 5th order approximations is proposed for a maximum error of 8×10^{-5} , using the sigmoid function.

In this paper we present an Automatic Neural GEnerator (ANGE) that programs ANNs in an FPGA with the characteristics defined by the user. The ANGE tool uses Matlab and Simulink (from Mathworks) and Integrated Software Environment (ISE), also from Xilinx. Matlab and Simulink tools are widely used in the ANN field and the ISE and System Generator are tools from the Xilinx company to work with programmable hardware which is the preferred solution for a large part of the ANN implementations due to the acceptable price for a reduced number of copies [9]. The initial part of this work was presented in [1].

The rest of the paper is organized as follows: section 2 describes shortly the neuron implementation developed; section 3 introduces the main tool; section 4 shows some of the results obtained; section 5 draws the conclusions and section 6 points the directions for future work.

2 Neuron Implementation

The first step towards the automatic implementation of ANN was obtained implementing the neuron. Using the tools mentioned in the previous section, the neuron is configured with a mask that is represented in figure 1.

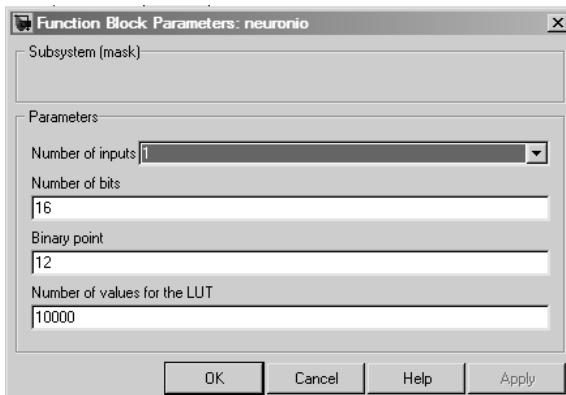


Fig. 1. Mask for selecting the neuron parameters

The parameters used in this mask are: the number of inputs, number of bits for the inputs, the position for the fixed point and the number of values for the Look Up Table (LUT) that implements the hyperbolic tangent activation function.

This mask hides the Matlab code that is responsible for placing the components in a Simulink model and connect them in order to complete the desired system. An example of a Simulink model for a neuron with four inputs can be seen in figure 2.

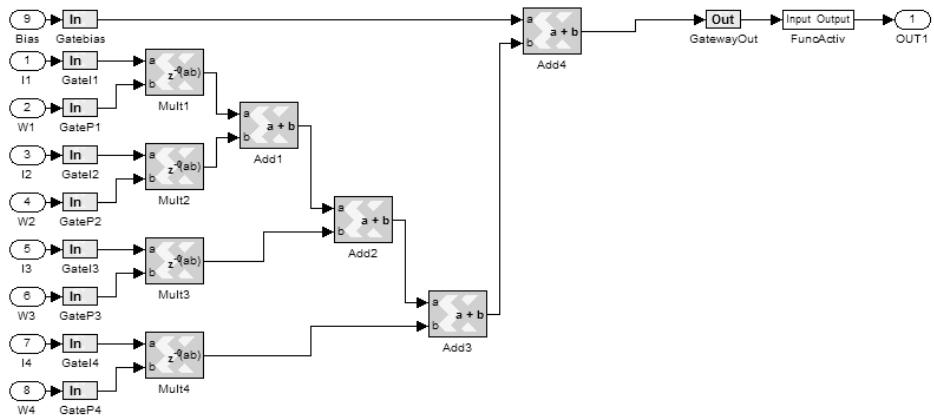


Fig. 2. Simulink model for a neuron with four inputs

Most of the blocks in this figure are common blocks. The FuncAtiv block holds the LUT that represents only part of the hyperbolic tangent since it is an odd function and transforms it in order to supply all the values necessary to the implementation.

The configuration windows of the ROM that implements the LUT can be seen in figure 3. The left side shows a ROM with 10000 values and the choice of the values that fill the memory. The right side shows the use of 32 bits, with only one for the integer part.

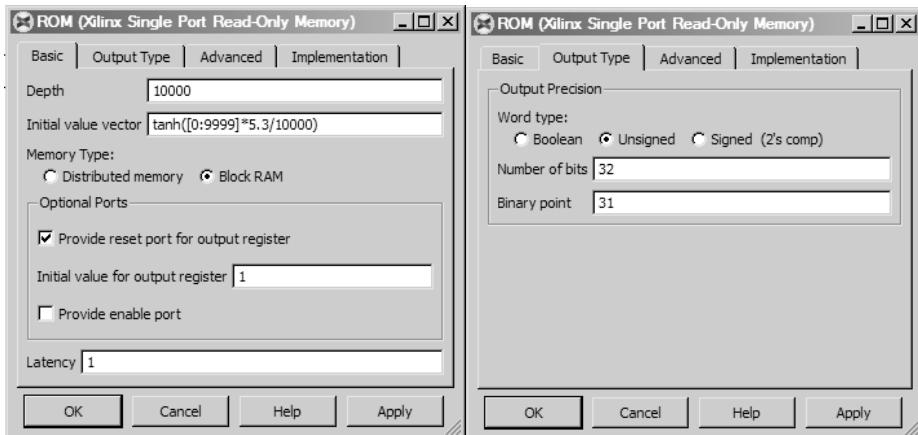


Fig. 3. Configuration of the ROM that implements the LUT

3 ANGE Details

ANGE, for its first version, is prepared to work with Multi-layer Perceptron or Feedforward Neural Networks with linear activation functions or hyperbolic tangents.

The hyperbolic tangent is implemented in its simplest way though trying to maximize its performance and minimize the error obtained: using a LUT and reduced to the smallest part that can be used to represent the whole function.

ANGE runs over Matlab R2007b, with System Generator 10.1 and ISE 10.1 and is capable of configuring hardware in a Field Programmable Gate Array (FPGA) for an ANN as large as the FPGA available allows. ANGE main window is in figure 4.

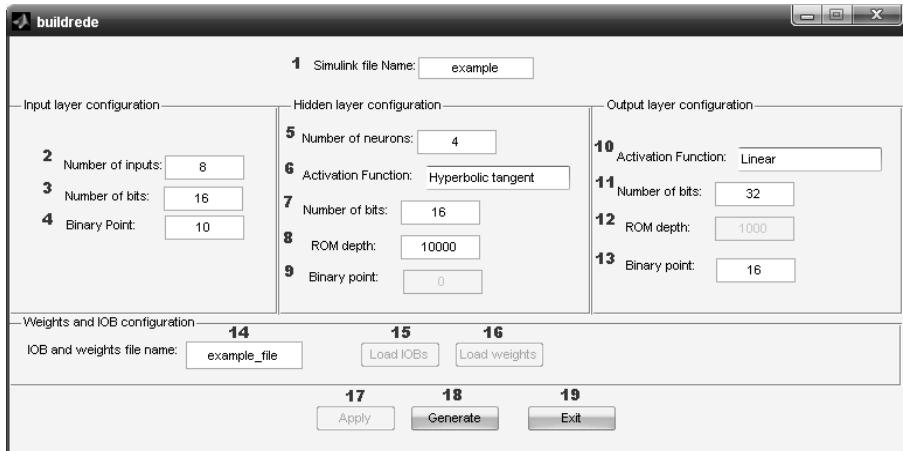


Fig. 4. ANGE main window

As can be seen, the number of bits for the inputs, outputs and activation function can be selected by the user to accommodate the needs and capacity of the hardware available, using 3, 7 and 11. The position of the binary point (System Generator uses fixed point) can also be selected, using 4,9 and 13, in order to maximize the number of bits available after the point to increase the resolution.

The weights can be uploaded to configure all the network at once and it is also possible to upload information about which of the Input/Output Blocks (IOB) to use and what to connect to each of them, providing the file name in 14 and pushing 15, 16 or both.

After selecting the configuration and characteristics of the network, ANGE will automatically generate a Simulink Model file, similar to the one represented in figure 5. The large blocks represented in this figure are the neurons, the inputs are identified by the word “In” and the weights are introduced using the constant blocks. As can be seen the ANN is implemented using full neurons and has a full parallel structure.

ANGE can also be used to create co-simulation blocks. These blocks can be inserted in a Simulink library and used in Simulink models, as shown in figure 6, inserting the FPGA in the loop and approaching the simulation to the real functioning of the system.

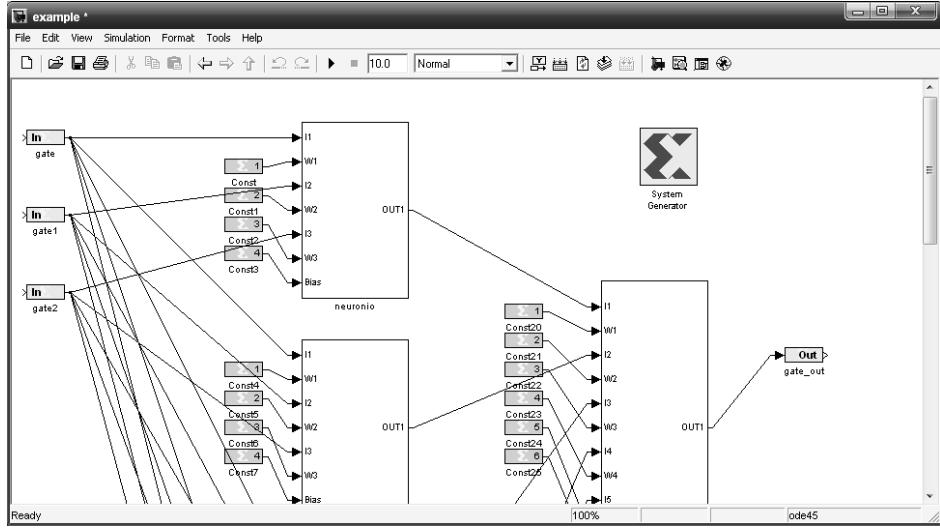


Fig. 5. Example of an ANN generated by ANGE with the weights loaded

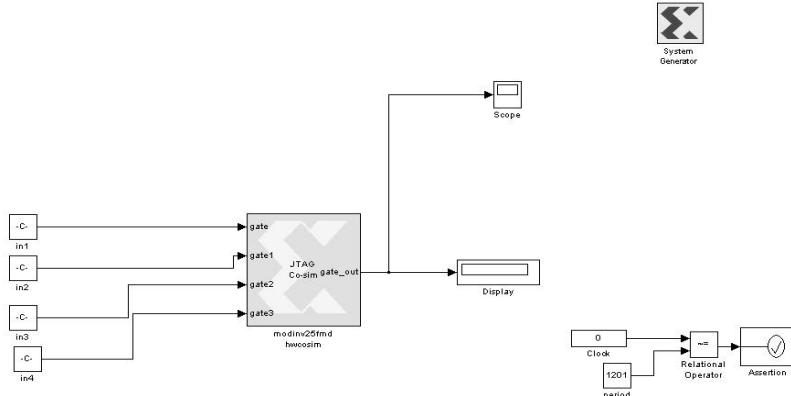


Fig. 6. Example of an ANN generated by ANGE used as a co-simulation block

4 Test Results

ANN models serve a purpose only when they represent some system or behaviour. To test ANGE's implementation and evaluate the error introduced by its fixed point notation, models from a reduced scale kiln were used. This kiln was prepared for the ceramic industry and further details can be seen in [10].

For these tests two sets of models that represent a direct and an inverse model of the system were used in order to construct a Direct Inverse Control loop, as represented in figure 7.

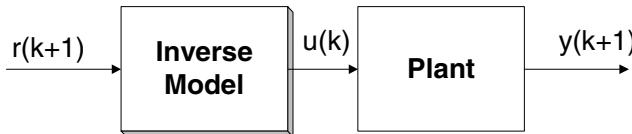


Fig. 7. Block diagram of the Direct Inverse Control

This kind of loop, though very simple and easy to understand, requires the models to have a very good match and therefore is the best loop to be used when testing hardware implemented models because if the implementation reduces the quality of the models it will be seen directly on the control results. The two sets of models, though representing the same system, were trained under different conditions and are different in the number of neurons in the hidden layer.

The ANNs were implemented using 16 bits for the inputs and representation of the hyperbolic tangent and 32 bits for the output. The LUT that holds the hyperbolic tangent values contains 10000 values.

The models were tested using two reference signals and they compare a result obtained implementing both models in Matlab with another one where the inverse model is implemented in an FPGA, using co-simulation. Some of the results can be seen in figure 8 and they were measured in terms of Mean Square Error (MSE) and are summarized in table 1.

Table 1. Mean square error in co-simulation and Matlab

Model and reference	Co-simulation	Matlab	Error change
Fmdb1 – Ref1	2,5155	2,5173	0,07%
Fmdb1 – Ref2	72,4492	72,3548	0,13%
25fmd – Ref1	0,01507	0,01307	15,31%
25fmd – Ref2	8,1688	8,1542	0,18%

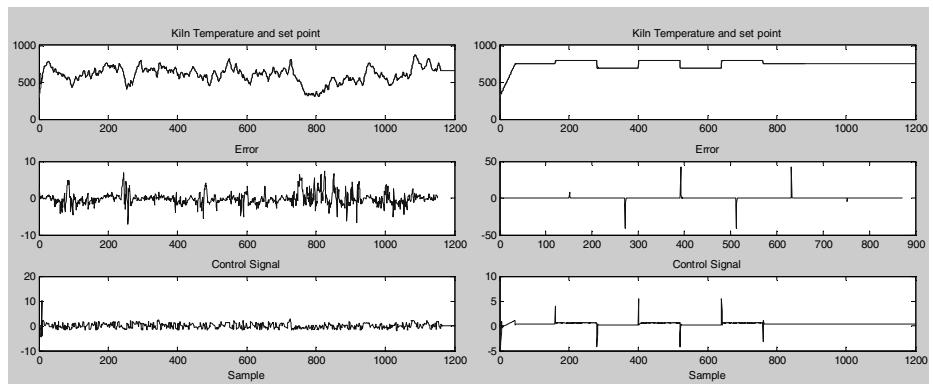


Fig. 8. Results of model fmd1b with reference 1 and model 25fmd with reference 2

As can be seen, the maximum error introduced in the hardware implementation of the models was of 15,31%. This value is not very low but there are important aspects that should be mentioned: the control loop maintained stability and the error introduced seems to be constant and therefore represents a larger percentage when the error in Matlab is smaller. The error introduced results from using less bits, fixed point notation and a LUT to represent the hyperbolic tangent. Its maximum value can be derived by the number of bits truncated and the maximum step between consecutive values in the LUT. As a result the error introduced should be bounded and small and be reduced with the increase of the number of bits used in the solution.

In hardware implementations besides precision it is important that the final solution does not use too many resources. To evaluate this, table 2 shows a resume of the resources used in proportion to the capacity of the FPGA, a Virtex 5 5VFX30TFF665-2S. The ANNs used are not very big (25fmd has 4 inputs and 8 neurons, while fmdb1 has 4 inputs and 6 neurons) and the FPGA is a small Virtex 5, which means that with a more recent FPGA it is possible to implement an ANN more than 20 times larger than the ones used as example here.

Table 2. Resume of the resources used in the implementation of both neural models for LUT with 10000 values

Resources	RAM16 % of 68	Registers % of 20480	LUTS % of 20480	DSP48ES % of 64
25fmd	58	22	31	62
fmdb1	44	17	22	46

5 Conclusion

This paper presents ANGE, an Automatic Neural Hardware Generator and shows some of the results obtained with it.

A test is presented with a small network with a medium resolution fitted in a small Virtex 5 FPGA. The control loop shown presents acceptable error with medium resolution and keeps the loop stable. With larger number of bits (possible with ANGE's present version) and more accurate implementation of the hyperbolic tangent these results will be further improved.

ANGE will allow fast prototyping using the preferred hardware target for the Neural community and is expected to contribute to a new growth of hardware implementations of ANN.

6 Further work

The work presented, although represents an important step, can be improved specially regarding the activation function. The next version of ANGE will have more options for the implementation of the hyperbolic tangent.

References

1. Aguiar, L., Reis, L., Morgado Dias, F.: Neuron Implementation Using System Generator, 9th Portuguese Conference on Automatic Control (2010)
2. Moctezuma Eugenio, J. C., Huitzil, C. T.: Estudio Sobre la Implementación de Redes Neuronales Artificiales Usando XILINX System Generador, XII Workshop Iberchip, Costa Rica (2006)
3. Tisan, A., Buchman, A., Oniga, S. Gavrincea C.: A Generic Control Block for Feedforward Neural Network with On-Chip Delta Rule Learning Algorithm, North Univ. of Baia Mare, Baia Mare, 30th International Spring Seminar on Electronics Technology, Romania (2007)
4. Rosado-Muñoz, A., Soria-Olivas, E., Gomez-Chova, L., Vila Francés, J.: An IP Core and GUI for Implementing Multilayer Perceptron with a Fuzzy Activation Function on Configurable Logic Devices, Journal of Universal Computer Science, vol. 14, no. 10, pp. 1678--1694 (2008)
5. Ferreira, P., Ribeiro, P., Antunes, A., Morgado Dias, F.: A high bit resolution FPGA implementation of a FNN with a new algorithm for the activation function, Neurocomputing, vol. 71, Issues 1-3, pp 71--77 (2007)
6. Arroyo Leon, M. A., Ruiz Castro, A., Leal Ascencio, R. R.: An artificial neural network on a field programmable gate array as a virtual sensor, Proceedings of The Third International Workshop on Design of Mixed-Mode Integrated Circuits and Applications, Puerto Vallarta, pp. 114--117, Mexico (1999)
7. Ayala, J. L., Lomeña, A. G., López-Vallejo, M., Fernández, A.: Design of a Pipelined Hardware Architecture for Real-Time Neural Network Computations, IEEE Midwest Symposium on Circuits and Systems, USA (2002)
8. Soares, A.M., Pinto, J.O.P., Bose, B.K., Leite, L.C., da Silva, L.E.B., Romero, M.E.: Field Programmable Gate Array (FPGA) Based Neural Network Implementation of Stator Flux Oriented Vector Control of Induction Motor Drive, IEEE International Conference on Industrial Technology (2006)
9. Morgado Dias, F., Antunes, A., Mota, A.: Artificial Neural Networks: a Review of Commercial Hardware, Engineering Applications of Artificial Intelligence, IFAC, vol. 17/8, pp. 945--952 (2004)
10. Morgado Dias, F., Mota, A.: Direct Inverse Control of a Kiln, 4th Portuguese Conference on Automatic Control (2000)

Uncertainty Sampling-Based Active Selection of Datasetoids for Meta-learning

Ricardo B.C. Prudêncio¹, Carlos Soares², and Teresa B. Ludermir¹

¹ Center of Informatics, Federal University of Pernambuco,
Cidade Universitária, CEP 50732-970, Recife (PE), Brazil
`{rbcp,tbl}@cin.ufpe.br`

² LIAAD-INESC Porto L.A., Faculdade de Economia, Universidade do Porto
Rua de Ceuta 118-6, 4050-190, Porto, Portugal
`csoares@fep.up.pt`

Abstract. Several meta-learning approaches have been developed for the problem of algorithm selection. In this context, it is of central importance to collect a sufficient number of datasets to be used as meta-examples in order to provide reliable results. Recently, some proposals to generate datasets have addressed this issue with successful results. These proposals include datasetoids, which is a simple manipulation method to obtain new datasets from existing ones. However, the increase in the number of datasets raises another issue: in order to generate meta-examples for training, it is necessary to estimate the performance of the algorithms on the datasets. This typically requires running all candidate algorithms on all datasets, which is computationally very expensive. One approach to address this problem is the use of an active learning approach to meta-learning, termed active meta-learning. In this paper we investigate the combined use of an active meta-learning approach based on an uncertainty score and datasetoids. Based on our results, we conclude that the accuracy of our method is very good results with as little as 10% to 20% of the meta-examples labeled.

1 Introduction

The goal of data analysts is to select the best algorithm for every problem, from the large number of algorithms that are available nowadays. *Meta-learning* can be used to assist the data analyst by predicting which is the best algorithm on a problem [1,8]. The problem is addressed as a supervised machine learning task. A learning algorithm is used to model the relation between the characteristics of learning problems (e.g., application domain, number of examples, proportion of symbolic attributes) and the relative performance of a set of algorithms.

An important issue in the development of meta-learning systems for algorithm recommendation is the computational cost of generating the meta-data [1]. This implies running the candidate algorithms on all the training datasets. As the number of datasets used in most studies is limited (typically less than 100), this aspect has not been a very serious problem so far and, thus, has not

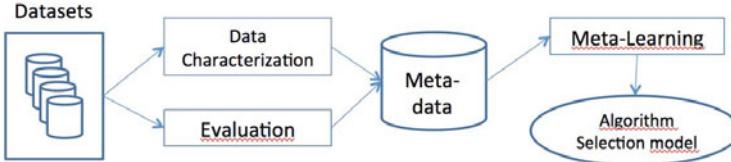


Fig. 1. The meta-learning process for algorithm selection (adapted from [1])

received much attention. One exception is the work on active meta-learning [7]. It shows that active learning can reduce the computational cost of obtaining meta-learning models for algorithm selection. In this paper, we further investigate a recent combination [6] of an uncertainty sampling-based active meta-learning approach with a dataset generation method, datasetoids [9]. This combination addresses two of the major issues in meta-learning: obtaining sufficient datasets for reliable meta-learning and reducing the computational cost of collecting meta-data [1]. It is tested on an algorithm selection task. However, despite the usefulness of datasetoids for meta-learning, they are known to be somewhat noisy [9]. Our goal in this work is, thus, to understand the effect of the noise in datasetoids on the performance of active-metalearning method.

We start by describing background information on meta-learning, including datasetoids (Section 2) and active learning (Section 3). Next, we summarize the experimental setup used to evaluate the approach empirically (Section 4). We close the paper with conclusions and some ideas for future work (Section 5).

2 Meta-learning for Algorithm Selection

The meta-learning approach to algorithm selection is summarized in Figure 1. A database is created with meta-data descriptions of a set of datasets. These meta-data contain estimates of the performance of a set of candidate algorithms on those datasets as well as some meta-features describing their characteristics (e.g., number of examples in the dataset, entropy of the class attribute, mean correlation between the numerical attributes). A machine learning algorithm (the so-called *meta-learner*) is applied to this database to induce a model that relates the value of the meta-features to the performance of the candidate algorithms (e.g., the best algorithm). For more information on meta-learning for algorithm recommendation, the reader is referred to [1,8] and references therein.

An important issue in meta-learning is related to the availability of a number of datasets that is sufficient to enable reliable (meta-)induction. The UCI Repository is the most common source of examples for meta-learning, however it contains slightly over 100 classification datasets. Given that each dataset represents one meta-example, most meta-learning research is based on approximately 100 meta-examples. This is a small number to ensure that highly reliable models are obtained, particularly in such a complex application such as meta-learning.

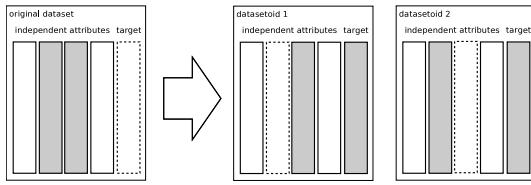


Fig. 2. Illustration of the generation of two classification datasetoids from a dataset with two symbolic attributes (reproduced from [9])

This problem is receiving an increasing amount of attention recently. Two common approaches are the generation of synthetic datasets and the manipulation of existing ones [3,5].

In this work we use *datasetoids*, a very simple data manipulation approach to generate new datasets which was recently proposed [9]. A datasetoid is generated from a given dataset by switching the target attribute with an independent attribute (Figure 2). Thus, the target attribute of the dataset becomes an independent attribute in the datasetoid and the independent attribute selected in the dataset becomes the target attribute in the datasetoid. To generate datasetoids for classification, the process is repeated for every symbolic attribute of the dataset, thus creating as many datasetoids as there are symbolic attributes in the dataset. Experiments on the problem of deciding whether to prune a decision tree using training meta-data containing datasetoids obtained significant improvements when compared to the case where only datasets were used [9].

3 Active Learning and Meta-Learning

Active learning is a paradigm of Machine Learning in which the learning algorithm has some control over the examples on which it trains [2]. It has been used in many learning tasks to reduce the number of training examples, while maintaining (or in many cases improving) the performance of the learning algorithms (e.g., [4]). Active learning is ideal for learning domains in which the acquisition of labeled examples is a costly process.

The cost of acquiring labels for meta-learning is computationally expensive, as it is necessary to execute the candidate algorithms on the datasets used for training. This makes meta-learning a good candidate problem for active learning techniques. In [7], the authors proposed the use of active learning to improve the generation of meta-examples. This proposal, termed as *Active Meta-learning*, is illustrated in Figure 3.

The method starts with a small set of one or more labelled examples and a large set of unlabeled ones. An active learning module receives these two sets as input and selects, from the latter, the next example to be labeled. The selection of unlabeled meta-examples is performed based on a pre-defined criterion which takes into account the meta-features of the problems and the current set of labeled examples. Labeling is done by evaluating the candidate algorithms on the

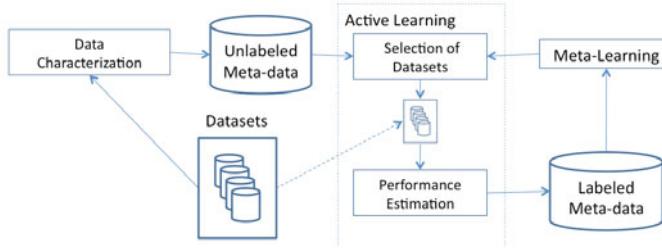


Fig. 3. Active meta-learning process

selected problems and the best algorithm on each of them becomes the label of the corresponding meta-example. The process iterates until some stopping condition occurs. The algorithm selection model is then obtained by meta-learning on the labeled examples.

The Active Meta-learning approach was empirically evaluated in [7]. The active learning algorithm used was an *uncertainty sampling method* originally proposed in [4] and based on the k-NN algorithm. This method selects unlabeled examples for which the current k-NN learner has high uncertainty in its prediction. The uncertainty of k-NN was defined in [4] as the ratio of: (1) the distance between the unlabeled example and its nearest labeled neighbor; and (2) the sum of the distances between the unlabeled example and its nearest labeled neighbor of every class. A high value of uncertainty indicates that the unlabeled example has nearest neighbors with similar distances but conflicting labeling. Hence, once an uncertain unlabeled example is labeled, it is expected that the uncertainty in its neighborhood is reduced.

In the context of meta-learning, let E be the set of labeled meta-examples. Let $\mathcal{C} = \{c_1, \dots, c_L\}$ be the domain of the class attribute C , with L possible class labels, representing the set of candidate algorithms. Each labeled meta-example e_i is represented as the pair $(\mathbf{x}_i, C(e_i))$ storing: (1) the description \mathbf{x}_i of the problem e_i , where $\mathbf{x}_i = (x_i^1, \dots, x_i^m)$ is a vector of m meta-features; and (2) the class attribute C associated to e_i , i.e., $C(e_i) = c_l$, where $c_l \in \mathcal{C}$.

Let \tilde{E} be the set of unlabeled meta-examples. Let E_l be the subset of labeled meta-examples associated to the class label c_l , i.e., $E_l = \{e_i \in E | C(e_i) = c_l\}$. Given E , the classification uncertainty of k-NN for each $\tilde{e} \in \tilde{E}$ is defined as:

$$\mathcal{S}(\tilde{e}|E) = \frac{\min_{e_i \in E} \text{dist}(\tilde{\mathbf{x}}, \mathbf{x}_i)}{\sum_{l=1}^L \min_{e_i \in E_l} \text{dist}(\tilde{\mathbf{x}}, \mathbf{x}_i)}$$

where $\tilde{\mathbf{x}}$ is the description of the unlabeled meta-example \tilde{e} and dist is the distance function adopted by the k-NN algorithm. The unlabeled meta-examples in \tilde{E} are selected according to the following probabilities:

$$p(\tilde{e}) = \frac{\mathcal{S}(\tilde{e}|E)}{\sum_{\tilde{e}_i \in \tilde{E}} \mathcal{S}(\tilde{e}_i|E)}$$

which are essentially a normalized value of the uncertainty.

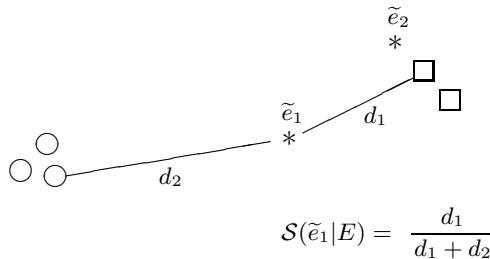


Fig. 4. Illustration of Uncertainty Sampling

Figure 4 illustrates the use of the uncertainty sampling method. Circles and squares represent two different classes of labeled examples. The stars named as \tilde{e}_1 and \tilde{e}_2 represent two unlabeled examples which are candidates to be labeled. The example \tilde{e}_2 would be less relevant since it is very close to the labeled examples of one specific class. By deploying the uncertainty sampling method, the \tilde{e}_1 would have a higher probability to be sampled since it is more equally distant from labeled examples of different classes.

4 Experiments and Results

Meta-data. We use the same experimental setup that was used in [9], which we summarize here due to lack of space. It consists of predicting, a priori, if pruning a decision tree will improve the quality of the model or not. There are three classes, **p**, **u** or **t**, meaning, respectively, the winner is the pruned tree, the unpruned tree or that they are tied.

Concerning the meta-features to characterize the datasets (and the datasetoids), we used (1) the entropy of the class and (2) the average entropy of the attributes [9]. These meta-features are expected to contain some information about the behavior of decision trees because this algorithm uses the concept of entropy. But, most importantly, although there are certainly other meta-features that could contribute to improve the meta-learning results, the use of measures that previously obtained good results enables us to focus on the main goal of this paper, which is to test the combination of active meta-learning and datasetoids.

The set of problems used to generate meta-examples are 64 UCI classification datasets. By swapping the target attribute with every nominal attribute, 983 datasetoids were obtained. Table 1 presents the class distribution, both in the meta-data obtained from the datasets as in the meta-data obtained from the datasetoids.

Meta-level performance estimation. Given that the datasetoids are generated from datasets, they cannot be treated as independent problems. In other words, datasetoids generate meta-examples which are not independent from the meta-examples representing the corresponding datasets. Therefore, to estimate meta-learning

Table 1. Class distribution (%) of the meta-data corresponding to the datasets and to the datasetoids

metadata	pruned tree wins (p)	unpruned tree wins (u)	tie (t)
datasets	36	23	41
datasetoids	37	10	53

performance, we adopted the same methodology as in [9], which is based on the following principles:

- predicting which algorithm is the best on the datasetoids is not relevant. As stated above, datasetoids are not interesting as applications *per se*. Therefore, only the original UCI datasets are used in the test set.
- to ensure independence between the training and test sets, we must guarantee that the datasetoids generated from the test datasets are not included in the training set. Thus, the meta-examples corresponding to the datasetoids obtained from test datasets are removed from the training meta-data.

By removing the datasets which are obtained from test datasets, the amount of meta-data available for learning can reduce significantly. To minimize this, we use a leave-one-out (LOO) approach as in [9], which means that, at each iteration, a single dataset is used as test and the corresponding datasetoids are removed from the training set.

The measure of meta-learning performance is the classification accuracy, i.e., the proportion of datasets for which a correct decision was made, in terms of whether it is best to prune or not, or if there is a tie.

Active Meta-learning Setting and Baseline. Every experiment starts with a single labeled meta-example which is selected randomly. Then, we allow the active meta-learning method to sample and label up to 300 training meta-examples (about 30% of the available candidate problems). Given that the methods has two random components (selection of the first labelled example and roulette wheel), we repeat each experiment 30 times to reduce the variance of the results.

We test the k-NN is run with a few different values of the number of neighbors, $k \in \{1, 3, 5, 7, 9\}$.

As a basis for comparison, we tested the Random Sampling method for selecting unlabeled problems. According to [4], despite its simplicity, the random method has the advantage of performing a uniform exploration of the example space. The Random Sampling method is repeated 50 times.

Another reference is the execution of the meta-learning method without active learning. In this case, we compare the active meta-learning method with two results, obtained by applying the meta-learning method on two different training sets: only datasets and the set of datasets and datasetoids together. The accuracies of those methods were 52% and 58%, respectively.

Results. The accuracy of the k-NN meta-learner increases as the number of labeled meta-examples increases, for both sampling methods and for all values

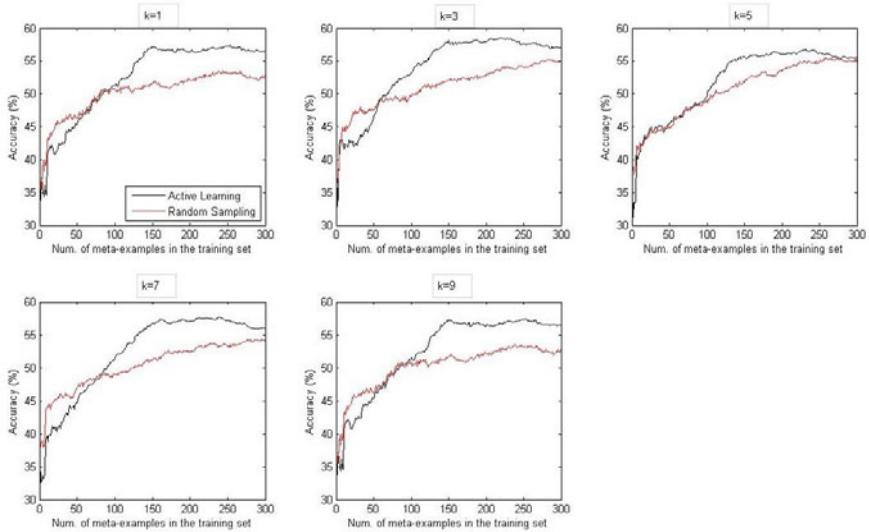


Fig. 5. Average accuracy obtained by active meta-learning with uncertainty and random sampling, for $k \in \{1, 3, 5, 7, 9\}$

of k . When comparing the two sampling methods, we observe two different behaviors. When the number of labeled meta-examples is small (up to 10%), the active meta-learning approach is unable to beat the random sampling method. On the other hand, when the number of labeled examples is approximately in the range between 10% and 20%, then the active learning approach beats the random method. These results indicate that with a very small number of meta-examples, the uncertainty score is not estimating the utility of the examples accurately. This may be due to the noise in the datasetoids [9].

When compared to the traditional meta-learning approach, the active meta-learning method is able to achieve similar accuracies that were obtained by meta-learning with all datasets and datasetoids (58%) using less than 200 labeled meta-examples. This is approximately 20% of the available problems, which represents a significant gain in computational costs.

5 Conclusion

We empirically evaluate the combination of Active Meta-Learning and datasetoids that was recently proposed to simultaneously address two of the most important issues of meta-learning for algorithm recommendation: augmenting the number of datasets to produce meta-examples and reducing the computational cost of collecting meta-data [6].

Our results confirm that it is possible to take advantage of the large number of meta-examples provided by datasetoids methods without incurring into significant extra computational costs. But the most important conclusion in this

work is that the results obtained indicate that the noise in the datasetoids may be affecting the quality of the uncertainty score when the number of labeled examples is small. One way to address this problem is to add a random element to the selection using the uncertainty score (e.g. roulette wheel or tournament selection).

Finally, we point out that our experiments were restricted to a single meta-learning problem. Although this is still common practice in the field, we will test the approach on other meta-learning problems, with different learning problems (e.g., regression), sets of base-level algorithms and meta-features.

Acknowledgements. The authors would like to thank CNPq, CAPES, FACEPE (Brazilian Agencies), FCT (Programa de Financiamento Plurianual de Unidades de I&D and project Rank! - PTDC/EIA/81178/2006) their financial support.

References

1. Brazdil, P., Giraud-Carrier, C., Soares, C., Vilalta, R.: *Metalearning: Applications to Data Mining*. Cognitive Technologies, Springer (2009)
2. Cohn, D., Atlas, L., Ladner, R.: Improving generalization with active learning. *Machine Learning* 15, 201–221 (1994)
3. Hilario, M., Kalousis, A.: Quantifying the resilience of inductive classification algorithms. In: Zighed, D., Komorowski, J., Zytkow, J. (eds.) *Proceedings of the Fourth European Conference on Principles of Data Mining and Knowledge Discovery*. pp. 106–115. Springer-Verlag (2000)
4. Lindenbaum, M., Markovitch, S., Rusakov, D.: Selective sampling for nearest neighbor classifiers. *Machine Learning* 54, 125–152 (2004)
5. Macià, N., Orriols-Puig, A., Bernadó-Mansilla, E.: Genetic-based synthetic data sets for the analysis of classifiers behavior. In: *Proceedings of 15th International Conference on Hybrid Intelligent Systems*. pp. 507–512 (2008)
6. Prudêncio, R., Soares, C., Ludermir, T.B.: Combining meta-learning and active selection of datasetoids for algorithm selection. In: Corchado, E., Kurzynski, M., Wozniak, M. (eds.) *Proc. of the 6th Int. Conf. on Hybrid Artificial Intelligent Systems (HAIS 2011)*. vol. 6679, pp. 164–171. Springer (2011)
7. Prudêncio, R.B.C., Ludermir, T.B.: Selective generation of training examples in active meta-learning. *International Journal of Hybrid Intelligent Systems* 5, 59–70 (2008)
8. Smith-Miles, K.: Cross-disciplinary perspectives on meta-learning for algorithm selection. *ACM Computing Surveys* 41(1), 1–25 (2008)
9. Soares, C.: UCI++: Improved support for algorithm selection using datasetoids. *Lecture Notes in Computer Science* 5476, 499–506 (2009)

Learning Scheme for Complex Neural Networks Using Simultaneous Perturbation

Yutaka Maeda*, Takahiro Yamada, Seiji Miyoshi, and Hiroomi Hikawa

Kansai University, Faculty of Engineering Science
Yamate-cho 3-3-35, 564-8680 Suita, Japan
maedayut@kansai-u.ac.jp
<http://www.ec.ee.kansai-u.ac.jp>

Abstract. Usually, the back-propagation learning rule is widely used for complex-valued neural networks as well. On the other hand, in this paper, learning rule for complex-valued neural networks using the simultaneous perturbation optimization method is proposed. Comparison between the back-propagation method and the proposed simultaneous perturbation learning rule is made for some test problems. Simplicity of the proposed method results in faster learning speed.

1 Introduction

Neural networks are widely studied very well. Above all, complex-valued neural networks have received strong attention, because of its capabilities; graphics transformation, signal processing and so on(1-3). Real-valued variances used as inputs, outputs and weight values in the ordinary neural networks are replaced by complex-valued variances. Then, learning scheme is of interest. Usually, the well-known back-propagation learning method is applied to complex-valued neural networks as well. The back-propagation learning method extended to complex values is considered in some references(4-6).

On the one hand, the simultaneous perturbation optimization method, which is a stochastic gradient method, was introduced by J. C. Spall (7; 8). Y. Maeda also independently proposed a learning rule using simultaneous perturbation for neural networks, and reported on the feasibility of the learning rule in neural networks and control problems (9-11). The merit of the learning rule was demonstrated in hardware implementation for many types of NNs (12-16). Analogue and pulse density types of NNs via the simultaneous perturbation learning rule were fabricated in these works.

The main advantage of the simultaneous perturbation method is its simplicity. The simultaneous perturbation can estimate the gradient of a function using only two values of the function itself. Therefore, it is relatively easy to implement as a learning rule of NNs compared with other learning rules such as the

* This work is financially supported by Grant-in-Aid for Scientific Research of the Ministry of Education, Culture, Sports, Science and Technology(MEXT) of Japan, and "Academic Frontier" Project of MEXT of Japan for Kansai University.

back-propagation learning rule. At the same time, this learning rule is easily applicable to recurrent types of NNs, since only the final error values are required to estimate the gradient of the error function with respect to the weights.

The idea of the simultaneous perturbation is applicable to complex-valued neural networks. Unlike the back-propagation method, complicated error propagation for real part and imaginary part is carried out very easily. Thus implementation is very clear.

2 Complex-Valued Neural Networks and the Simultaneous Perturbation Method

We consider neural networks with complex-valued inputs, outputs and weights. Weighted input for a certain neuron is described as follows;

$$z_l = \sum_{i=1}^n w_{li} x_i - \theta_l \quad (1)$$

Where, z_l denotes weighted input for the ℓ -th neuron. w_{li} represents a weight value from the i -th neuron in the previous layer to the ℓ -th neuron in the layer. x_i is an output of the i -th neuron in the previous layer. θ_l means a threshold of the neuron.

Output-input characteristic of the neuron is the following sigmoid function.

$$f(z) = \frac{1 - \exp(-z)}{1 + \exp(-z)} \quad (2)$$

2.1 Complex Back-Propagation Learning Rule

The complex back-propagation learning rule for complex-valued neural networks is proposed(4–6). The learning method is direct extension of the ordinary back-propagation learning rule. Using this method, learning of complex-valued neural networks became possible.

Like the ordinary real-valued back-propagation, the complex back-propagation is described as follows;

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \Delta \mathbf{w} \quad (3)$$

$$\Delta w_t^i = \alpha \frac{\partial J}{\partial \text{Re}[w_t^i]} + i\alpha \frac{\partial J}{\partial \text{Im}[w_t^i]} \quad (4)$$

Where \mathbf{w} denotes weight vector with complex-valued weight components w^i . $\Delta \mathbf{w}$ is modifying quantity. Δw_t^i is the i -th component of $\Delta \mathbf{w}$. Subscript t denotes iteration. J represents an error or evaluation function defined as follow;

$$J = \sum_{i=1}^n J_i = \sum_{i=1}^n \frac{1}{2} \{ (\text{Re}[y_{di} - y_i])^2 + (\text{Im}[y_{di} - y_i])^2 \} \quad (5)$$

Where n , y_i and y_{di} are number of output of the neural network, i -th output and its corresponding desired one, respectively.

2.2 Simultaneous Perturbation Learning Rule for Complex-Valued Neural Networks

Unlike the complex back-propagation learning rule, the learning rule using the simultaneous perturbation does not carried out so-called error propagation. Two values of error function or evaluation function are used to estimate gradient of the function at the present position. The ordinary simultaneous perturbation learning rule for real-valued neural networks is described as follows;

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \Delta \mathbf{w}_t \quad (6)$$

$$\Delta \mathbf{w}_t^i = \frac{J(\mathbf{w}_t + c \mathbf{s}_t) - J(\mathbf{w}_t)}{c s_t^i} \quad (i = 1, \dots, n) \quad (7)$$

Where, α is a positive gain coefficient, c is a magnitude of perturbation. \mathbf{s}_t and s_t^i are a sign vector and its i -th element whose value, which is determined randomly, is +1 or -1. $\Delta \mathbf{w}_t$ is an estimator of the gradient of the function. $\Delta \mathbf{w}_t^i$ represents the i -th element of $\Delta \mathbf{w}_t$.

In this method, the perturbation has important role. If the perturbations are same for all parameters, the simultaneous perturbation method cannot estimate the gradient properly. The perturbation should obey certain conditions. The conditions are described as follows;

$$\begin{aligned} \mathbb{E}(\mathbf{s}_t) &= \mathbf{0} \\ \mathbb{E}(s_{t1}^i s_{t2}^j) &= \delta_{ij} \delta_{t1t2} \end{aligned} \quad (8)$$

Where, \mathbb{E} denotes the expectation. δ is the Kronecker's delta. Instead of the $+c$ or $-c$ perturbation, we can also use uniform distribution except neighbor of zero. For hardware implementation, the above mentioned Bernoulli distribution is beneficial, since the perturbation mechanism is very simple.

As is shown in Eq.(7), the algorithm requires only two values of the target function despite of dimension of the function. That is, even if the dimension n of the target function is so large, two value of the function gives the modifying quantities for all parameter unlike the conventional finite difference.

This method does not directly require derivative of the objective function. The method estimates the gradient using a kind of finite difference technique. Note that the simultaneous perturbation optimization method requires only values of an objective function. The simplicity yields many benefits from many points of view.

Now, we extend the real-valued simultaneous perturbation learning rule to complex one. All variables are complex-valued. Then, we propose the following algorithm.

$$\Delta \mathbf{w}_t^i = \frac{J(\mathbf{w}_t + c \mathbf{s}_t) - J(\mathbf{w}_t)}{c \operatorname{Re}[s_t^i]} + \frac{J(\mathbf{w}_t + c \mathbf{s}_t) - J(\mathbf{w}_t)}{c \operatorname{Im}[s_t^i]} i(i = 1, \dots, n) \quad (9)$$

First, value of an error function defined in Eq.(5) is evaluated without perturbation. Perturbations are added to all weights. The perturbations are, of course, complex values. Therefore, different perturbations are superimposed to real part and imaginary part of the weights independently. Then, the error is calculated with the perturbations. Similar to the ordinary simultaneous perturbation, only these two values of the error function are used to produce modifying quantity. Difference of values of the error function is divided by the corresponding perturbation to estimate a partial derivative of the function. The flow of the learning procedure is given in Fig.1.

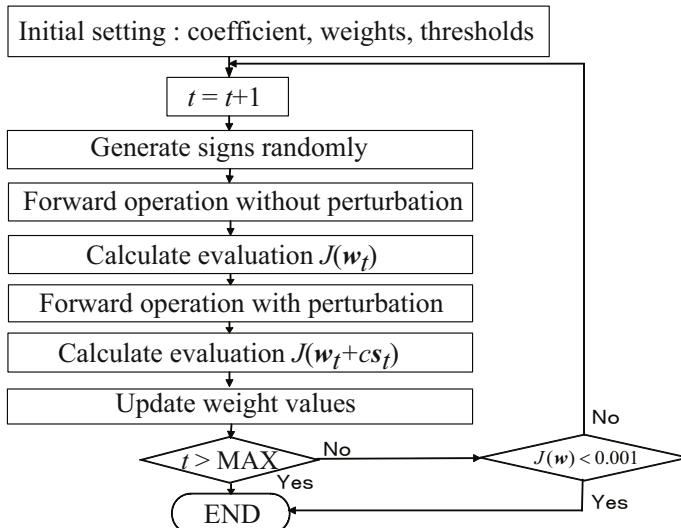


Fig. 1. Procedure of the proposed learning scheme

3 Comparison between Simultaneous Perturbation and Back-Propagation

We consider performance of the proposed simultaneous perturbation learning method for complex-valued neural networks, comparing with the usual complex back-propagation learning method. We use the exclusive OR problem, simple affine transformation of scaling and rotation.

We evaluate average converge rate and average number of learning for converge. Initial weights and thresholds are generated by uniform random number with range of $[-1 + 1]$. If the error is getting smaller than 0.001 within a certain maximum number of modifications, we find the trial successive. We make 1000 times trials and the average of the trials are discussed. Matlab on Windows XP using Core i7 CPU with 3.07GHz is used. The perturbation and learning coefficient were determined empirically through preliminary experiments.

3.1 Exclusive OR Problem

1-2-1 complex-valued neural network is used. Table 1 is the result for the problem.

Table 1. Result for the EOR problem.

Average for 1000 times trials	The proposed SP learning	Complex BP
Perturbation c	0.5	—
Learning coefficient α	0.1	1.5
Converge rate [%]	99.8	96.9
Average learning number [iteration]	2368	3815
CPU time [sec.]	31.34	37.34

The proposed method and the usual back-propagation method have equivalent learning capability for the problem. However, for actual calculation time, the proposed method is better.

3.2 Scaling Problem

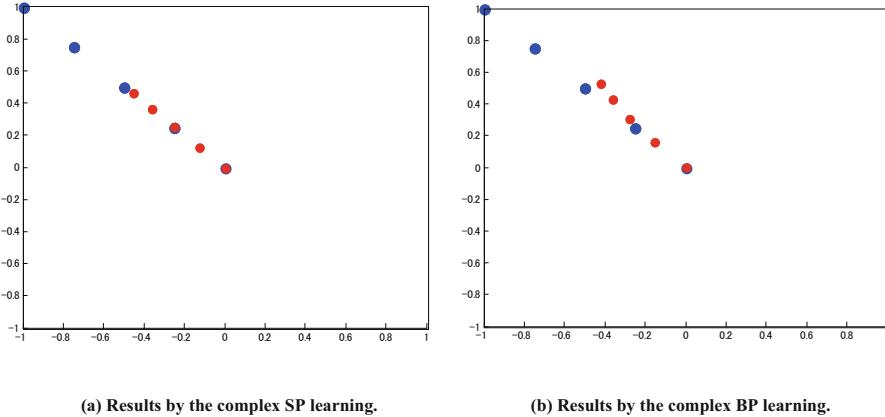
Now, 1/2 scaling is considered. Maximum learning number is 50,000. Input data is [-1-i; -0.75-0.75i; -0.5-0.5i; -0.25-0.25i; 0+0i; 0.25+0.25i; 0.5+0.5i; 0.75+0.75i; 1+i]. These nine points in complex plane are mapped to the following points; [-0.5-0.5i;-0.38-0.38i;-0.25-0.25i;-0.13-0.13i;0+0i;0.13+0.13i;0.25+0.25i;0.38+0.38i; 0.5+0.5i]. That is, 1/2 scaling is realized. 1-3-1 complex-valued neural network is used for the problem. Table 2 shows the result.

Table 2. Result for the scaling problem.

Average for 1000 times trials	The proposed SP learning	Complex BP
Perturbation c	0.00001	—
Learning coefficient α	0.09	0.8
Converge rate [%]	90.7	75.5
Average learning number [iteration]	11700	9630
CPU time [sec.]	74.12	92.43

For the converge rate, the proposed simultaneous perturbation learning method gives better result than the ordinary back-propagation learning. The complex back-propagation learning is better than the complex simultaneous perturbation learning on learning number. However, for actual CPU time, the proposed method is better than the usual back-propagation method.

Fig.2 shows generalization capability of the learnt complex-valued neural network. For untrained points, we measured accuracy of the transformation. Blue points show inputs for the network. Red points are transformed result.

**Fig. 2.** Result for the scaling

Transformation by the proposed learning is superior to the back-propagation learning.

3.3 Rotation Problem

Next, we considered +45 degree rotation around original point. We prepared the following nine points; $[-0.5-0.5i; -0.5+0i; 0.5-0.5i; -0.5+0i; 0+0i; 0.5+0i; -0.5+0.5i; 0+0.5i; 0.5+0.5i]$. Moved points are $[0-0.7i; -0.35+0.35i; 0.7-0i; 0.35+0.35i; 0+0i; 0.35+0.35i; -0.7+0i; -0.35+0.35i; 0+0.7i]$. 1-8-1 complex-valued neural network is used. Table 3 gives the result for the rotation problem. Maximum learning number is 100,000.

Table 3. Result for the rotation problem

Average for 1000 times trials	The proposed SP learning	Complex BP
Perturbation c	0.00001	—
Learning coefficient α	0.05	0.5
Converge rate [%]	62.8	71.6
Average learning number [iteration]	33200	26500
CPU time [sec.]	269.73	308.23

The usual back-propagation method is better than the proposed method on converge rate and required learning number. However, for CPU time, the simultaneous perturbation method is better than the back-propagation.

Fig.3 shows generalization capability of the learnt complex-valued neural network. For untrained points, we measured accuracy of the transformation. Blue points show inputs for the network. Red points are transformed result. We can see that the proper transformation is carried out.

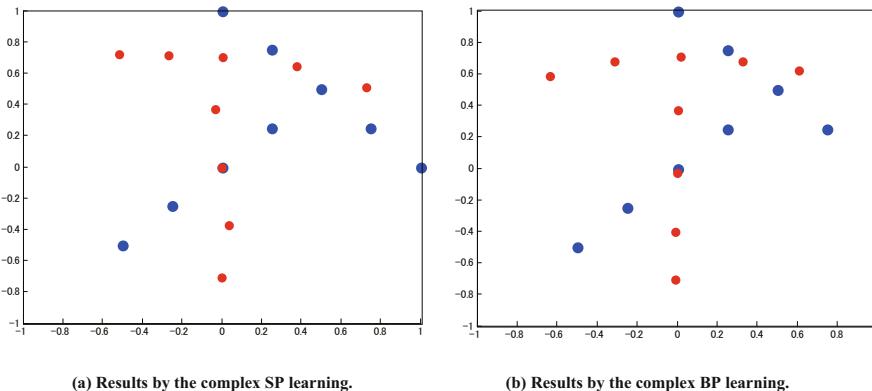


Fig. 3. Result for the rotation

4 Conclusion

In this paper, we propose a learning scheme for complex-valued neural networks using the simultaneous perturbation optimization method. Details of the learning scheme were described.

Using some examples of the exclusive OR, scaling and rotation problems, we compared performances of the proposed method and the usual complex back-propagation learning method. As a result, we could see that the proposed method has enough capability for complex-valued neural networks. Moreover, this learning scheme does not directly use the gradient of the error function. Therefore, it is widely applicable to many complex-valued recurrent neural networks.

References

1. Hirose, A.: Dynamics of Fully Complex-valued Neural Networks. *Electronics Letters* 28, 1492–1494 (1992)
2. Jankowski, S., Lozowski, A., Zurada, J.M.: Complex-valued Multistate Neural Associative Memory. *IEEE Trans. Neural Networks* 7, 1491–1496 (1996)
3. Nitta, T. (ed.): Complex-valued Neural Networks. *Information Science Reference*, Hershey (2009)
4. Nitta, T.: An Extension of the Back-propagation Algorithm to Complex Numbers. *Neural Networks* 10, 1391–1415 (1997)
5. Hirose, A.: Continuous Complex-valued Back-propagation Learning. *Electronics Letters* 28, 1854–1855 (1992)
6. Georgiou, G.M., Koutsougeras, C.: Complex Domain Backpropagation. *IEEE Trans. Circuits and Systems II: Analog and Digital Signal Processing* 39, 330–334 (1992)
7. Spall, J.C.: A Stochastic Approximation Technique for Generating Maximum Likelihood Parameter Estimates. In: Proc. of the 1987 American Control Conference, pp. 1161–1167 (1987)

8. Spall, J.C.: Multivariable Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation. *IEEE Trans. Automatic Control*, AC 37, 332–341 (1992)
9. Maeda, Y., Kanata, Y.: Learning Rules for Recurrent Neural Networks Using Perturbation and Their Application to Neuro-control. *Trans. of the Institute of Electrical Engineers of Japan* 113-C, 402–408 (1993) (in Japanese)
10. Maeda, Y., Kanata, Y.: A Learning Rule of Neural Networks for Neuro-controller. In: *Proc. of the 1995 World Congress of Neural Networks*, vol. 2, p. II-402-II-405 (1995)
11. Maeda, Y., de Figueiredo, R.J.P.: Learning Rules for Neuro-controller Via Simultaneous Perturbation. *IEEE Trans. Neural Networks* 8, 1119–1130 (1997)
12. Maeda, Y., Hirano, H., Kanata, Y.: A Learning Rule of Neural Networks Via Simultaneous Perturbation and Its Hardware Implementation. *Neural Networks* 8, 251–259 (1995)
13. Cauwenberghs, G.: An Analog VLSI Recurrent Neural Network Learning a Continuous-time Trajectory. *IEEE Trans. Neural Networks* 7, 346–361 (1996)
14. Maeda, Y., Nakazawa, A., Kanata, Y.: Hardware Implementation of a Pulse Density Neural Network Using Simultaneous Perturbation Learning Rule. *Analog Integrated Circuits and Signal Processing* 18, 1–10 (1999)
15. Maeda, Y., Tada, T.: FPGA Implementation of a Pulse Density Neural Network with Learning Ability Using Simultaneous Perturbation. *IEEE Trans. Neural Networks* 14, 688–695 (2003)
16. Maeda, Y., Wakamura, M.: Simultaneous Perturbation Learning Rule for Recurrent Neural Networks and Its FPGA Implementation. *IEEE Trans. Neural Networks* 16, 1664–1672 (2005)

Author Index

- Abe, Shigeo II-143
Aguiar, Luis II-446
Ahn, Kyoung Kwan II-229
Ahumada, Hernán C. I-117
Aiolli, Fabio I-142
Airola, Antti I-134
Ajanki, Antti II-357
Akaho, Shotaro II-277
Aksanova, Tatiana II-17
Alexandre, Luís A. I-331
Alippi, Cesare II-173
Arvanitopoulos, Nikolaos II-79
Asai, Yoshiyuki I-258
Ayad, Omar II-165
- Baptista, Dario II-446
Barbero, Álvaro II-135
Basso, Curzio I-379
Benabid, Alim-Louis II-17
Bennani, Younès II-87
Benuskova, Lubica II-341
Bicho, Estela II-325
Billaudel, Patrice II-165
Birlutiu, Adriana II-159
Blachnik, Marcin II-71
Bogdan, Martin II-40, II-56
Bohte, Sander M. I-60
Boniface, Yann I-93
Boracchi, Giacomo II-173
Bouchain, David II-317
Bouzas, Dimitrios II-79
Brænne, Ingrid II-111
Buessler, Jean-Luc II-95
- Carrillo, Snaider I-77
Cawley, Seamus I-77
Cerezuela-Escudero, Elena II-389
Chenot, Jean-Hugues II-87
Cho, KyungHyun I-10
Chorowski, Jan II-309
Cireşan, Dan I-52
Cook, Matthew I-101
Cortez, Célia II-103
Cox, Gregory E. I-307
- Csató, Lehel II-221
Cuadrado, Abel A. II-285
- Dähne, Sven I-36
Dai, Andrew M. I-241
Danóczy, Márton I-26
Da San Martino, Giovanni I-142
Daum, Paméla II-95
Dávila, Saylisse II-245
Deng, Houtao II-293
De Vos, Maarten I-233
Díaz, Ignacio II-285
Diez, Alberto B. II-285
Dinh, Quang Truong II-229
Doan, Ngoc Chi Nam II-229
Domínguez, Manuel II-285
Domínguez-Morales, Manuel II-389
Dorronsoro, José R. II-135
Dozono, Hiroshi II-197
- Eliseyev, Andrey II-17
El-Laithy, Karim II-40, II-56
Erdmann, Jeanette II-111
Erlhagen, Wolfram II-325
- Faigl, Jan I-85
Fazli, Siamac I-26
Ferreira, Flora II-325
Frick, Josef II-32
- Girau, Bernard I-93
Gisbrecht, Andrej I-150
Gnecco, Giorgio I-126
Granitto, Pablo M. I-117
Grinblat, Guillermo L. I-117
Groot, Perry II-159
Gross, Horst-Michael II-181
Gutmann, Michael U. I-323, II-213
- Hammer, Barbara I-150
Hara, Kazuyuki I-363
Harkin, Jim I-77
Hartikainen, Jouni I-193
Hauberg, Søren I-347
Hauser, Florian II-317

- Heess, Nicolas II-9
 Heskes, Tom II-1, II-159
 Hikawa, Hiroomi II-462
 Hino, Hideitsu II-301
 Hinton, Geoffrey E. I-44
 Hirose, Saori I-283
 Hochleitner, Christina II-373
 Höhne, Johannes I-36
 Hollmén, Jaakko II-405
 Hourdakis, Emmanouil II-48
 Hultén, Annika I-275
 Hunyadi, Borbala I-233
 Huopaniemi, Ilkka I-209
 Huttunen, Heikki II-189
 Hyvärinen, Aapo I-323, II-205, II-213
- Ilin, Alexander I-10
 Ito, Shinsuke II-197
- Jääskeläinen, Iiro P. II-189
 Jakab, Hunor II-221
 Jaziri, Rakia II-87
 Jimenez, Gabriel II-389
 Jimenez-Fernandez, Angel II-389
 Jones, Michael N. I-307
 Jug, Florian I-101
- Kachergis, George I-307
 Kamimura, Ryotaro I-109
 Kammer, Marc II-349
 Karandashev, Iakov M. II-25
 Kaski, Samuel I-209, II-357
 Kauppi, Jukka-Pekka II-189
 Kawanabe, Motoaki II-397
 Kivinen, Jyri J. I-1
 Klement, Sascha I-315
 Kmet, Tibor II-261
 Knott, Alistair II-341
 Kollias, Stefanos D. I-291
 Koprinkova-Hristova, Petia I-69
 Kordos, Miroslaw II-71
 Korkala, Heikki II-189
 Kortkamp, Marco II-437
 Krautz, Christoph I-101
 Krizhevsky, Alex I-44
 Kryzhanovskiy, Vladimir II-119
 Kryzhanovsky, Boris V. II-25
 Kuriyama, Naoko I-283
 Kůrková, Věra I-126
- Kurokawa, Hiroaki I-217
 Kusherbaeva, Victoria I-185
- Laaksonen, Jorma II-373
 Lagus, Krista I-275
 Laparra, Valero II-213
 Lazar, Andreea II-127
 Lebbah, Mustapha II-87
 Lefort, Mathieu I-93
 Lehtonen, Minna I-275
 Le Roux, Nicolas II-9
 Linares-Barranco, Alejandro II-389
 Liu, Qingshan II-253
 López, Jorge II-135
 Ludermir, Teresa B. II-454
- Maeda, Yutaka II-462
 Magimai-Doss, Mathew I-299
 Malo, Jesús II-213
 Mamlouk, Amir Madany II-111
 Maris, Eric II-1
 Martinetz, Thomas I-315
 Masci, Jonathan I-52
 Matsuoka, Kiyotoshi II-269
 May, Patrick J.C. II-421
 McDaid, Liam I-77
 Meier, Ueli I-52
 Meinecke, Frank C. II-397
 Minin, Alexey I-185
 Mitsuishi, Takashi II-237
 Miyoshi, Seiji I-363, II-462
 Morgado Dias, Fernando II-446
 Morgan, Fearghal I-77
 Moya-Sánchez, E. Ulises II-429
 Müller, Klaus Robert I-26
 Murata, Noboru II-301
 Myllymäki, Petri I-355
 Mysling, Peter I-347
- Nagai, Yukie II-349
 Nakada, Kazuki II-269
 Nakagawa, Masanori I-283
 Nakakuni, Masanori II-197
 Neumann, Klaus I-339
 Nikolaev, Nikolay I-176
 Nishide, Shun I-167
- Oakes, Michael II-64
 Ogata, Tetsuya I-167
 Oja, Erkki I-250

- Okamoto, Hiroshi I-371
 Okuno, Hiroshi G. I-167
 Orešić, Matej I-209
 Osana, Yuko I-266
 Oubbati, Mohamed II-32
- Pahikkala, Tapio I-134
 Palm, Guenther I-69
 Palm, Günther II-32, II-317
 Pande, Sandeep I-77
 Paz-Vicente, Rafael II-389
 Pedersen, Kim Steenstrup I-347
 Pinto, Thiago M. II-103
 Pipa, Gordon II-127
 Přeučil, Libor I-85
 Prudêncio, Ricardo B.C. II-454
 Pulkkinen, Teemu I-355
 Puuronen, Jouni II-205
- Raftopoulos, Konstantinos A. I-291
 Raiko, Tapani I-10
 Rasipuram, Ramya I-299
 Reichert, David P. I-18
 Reinhart, René Felix I-159
 Reis, Leonardo II-446
 Riihimäki, Jaakko I-193
 Romero, Enrique I-225
 Roos, Teemu I-355
 Roveri, Manuel II-173
 Runger, George II-245, II-293
 Ruokolainen, Teemu II-373
- Salakoski, Tapio I-134
 Salmelin, Riitta I-275
 Samek, Wojciech II-397
 Sams, Mikko II-189
 Sanguineti, Marcello I-126
 Santoro, Matteo I-379
 Särkkä, Simo I-193, I-201, II-151
 Sato, Yasuomi D. II-269
 Sayed-Mouchaweh, Moamar II-165
 Schack, Thomas II-349
 Schaffernicht, Erik II-181
 Schelldorfer, Jürg I-26
 Schleif, Frank-Michael I-150
 Schmidhuber, Jürgen I-52
 Schreuder, Martijn I-36
 Series, Peggy I-18
- Shidama, Yasunari II-237
 Signoretto, Marco I-233
 Sirola, Miki II-381
 Sirvio, Konsta II-405
 Smirnov, Evgeni I-176
 Soares, Carlos II-454
 Sovilj, Dušan II-413
 Sperduti, Alessandro I-142
 Steil, Jochen Jakob I-159, I-339
 Storkey, Amos J. I-18, I-241
 Suetani, Hiromichi II-277
 Suvitaival, Tommi I-209
 Suykens, Johan A.K. I-233
- Takac, Martin II-341
 Takahashi, Toru I-167
 Talonen, Jaakko II-381
 Tangermann, Michael I-36
 Tefas, Anastasios II-79
 Terai, Asuka I-283
 Tiitinen, Hannu II-421
 Tino, Peter I-176
 Tohka, Jussi II-189
 Trahanias, Panos II-48
 Traunmüller, Rudolf II-373
 Triesch, Jochen II-127
 Tripathi, Nandita II-64
 Tscherpanow, Marko II-349
 Tuv, Eugene II-245, II-293
- Urban, Jean-Philippe II-95
- van Gerven, Marcel A.J. II-1
 Van Huffel, Sabine I-233
 Van Paesschen, Wim I-233
 Vázquez-Santacruz, Eduardo II-429
 Verri, Alessandro I-379
 Villa, Alessandro E.P. I-258
 Villa, Silvia I-379
 Virpioja, Sami I-275
 von Bünau, Paul II-397
- Wachsmuth, Sven II-437
 Wang, Jun II-253
 Wang, Sida D. I-44
 Weber, Cornelius II-333, II-365
 Wedemann, Roseli S. II-103
 Wermter, Stefan II-64, II-333, II-365

- Wieczorek, Tadeusz II-71
Williams, Christopher K.I. I-1
Winn, John II-9
- Yamada, Takahiro II-462
Yan, Wenjie II-365
Yang, Zhirong I-250
Yonekawa, Masato I-217
Yoon, Jong Il II-229
- Yoshida, Akio I-266
Yuan, Zhijian I-250
- Zhang, He I-250, II-373
Zhang, Yang I-167
Zhong, Junpei II-333
Zimmermann, Hans-Georg I-185
Zurada, Jacek M. II-309