

José Luis Balcázar  
Francesco Bonchi  
Aristides Gionis  
Michèle Sebag (Eds.)

LNAI 6323

# Machine Learning and Knowledge Discovery in Databases

European Conference, ECML PKDD 2010  
Barcelona, Spain, September 2010  
Proceedings, Part III

3 Part III



Springer

Lecture Notes in Artificial Intelligence 6323

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

José Luis Balcázar  
Francesco Bonchi Aristides Gionis  
Michèle Sebag (Eds.)

# Machine Learning and Knowledge Discovery in Databases

European Conference, ECML PKDD 2010  
Barcelona, Spain, September 20-24, 2010  
Proceedings, Part III

## Series Editors

Randy Goebel, University of Alberta, Edmonton, Canada  
Jörg Siekmann, University of Saarland, Saarbrücken, Germany  
Wolfgang Wahlster, DFKI and University of Saarland, Saarbrücken, Germany

## Volume Editors

José Luis Balcázar  
Universidad de Cantabria  
Departamento de Matemáticas, Estadística y Computación  
Avenida de los Castros, s/n, 39071 Santander, Spain  
E-mail: joseluis.balcazar@unican.es

Francesco Bonchi  
Aristides Gionis  
Yahoo! Research Barcelona  
Avinguda Diagonal 177, 08018 Barcelona, Spain  
E-mail: {bonchi, gionis}@yahoo-inc.corp

Michèle Sebag  
TAO, CNRS-INRIA-LRI, Université Paris-Sud  
91405, Orsay, France  
E-mail: sebag@lri.fr

Cover illustration: Decoration detail at the Park Güell, designed by Antoni Gaudí, and one of the landmarks of modernist art in Barcelona. Licence Creative Commons, Jon Robson.

Library of Congress Control Number: 2010934301

CR Subject Classification (1998): I.2, H.3, H.4, H.2.8, J.1, H.5

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743  
ISBN-10 3-642-15938-9 Springer Berlin Heidelberg New York  
ISBN-13 978-3-642-15938-1 Springer Berlin Heidelberg New York

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

springer.com

© Springer-Verlag Berlin Heidelberg 2010  
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India  
Printed on acid-free paper 06/3180

# Preface

The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ECML PKDD 2010, was held in Barcelona, September 20–24, 2010, consolidating the long junction between the European Conference on Machine Learning (of which the first instance as European workshop dates back to 1986) and Principles and Practice of Knowledge Discovery in Data Bases (of which the first instance dates back to 1997). Since the two conferences were first collocated in 2001, both machine learning and data mining communities have realized how each discipline benefits from the advances, and participates to defining the challenges, of the sister discipline. Accordingly, a single ECML PKDD Steering Committee gathering senior members of both communities was appointed in 2008.

In 2010, as in previous years, ECML PKDD lasted from Monday to Friday. It involved six plenary invited talks, by Christos Faloutsos, Jiawei Han, Hod Lipson, Leslie Pack Kaelbling, Tomaso Poggio, and Jürgen Schmidhuber, respectively. Monday and Friday were devoted to workshops and tutorials, organized and selected by Colin de la Higuera and Gemma Garriga. Continuing from ECML PKDD 2009, an industrial session managed by Taneli Mielikainen and Hugo Zaragoza welcomed distinguished speakers from the ML and DM industry: Rakesh Agrawal, Mayank Bawa, Ignasi Belda, Michael Berthold, José Luis Flórez, Thore Graepel, and Alejandro Jaimés. The conference also featured a discovery challenge, organized by András Benczúr, Carlos Castillo, Zoltán Gyöngyi, and Julien Masanès.

From Tuesday to Thursday, 120 papers selected among 658 submitted full papers were presented in the technical parallel sessions. The selection process was handled by 28 area chairs and the 282 members of the Program Committee; additional 298 reviewers were recruited. While the selection process was made particularly intense due to the record number of submissions, we heartily thank all area chairs, members of the Program Committee, and additional reviewers for their commitment and hard work during the short reviewing period. The conference also featured a demo track, managed by Ulf Brefeld and Xavier Carreras; 12 demos out of 24 submitted ones were selected, attesting to the high impact technologies based on the ML and DM body of research.

Following an earlier tradition, seven ML and seven DM papers were distinguished by the program chairs on the basis of their exceptional scientific quality and high impact on the field, and they were directly published in the Machine Learning Journal and the Data Mining and Knowledge Discovery Journal, respectively. Among these papers, some were selected by the Best Paper Chair Hiroshi Motoda, and received the Best Paper Awards and Best Student Paper Awards in Machine Learning and in Data Mining, sponsored by Springer.

A topic widely explored from both ML and DM perspectives was graphs, with motivations ranging from molecular chemistry to social networks. The point of matching or clustering graphs was examined in connection with tractability and domain knowledge, where the latter could be acquired through common patterns, or formulated through spectral clustering. The study of social networks focused on how they develop, overlap, propagate information (and how information propagation can be hindered). Link prediction and exploitation in static or dynamic, possibly heterogeneous, graphs, was motivated by applications in information retrieval and collaborative filtering, and in connection with random walks.

Frequent itemset approaches were hybridized with constraint programming or statistical tools to efficiently explore the search space, deal with numerical attributes, or extract locally optimal patterns. Compressed representations and measures of robustness were proposed to optimize association rules. Formal concept analysis, with applications to pharmacovigilance or Web ontologies, was considered in connection with version spaces.

Bayesian learning features new geometric interpretations of prior knowledge and efficient approaches for independence testing. Generative approaches were motivated by applications in sequential, spatio-temporal or relational domains, or multi-variate signals with high dimensionality. Ensemble learning was used to support clustering and biclustering; the post-processing of random forests was also investigated.

In statistical relational learning and structure identification, with motivating applications in bio-informatics, neuro-imagery, spatio-temporal domains, and traffic forecasting, the stress was put on new learning criteria; gradient approaches, structural constraints, and/or feature selection were used to support computationally effective algorithms.

(Multiple) kernel learning and related approaches, challenged by applications in image retrieval, robotics, or bio-informatics, revisited the learning criteria and regularization terms, the processing of the kernel matrix, and the exploration of the kernel space. Dimensionality reduction, embeddings, and distance were investigated, notably in connection with image and document retrieval.

Reinforcement learning focussed on ever more scalable and tractable approaches through smart state or policy representations, a more efficient use of the available samples, and/or Bayesian approaches.

Specific settings such as ranking, multi-task learning, semi-supervised learning, and game-theoretic approaches were investigated, with some innovative applications to astrophysics, relation extraction, and multi-agent systems. New bounds were proved within the active, multi-label, and weighted ensemble learning frameworks.

A few papers aimed at efficient algorithms or computing environments, e.g., related to linear algebra, cutting plane algorithms, or graphical processing units, were proposed (with available source code in some cases). Numerical stability was also investigated in connection with sparse learning.

Among the applications presented were review mining, software debugging/process modeling from traces, and audio mining.

To conclude this rapid tour of the scientific program, our special thanks go to the local chairs Ricard Gavaldà, Elena Torres, and Estefania Ricart, the Web and registration chair Albert Bifet, the sponsorship chair Debora Denato, and the many volunteers that eagerly contributed to make ECML PKDD 2010 a memorable event.

Our last and warmest thanks go to all invited speakers and other speakers, to all tutorial, workshop, demo, industrial, discovery, best paper, and local chairs, to the area chairs and all reviewers, to all attendees — and overall, to the authors who chose to submit their work to the ECML PKDD conference, and thus enabled us to build up this memorable scientific event.

July 2010

José L Balcázar  
Francesco Bonchi  
Aristides Gionis  
Michèle Sebag

# Organization

## Program Chairs

José L Balcázar  
Universidad de Cantabria and  
Universitat Politècnica de Catalunya, Spain  
<http://personales.unican.es/balcazarjl/>

Francesco Bonchi  
Yahoo! Research  
Barcelona, Spain  
<http://research.yahoo.com>

Aristides Gionis  
Yahoo! Research  
Barcelona, Spain  
<http://research.yahoo.com>

Michèle Sebag  
CNRS  
Université Paris Sud, Orsay Cedex, France  
<http://www.lri.fr/~sebag/>

## Local Organization Chairs

Ricard Gavaldà	Universitat Politècnica de Catalunya
Estefania Ricart	Barcelona Media
Elena Torres	Barcelona Media

## Organization Team

Ulf Brefeld	Yahoo! Research
Eugenia Fuenmayor	Barcelona Media
Mia Padullés	Yahoo! Research
Natalia Pou	Barcelona Media

## Workshop and Tutorial Chairs

Gemma C. Garriga	University of Paris 6
Colin de la Higuera	University of Nantes



## Best Papers Chair

Hiroshi Motoda AFOSR/AOARD and Osaka University

## Industrial Track Chairs

Taneli Mielikainen Nokia  
Hugo Zaragoza Yahoo! Research

## Demo Chairs

Ulf Brefeld Yahoo! Research  
Xavier Carreras Universitat Politècnica de Catalunya

## Discovery Challenge Chairs

András Benczúr Hungarian Academy of Sciences  
Carlos Castillo Yahoo! Research  
Zoltán Gyöngyi Google  
Julien Masanès European Internet Archive

## Sponsorship Chair

Debora Donato Yahoo! Labs

## Web and Registration Chair

Albert Bifet University of Waikato

## Publicity Chair

Ricard Gavaldà Universitat Politècnica de Catalunya

## Steering Committee

Wray Buntine  
Walter Daelemans  
Bart Goethals  
Marko Grobelnik  
Katharina Morik  
Joost N. Kok  
Stan Matwin  
Dunja Mladenic  
John Shawe-Taylor  
Andrzej Skowron

## Area Chairs

Samy Bengio	George Karypis
Bettina Berendt	Laks V.S. Lakshmanan
Paolo Boldi	Katharina Morik
Wray Buntine	Jan Peters
Toon Calders	Kai Puolamäki
Luc de Raedt	Yucel Saygin
Carlotta Domeniconi	Bruno Scherrer
Martin Ester	Arno Siebes
Paolo Frasconi	Soeren Sonnenburg
Joao Gama	Alexander Smola
Ricard Gavaldà	Einoshin Suzuki
Joydeep Ghosh	Evimaria Terzi
Fosca Giannotti	Michalis Vazirgiannis
Tu-Bao Ho	Zhi-Hua Zhou

## Program Committee

Osman Abul	Jean-Francois Boulicaut
Gagan Agrawal	Ulf Brefeld
Erick Alphonse	Laurent Brehelin
Carlos Alzate	Bjoern Bringmann
Massih Amini	Carla Brodley
Aris Anagnostopoulos	Rui Camacho
Annalisa Appice	Stéphane Canu
Thierry Artières	Olivier Cappé
Sitaram Asur	Carlos Castillo
Jean-Yves Audibert	Jorge Castro
Maria-Florina Balcan	Ciro Cattuto
Peter Bartlett	Nicolò Cesa-Bianchi
Younes Bennani	Nitesh Chawla
Paul Bennett	Sanjay Chawla
Michele Berlingerio	David Cheung
Michael Berthold	Sylvia Chiappa
Albert Bifet	Boris Chidlovski
Hendrik Blockeel	Flavio Chierichetti
Mario Boley	Philipp Cimiano
Antoine Bordes	Alexander Clark
Gloria Bordogna	Christopher Clifton
Christian Borgelt	Antoine Cornuéjols
Karsten Borgwardt	Fabrizio Costa
Henrik Boström	Bruno Crémilleux
Marco Botta	James Cussens
Guillaume Bouchard	Alfredo Cuzzocrea

Florence d'Alché-Buc  
Claudia d'Amato  
Gautam Das  
Jeroen De Knijf  
Colin de la Higuera  
Krzysztof Dembczynski  
Ayhan Demiriz  
Francois Denis  
Christos Dimitrakakis  
Josep Domingo Ferrer  
Debora Donato  
Dejing Dou  
Gérard Dreyfus  
Kurt Driessens  
John Duchi  
Pierre Dupont  
Saso Dzeroski  
Charles Elkan  
Damien Ernst  
Floriana Esposito  
Fazel Famili  
Nicola Fanizzi  
Ad Feelders  
Alan Fern  
Daan Fierens  
Peter Flach  
George Forman  
Vojtech Franc  
Eibe Frank  
Dayne Freitag  
Elisa Fromont  
Patrick Gallinari  
Auroop Ganguly  
Fred Garcia  
Gemma Garriga  
Thomas Gärtner  
Eric Gaussier  
Floris Geerts  
Matthieu Geist  
Claudio Gentile  
Mohammad Ghavamzadeh  
Gourab Ghoshal  
Chris Giannella  
Attilio Giordana  
Mark Girolami

Shantanu Godbole  
Bart Goethals  
Sally Goldman  
Henrik Grosskreutz  
Dimitrios Gunopulos  
Amaury Habrard  
Eyke Hüllermeier  
Nikolaus Hansen  
Iris Hendrickx  
Melanie Hilario  
Alexander Hinneburg  
Kouichi Hirata  
Frank Hoepfner  
Jaakko Hollmen  
Tamas Horvath  
Andreas Hotho  
Alex Jaimes  
Szymon Jaroszewicz  
Daxin Jiang  
Felix Jungermann  
Frederic Jurie  
Alexandros Kalousis  
Panagiotis Karras  
Samuel Kaski  
Dimitar Kazakov  
Sathiya Keerthi  
Jens Keilwagen  
Roni Khardon  
Angelika Kimmig  
Ross King  
Marius Kloft  
Arno Knobbe  
Levente Kocsis  
Jukka Kohonen  
Solmaz Kolahi  
George Kollios  
Igor Kononenko  
Nick Koudas  
Stefan Kramer  
Andreas Krause  
Vipin Kumar  
Pedro Larrañaga  
Mark Last  
Longin Jan Latecki  
Silvio Lattanzi

Anne Laurent  
 Nada Lavrac  
 Alessandro Lazaric  
 Philippe Leray  
 Jure Leskovec  
 Carson Leung  
 Chih-Jen Lin  
 Jessica Lin  
 Huan Liu  
 Kun Liu  
 Alneu Lopes  
 Ramón López de Mántaras  
 Eneldo Loza Mencía  
 Claudio Lucchese  
 Elliot Ludvig  
 Dario Malchiodi  
 Donato Malerba  
 Bradley Malin  
 Giuseppe Manco  
 Shie Mannor  
 Stan Matwin  
 Michael May  
 Thorsten Meinl  
 Prem Melville  
 Rosa Meo  
 Pauli Miettinen  
 Lily Mihalkova  
 Dunja Mladenic  
 Ali Mohammad-Djafari  
 Fabian Morchen  
 Alessandro Moschitti  
 Ion Muslea  
 Mirco Nanni  
 Amedeo Napoli  
 Claire Nedellec  
 Frank Nielsen  
 Siegfried Nijssen  
 Richard Nock  
 Sebastian Nowozin  
 Alexandros Ntoulas  
 Andreas Nuernberger  
 Arlindo Oliveira  
 Balaji Padmanabhan  
 George Paliouras  
 Themis Palpanas

Apostolos Papadopoulos  
 Andrea Passerini  
 Jason Pazis  
 Mykola Pechenzkiy  
 Dmitry Pechyony  
 Dino Pedreschi  
 Jian Pei  
 Jose Peña  
 Ruggero Pensa  
 Marc Plantevit  
 Enric Plaza  
 Doina Precup  
 Ariadna Quattoni  
 Predrag Radivojac  
 Davood Rafiei  
 Chedy Raissi  
 Alain Rakotomamonjy  
 Liva Ralaivola  
 Naren Ramakrishnan  
 Jan Ramon  
 Chotirat Ratanamahatana  
 Elisa Ricci  
 Bertrand Rivet  
 Philippe Rolet  
 Marco Rosa  
 Fabrice Rossi  
 Juho Rousu  
 Céline Rouveirol  
 Cynthia Rudin  
 Salvatore Ruggieri  
 Stefan Rüping  
 Massimo Santini  
 Lars Schmidt-Thieme  
 Marc Schoenauer  
 Marc Sebban  
 Nicu Sebe  
 Giovanni Semeraro  
 Benyah Shaparenko  
 Jude Shavlik  
 Fabrizio Silvestri  
 Dan Simovici  
 Carlos Soares  
 Diego Sona  
 Alessandro Sperduti  
 Myra Spiliopoulou

Gerd Stumme  
Jiang Su  
Masashi Sugiyama  
Johan Suykens  
Domenico Talia  
Pang-Ning Tan  
Tamir Tassa  
Nikolaj Tatti  
Yee Whye Teh  
Maguelonne Teisseire  
Olivier Teytaud  
Jo-Anne Ting  
Michalis Titsias  
Hannu Toivonen  
Ryota Tomioka  
Marc Tommasi  
Hanghang Tong  
Luis Torgo  
Fabien Torre  
Marc Toussaint  
Volker Tresp  
Koji Tsuda  
Alexey Tsymbal  
Franco Turini

Antti Ukkonen  
Matthijs van Leeuwen  
Martijn van Otterlo  
Maarten van Someren  
Celine Vens  
Jean-Philippe Vert  
Ricardo Vilalta  
Christel Vrain  
Jilles Vreeken  
Christian Walder  
Louis Wehenkel  
Markus Weimer  
Dong Xin  
Dit-Yan Yeung  
Cong Yu  
Philip Yu  
Chun-Nam Yue  
Francois Yvon  
Bianca Zadrozny  
Carlo Zaniolo  
Gerson Zaverucha  
Filip Zelezny  
Albrecht Zimmermann

## Additional Reviewers

Mohammad Ali Abbasi  
Zubin Abraham  
Yong-Yeol Ahn  
Fabio Aioli  
Dima Alberg  
Salem Alelyani  
Aneeth Anand  
Sunil Aryal  
Arthur Asuncion  
Gowtham Atluri  
Martin Atzmueller  
Paolo Avesani  
Pranjal Awasthi  
Hanane Azzag  
Miriam Baglioni  
Raphael Bailly  
Jaume Baixeries  
Jorn Bakker

Georgios Balkanas  
Nicola Barbieri  
Teresa M.A. Basile  
Luca Bechetti  
Dominik Benz  
Maxime Berar  
Juliana Bernardes  
Aur lie Boisbunon  
Shyam Boriah  
Zoran Bosnic  
Robert Bossy  
Lydia Boudjeloud  
Dominique Bouthinon  
Janez Brank  
Sandra Bringay  
Fabian Buchwald  
Krisztian Buza  
Matthias B ck

José Caldas  
Gabriele Capannini  
Annalina Caputo  
Franco Alberto Cardillo  
Xavier Carreras  
Giovanni Cavallanti  
Michelangelo Ceci  
Eugenio Cesario  
Pirooz Chubak  
Anna Ciampi  
Ronan Collobert  
Carmela Comito  
Gianni Costa  
Bertrand Cuissart  
Boris Cule  
Giovanni Da San Martino  
Marco de Gemmis  
Kurt De Grave  
Gerben de Vries  
Jean Decoster  
Julien Delporte  
Christian Desrosiers  
Sanjoy Dey  
Nicola Di Mauro  
Joshua V. Dillon  
Huyen Do  
Stephan Doerfel  
Brett Drury  
Timo Duchrow  
Wouter Duivesteyn  
Alain Dutech  
Ilenia Epifani  
Ahmet Erhan Nergiz  
Rémi Eyraud  
Philippe Ezequel  
Jean Baptiste Faddoul  
Fabio Fassetti  
Bruno Feres de Souza  
Remi Flamary  
Alex Freitas  
Natalja Friesen  
Gabriel P.C. Fung  
Barbara Furletti  
Zeno Gantner  
Steven Ganzert

Huiji Gao  
Ashish Garg  
Aurelien Garviev  
Gilles Gasso  
Elisabeth Georgii  
Edouard Gilbert  
Tobias Girschick  
Miha Grcar  
Warren Greiff  
Valerio Grossi  
Nistor Grozavu  
Massimo Guarascio  
Tias Guns  
Vibhor Gupta  
Rohit Gupta  
Tushar Gupta  
Nico Görnitz  
Hirotaka Hachiya  
Steve Hanneke  
Andreas Hapfelmeier  
Daniel Hsu  
Xian-Sheng Hua  
Yi Huang  
Romain Hérault  
Leo Iaquinta  
Dino Ienco  
Elena Ikonomovska  
Stéphanie Jacquemont  
Jean-Christophe Janodet  
Frederik Janssen  
Baptiste Jeudy  
Chao Ji  
Goo Jun  
U Kang  
Anuj Karpatne  
Jaya Kawale  
Ashraf M. Kibriya  
Kee-Eung Kim  
Akisato Kimura  
Arto Klami  
Suzan Koknar-Tezel  
Xiangnan Kong  
Arne Koopman  
Mikko Korpela  
Wojciech Kotlowski

Alexis Kotsifakos  
Petra Kralj Novak  
Tetsuji Kuboyama  
Matjaz Kukar  
Sanjiv Kumar  
Shashank Kumar  
Pascale Kuntz  
Ondrej Kuzelka  
Benjamin Labbe  
Mathieu Lajoie  
Hugo Larochelle  
Agnieszka Lawrynowicz  
Gregor Leban  
Mustapha Lebbah  
John Lee  
Sau Dan Lee  
Gayle Leen  
Florian Lemmerich  
Biao Li  
Ming Li  
Rui Li  
Tiancheng Li  
Yong Li  
Yuan Li  
Wang Liang  
Ryan Lichtenwalter  
Haishan Liu  
Jun Liu  
Lei Liu  
Xu-Ying Liu  
Corrado Loglisci  
Pasquale Lops  
Chuan Lu  
Ana Luisa Duboc  
Panagis Magdalinos  
Sebastien Mahler  
Michael Mampaey  
Prakash Mandayam  
Alain-Pierre Manine  
Patrick Marty  
Jeremie Mary  
André Mas  
Elio Masciari  
Emanuel Matos  
Andreas Maunz

John McCrae  
Marvin Meeng  
Wannes Meert  
Joao Mendes-Moreira  
Aditya Menon  
Peter Mika  
Folke Mitzlaff  
Anna Monreale  
Tetsuro Morimura  
Ryoko Morioka  
Babak Mougouie  
Barzan Mozafari  
Igor Mozetic  
Cataldo Musto  
Alexandros Nanopoulos  
Fedelucio Narducci  
Maximilian Nickel  
Inna Novalija  
Benjamin Oatley  
Marcia Oliveira  
Emauele Olivetti  
Santiago Ontañón  
Francesco Orabona  
Laurent Orseau  
Riccardo Ortale  
Aomar Osmani  
Aline Paes  
Sang-Hyeun Park  
Juuso Parkkinen  
Ioannis Partalas  
Pekka Parviainen  
Krishnan Pillaipakkamnatt  
Fabio Pinelli  
Cristiano Pitangui  
Barbara Poblete  
Vid Podpecan  
Luigi Pontieri  
Philippe Preux  
Han Qin  
Troy Raeder  
Subramanian Ramanathan  
Huzefa Rangwala  
Guillaume Raschia  
Konrad Rieck  
François Rioult

Ettore Ritacco  
Mathieu Roche  
Christophe Rodrigues  
Philippe Rolet  
Andrea Romei  
Jan Rupnik  
Delia Rusu  
Ulrich Rückert  
Hiroshi Sakamoto  
Vitor Santos Costa  
Kengo Sato  
Saket Saurabh  
Francois Scharffe  
Leander Schietgat  
Jana Schmidt  
Constanze Schmitt  
Christoph Scholz  
Dan Schrider  
Madeleine Seeland  
Or Sheffet  
Noam Shental  
Xiaoxiao Shi  
Naoki Shibayama  
Nobuyuki Shimizu  
Kilho Shin  
Kaushik Sinha  
Arnaud Soulet  
Michal Sramka  
Florian Steinke  
Guillaume Stempfél  
Liwen Sun  
Umar Syed  
Gabor Szabo  
Yasuo Tabei  
Nima Taghipour  
Hana Tai  
Frédéric Tantini  
Katerina Tashkova  
Christine Task  
Alexandre Termier  
Lam Thoang Hoang

Xilan Tian  
Xinmei Tian  
Gabriele Tolomei  
Aneta Trajanov  
Roberto Trasarti  
Abhishek Tripathi  
Paolo Trunfio  
Ivor Tsang  
Theja Tulabandhula  
Boudewijn van Dongen  
Stijn Vanderlooy  
Joaquin Vanschoren  
Philippe Veber  
Sriharsha Veeramachaneni  
Sebastian Ventura  
Alessia Visconti  
Jun Wang  
Xufei Wang  
Osamu Watanabe  
Lorenz Weizsäcker  
Tomas Werner  
Jörg Wicker  
Derry Wijaya  
Daya Wimalasuriya  
Adam Woznica  
Fuxiao Xin  
Zenglin Xu  
Makoto Yamada  
Liu Yang  
Xingwei Yang  
Zhirong Yang  
Florian Yger  
Reza Bosagh Zadeh  
Reza Zafarani  
Amelia Zafra  
Farida Zehraoui  
Kai Zeng  
Bernard Zenko  
De-Chuan Zhan  
Min-Ling Zhang  
Indre Zliobaite



## Sponsors

We wish to express our gratitude to the sponsors of ECML PKDD 2010 for their essential contribution to the conference: the French National Institute for Research in Computer Science and Control (INRIA), the Pascal2 European Network of Excellence, Nokia, Yahoo! Labs, Google, KNIME, Aster data, Microsoft Research, HP, MODAP (Mobility, Data Mining, and Privacy) a Coordination Action type project funded by EU, FET OPEN, the Data Mining and Knowledge Discovery Journal, the Machine Learning Journal, LRI (Laboratoire de Recherche en Informatique, Université Paris-Sud -CNRS), ARES (Advanced Research on Information Security and Privacy) a national Spanish project, the UNESCO Chair in Data Privacy, Xerox, Universitat Politècnica de Catalunya, IDESCAT (Institut d'Estadística de Catalunya), and the Ministerio de Ciencia e Innovación (Spanish government).



# Table of Contents – Part III

## Regular Papers

Efficient Planning in Large POMDPs through Policy Graph Based Factorized Approximations . . . . .	1
<i>Joni Pajarinen, Jaakko Peltonen, Ari Hottinen, and Mikko A. Usitalo</i>	
Unsupervised Trajectory Sampling . . . . .	17
<i>Nikos Pelekis, Ioannis Kopanakis, Costas Panagiotakis, and Yannis Theodoridis</i>	
Fast Extraction of Locally Optimal Patterns Based on Consistent Pattern Function Variations . . . . .	34
<i>Frédéric Pennerath</i>	
Large Margin Learning of Bayesian Classifiers Based on Gaussian Mixture Models . . . . .	50
<i>Franz Pernkopf and Michael Wohlmayr</i>	
Learning with Ensembles of Randomized Trees: New Insights . . . . .	67
<i>Vincent Pisetta, Pierre-Emmanuel Jouve, and Djamel A. Zighed</i>	
Entropy and Margin Maximization for Structured Output Learning . . . . .	83
<i>Patrick Pletscher, Cheng Soon Ong, and Joachim M. Buhmann</i>	
Virus Propagation on Time-Varying Networks: Theory and Immunization Algorithms . . . . .	99
<i>B. Aditya Prakash, Hanghang Tong, Nicholas Valler, Michalis Faloutsos, and Christos Faloutsos</i>	
Adapting Decision DAGs for Multipartite Ranking . . . . .	115
<i>José Ramón Quevedo, Elena Montañés, Oscar Luaces, and Juan José del Coz</i>	
Fast and Scalable Algorithms for Semi-supervised Link Prediction on Static and Dynamic Graphs . . . . .	131
<i>Rudy Raymond and Hisashi Kashima</i>	
Modeling Relations and Their Mentions without Labeled Text . . . . .	148
<i>Sebastian Riedel, Limin Yao, and Andrew McCallum</i>	
An Efficient and Scalable Algorithm for Local Bayesian Network Structure Discovery . . . . .	164
<i>Sérgio Rodrigues de Morais and Alex Aussem</i>	

Selecting Information Diffusion Models over Social Networks for Behavioral Analysis . . . . .	180
<i>Kazumi Saito, Masahiro Kimura, Kouzou Ohara, and Hiroshi Motoda</i>	
Learning Sparse Gaussian Markov Networks Using a Greedy Coordinate Ascent Approach . . . . .	196
<i>Katya Scheinberg and Irina Rish</i>	
Online Structural Graph Clustering Using Frequent Subgraph Mining . . . . .	213
<i>Madeleine Seeland, Tobias Girschick, Fabian Buchwald, and Stefan Kramer</i>	
Large-Scale Support Vector Learning with Structural Kernels . . . . .	229
<i>Aliaksei Severyn and Alessandro Moschitti</i>	
Synchronization Based Outlier Detection . . . . .	245
<i>Junming Shao, Christian Böhm, Qinli Yang, and Claudia Plant</i>	
Laplacian Spectrum Learning . . . . .	261
<i>Pannagadatta K. Shivaswamy and Tony Jebara</i>	
$k$ -Version-Space Multi-class Classification Based on $k$ -Consistency Tests . . . . .	277
<i>Evgueni Smirnov, Georgi Nalbantov, and Nikolay Nikolaev</i>	
Complexity Bounds for Batch Active Learning in Classification . . . . .	293
<i>Philippe Rolet and Olivier Teytaud</i>	
Semi-supervised Projection Clustering with Transferred Centroid Regularization . . . . .	306
<i>Bin Tong, Hao Shao, Bin-Hui Chou, and Einoshin Suzuki</i>	
Permutation Testing Improves Bayesian Network Learning . . . . .	322
<i>Ioannis Tsamardinou and Giorgos Borboudakis</i>	
Example-dependent Basis Vector Selection for Kernel-Based Classifiers . . . . .	338
<i>Antti Ukkonen and Marta Arias</i>	
Surprising Patterns for the Call Duration Distribution of Mobile Phone Users . . . . .	354
<i>Pedro O.S. Vaz de Melo, Leman Akoglu, Christos Faloutsos, and Antonio A.F. Loureiro</i>	
Variational Bayesian Mixture of Robust CCA Models . . . . .	370
<i>Jaakko Viinikanoja, Arto Klami, and Samuel Kaski</i>	
Adverse Drug Reaction Mining in Pharmacovigilance Data Using Formal Concept Analysis . . . . .	386
<i>Jean Villerd, Yannick Toussaint, and Agnès Lillo-Le Louët</i>	

Topic Models Conditioned on Relations . . . . .	402
<i>Mirwaes Wahabzada, Zhao Xu, and Kristian Kersting</i>	
Shift-Invariant Grouped Multi-task Learning for Gaussian Processes . . . .	418
<i>Yuyang Wang, Roni Khardon, and Pavlos Protopapas</i>	
Nonparametric Bayesian Clustering Ensembles . . . . .	435
<i>Pu Wang, Carlotta Domeniconi, and Kathryn Blackmond Laskey</i>	
Directed Graph Learning via High-Order Co-linkage Analysis . . . . .	451
<i>Hua Wang, Chris Ding, and Heng Huang</i>	
Incorporating Domain Models into Bayesian Optimization for Reinforcement Learning . . . . .	467
<i>Aaron Wilson, Alan Fern, and Prasad Tadepalli</i>	
Efficient and Numerically Stable Sparse Learning . . . . .	483
<i>Sihong Xie, Wei Fan, Olivier Verscheure, and Jiangtao Ren</i>	
Fast Active Exploration for Link-Based Preference Learning Using Gaussian Processes . . . . .	499
<i>Zhao Xu, Kristian Kersting, and Thorsten Joachims</i>	
Many-to-Many Graph Matching: A Continuous Relaxation Approach . . . .	515
<i>Mikhail Zaslavskiy, Francis Bach, and Jean-Philippe Vert</i>	
Competitive Online Generalized Linear Regression under Square Loss . . . . .	531
<i>Fedor Zhdanov and Vladimir Vovk</i>	
Cross Validation Framework to Choose amongst Models and Datasets for Transfer Learning . . . . .	547
<i>Erheng Zhong, Wei Fan, Qiang Yang, Olivier Verscheure, and Jiangtao Ren</i>	
Fast, Effective Molecular Feature Mining by Local Optimization . . . . .	563
<i>Albrecht Zimmermann, Björn Bringmann, and Ulrich Rückert</i>	
<b>Demo Papers</b>	
AnswerArt - Contextualized Question Answering . . . . .	579
<i>Lorand Dali, Delia Rusu, Blaž Fortuna, Dunja Mladenić, and Marko Grobelnik</i>	
Real-Time News Recommender System . . . . .	583
<i>Blaž Fortuna, Carolina Fortuna, and Dunja Mladenić</i>	
CET: A Tool for Creative Exploration of Graphs . . . . .	587
<i>Stefan Hawn, Andreas Nürnberger, Tobias Kötter, Kilian Thiel, and Michael R. Berthold</i>	

NewsGist: A Multilingual Statistical News Summarizer . . . . .	591
<i>Mijail Kabadjov, Martin Atkinson, Josef Steinberger, Ralf Steinberger, and Erik van der Goot</i>	
QUEST: Query Expansion Using Synonyms over Time . . . . .	595
<i>Nattiya Kanhabua and Kjetil Nørvåg</i>	
Flu Detector - Tracking Epidemics on Twitter . . . . .	599
<i>Vasileios Lampos, Tijl De Bie, and Nello Cristianini</i>	
X-SDR: An Extensible Experimentation Suite for Dimensionality Reduction . . . . .	603
<i>Panagis Magdalinos, Anastasios Kapernekas, Alexandros Mpiratsis, and Michalis Vazirgiannis</i>	
SOREX: Subspace Outlier Ranking Exploration Toolkit . . . . .	607
<i>Emmanuel Müller, Matthias Schiffer, Patrick Gerwert, Matthias Hannen, Timm Jansen, and Thomas Seidl</i>	
KDTA: Automated Knowledge-Driven Text Annotation . . . . .	611
<i>Katerina Papantoniou, George Tsatsaronis, and Georgios Paliouras</i>	
Detecting Events in a Million New York Times Articles . . . . .	615
<i>Tristan Snowsill, Ilias Flaounas, Tijl De Bie, and Nello Cristianini</i>	
Experience STORIES: A Visual News Search and Summarization System . . . . .	619
<i>Ilija Subašić and Bettina Berendt</i>	
Exploring Real Mobility Data with M-Atlas . . . . .	624
<i>R. Trasarti, S. Rinzivillo, F. Pinelli, M. Nanni, A. Monreale, C. Renso, D. Pedreschi, and F. Giannotti</i>	
<b>Author Index</b> . . . . .	629

# Efficient Planning in Large POMDPs through Policy Graph Based Factorized Approximations

Joni Pajarinen<sup>1</sup>, Jaakko Peltonen<sup>1</sup>, Ari Hottinen<sup>2</sup>, and Mikko A. Uusitalo<sup>2</sup>

<sup>1</sup> Aalto University School of Science and Technology,  
Department of Information and Computer Science,  
P.O. Box 15400, FI-00076 Aalto, Finland  
{Joni.Pajarinen,Jaakko.Peltonen}@tkk.fi

<sup>2</sup> Nokia Research Center, P.O. Box 407, FI-00045 NOKIA GROUP, Finland  
{Ari.Hottinen,Mikko.A.Uusitalo}@nokia.com

**Abstract.** Partially observable Markov decision processes (POMDPs) are widely used for planning under uncertainty. In many applications, the huge size of the POMDP state space makes straightforward optimization of plans (policies) computationally intractable. To solve this, we introduce an efficient POMDP planning algorithm. Many current methods store the policy partly through a set of “value vectors” which is updated at each iteration by planning one step further; the size of such vectors follows the size of the state space, making computation intractable for large POMDPs. We store the policy as a graph only, which allows tractable approximations in each policy update step: for a state space described by several variables, we approximate beliefs over future states with factorized forms, minimizing Kullback-Leibler divergence to the non-factorized distributions. Our other speedup approximations include bounding potential rewards. We demonstrate the advantage of our method in several reinforcement learning problems, compared to four previous methods.

## 1 Introduction

Planning under uncertainty is a central task in many applications, such as control of various robots and machines, medical diagnosis, dynamic spectrum access for cognitive radio, and many others. Such planning can often be described as a reinforcement learning problem where an agent must decide a behavior (action policy), and the quality of any policy can be evaluated in terms of a reward function. *Partially observable Markov decision processes* (POMDPs) [1] are a widely used class of models for planning (choosing good action policies) in such scenarios. In brief, in a POMDP the latent state of the world evolves according to a Markov model given each action chosen; the state is not directly observable, and end results of potential actions are not known, but the agent receives observations that depend on the state, and can plan ahead based on probabilistic beliefs about current and future states. For a survey of POMDP applications see, e.g., [2].

Policies are optimized in POMDPs by iterative algorithms. A central problem is that the optimization becomes computationally intractable when the size of the

underlying state space is large. We consider POMDPs with discrete state spaces; if the state is described by  $N$  variables, the number of states is at worst exponential in  $N$  and the number of state transitions is at worst exponential in  $2N$ . To combat computational intractability, many planning algorithms have been introduced with various approaches for improving efficiency [3–5]; we describe several approaches in Section 2.1. Overall, however, computational intractability remains a large problem which limits current applicability of POMDPs.

We present a novel method for efficient planning with POMDPs. In POMDPs the state can often be described by several variables whose individual transition probabilities do not depend on the whole state but only on a subset of variables. However, this does not yet ensure tractability: POMDP planning requires posterior probabilities of current and future states integrated over a distribution (belief) about previous states. Such integration is done over values of the whole previous state, and does not reduce to a computationally tractable form. However, the result can be *approximated* by a tractable form: in each such computation we use a factorized approximation optimized to minimize Kullback-Leibler divergence to the non-factorized intractable belief. We apply such factorized approximation in several central parts of the computation, which is organized based on a policy graph. The approximate computations ensure that beliefs over states remain in a factorized form, which crucially reduces complexity of evaluating and optimizing plans. We use a speedup based on computing bounds for potential policy rewards, to avoid evaluating policy alternatives that cannot compete with best existing policies; effectiveness of such pruning can be further increased with suitable ordering of evaluations. We describe our method in Section 3.

We compare the performance of our method to four existing POMDP solutions: two traditional approaches (Perseus [3] and HSVI [4]) and two methods designed for large problems (Symbolic Perseus [5] and Truncated Krylov Iteration combined with Perseus [5]). We compare the methods on four POMDP benchmark problems of scalable size, including two new benchmarks introduced here: the *Uncertain RockSample* problem, which is a more difficult variant of the traditional RockSample benchmark, and *Spectrum Access* which is adapted from a cognitive radio application and is described further below. Our method gets better policies than others in the same running time, and can handle large problems where other methods run out of memory, disk space, or time.

One increasingly important application area of reinforcement learning is *opportunistic spectrum access*, where devices such as cognitive radios detect available unused radio channels and exploit them for communication, avoiding collisions with existing users of the channels. This task can be formulated as a POMDP problem, and various POMDP solutions with different levels of model detail exist [6, 7]. Computational intractability is a problem for POMDP solutions: if the status of each channel (describing ongoing packet trains) is modeled with a detailed model having several states, like 15 in [6], state space grows exponentially as  $15^N$  with respect to the number of channels  $N$  used by the model; this makes POMDP computation challenging. The simple solution, restricting policies to few channels only, is not desirable: the more channels one can take

into account, the more efficient will be the use of spectrum and the more benefit the users will get. We present a new benchmark problem for POMDPs called *Spectrum Access* which is directly adapted from our proposal for a cognitive radio solution [6]. We use spectrum access as one of the benchmark problems in the experiments, and show that our method yields the best results for it.

In the following, we first describe the basic concepts of POMDPs and review existing methods for planning in POMDP problems in Section 2; in Section 3 we present our solution; and in Section 4 we describe the comparison experiments including the two new benchmark problems. In Section 5 results are discussed. We give conclusions in Section 6.

## 2 Partially Observable Markov Decision Processes

A partially observable Markov decision process (POMDP) is defined completely by (1) a *Markov model* describing the possible state transitions and observation probabilities given each action of the agent, and (2) a *reward model* defining how much reward is given when performing an action in a certain state. Formally a POMDP consists of the finite sets of states  $S$ , actions  $A$ , observations  $O$ , and rewards  $R : S \times A \rightarrow \mathcal{R}$ . At each time step, the agent performs action  $a$ , the world transitions from its current state  $s$  to a new state  $s'$  chosen according to the transition probabilities  $P(s'|s, a)$ , and the agent receives observation  $o$  according to the observation probability  $P(o|s', a)$ . The reward at each time step is  $R(s, a)$ .

The goal of the agent is to choose her actions to maximize a cumulative reward over time. We discuss the typical infinite-horizon discounted objective [1]  $E(\sum_{t=0}^{\infty} \gamma^t R_t)$ , where  $\gamma$  is the discount factor,  $0 < \gamma < 1$ , and  $R_t$  is the reward at time step  $t$ . The exact state of the world is not known to the agent, but a probability distribution, which tells the probability for being in state  $s$  can be maintained. This distribution is the so-called belief  $b$ : we denote the whole distribution by  $b$ , and the belief (probability) of being in state  $s$  is  $b(s)$ . The Bayes formula for updating the belief, after doing action  $a$  and getting observation  $o$  is

$$b'(s'|b, a, o) = P(o|s', a) \sum_s P(s'|s, a) b(s) / P(o|b, a), \quad (1)$$

where  $b'(s'|b, a, o)$  is the posterior belief, given  $o$ , that after action  $a$  the world is in state  $s'$ . The normalization term is the overall observation probability given  $a$  and the belief about the starting state,  $P(o|b, a) = \sum_{s'} P(o|s', a) \sum_s P(s'|s, a) b(s)$ . An optimal action  $a$  maximizes expected total discounted reward over possible futures; a precise definition is given later. Choosing  $a$ , given belief  $b$  over current states  $s$ , entails considering all possible action–observation sequences into the future. A function choosing an action for each  $b$  is called a *plan* or *policy*.

A brute force approach to choosing optimal actions would yield exponential complexity for planning with respect to how many steps ahead it looks. Many state-of-the-art algorithms exploit the Bellman equation for planning:

$$V^*(b) = \max_{a \in A} \left[ \sum_{s \in S} R(s, a) b(s) + \gamma \sum_{o \in O} P(o|b, a) V^*(b'(s'|b, a, o)) \right], \quad (2)$$



where  $V^*(b)$  is the value (total expected discounted reward) that can be attained when acting optimally if the current state is distributed according to belief  $b$ ; we call this the “value of  $b$ ” for short. Above, differently from our usual notation,  $b'(s'|b, a, o)$  denotes the whole posterior distribution over states  $s'$ , rather than a single value. The action  $a$  giving the maximum at right in (2) is optimal for belief  $b$ . State-of-the-art methods use Equation 2 iteratively to optimize policies.

*Factored definition of POMDPs.* POMDP problems can have millions of states. It would be computationally intractable to define such problems in a “flat” format with transitions as  $|S| \times |S|$  probability tables for each action. Luckily, many POMDP problems can be defined in factored form [8]. In a factored POMDP the state  $s$  is described as a combination of several variables  $s_i$ ; an observation  $o$  is described by several variables  $o_i$ . There can be millions of states  $s$ , and a large number of observations  $o$ , but the POMDP can be defined in terms of the individual  $s_i$  and  $o_i$ , which have only few elements each. The transition probabilities of each  $s_i$  and observation probabilities of each  $o_i$  depend only on a subset of the state variables:  $\text{Parents}(s_i)$  denotes the set of state variables affecting transitions of  $s_i$ , and  $\text{Parents}(o_i)$  is the set of state variables affecting the observation probabilities of  $o_i$ . The transition probabilities are then written as  $P(s'|s, a) = \prod_i P(s'_i | \text{Parents}(s'_i), a)$ , and observation probabilities are  $P(o | \text{Parents}(o), a) = \prod_i P(o_i | \text{Parents}(o_i), a)$ . The reward functions are defined as functions over subsets  $S_i$  of state variables:  $R(s, a) = \sum_i R_i(S_i, a)$ , where  $R_i$  is a reward function operating on subset  $S_i$ .

## 2.1 POMDP Methods

Several methods exist for planning in POMDPs. In principle, *exact optimal planning* in POMDPs can be done using incremental algorithms based on linear programming [9] to cover all possible beliefs. However, such algorithms can handle only problems with few states. *Point based value iteration* algorithms do not compute a solution for all beliefs, but either sample a set of beliefs (as in Perseus [3]) before the main algorithm or select beliefs during planning (as in HSVI [4]). ‘Value iteration’ refers to updating the value for a belief using a form of the Bellman equation (2). Point based value iteration algorithms scale to problems with thousands of states [4], which can still be insufficient. To cope with large state spaces, *POMDP compression methods* [10, 11] reduce the size of the state space. A linear static compression is computed and used to compress the transition and observation probabilities and the reward function. These compressions can then be fed to modified versions of POMDP algorithms like Perseus, to compute policies; this can give good performance [5] despite the (lossy) compression. In all these methods the value function is stored explicitly in vector form or in algebraic decision diagram (ADD) form; even for factored problems, its storage size grows exponentially with the number of state variables.

Some algorithms [5] store the policy as a finite state controller (FSC), a graph with actions as nodes and observations as edges. Any two nodes can be connected. ‘Policy iteration’ can improve the FSC by repeated policy improvement

and evaluation. Even if the FSC can be kept compact, the computed value functions have the same size as in the value iteration algorithms above. We maintain the value function as a policy graph using value iteration one layer at a time; nodes in each layer are connected only to the next layer.

Some methods *approximate belief updates*. In [12] it is shown that approximation errors in belief updates are bounded; a bound is given when minimizing Kullback-Leibler divergence between approximated and true belief. In [13] theoretical bound analysis of POMDP planning with approximated beliefs is done. Dynamic Bayesian network inference is performed in [14] using approximations similar to two of our approximations in Section 3.1.

The online POMDP method in [15] uses a factorized belief similar to ours; their planning involves searching different action-observation paths using a pruning heuristic; the worst case complexity is exponential in the search depth.

We lastly note that we use a policy graph based approach; a similar approach has been discussed in [16] for optimizing POMDP policy for unknown stationary transition probabilities but the setting is very different and approximation is not considered. We next describe our approach.

### 3 Our Algorithm: Factorized Belief Value Projection

We present our novel POMDP planning algorithm called Factorized Belief Value Projection (FBVP). Similarly to Perseus [3], FBVP starts by sampling a set of beliefs  $B$  before the actual planning. The main planning algorithm (Algorithm 1) takes the belief set  $B$  as input and produces a *policy graph* that can be used for decision making during online operation.

On a high level, Algorithm 1 works as follows. Initially, a simple policy  $\alpha_0$  is created, and all beliefs in  $B$  are associated with it. Then, at each iteration, beliefs are picked in random order, and for each belief  $b$ , a new policy  $\alpha$  is optimized that looks one step further than previously; this optimization is called the *backup* operation. The belief  $b$  is associated with this new policy  $\alpha$ . The policy  $\alpha$  may have been optimal also for other beliefs earlier during this iteration; if not ( $\alpha$  is new at this iteration), we check if any beliefs that haven't been processed yet during this iteration could get improved value from policy  $\alpha$ . If they do get improved value, we simply associate those beliefs with  $\alpha$  instead of separately picking them for optimization during this iteration. A speedup is to run *backup* for a small set of beliefs concurrently (randomly chosen set with heuristically chosen size) and check if any returned policy improves the values of the unprocessed beliefs; in the following description we omit this speedup for clarity. When all beliefs have been associated to some improved policy, the iteration ends and the new policies found during the iteration become the nodes of a new layer in the policy graph. The main algorithm is similar to the main algorithm of Perseus [3].

The key operations are the *backup* operation and evaluation (called *eval*) of the value of a policy for a belief. To perform them tractably, we exploit the policy graph from previous iterations, and use approximations; we describe the equations and approximations in Section 3.1 after the algorithms.

---

**Algorithm 1.** The main algorithm of our planning method FBVP
 

---

```

1: Input:  $P(s'_i|Parents(s'_i), a)$ ,  $P(o_i|Parents(o_i), a)$ ,  $R(S_i, a)$ ,  $V_0$ ,  $B$ 
2: Output: policy graph  $G$ 
3: Initialize  $n = 0$ ,  $\alpha_0(b) = V_0$ ,  $G_0 = \alpha_0$ 
4: repeat
5:   Initialize  $\tilde{B} = B$  and  $H = \emptyset$ .
6:   repeat
7:     Sample belief  $b$  from  $\tilde{B}$  and compute  $\alpha = \mathbf{backup}(b, G, P, R)$ .
8:     if  $\alpha \notin H$  then
9:        $H = (H, \alpha)$ 
10:    if  $\alpha \notin G$  then
11:      for  $b \in \tilde{B}$  do
12:        Compute  $V_{n+1}(b) = \mathbf{eval}(\alpha, b, G, P, R)$ 
13:      end for
14:    end if
15:     $\tilde{B} = (b \in \tilde{B} : V_{n+1}(b) < V_n(b))$ 
16:  end if
17: until  $\tilde{B}$  is  $\emptyset$ 
18:    $n = n + 1$ 
19:    $G_n = H$ 
20: until convergence

```

---

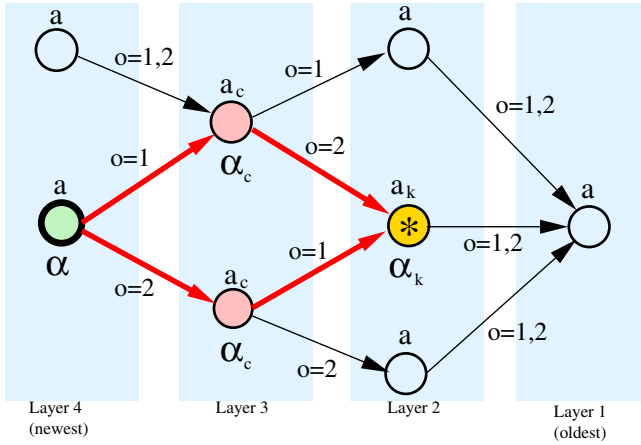
The *eval* algorithm (Algorithm 2) evaluates the value of a belief at a graph node  $\alpha$ , as follows: we proceed through the graph from the newest layer where  $\alpha$  is to the oldest. At each layer, we process all nodes: for each node  $\alpha_k$ , we know which younger nodes link to it (we call them “caller nodes”), and which observation is associated to each such link. At each caller node  $\alpha_c$  we have previously computed a projected belief  $b_c^{a_c}$  with values  $b_c^{a_c}(s') = b'_c(s'|b_c, a_c)$ ; here  $b_c$  is the belief projected from the starting node  $\alpha$  all the way to  $\alpha_c$ , and  $a_c$  is the action given by node  $\alpha_c$ . The observations when leaving  $\alpha_c$  are taken into account next: at each caller node  $\alpha_c$  we have previously computed the *path probability*,  $p_c$ , of arriving at that node. The belief at  $\alpha_k$  is a sum of caller beliefs, weighted by their path probabilities and the probabilities  $p(\alpha_c \rightarrow \alpha_k)$  of continuing their paths to  $\alpha_k$ . We have  $p(\alpha_c \rightarrow \alpha_k) = p(o_{c,k}|a_c, b_c)$  where  $o_{c,k}$  is the observation in the link ( $\alpha_c \rightarrow \alpha_k$ ). Similarly, the path probability  $p_k$  at  $\alpha_k$  is a sum of incoming path probabilities;  $p_k$  is used to normalize the beliefs at  $\alpha_k$ . As a computational trick, we multiply the path probability by the discount factor. The newest layer is a special case since it has no caller nodes: there, we just set the belief at the starting node and set the path probability to one for the starting node and zero for others. Having the belief at  $\alpha_k$ , we compute expected direct reward over the belief, given the action  $a_k$  chosen by  $\alpha_k$ . We multiply the expected reward by the path probability at  $\alpha_k$ , and add it to total reward for the original node  $\alpha$ ; since we incorporated the discount factor into path probabilities, the added rewards get properly discounted. Lastly, we compute the distribution  $b_k^{a_k}$  with values  $b_k^{a_k}(s') = b'_k(s'|b_k, a_k)$ . Now the node  $\alpha_k$  can be used as a “caller node” for further layers. We process all nodes in the layer, then proceed to the next layer, and so on through the graph.

**Algorithm 2.** Evaluate belief value at an  $\alpha$  node:  $\mathbf{eval}(\alpha_{n,j}, b, G, P, R)$ 

- 
- 1: Input: node  $\alpha_{n,j}$  (where  $n$  is the index of the newest graph layer and  $j$  is the index of the node within the layer), belief  $b$  at the node, graph  $G$ , probability tables  $P$ , reward tables  $R$
  - 2: Output: value  $V$  of using  $\alpha_{n,j}$  as policy for belief  $b$
  - 3: At the newest layer initialize beliefs :  $b_{n,j} = b$ , compute  $b_{n,j}^a$  (Equation 3) using action  $a_{n,j}$ , and initialize path probabilities:  $p_{n,j} = 1$  and  $p_{n,i} = 0$  for all  $i \neq j$ .
  - 4: Compute immediate reward at start:  $V = R(b_{n,j}, a_{n,j})$  (by Equation 7).
  - 5: **for** layers  $i = n - 1$  to 1 **do**
  - 6:   **for** each node  $k$  in layer  $i$  **do**
  - 7:     For this node  $\alpha_{i,k}$  (here denoted  $\alpha_k$  for short) do the following:
  - 8:     1. For each caller node  $\alpha_c$  linked to  $\alpha_k$  from the previous layer, compute  $b_c^{a_c, o_{c,k}} p_c p(\alpha_c \rightarrow \alpha_k)$ , where  $p(\alpha_c \rightarrow \alpha_k) = p(o_{c,k} | a_c, b_c)$ ,  $o_{c,k}$  is the observation associated with the link ( $\alpha_c \rightarrow \alpha_k$ ),  $a_c$  and  $b_c$  are the action and belief at  $\alpha_c$ , and  $b_c^{a_c, o_{c,k}}$  is the belief conditioned on  $a_c$  and  $o_{c,k}$  using Equation 4.
  - 9:     2. The path probability  $p_{i,k}$  at this node is a weighted sum over incoming path probabilities from caller nodes:  $p_{i,k} = \sum_c p_c p(o_{c,k} | a_c, b_c)$ .
  - 10:     3. The belief at this node  $\alpha_k$  is a weighted sum of incoming beliefs from caller nodes:  $b_{i,k} = \sum_c b_c^{a_c, o_{c,k}} p_c p(o_{c,k} | a_c, b_c) / p_{i,k}$ . Approximate the sum by a single factorized belief by minimizing KL-divergence (Equation 6).
  - 11:     4. Calculate expected immediate reward  $R_{i,k}$  for the belief  $b_{i,k}$  and the action  $a_{i,k}$  at this node (Equation 7).
  - 12:     5. Add the path-weighted reward  $R_{i,k} p_{i,k}$  to the total value  $V$ .
  - 13:     6. Project the belief  $b_{i,k}$  using the action  $a_{i,k}$  but not conditioning on observations, to obtain  $b_{i,k}^a$  with values  $b_{i,k}^a = b'_{i,k}(s' | b_{i,k}, a_{i,k})$  (Equation 3).
  - 14:     7. Multiply the path probability  $p_{i,k}$  by the discount factor  $\gamma$ .
  - 15:   **end for**
  - 16: **end for**
- 

Figure 1 illustrates *eval*, when computing value of the node  $\alpha$  (marked with green color and bold outline) for a given belief. The algorithm is currently processing node  $\alpha_k$  (marked with yellow color and an asterisk) in Layer 2. Nodes  $\alpha_c$  in Layer 3 are caller nodes for  $\alpha_k$ . Bold red arrows show where the belief at  $\alpha_k$  originates. The  $a$  indicate actions chosen in each state (many indices omitted for clarity) and  $o$  are observations attached to each link.

The *backup* (Algorithm 3) finds an improved policy for belief  $b$  as follows. The candidate improved value  $V$  is first  $-\infty$ . We plan to create a new node into the graph, and we must decide which action to choose there, and which further nodes it should link to for each observation. A posterior belief  $b'$  with values  $b'(s' | a, o)$  is computed for all actions  $a$  and observations  $o$ . For each action, we go through all observations, and for each observation we try nodes from the next graph layer, as candidates the new node could link to. We use *eval* to get the value given by each candidate for the belief  $b'$  we computed above. We choose the candidate with the highest value as the link for this action-observation pair. Having processed all observations for action  $a$ , we compute the value given by  $a$  for the original belief  $b$ : it is the sum of immediate rewards for this belief-action pair, and values given by the links chosen above for observation of this



**Fig. 1.** Illustration of the *eval* algorithm, which proceeds through a fixed policy graph from the left to the right, evaluating the reward acquired by starting from a certain node. The graph that *eval* operates on is constructed by Algorithm 1, adding one layer to the left end in each iteration and creating nodes into it using the *backup* algorithm.

$a$ , which are weighted by observation probabilities and by the discount factor. If this value is the best found so far, action  $a$  (along with the associated links for each observation from  $a$ ) replaces the previous best action.

### 3.1 Equations and Approximations

We keep the belief values always in a fully factored form  $b(s) = \prod_i b(s_i)$ . We maintain an exact lower bound for the value function in the form of the policy graph. However, when using the graph for belief value evaluation, there are three operations that can potentially break the fully factored form of beliefs. For each of these operations we find a fully factorized approximation (sometimes called a mean-field approximation).

We approximate the transition from the current belief to the next without observation conditioning with Equation 3. Let state variable  $s_i$  have  $K$  parents out of all  $N$  state variables; denote them  $Parents(s'_i) = s_1, \dots, s_K$ . It is easy to show that when we want to approximate the posterior distribution of several state variables by a fully factored form, the best approximation minimizing KL divergence from the real posterior is simply the product of marginals of the real posterior; for our model each marginal depends on the parents of the corresponding variable. The approximation is

$$b^a(s'_i) = b'(s'_i|b, a) = \sum_{s_1} b(s_1) \cdots \sum_{s_K} b(s_K) P(s'_i|s_1, \dots, s_K, a). \quad (3)$$

This approximation minimizes KL divergence from the real posterior belief  $b^a(s')$  to the factorized approximation  $\prod_i b^a(s'_i)$ , where  $b^a(s') = b'(s'|b, a) =$

**Algorithm 3.** Backup operation  $\alpha = \text{backup}(b, G, P, R)$ 

- 
- 1: **Input:** belief  $b$  for which an improved policy is wanted, policy graph  $G$ , Markov model  $P$ , reward function  $R$
  - 2: **Output:** policy graph node  $\alpha$  corresponding to an improved policy for  $b$
  - 3: Initialize the value of the improved policy to  $V_{max} = -\infty$ .
  - 4: Compute posterior belief  $b^{a,o}$  (having values  $b(s'|a, o, b)$ ) for all actions  $a$  and observations  $o$
  - 5: **for all**  $a$  **do**
  - 6:   **for all**  $o$  **do**
  - 7:     **for** candidate link target nodes  $\tilde{\alpha} \in G_{n-1}$  **do**
  - 8:       compute value  $V(b^{a,o}, \tilde{\alpha}) = \text{eval}(\tilde{\alpha}, b^{a,o}, G_{n-1, \dots, 1}, P, R)$
  - 9:     **end for**
  - 10:    choose the link target as the best candidate,  $\hat{\alpha}^{a,o} = \arg \max_{\tilde{\alpha}} V(b^{a,o}, \tilde{\alpha})$
  - 11:   **end for**
  - 12: Compute value of this action  $a$ ,  $V = R(b) + \gamma \sum_o P(o|b, a) V(b^{a,o}, \hat{\alpha}^{a,o})$
  - 13: **if**  $V > V_{max}$  **then**
  - 14:   This action becomes the best candidate so far,  $V_{max} = V$
  - 15:   Set the new node to use this action and its associated link targets,  $\alpha = (a, (\hat{\alpha}^{a,o} : \forall o \in O))$
  - 16: **end if**
  - 17: **end for**
- 

$\sum_{s_1, \dots, s_N} (\prod_i P(s'_i | \text{Parents}(s'_i), a)) b(s_1) \dots b(s_N)$ . We next approximate the conditioning of the above approximated belief on an observation, with Equation 4. Let observation  $o$  have  $L$  parents out of  $N$  state variables, call them  $\text{Parents}(o) = s'_1, \dots, s'_L$ . We minimize KL divergence to a factored approximation  $\prod_i b^{a,o}(s'_i)$ ; we set the gradient of the divergence with respect to each factor (distribution) to zero, which yields the distribution

$$b^{a,o}(s'_i) = \frac{1}{p(o|b, a)} \sum_{s'_1} b^a(s'_1) \dots \sum_{s'_{i-1}} b^a(s'_{i-1}) \sum_{s'_{i+1}} b^a(s'_{i+1}) \dots \sum_{s'_L} b^a(s'_L) P(o|s'_1, \dots, s'_L, a), \quad (4)$$

where  $p(o|b, a) = \sum_{s'_1} b^a(s'_1) \dots \sum_{s'_L} b^a(s'_L) P(o|s'_1, \dots, s'_L, a)$ . The approximations in Equations 3 and 4 are used in 14 for inference in Bayesian networks.

We also update the path probability (used in Algorithm 2) for each path arriving at a node  $\alpha_k$  from a node  $\alpha_c$ , associated with a belief  $b_c$ , an action  $a_c$  and an observation  $o_{c,k}$ :

$$p(c, c \rightarrow k) = p_c p(o_{c,k} | a_c, b_c) \quad (5)$$

When multiple beliefs “arrive” at a node  $\alpha_k$  from previous nodes  $\alpha_c$  (see Algorithm 2 for details), the belief at  $\alpha_k$  is a sum  $\sum_c (p(c, c \rightarrow k) / p_k) \prod_{i=1}^N b_c^{a_c, o_{c,k}}(s_i)$  where  $p_k$  is the path probability at  $\alpha_k$ ; this belief is approximated as a factored

form  $\prod_{i=1}^N b_k(s_i)$  using Equation 6; minimizing KL divergence (by again setting the gradient with respect to each discrete distribution to zero) we have

$$b_k(s_i) = \sum_c (p(c, c \rightarrow k) / p_k) b_c^{a_c, o_c, k}(s_i), \quad (6)$$

where  $p_k = \sum_c p(c, c \rightarrow k)$ .

The reward at the  $\alpha$  nodes can be calculated exactly using Equation 7:

$$R(b, a) = \sum_{s_1} b(s_1) \cdots \sum_{s_N} b(s_N) R(s_1, \dots, s_N, a). \quad (7)$$

Note that the approximation error in FBVP can reduce further into the policy graph, with a rate depending on several factors; see [12] for analysis. The value function is a convex piecewise linear function [1] corresponding to the policy graph; beliefs that share the same optimal linear function, i.e. same optimal policy graph node, are more likely to be “near” each other in the belief simplex. Thus the error in the approximation in Equation 6 is usually small, because beliefs in the approximated sum are usually similar.

### 3.2 Pruning

In the worst case, in iteration  $t$  each belief is evaluated  $\mathcal{O}(|A||O||G_{t-1}|)$  times, where  $|G_{t-1}|$  is the number of policy graph nodes in layer  $t - 1$ ; for each belief, the maximum number of calls to Equations 3, 6, and 7 is  $\mathcal{O}(\sum_{i=1}^{t-1} |G_i|)$  and  $\mathcal{O}(|O| \sum_{i=1}^{t-1} |G_i|)$  for Equation 4. The algorithm has polynomial complexity with respect to the number of state variables and to the horizon (maximum plan-ahead depth) and scales well to large state spaces, but evaluating the whole belief tree yields significant computational overhead. When the number of  $\alpha$  nodes is large, the *backup* algorithm (see Algorithm 3) dominates. To eliminate a part of policy graph evaluation we compute an approximate upper bound for the value at each graph node. This bound is used to compute maximum values during policy graph evaluation. Evaluation can be stopped if the value accumulated so far, added to the maximum value possible, is smaller than the best found solution’s value.

The requirements for a *policy graph node upper bound* are: it should not underestimate the value of a node, should be tight, fast to evaluate, and to a lesser extent fast to compute.

Each of our policy graph nodes would, in a traditional approach [3], correspond to a vector, whose dot product with a belief, would yield the value for the belief. Because of the state space size, we use a sum of linear functions of individual state variables as an approximation. Then  $B\mathbf{v} = V$ , where  $B$  is the matrix of beliefs in factored form for which we have computed values  $V$  and  $\mathbf{v}$  is a vector of concatenated linear state variable functions. Each row of the matrix  $B$  has the probabilities of the single state variables concatenated. The approximation is not exponential in the number of state variables and is tractable. To guarantee  $\mathbf{v}$  does not underestimate the value, all extreme points of the beliefs would have to be added to  $B$ . But as the number of extreme points equals the size of the

state space, we calculate additional values for a randomly selected set of beliefs, with individual belief component values set to unity.

In order to find a tight bound we can reformulate the problem as  $\mathbf{v}^T B^T B \mathbf{v} - V^T B^T \mathbf{v} = \epsilon$ ,  $B \mathbf{v} \geq V$  and find the  $\mathbf{v}$  that minimizes  $\epsilon$  using quadratic programming or fit  $\mathbf{v}$  using least squares. The quadratic programming approach is possible in many problems, because the individual state variables can be very small. For example in the computer network problem (see Section 4 for more details) there are 16 computers each having a binary state variable.

We either compute a fit with least squares or a bound with quadratic programming (with a time limit), and then use cross-validation to estimate the maximum error of the bound and add that to the bound to ensure optimal actions for the current belief set are not pruned. The procedure does not guarantee an exact bound for future belief sets but performed well in the experiments; we used 5-fold cross-validation and quadratic programming to compute the bounds.

A further optimization can be done by *observation ordering*. The value for an action is the sum of the immediate reward for the belief and the values for the beliefs projected for each observation. When we have accumulated the sum of the immediate reward and the rewards for part of the observations, the remaining observations must produce at least a “minimum” value that is greater than the best found action value so far, minus the accumulated value, in order for the current action to exceed previous actions. As we can compute maximum values for policy graph nodes, we can use the probability of the observation under consideration, the required “minimum” value, and the maximum values for the remaining observations to compute the smallest possible value for an observation that can make the current action exceed previous actions. If the observation does not reach this smallest possible value during value evaluation, the action under consideration can be pruned. By ordering projected beliefs according to their observation probability (see line 6 of Algorithm 3) this early stopping can be made more likely and the effectiveness of the upper bound pruning increased.

## 4 Experiments

We compare our method against four others on several benchmark problems. We next describe the comparison methods, benchmark problems, and results.

### 4.1 Comparison Methods

We use the following POMDP algorithms as comparison methods:

1. Perseus<sup>1</sup> iteratively improves a value function lower bound processing a fixed set of belief samples in random order. The proposed method FBVP resembles Perseus in that it also samples a fixed set of beliefs and then improves the value lower bound for each belief.

---

<sup>1</sup> Code available at <http://staff.science.uva.nl/~mtjspaans/software/approx/>



2. HSVI2<sup>2</sup> maintains both an upper and lower bound on the value function. In contrast to Perseus it generates beliefs during training and applies a depth-first type of search, while Perseus uses a breadth-first type of search.
3. Symbolic Perseus<sup>3</sup> is a version of Perseus that uses abstract decision diagrams (ADDs) for representing the POMDP data structures and is configured in factored form. It uses a mean-field approximation on the beliefs in Bellman backups and cuts of non-significant ADD leaves. Symbolic Perseus has been applied on POMDP problems with millions of states [17].
4. In truncated Krylov iteration [5] the POMDP problem is compressed into a linear approximation with smaller dimension than the original problem and a policy is found by using a (slightly modified) standard POMDP solver with the compressed problem. We use Perseus [3] as the POMDP solver. For factored problems the problem structure can potentially be exploited to perform Krylov iteration efficiently. We have a generic truncated Krylov iteration implementation using ADD data structures as suggested in [5] to provide a fair comparison. In our implementation the basis vectors are stored as sums of products of state variable functions. With this implementation we were able to compute compressions even for the largest problems.

## 4.2 POMDP Problems

We use two traditional benchmarks: *RockSample* [4] (a rover moves on a grid and tries to sample rocks that are “good”; e.g. RS(5,15) denotes a  $5 \times 5$  field with 15 rocks) and *Computer Network* (computers are up or down and administrators can ping or reboot them; included with Symbolic Perseus software; e.g. CN(16) denotes 16 computers). In *RockSample*, the rover’s position is deterministic; we introduce a new *Uncertain RockSample* problem, where the rover stays at the same location (except when moving to a terminal state) with 0.05 probability, when it should move. The uncertain location makes the problem harder. E.g. URS(5,15) again denotes a  $5 \times 5$  field with 15 rocks.

We also present a new benchmark problem for POMDPs which we call *Spectrum Access* (SA) [4]. It is motivated by real-life needs of growing wireless communication: the number of devices communicating over wireless connections is growing, and ever more data is transmitted due to e.g. increasing video communication. To avoid congestion over the limited amount of available spectrum, it is important to allocate resources over the spectrum efficiently; here a cognitive radio [18] must predict when a radio channel will be free of traffic, and use such “time slots” for communication but avoid access conflicts with existing (primary) users of the channels. Each channel evolves according to a 15-state Markov model estimated from simulated data; it describes packet burst lengths, pauses etc. The cognitive radio device can only sense three channels at a time and transmit on one at each step. Observations tell if a channel is busy/idle but

<sup>2</sup> Software available from <http://www.cs.cmu.edu/~trey/zmdp/>

<sup>3</sup> Software available from <http://www.cs.uwaterloo.ca/~ppoupart/software.html>

<sup>4</sup> See SA and URS specifications at [www.cis.hut.fi/jpajarin/pomdp/problems/](http://www.cis.hut.fi/jpajarin/pomdp/problems/)

not the exact channel state (burst type etc.). Rewards are given for successful transmissions penalties for using energy for listening ( $-0.01$  per channel) and strong penalties for access conflicts. E.g. SA(4) denotes four radio channels.

## 5 Results and Discussion

Table 1 shows discounted expected rewards for all the problems and algorithms tested. The discount factor was 0.95. In the classic RockSample (RS) problem, because of time constraints, algorithms were run for two days and in the other problems for three days. Part of the algorithms converged on some of the problems before maximum running time. The algorithms were initialized using their default initialization and FBVP was initialized with zero valued initial values for all problems. In spectrum access 3000 beliefs were used and 10000 in the other problems for Perseus, FBVP, and Symbolic Perseus. If an algorithm ran out of memory, then the intermediate policy output (if any) was used for evaluation. Evaluation was run for 500 runs of 60 steps. The methods were evaluated using their default evaluation method. Symbolic Perseus and FBVP were evaluated by following their policy graphs after belief evaluation, and Perseus, HSVI, and truncated Krylov iteration with Perseus were evaluated by using the computed value vectors.

Each maximum time experiment was run on one core of a “AMD Opteron 2435” processor with 8GB allocated memory. In only few cases such as Symbolic Perseus in the 16-machine computer network problem, all 8GB was needed.

Perseus and HSVI performed best in the smallest RS problem, but FBVP was very close to them. In the other RS problems Perseus and HSVI could not be configured. Truncated Krylov iteration together with Perseus did not perform well in any of the RS problems. In the 15-rock RS problem Symbolic Perseus achieved best results. Symbolic Perseus seems to be able to exploit the deterministic location of the rover using its ADD data structures. In the largest RS problems only FBVP had good results.

In the Computer Network (CN) problems we were not able to reproduce the results of Poupart et al. reported in [5] with our truncated Krylov iteration with Perseus implementation. This can be due to truncated Krylov iteration selecting basis vectors using an Euclidean metric. At what point the exact L1-normalization of basis vectors suggested in [5] is done in the truncated Krylov iteration algorithm may change the order of basis vectors added to the compression matrix. Also, even if the required adding of a constant to rewards does not change the optimal policy for the objective function, it changes the Euclidean distance between vectors. We used the results from [5] as an additional comparison and ran additional evaluation for Symbolic Perseus and FBVP for the reported running times. Note that the running times are not directly comparable. For the full training time Symbolic Perseus gave a policy that was very slow to evaluate and thus was limited to 276 evaluation runs.

In the CN problems Perseus and HSVI could not be configured due to size of the POMDP configurations. For the 16-machine problem FBVP and Symbolic Perseus gave better results than truncated Krylov iteration with Perseus. For the shorter training times Symbolic Perseus was better than FBVP and for the

**Table 1.** Performance of selected POMDP algorithms

Problem (states /actions/obs.)	Reward	Time	Problem (states /actions/obs.)	Reward	Time
<b>RS(5,5)</b> (801s 10a 2o)			<b>URS(5,5)</b> (801s 10a 2o)		
Perseus	19.05	2days	Perseus	16.25	3days
HSVI2	19.05	converged	HSVI2	16.43	out of mem
SymbolicPerseus	17.78	2days	SymbolicPerseus	15.47	3days
Tr.Kry.+Perseus	7.02	2days	Tr.Kry.+Perseus	16.73	3days
FBVP	18.67	2days	FBVP	15.64	3days
<b>RS(5,15)</b> (819201s 20a 2o)			<b>URS(5,15)</b> (819201s 20a 2o)		
Perseus	-	config. fail	Perseus	-	config. fail
HSVI2	-	config. fail	HSVI2	-	config. fail
SymbolicPerseus	31.66	2days	SymbolicPerseus	13.21	3days
Tr.Kry.+Perseus	-14.81	2days	Tr.Kry.+Perseus	-8.37	3days
FBVP	24.29	2days	FBVP	21.15	3days
<b>RS(5,20)</b> (26214401s 25a 2o)			<b>URS(5,20)</b> (26214401s 25a 2o)		
Perseus	-	config. fail	Perseus	-	config. fail
HSVI2	-	config. fail	HSVI2	-	config. fail
SymbolicPerseus	-	out of mem	SymbolicPerseus	-	out of mem
Tr.Kry.+Perseus.	-20.70	2days	Tr.Kry.+Perseus	2.25	3days
FBVP	23.89	2days	FBVP	18.55	3days
<b>RS(6,25)</b> ( $\sim 1.2G$ s 30a 2o)			<b>URS(6,25)</b> ( $\sim 1.2G$ s 30a 2o)		
Perseus	-	config. fail	Perseus	-	config. fail
HSVI2	-	config. fail	HSVI2	-	config. fail
SymbolicPerseus	-	out of mem	SymbolicPerseus	-	out of mem
Tr.Kry.+Perseus	0.00	2days	Tr.Kry.+Perseus	0.00	3days
FBVP	18.05	2days	FBVP	19.22	3days
<b>CN(16)</b> ( $2^{16}$ s 33a 2o)			<b>SA(4)</b> ( $15^4$ s 9a 8o)		
Perseus	-	config. fail	Perseus	-	out of mem
HSVI2	-	config. fail	HSVI2	13.71	out of mem
SymbolicPerseus	107.98	12658sec	SymbolicPerseus	14.19	3days
SymbolicPerseus	108.14	3days	Tr.Kry.+Perseus	13.14	3days
Tr.Kry.+Perseus	103.6	[Poupart05]	FBVP	14.52	3days
Tr.Kry.+Perseus	88.84	3days	<b>SA(8)</b> ( $15^8$ s 25a 8o)		
FBVP	105.97	12658sec	Perseus	-	config. fail
FBVP	109.05	3days	HSVI2	-	config. fail
<b>CN(25)</b> ( $2^{25}$ s 51a 2o)			SymbolicPerseus	-	out of mem
Perseus	-	config. fail	Tr.Kry.+Perseus	-43.07	3days
HSVI2	-	config. fail	FBVP	13.86	3days
SymbolicPerseus	-	out of mem			
Tr.Kry.+Perseus	148	[Poupart05]			
Tr.Kry.+Perseus	136.69	3days			
FBVP	152.01	8574sec			
FBVP	154.66	3days			

longer training times at the same level. In the 25-machine problem Symbolic Perseus ran out of memory and FBVP performed best.

In the smallest Uncertain RockSample (URS) problem Perseus and HSVI could be configured and trained successfully. All five methods got discounted rewards that were roughly at the same level. The second smallest URS problem had hundreds of thousands of states and FBVP performed best. Most likely the uncertain location of the rover makes the abstract decision diagram presentation in Symbolic Perseus less efficient than in the original RS problem of same size. For the two largest URS problems only FBVP had acceptable results. Symbolic Perseus ran out of memory in these problems. Truncated Krylov iteration with Perseus did not produce good results.

In the four channel Spectrum Access (SA) problem Perseus ran out of memory before actual training, but HSVI could be configured and trained for a while before memory ran out. FBVP and Symbolic Perseus had the best results. Truncated Krylov iteration with Perseus and HSVI had results that were not bad for such a big problem. In this problem truncated Krylov iteration was able to produce a compression close to a lossless compression in 1000 dimensions, when the original state space had 50625 dimensions. Only FBVP got good performance in the eight channel SA problem. Perseus, HSVI, and Symbolic Perseus did not yield any policy, and the reward obtained by truncated Krylov iteration with Perseus was much lower than that of FBVP.

## 6 Conclusions

We have presented a novel efficient POMDP algorithm policy graph based computation with factorized approximations and bounding. The approximations and policy graph approach ensure polynomial complexity with respect to number of state variables and look ahead depth. The results show that our algorithm, called Factorized Belief Value Projection (FBVP), scales well to very large problems and produces adequate rewards for smaller problems compared to algorithms that do not employ similar approximations. FBVP does not require a domain expert for specific tasks such as grouping state variables.

In the future it may be interesting to extend FBVP to problems where using the policy graph as the only value function representation can be inherently advantageous such as for POMDPs with unknown transition probabilities [16]. The effect of pruning and observation ordering also needs further study.

**Acknowledgments.** JoP belongs to AIRC; JaP to AIRC and HIIT. The work was supported by TEKES, PASCAL2, and Academy of Finland decision 123983.

## References

1. Sondik, E.J.: The optimal control of partially observable Markov processes over the infinite horizon: Discounted costs. *Operations Research* 26(2), 282–304 (1978)
2. Cassandra, A.R.: A Survey of POMDP Applications. Technical report, Austin, USA (1998) Presented at the AAAI Fall Symposium (1998)

3. Spaan, M., Vlassis, N.: Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research* 24, 195–220 (2005)
4. Smith, T., Simmons, R.: Point-Based POMDP Algorithms: Improved Analysis and Implementation. In: *Twenty-First Annual Conf. on Uncertainty in Artif. Int.*, Arlington, Virginia, pp. 542–549. AUAI Press (2005)
5. Poupart, P.: Exploiting structure to efficiently solve large scale partially observable Markov decision processes. PhD thesis, Univ. of Toronto, Toronto, Canada (2005)
6. Pajarinen, J., Peltonen, J., Uusitalo, M.A., Hottinen, A.: Latent state models of primary user behavior for opportunistic spectrum access. In: *20th Intl. Symposium on Personal, Indoor and Mobile Radio Communications*, pp. 1267–1271. IEEE, Los Alamitos (2009)
7. Zhao, Q., Tong, L., Swami, A., Chen, Y.: Decentralized cognitive MAC for opportunistic spectrum access in ad hoc networks: A POMDP framework. *IEEE J. Sel. Areas Commun.* 25(3), 589–600 (2007)
8. Boutilier, C., Poole, D.: Computing optimal policies for partially observable decision processes using compact representations. In: *Thirteenth National Conf. on Artif. Int.*, pp. 1168–1175. The AAAI Press, Menlo Park (1996)
9. Cassandra, A., Littman, M., Zhang, N.: Incremental pruning: a simple, fast, exact method for partially observable Markov decision processes. In: *13th Annual Conf. on Uncertainty in Artif. Int.*, pp. 54–61. Morgan Kaufmann, San Francisco (1997)
10. Poupart, P., Boutilier, C.: Value-directed compression of POMDPs. In: Becker, S., Obermayer, K. (eds.) *Advances in Neural Information Processing Systems*, vol. 15, pp. 1547–1554. MIT Press, Cambridge (2003)
11. Li, X., Cheung, W., Liu, J., Wu, Z.: A novel orthogonal NMF-based belief compression for POMDPs. In: Ghahramani, Z. (ed.) *24th Annual International Conference on Machine Learning*, pp. 537–544. Omnipress (2007)
12. Boyen, X., Koller, D.: Tractable inference for complex stochastic processes. In: *Fourteenth Annual Conf. on Uncertainty in Artif. Int.*, pp. 33–42. Morgan Kaufmann, San Francisco (1998)
13. McAllester, D., Singh, S.: Approximate planning for factored POMDPs using belief state simplification. In: *Fifteenth Annual Conf. on Uncertainty in Artif. Int.*, pp. 409–417. Morgan Kaufmann, San Francisco (1999)
14. Murphy, K., Weiss, Y.: The factored frontier algorithm for approximate inference in DBNs. In: *Seventeenth Annual Conf. on Uncertainty in Artif. Int.*, pp. 378–385. Morgan Kaufmann, San Francisco (2001)
15. Paquet, S., Tobin, L., Chaib-draa, B.: An online POMDP algorithm for complex multiagent environments. In: *Fourth International Joint Conference on Autonomous Agents and Multiagent systems*, pp. 970–977. ACM, New York (2005)
16. Poupart, P., Vlassis, N.: Model-based Bayesian reinforcement learning in partially observable domains. In: *Tenth Intl. Symp. on Artif. Intelligence and Math.* (2008)
17. Boger, J., Poupart, P., Hoey, J., Boutilier, C., Fernie, G., Mihailidis, A.: A decision-theoretic approach to task assistance for persons with dementia. In: *Nineteenth Intl. Joint Conf. on Artif. Int.*, vol. 19, pp. 1293–1299 (2005)
18. Haykin, S.: Cognitive radio: brain-empowered wireless communications. *IEEE J. Sel. Areas Commun.* 23, 201–220 (2005)

# Unsupervised Trajectory Sampling

Nikos Pelekis<sup>1</sup>, Ioannis Kopanakis<sup>2</sup>, Costas Panagiotakis<sup>3</sup>, and Yannis Theodoridis<sup>4</sup>

<sup>1</sup> Dept. of Statistics and Insurance Science, Univ. of Piraeus, Greece  
npelekis@unipi.gr

<sup>2</sup> Tech. Educational Inst. of Crete, Greece  
i.kopanakis@emark.teicrete.gr

<sup>3</sup> Dept. of Computer Science, Univ. of Crete, Greece  
cpanag@csd.uoc.gr

<sup>4</sup> Dept. of Informatics, Univ. of Piraeus, Greece  
ytheod@unipi.gr

**Abstract.** A novel methodology for efficiently sampling Trajectory Databases (TD) for mobility data mining purposes is presented. In particular, a three-step unsupervised trajectory sampling methodology is proposed, that initially adopts a symbolic vector representation of a trajectory which, using a similarity-based voting technique, is transformed to a continuous function that describes the representativeness of the trajectory in the TD. This vector representation is then relaxed by a merging algorithm, which identifies the maximal representative portions of each trajectory, at the same time preserving the space-time mobility pattern of the trajectory. Finally, a novel sampling algorithm operating on the previous representation is proposed, allowing us to select a subset of a TD in an unsupervised way encapsulating the behavior (in terms of mobility patterns) of the original TD. An experimental evaluation over synthetic and real TD demonstrates the efficiency and effectiveness of our approach.

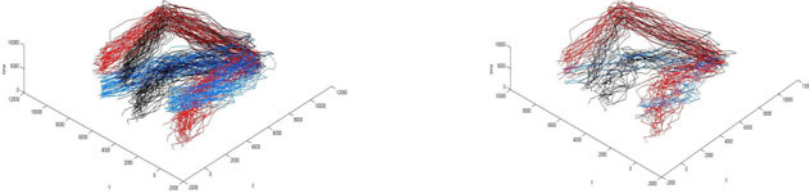
**Keywords:** Trajectory Databases, Sampling, Symbolic Trajectories.

## 1 Introduction

Data analysis and knowledge discovery over trajectory databases [8] discovers behavioral patterns of moving objects that can be exploited in several fields. Example domains include traffic engineering, climatology, social anthropology and zoology, studying vehicle position data, hurricane track data, human and animal movement data, respectively. In the literature, there have been proposed several works that try to analyze trajectory data either in an exploratory way [1], [3], [4] or by mining movement-aware patterns, such as clusters of moving objects [5], [15], [12], [19], [14], [10], sequential trajectory patterns [7], and flock patterns [9].

All of these approaches usually operate on large TD and it is natural for an analyst to wonder whether she could extract the same patterns operating on a much smaller and representative subset of the TD. In other words, the question is rephrased to whether one can appropriately reduce a large TD, taking only a sample of it, whose size is automatically computed, in an optimized and unsupervised way, and which

encapsulates the mobility patterns hidden in the whole TD. Fig. 1 illustrates the above discussion. Let us assume that mining a TD (Fig. 1(a)) results in a set of movement patterns (say, those illustrated by different colours in Fig. 1(a)). The question is whether we could extract an appropriate subset of the original TD (Fig. 1(b)) that would again capture the same patterns. If the answer to the above question is yes, such a methodology would radically speed up several analysis and mining tasks in the field.



**Fig. 1.** (a) original TD, (b) sample TD

The problem of trajectory sampling is very challenging due to the complexity of movement data (e.g. lack of ordering, lack of compact representation) that makes standard (e.g. uniform or stratified sampling) or point density-biased sampling techniques (e.g. [17], [11], [16]) inappropriate for the TD domain. A naïve solution for trajectory sampling would first cluster the TD and select the centroids of each cluster as the TD samples. However, existing trajectory-oriented clustering algorithms provide a single representation associated with each cluster (called *Representative* [12] or *Centroid* trajectory [19]). In addition, the aim of clustering is to partition the data into groups and not to downsize the TD. More importantly, such a solution would fail to select trajectories important for mobility patterns other than clusters (e.g. sequential trajectory patterns [7], or trajectory outliers [13]). To the best of our knowledge, it is only explorative, supervised sampling techniques that have been proposed for TD [3], [4], which however suffer from the above mentioned problems, as they imply a-priori knowledge of the underlying trajectory clusters.

In this paper, we propose a three-step approach to tackle the problem of trajectory sampling. Initially, we adopt a symbolic representation of trajectories that allows us to model all trajectories of the TD in an approximate way as vectors (i.e., sequences of regions wherefrom a moving object passes), whose dimensionality implies the intended space-time granularity of the application. This symbolic representation preserves the mobility pattern of each trajectory (by vanishing jerky movements), speeds up computations, and, moreover, turns out to be lossless in terms of mobility patterns, as shown in our experimental study.

On top of this representation, we propose an effective method to represent each trajectory by a continuous function that implicitly describes the *representativeness* of each constituent part of it (i.e., dimension) with respect to the whole TD. Given such an intuitive representation, we devise an algorithm, called *SyTra* (*Symbolic Trajectory*), which improves the initial representation of each trajectory by relaxing its vector representation. The idea is to adopt a merging algorithm that identifies the maximal time period wherein the mobility pattern of each trajectory is preserved, while in this augmented period it presents uniform behavior in terms of *representativeness*.

In other words, this merging process is an optimization trade-off between local (trajectory’s mobility pattern) and global (representativeness in TD) criteria.

At the third step of our approach, we propose an automatic method for trajectory sampling, called *T-Sampling*, based on the representativeness of the trajectories. An important aspect of this method is that it takes into account not only the most (i.e., dense, frequent) but also the least representatives, which are quite interesting in various application scenarios (e.g. detecting movement outliers or sparse clusters), thus they should somehow survive in the sample TD.

The contributions and merits of our work are summarized below:

- Based on a symbolic vector representation of a trajectory, we propose a global voting method allowing us to quantify the *representativeness* of each trajectory in a TD as a continuous descriptor.
- We devise an algorithm (*SyTra*) that relaxes the time-based point representation of the centroid trajectories, allowing the modeling of the mobility pattern of each trajectory at a higher level abstraction, as well as in a ‘homogenous’ way according to its representativeness in TD.
- We propose a novel unsupervised method (*T-Sampling*) for sampling the representative trajectories in a TD. The cardinality of the sample is either given as input or selected by the method itself.
- Finally, we conduct a comprehensive set of experiments over synthetic and real trajectory datasets, in order to thoroughly evaluate our approach.

The rest of this paper is structured as follows: Section 2 discusses related work. In Section 3, we set the scene by formulating the problem. The proposed methodology for trajectory sampling is presented in Section 4. In Section 5, we conduct an experimental study over synthetic and real TD in order to evaluate our approach. Finally, Section 6 concludes the paper.

## 2 Related Work

Recently there is an increasing interest in TD mining. Trajectory sampling is a very challenging task with great potential applications in TD mining, however very limited work has been carried out. Among the various proposals for the discovery of mobility patterns, the works in [12], [19] further provide aggregate representation of the extracted patterns (representative trajectories).

In [12], the authors proposed TRACCLUS, a partition-and-group framework for clustering 2D trajectories which enables the discovery of common sub-trajectories. TRACCLUS clusters trajectories as line segments (sub-trajectories). The notion of the *representative trajectory* of a cluster is also defined. In this approach, the temporal information is not taken into consideration, while the partitioning is performed per trajectory and it does not use global criteria. This is in contrast to our approach where our merging algorithm, which can also be considered as a segmentation method, views trajectories as sequences of sub-trajectories that are identified using global criteria. Furthermore, in contrast to our work, the representative sub-trajectories identified by TRACCLUS conform to straight movement and cannot identify complex (e.g. snakelike) patterns, which are common in real world applications.



In [19], an approach for clustering trajectories as a whole is proposed, using a symbolic representation of trajectories and local criteria, also with special care for handling uncertainty. In particular, a density- as well as similarity-based algorithm, called *CenTra*, is proposed in order to discover the *centroid* of a group of trajectories. Then, an FCM variation, called *CenTR-I-FCM*, clusters trajectories by utilizing *CenTra*. In comparison to the present work where we target at sampling, this approach also uses a global but static and predefined temporal segmentation of trajectories, which are symbolically represented as intuitionistic fuzzy vectors.

For trajectory segmentation, related work includes [1], [18]. In [1], global distance-based criteria have been proposed for the segmentation of trajectories using spatiotemporal information. The distance-based segmentation problem is given as a solution of a maximization problem that attempts to create minimum bounding rectangles (used as simplifications of parts of trajectories) in such a way that the original pairwise distances between all trajectories are minimized. In comparison with our merging approach, we also use global criteria; however these are entailed by the representativeness of each trajectory with respect to the TD. At the same time, our merging process gives special attention to preserve the movement pattern of each trajectory, as our ultimate goal is to succeed effective sampling. In [18], an approach for expressing the representativeness in a TD via a voting process is proposed. This process is improved in the present work in order to support the first step of our methodology, which is to describe the representativeness of each trajectory in the TD.

Explorative TD analysis using visual techniques is proposed in [3], [4]. The authors use uniform sampling to minimize the volume of the trajectories that can be clustered by a density-based clustering algorithm [15]. This approach suffers from the limitations discussed in the introduction, as sampling is supervised by the user and solely depends on the results of the clustering.

Finally, in [17], [11], [16], density biased sampling (DBS) techniques for point sets are proposed, which, obviously, cannot find straightforward application in TD. Nevertheless, in our approach we extend the idea of DBS in a way that density properties as well as the similarity of trajectories segments are taken into account.

Summarizing, although very interesting as a problem and with great potentials in the TD domain, to the best of our knowledge there is no related work on unsupervised trajectory sampling taking the complex nature of TD into consideration.

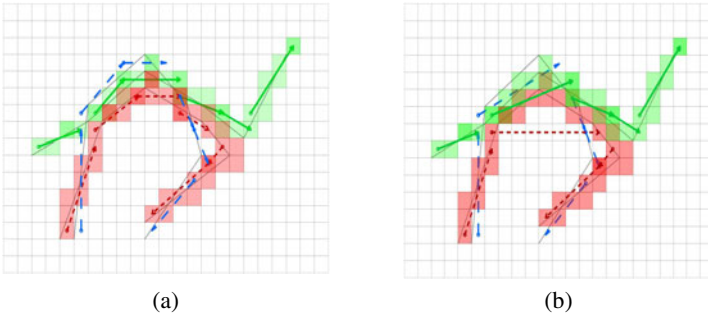
### 3 Setting the Scene

Let  $D = \{T_1, T_2, \dots, T_N\}$  be a dataset of  $N$  trajectories. Assuming linear interpolation between consecutive time-stamped positions, a trajectory  $T_i = \langle (x_{i,0}, y_{i,0}, t_{i,0}), \dots, (x_{i,n_i}, y_{i,n_i}, t_{i,n_i}) \rangle$ , consists of a sequence of  $n_i > 0$  line segments in 3D space, where the  $j$ -th segment interpolates positions sampled at time  $t_{i,j-1}$  and  $t_{i,j}$ . In this paper, we adopt an approximate representation of trajectories, originally proposed in [19], by transforming trajectories in a space where each  $T_i$  is represented as a  $p$ -dimensional point. (In this way, as shown in [19], we speed up computations, at the same time vanishing negligent movements while preserving the shape designating the mobility pattern of each trajectory.) More specifically, we use an approximation technique and define the dimensionality of trajectories by dividing the lifespan of each trajectory in  $p$  sub-intervals (e.g. 1 min. periods). Regarding the spatial dimension, we assume a regular grid of equal rectangular

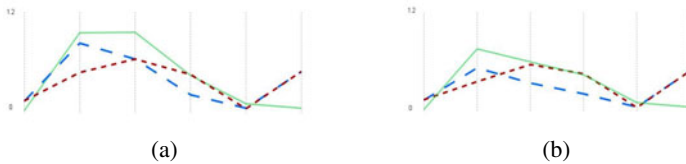
cells with user-defined size (e.g.  $100 \times 100 \text{ m}^2$ ). Given this setting, we extend the technique used in [19] where each trajectory  $T_i$  is partitioned into  $p \ll n_i$  equi-sized temporal periods and substitutes the trajectory 3D line segments of each period with (a) the set of the grid cells that  $T_i$  crosses during this period and (b) a motion vector  $\vec{d}$  that specifies the direction of the trajectory during the period. Formally:

**Definition 1.** Given a regular grid  $G$  of granularity  $m \times n$  consisting of cells  $c_{k,l}$  ( $1 \leq k \leq m$  and  $1 \leq l \leq n$ ), the lifespan  $ls$  of all trajectories in the trajectory database  $D$ , a trajectory  $T_i$  in  $D$  as a sequence of  $n_i$  line segments, and a target dimension  $p \ll n_i$ , the approximate trajectory (ApTra)  $\bar{T}_i = \langle (r_{i,1}, \vec{d}_{i,1}), \dots, (r_{i,p}, \vec{d}_{i,p}) \rangle$  of  $T_i$  is the one resulted by  $T_i$  when all trajectory triplets  $(x_{i,j}, y_{i,j}, t_{i,j})$  of  $T_i$  found inside a temporal period  $p_j = \left[ \frac{ls \cdot (j-1)}{p}, \frac{ls \cdot j}{p} \right]$ ,  $1 \leq j \leq p$ , are replaced by a pair  $(r_{i,j}, \vec{d}_{i,j})$ , called directed region  $dr_{i,j}$ , consisting of a region  $r_{i,j}$ , which is composed by the set of cells  $c_{k,l}$  crossed by  $T_i$  during  $p_j$  and a directed segment  $\vec{d}_{i,j}$  starting from the center of the first cell and ending at the center of the last crossed cell. ■

This technique allows us to view all trajectories in  $D$  as vectors in the same, user-defined dimensionality  $p$ , where each value of the vector corresponds to a dynamic time-ordered list of cells crossed by the trajectory, accompanied with the corresponding motion vector. Note that, depending on the choice of the space-time granularity of grid  $G$ , intended by the application requirements, a trajectory may introduce *gaps* (i.e., regions with empty set of cells) when there is no motion during the particular period of time. Fig. 2(a) depicts three trajectories and their corresponding ApTra illustrated in different colours (red, green, blue). The dimensionality  $p$  is set to  $p = 6$ , the arrows imply direction, while, regarding the blue trajectory it is only the motion vectors that are coloured, for clarity.



**Fig. 2.** (a) ApTra for 3 trajectories (b) Merged ApTra



**Fig. 3.** (a) ReTra before merging (b) ReTra after merging

The *representativeness* of a trajectory in TD is then defined by extending the definition of density biased sampling (DBS) in point sets [11], [17] for trajectories segments. According to DBS, the local density for each point of the set is approximated by the number of points in a region, divided by the volume of the region. In our case, the representativeness of a trajectory corresponds to the number of other trajectories that are similar to that in terms of time, space, and direction. Technically speaking, the representativeness is calculated by a voting process that is applied for each directed region pair  $dr_{i,j}$  of  $\bar{T}_i$ , extending and improving a preliminary version of the proposed method, presented in [18]. Thus,  $dr_{i,j}$  will be voted by the approximate trajectories of TD, according to the distance of  $dr_{i,j}$  to each trajectory of the TD. The sum of these votes is related to the number of trajectories that are similar to  $dr_{i,j}$ , called hereafter  $v(dr_{i,j})$ . More formally:

**Definition 2.** Given an  $ApTra$   $\bar{T}_i = \langle (r_{i,1}, \bar{d}_{i,1}), \dots, (r_{i,p}, \bar{d}_{i,p}) \rangle$  and a voting function  $v(\cdot)$  we define a representative trajectory ( $ReTra$ ) as a set of triplets  $(r_{i,j}, \bar{d}_{i,j}, v(dr_{i,j}))$  where the third value corresponds to the representativeness of each directed region (i.e., sub-trajectory)  $dr_{i,j}$  of  $ApTra$ . ■

Under this definition, the representativeness is formulated as a continuous descriptor  $v(dr_{i,j})$  over  $j = 1, \dots, p$ . Fig. 3(a) depicts the descriptors  $v(dr_{i,j})$  for the trajectories of Fig. 2(a) where one can see the votes each trajectory collected in each of the six periods (vertical lanes), as well as to realize that the red and blue trajectories appear to have similar mobility patterns, while this is not the case for the green trajectory. A novel approach for the extraction of  $ReTra$  is given in Section 4.1.

Viewing dimensionality from a different perspective, one may consider it as a segmentation of the trajectories in time axis. Although intuitive, this segmentation is static and predefined for the whole TD. As we aim at sampling whole trajectories, we argue that a more intuitive representation would be to aggregate each trajectory along the temporal dimension. Such a representation would model each trajectory with a (possibly) different number of time periods (less than  $p$ ) of varying longer duration. Each such period would be the result of merging successive periods (and the respective regions and motion vectors) of the initial representation, having as aim to identify the maximal temporal periods during which some uniformity criterions hold. These criteria are a trade-off between the preservation of the (local) mobility pattern of each trajectory and the uniformity of the (global) representativeness of the merged directed regions.

**Definition 3.** Given a representative trajectory ( $ReTra$ ) of  $p$  triplets  $\langle (r_{i,j}, \bar{d}_{i,j}, v(dr_{i,j})) \rangle$ ,  $j = 1, \dots, p$ , a threshold  $\Delta d$  over the distance  $D_d$  between successive motion vectors of  $ReTra$  (see Definition 7), and a threshold  $\Delta v$  over the difference between successive values of  $v(dr_{i,j})$ , we define its symbolic trajectory ( $SyTra$ ) as a set of triplets  $(sr_{i,k}, \bar{s}d_{i,k}, v(sdr_{i,k}))$ ,  $k = 1, \dots, K_i$ , where  $K_i \leq p$  denotes the number of directed regions  $dr_{i,j}$  that are merged to define a symbolic directed region  $sdr_{i,k}$ ,  $sr_{i,k}$  is the union of successive  $\Delta v$ -similar regions  $r_{i,j}$ ,  $\bar{s}d_{i,k}$  is the motion vector of  $sr_{i,k}$  where the motion vectors of the merged  $\bar{d}_{i,j}$  are  $\Delta d$ -similar, and  $v(sdr_{i,k})$  is the representativeness (see Definition 2) after the merging. ■

In other words, the *symbolic trajectory* (*SyTra*) can be considered as a trajectory segmentation of into  $K_i$  sub-trajectories, where each sub-trajectory contains successive regions that are similar to each other with respect to trajectory representativeness (*Av-similar*) and shape (*Ad-similar*), thus providing a compact representation of *ReTra*.

Similarly to Definition 2, the *SyTra based representativeness* descriptor shows how many objects follow the specific  $k$ -sub-trajectory at almost the same time, space and direction. Note here that, as the symbolic directed regions  $sdr_{i,k}$  are different from directed regions  $dr_{i,j}$  the voting process is applied again to *SyTra*. Fig. 2(b) illustrates the results of merging the three *ReTra* of Fig. 2(a), while Fig. 3(b) illustrates the updated *representativeness* of the new voting process. A novel algorithm for computing *SyTra* is provided in Section 4.2.

**Definition 4.** *The trajectory representativeness descriptor  $V_i$  of a SyTra is defined as the weighting average over  $v(sdr_{i,k})$ ,*

$$V_i(D) = \frac{\sum_{k=1}^{K_i} \delta_{i,k} \cdot v(sdr_{i,k})}{\sum_{k=1}^{K_i} \delta_{i,k}} \quad (1)$$

where weight  $\delta_{i,k}$  denotes the duration of  $k$ -sub-trajectory. ■

The trajectory representativeness descriptor  $V_i$  is to be used for the trajectory sampling problem. Actually, the goal of trajectory sampling is to select the most representative trajectories of the TD. However, by selecting the top-voted sub-trajectories, which sounds to be an obvious decision, the high density regions of the TD will be oversampled, resulting in a non-representative sample. On the contrary, we propose the sampling to be considered as an optimization problem that aims to maximize a formula (Equation 2), taking into account the representativeness  $V_i(D)$  in the original TD  $D$  as well as the representativeness  $V_i(S)$  in the sample  $S$  ( $V_i(S)$  is defined similarly to  $V_i(D)$ ). So, our goal is that  $S$  should contain trajectories of high representativeness trying at the same time to cover the full space of  $D$ . Recalling Fig. 2, this implies that a reasonable decision of an intuitive sampling algorithm would be to select the green and either the red or the blue trajectory in case that the top-2 representative trajectories are requested (i.e.,  $|S| = 2$ ).

**Definition 5.** *Let  $S_i$  be equal to one when trajectory  $T_i$  of dataset  $D$  belongs to the sampling set, and equal to zero otherwise. The Trajectory Sampling problem is defined as the optimization of finding an appropriate subset  $S$  of  $D$ , which maximizes the function  $SR(S)$ :*

$$SR(S) = \sum_{i=1}^N S_i \cdot V_i(D) \cdot (1 - V_i(S)) \quad (2)$$

This implies that the number of trajectories in  $D$  that find their representatives in  $S$  is maximized. ■

Obviously, the idea of  $SR(S)$  is to favour  $V_i(D)$  and penalize  $V_i(S)$ , at the same time producing a non-monotonously increasing  $SR(S)$  with  $N$ . A novel solution to the *Trajectory Sampling problem* is provided in Section 4.3.

## 4 Sampling Trajectories

In this section, we present our approach for trajectory sampling in detail. The proposed methodology (consisting of three steps) is sketched by an algorithm, called *T-Sampling* and illustrated in Fig. 4. The input of the algorithm is a TD  $D$ , the parameters of our methodology (namely, a grid  $G$ , the dimensionality  $p$ , a control parameter  $\sigma$  – to be discussed in Section 4.1 – and thresholds  $\Delta v$  and  $\Delta d$ ), and a number  $topk$  for the desired cardinality of the sampling set  $S$ ; the output is the sampling set  $S$ . Although in the presented algorithm,  $topk$  is given as input, it is important to note that an ‘optimal’ value of it can be also recommended by the method itself (this interesting property will be discussed in Section 4.3).

The *T-Sampling* algorithm works as follows. First, trajectories are transformed into a low-dimensional symbolic space and their representatives (*ReTra*) are elected using a similarity-based voting technique (lines 1-2). Second, the maximal symbolic representative portions of trajectories (*SyTra*) are extracted (lines 3-5; line 5 is included only for clarity as its computation is incrementally performed in line 4). Third, sampling is achieved by a novel sampling algorithm (*TrajectorySampling*) in an unsupervised way (line 6).

---

**Algorithm T-Sampling**  
(TD  $D$ , Grid  $G$ , int  $p$ , Real  $\sigma$ , Real  $\Delta v$ ,  
Real  $\Delta d$ , int  $topk$ )

---

```

// STEP 1, see Sec. 4.1
01. for  $i=1$  to  $N$   $\bar{T}_i = ApTra(G, p, T_i)$ ;
02. for  $i=1$  to  $N$   $R_i = ReTra(\bar{T}_i, \sigma)$ ;
    // STEP 2, see Sec. 4.2
03. for  $i=1$  to  $N$   $S_i = SyTra(R_i, G, \Delta v, \Delta d)$ ;
04. for  $i=1$  to  $N$   $R_i = ReTra(S_i, \sigma)$ ;
05. forall  $R_i$  compute  $V_i(D)$ ; //Eq.1
    // STEP 3, see Sec. 4.3
06.  $S = TrajectorySampling(V(D), topk)$ ;
07. return  $S$ ;

```

---

Fig. 4. *T-Sampling* Algorithm

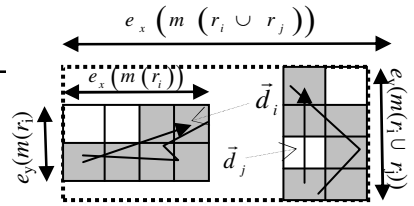


Fig. 5.  $D_r(r_i, r_j)$  distance function

The three steps of the algorithm are described in the following sections.

### 4.1 Voting for Trajectory Representatives

As already discussed, in order to define the representativeness of a trajectory we measure distance  $D_{dr}$  between two directed regions. The key observation is that such a distance function can be decomposed in two parts, distance  $D_r$  between two regions and distance  $D_d$  between two motion vectors; then, we combine them into a single distance function using an aggregator, e.g. the average of the two components (for clarity we suppress the second subscript notating the trajectory id):

$$D_{dr}(dr_i, dr_j) = (D_d(\vec{d}_i, \vec{d}_j) + D_r(r_i, r_j)) / 2 \quad (3)$$

Below we give the definition of the distance between two regions (i.e., sets of cells), taking also into account the time periods that they correspond to.

**Definition 6.** Given two 3D regions  $r_i$  and  $r_j$ , their distance  $D_r(r_i, r_j)$  is defined as follows:

$$D_r(r_i, r_j) = 1 - \frac{1}{3} \left( \sum_{\text{dim} \in \{x, y, z\}} \frac{e_{\text{dim}}(m(r_i)) + e_{\text{dim}}(m(r_j))}{2 \cdot e_{\text{dim}}(m(r_i \cup r_j))} \right), \quad (4)$$

where e.g.  $e_x(m(r_i))$  is the extent of the minimum bounding box (mbb) of  $r_i$  along the  $x$ -axis (similar for  $y$ - and  $z$ -axis). ■

Fig. 5 illustrates the 2D spatial projection of two regions (time is omitted for clarity), as well as the respective motion vectors. It is clear that  $D_r(r_i, r_j)$  is bounded in  $[0, 1]$ . Intuitively,  $D_r(r_i, r_j)$  takes into account both the Euclidean distance between two regions and their extents, while it produces non-zero results in the case of overlapping (in space or time) but non identical regions.

Similarly, we give the definition of the distance between two motion vectors (i.e., directed segments)  $\vec{d}_i$  and  $\vec{d}_j$  as the minimum normalized energy of rotation of line segment  $\vec{d}_i$  to  $\vec{d}_j$ , or vice-versa, which depends on the angle  $\theta$  formed between the two directed segments.

**Definition 7.** Given two motion vectors (directed segments)  $\vec{d}_i$  and  $\vec{d}_j$ , their distance  $D_d(\vec{d}_i, \vec{d}_j)$  is defined as the normalized distance, that a segment will cover, after its rotation of  $\theta$  rads,  $0 \leq \theta \leq \pi$ , where  $\theta$  is the angle between the two segments after they have been translated so as to have a common starting points. Formally: ■

$$D_d(\vec{d}_i, \vec{d}_j) = \frac{\theta}{\pi} \quad (5)$$

According to the problem formulation presented in Section 3, the *representativeness* of a directed region  $dr_i$  depends on its distance to every trajectory in TD. We define distance  $D_{dr}(dr_i, T_m)$  between  $dr_i$  and a trajectory  $T_m$  of the TD as the distance between two directed regions, namely  $dr_i$  and the closest directed region of  $T_m$  to  $dr_i$ . In the literature, several voting functions have been proposed, either step or continuous. In this work, we have selected to use the following continuous function of Gaussian kernel, which is widely used in a variety of applications in the field of pattern recognition [21].

$$v(dr_i, T_m) = e^{-\frac{D_{dr}^2(dr_i, T_m)}{2 \cdot \sigma^2}} \quad (6)$$

Since  $0 \leq D_{dr}(dr_i, T_m) \leq 1$ , the control parameter  $0 \leq \sigma \leq 1/2$  tunes how fast the function (i.e., voting influence) decreases with distance. According to this definition, it holds that  $0 \leq v(dr_i, T_m) \leq 1$ . The lower the  $D_{dr}(dr_i, T_m)$  the higher the value of the voting function and vice-versa.

Adopting a continuous voting function, like the Gaussian kernel, we get smooth results for slight changes on parameters ( $\sigma$  in our case), and getting decimal values as

results of the voting process increases the robustness of the method. Finally,  $v(dr_i)$  is computed by summing the votes from all trajectories  $T_m$ .

$$v(dr_i) = \sum_{m=1 \dots n, mk}^N v(dr_i, T_m) \quad (7)$$

**Lemma 1.** The time complexity of the *Trajectory Voting* step of our methodology is  $O(N \cdot p \cdot \log(N \cdot p))$ , where  $p$  is the dimensionality of the symbolic representation and  $N$  is the cardinality of  $D$ .

**Proof:** Assuming that  $L$  denotes the average number of directed regions per trajectory, the computational cost of representativeness  $v(dr_i)$  is  $O(N \cdot L)$ . This is measured  $L$  times on the average for each of  $N$  trajectories in TD. As such, the total computation cost is  $O(N^2 \cdot L^2)$ . However, in order to be able to execute the algorithm in large databases, this computation cost may be reduced using a spatial index, such as the R-tree-like structures used in [6]. Using R-trees reduces the cost of voting for each  $dr_i$  to  $O(\log(N \cdot L))$ , resulting in a total cost  $O(N \cdot L \cdot \log(N \cdot L))$ . The voting method is applied twice: first, on the  $p$ -dimensional  $ApTra$  and, second, on the merged  $SyTra$ . In the first case,  $L = p$  while in the second case  $L \ll p$ . Overall, the cost of the trajectory voting step is the sum of the two phases, hence  $O(N \cdot p \cdot \log(N \cdot p))$ . ■

## 4.2 Global Symbolic Trajectories

As already discussed, a more intuitive representation for *ReTra* that certainly has added value for an analyst would be to provide an aggregate representation along the temporal dimension (i.e., *SyTra*). Fig. 6 presents an algorithm that transforms *ReTra* to *SyTra*, according to Definition 3. The main idea of the algorithm is, starting from an initial time period to follow a local time-based clustering approach in the form of a greedy time-expansion (merging) technique that concatenates a set of successive periods, as long as distance  $D_d$  of the directed segments is low enough to preserve the (local) mobility pattern (sketch) of the trajectory, but also the *representativeness* of the candidate (for merging) regions remains more or less the same. More specifically, after the initialization (lines 1-2), the algorithm starts an iterative procedure until all time periods are used (lines 3-15). At each iteration, it appends a local merged *ReTra* to the transformed *SyTra* (line 14). The new local *ReTra* is initialized with the corresponding old local *ReTra* of the first time period not used so far (line 4). Subsequently, using the previous region as seed, the merging (M) begins (line 5) by searching the best among the candidates periods (lines 6-8), which if concatenated to local *ReTra* the resulted region will be  $\Delta d$ -similar, while the relative difference in representativeness, before and after the merging is low (i.e.,  $\Delta v$ -similar) (line 8). The best period to stop expansion is the one that minimizes the relative difference after merging (lines 9-12). The whole process continues until no more merging can be done (line 13).

**Lemma 2.** The time complexity of the *SyTra* algorithm is  $O(p^2)$ , where  $p$  is the dimensionality of the symbolic representation.

**Proof:** There are two loops that determine the complexity of the algorithm (lines 3 and 5). The loop (line 3) that revokes the merging process (line 5) may be repeated at most  $p$  times. This will happen if no actual merging occurs. This means that in this

---

**Algorithm SyTra**  
(ReTra  $retra$ , Grid  $G$ , Real  $\Delta v$ , Real  $\Delta d$ )

---

```

01. sytra= $\emptyset$ ; j=1; c=1;
02. forall i in [j, P] used( $p_i$ )=false;
03. repeat
04.    $retra_c = retra(p_j)$ ;
05.   repeat
06.     forall periods  $p_k$ , k in [j+1, p]
07.        $M_{j,k} = retra_c$  expanded with
          $retra(p_k)$ ;
08.    $M = \{M_{j,k} \mid D_d(retra_c, M_{j,k}) < \Delta d \text{ and}$ 
          $\frac{|v(M_{j,k}) - v(retra_c)|}{v(retra_c)} < \Delta v \}$ ;
09.   if  $M \neq \emptyset$ 
10.      $retra_c = \arg \min_{M_{j,k} \in M} \left( \frac{|v(M_{j,k}) - v(retra_c)|}{v(retra_c)} \right)$ ;
11.     forall i in [j, k] used( $p_i$ )=true;
12.     j=k+1;
13.   until  $M \neq \emptyset$ ;
14.   sytra = sytra  $\cup$   $retra_c$ ; c=c+1;
15. until used( $p_p$ )==true;
16. return sytra;

```

---

Fig. 6. SyTra Algorithm

---

**Algorithm TrajectorySampling**  
(Vector  $V(D)$ , int topk)

---

```

01. for i=1 to N  $S_i=0$ ;
02.  $V = \text{sort\_ascending}(V)$ ;
03. for k=1 to topk
04.    $SR_{gain}^{max} = -1$ ;
05.   for i=1 to N
06.     if  $S_i == 0$  AND
          $SR_{gain}(i) > SR_{gain}^{max}$ 
07.        $SR_{gain}^{max} = SR_{gain}(i)$ ;
08.       id = i;
09.     end
10.   if  $SR_{gain}^{max} > V_i$ 
11.     break;
12.   end
13. end
14. if  $SR_{gain}^{max} \leq 0$ 
15.   topk = i-1;
16.   return S;
17. end
18.  $S_{id}=1$ ;
19. end
20. return S;

```

---

Fig. 7. TrajectorySampling Algorithm

case the internal loop (line 5) will be revoked only once for each period (i.e.,  $M = \emptyset$ ) and, as such, its cost is only the cost of the loop through all (the remaining) time periods (lines 6-7), which is  $p$ ; plus the cost of the filtering (line 8), which is also  $p$ , as the  $D_d$  calculation has constant cost and the aggregated calculations are computed incrementally. Consequently, in this case the complexity of the algorithm is  $O(p(p+p))$ , hence  $O(p^2)$ . In the other extreme case, the loop (line 3) that revokes the time expansion process (line 5) is revoked only once, implying that all periods are merged in one. In this case, the cost of the algorithm is dominated by the internal loop (line 5), which, in the worst case, will be revoked  $p$  times. Again the cost of each iteration is  $O(p+p)$  as delineated in the previous paragraph, even if  $M = \emptyset$  in this case, as the actual merging (line 10) can also be calculated incrementally for all iterations of the merging process. So, the overall complexity is again  $O(p^2)$ . ■

### 4.3 Trajectory Sampling Based on Representativeness

At the final step of our methodology, we provide a solution to the trajectory sampling problem defined in Section 3. In detail, the *TrajectorySampling* algorithm (illustrated in Fig. 7) takes the representativeness vector descriptor of the trajectories and returns the sampled subset. Recall that the goal of sampling is the maximization of the number of trajectories  $SR(S)$  of the original TD that find their representatives in the sampling set (see Definition 5). The complexity of an exhaustive algorithm would search for all possible solutions in order to maximize Equation (2), resulting in a prohibitive cost  $O\binom{N}{topk}$ . On the other hand, the proposed algorithm suboptimally solves the problem in  $O(N \cdot topk)$  iterations adopting a greedy optimization approach.



In detail, the *TrajectorySampling* algorithm starts with an empty sampling set  $S_i = 0$  for each  $i = 1, \dots, N$  (line 1). In each iteration, an unselected trajectory of  $D$  that maximizes Equation (2) is added in  $S$ , which is equivalent with the maximization of  $SR_{gain}(i)$  that is defined as:

$$SR_{gain}(i) = V_i(D) \cdot (1 - V_i(S)) \quad (8)$$

$SR_{gain}(i)$  expresses the gain of  $SR(S)$  if the  $i$ -th trajectory of  $D$  is added in  $S$ . According to the proposed algorithm, it holds that  $SR_{gain}$  is a monotonically decreasing function as sampling size increases, while it also holds that  $SR_{gain}(i) \leq V_i(D)$ . Therefore, an efficient way to reduce the computation cost of the maximization of  $SR_{gain}(i)$ , is to keep the trajectories in a list sorted in ascending order by  $v_i(D)$  (line 2). Instead of computing  $SR_{gain}(i)$  for each trajectory in  $D$  in order to find the maximum, we get the  $i$ -th trajectory from the list and we compare its *representativeness* with the current highest  $SR_{gain}$  ( $SR_{gain}^{max}$ ) (lines 6-9). This loop terminates if  $SR_{gain}^{max}$  is higher than the *representativeness* of the  $i$ -th trajectory (line 10), because it holds that the  $SR_{gain}$  of each trajectory from the rest trajectories of the list would be lower than its *representativeness* and lower than  $SR_{gain}^{max}$ , since the contents of the list are in ascending order by trajectory *representativeness*. The algorithm terminates either when the size of sampling set reaches  $topk$  (which is given as an input parameter); or when  $SR_{gain}^{max}$  becomes a non positive number, which means that  $SR(S)$  has been maximized (lines 14-17), as the latter is increased in each step by  $SR_{gain}^{max}$ . In the second case, the algorithm has automatically found an ‘optimal’  $topk$ , assuming it is lower than the given input.

An advantage of the proposed method is that it provides a deterministic solution in contrary with other probabilistic techniques [11], [16] that provide a randomly constructed sampling set trying to fit it to a desired distribution.

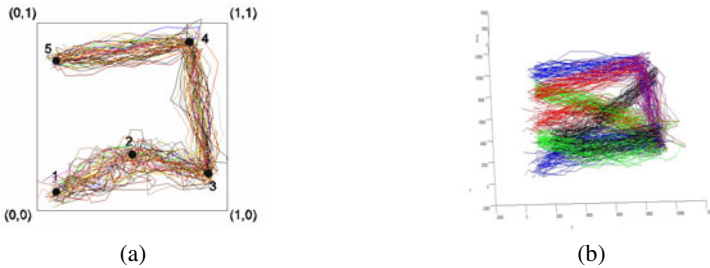
**Lemma 3.** Assuming  $topk \ll N$ , the time complexity of the *TrajectorySampling* is  $O(N \cdot \log(L \cdot topk) + N \cdot \log N)$ ,  $\Omega(N \cdot \log(N) + topk \cdot L \cdot \log(L \cdot topk))$ , where  $N$  is the number of trajectories in TD and  $L$  is the average number of directed regions per trajectory.

**Proof:** First, sorting  $V(D)$  costs  $O(N \cdot \log(N))$ . Next, in the worst case, the method computes  $SR_{gain}(k)$   $topk \cdot N$  times.  $SR_{gain}(k)$  computation requires the computation of  $V_k(S)$ . The cost for  $V_k(S)$  is  $O(L \cdot \log(topk \cdot L))$  according to Lemma 1, since the maximum size of sampling set is  $topk$ . Therefore, in the worst case the cost of the method is  $O(topk \cdot N \cdot \log(L \cdot topk) + N \cdot \log(N))$ , which results in  $O(N \cdot \log(L \cdot topk) + N \cdot \log N)$  if  $topk \ll N$ . In the best case, the break of line 11 of the algorithm can stop the intrinsic loop in two ( $O(1)$ ) instead of  $N$  steps ( $O(N)$ ). In this case, the cost is  $\Omega(N \cdot \log(N) + topk \cdot L \cdot \log(L \cdot topk))$ . ■

## 5 Experimental Study

In this section, we present an experimental study in order to evaluate our approach over real and synthetic TD. In particular, we have used the *Athens trucks* real dataset (available at <http://www.rtreeportal.org>), which consists of 112,300 GPS-tracked positions from 50 trucks transporting concrete in the metropolitan area of Athens,

partitioned in 1100 trajectories. For further experimentation, we have also used synthetic datasets generated by a custom generator based on the popular GSTD [20]. Specifically, this generator produces trajectory datasets based on a given distribution of *spatio-temporal focal points*, to be visited by each trajectory in a specific order. The generated dataset then forms a natural cluster, since all trajectories follow more or less the same behavior. For example, Fig. 8(a) illustrates the 2D projection of a cluster generated with the above generator, using points 1 to 5 as focal points. The generator also allows adjusting the speed of each moving point (which may follow random or normal distribution), and also the temporal periods between sampled points (e.g., temporal gap between two sampled positions). Finally, by choosing focal points of varying distributions one may produce parts of a cluster with varying density (e.g. in Fig. 8(a) motions from focal point 1 to point 2 are sparser than those from point 4 to point 5). Using this methodology, we produced four synthetic clusters of trajectories, called  $C_1$  to  $C_4$ , whose 3D visualization is presented in Fig. 8(b). Each cluster contains 50 trajectories, while each trajectory has 50 segments in average. It is obvious that the produced clusters were produced by mixing the focal points, and as such hiding the latent mobility patterns. Moreover, we produced three clusters, called  $C_2^{40}$ ,  $C_3^{30}$ ,  $C_4^{20}$ , by randomly removing trajectories from the original clusters. For example,  $C_2^{40}$  has been produced by removing 10 trajectories from  $C_2$ . Finally we concatenate clusters (i.e.  $C_1C_2$ ,  $C_1C_2^{40}$ ,  $C_1C_2^{40}C_3^{30}$ ,  $C_1C_2^{40}C_3^{30}C_4^{20}$ ) as such producing datasets having various numbers of clusters with diverse density.



**Fig. 8.** Synthetic data: (a) cluster  $C_1$  in 2D, (b) 4 clusters in 3D

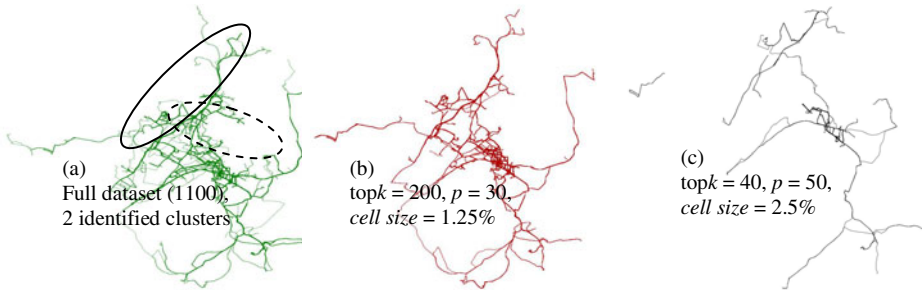
The parameters introduced in our approach are: (a) those regarding the approximate representation of the trajectories (i.e.,  $p$  and  $G$ ), and which mainly affect the execution time of our approach as trajectories are initially transformed to vectors in various spatio-temporal granularities; (b) the control parameter  $0 \leq \sigma \leq 1/2$  that in our experiments was set to 0.1, which corresponds to a smooth voting influence; (c) the thresholds  $\Delta v$  and  $\Delta d$  of *SyTra* algorithm that in our experiments were both set to 0.1, which corresponds to 10% relative difference to *representativeness* and  $18^\circ$  directional deviation, respectively; (d) the *topk* parameter.

The experiments were run on a PC with Intel Core Duo at 2.53 GHz, 4 GB RAM and 240 GB hard disk. We implemented the proposed algorithms using C++.

## 5.1 Experimenting with Real Data

In this section, we evaluate the effectiveness and efficiency of our approach in the *Athens trucks* TD, which is visualized in Fig. 9(a) and where the majority of motions are around two dense areas (see the two ellipsoids in Fig. 9(a)).

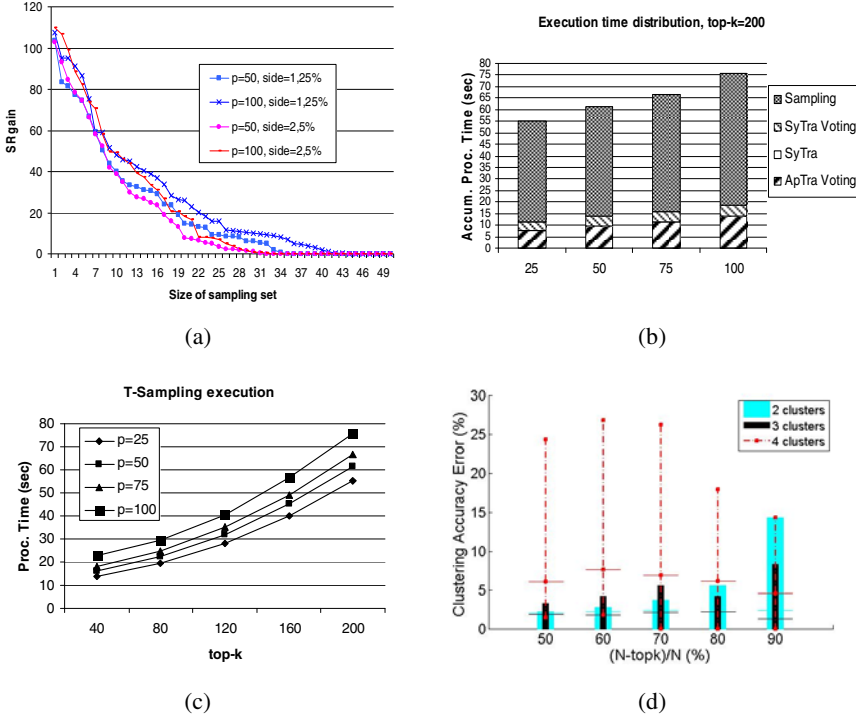
In our first experiment, we sample the TD scaling the  $\text{top}k$  parameter, and we test the ability of our approach to capture the sketch of the whole TD by visualizing the sampled trajectories. Fig. 9(b) and (c) prove that our approach succeeds to preserve this sketch with various diverse granularities of approximation. The *cell size* is shown as percentage of the size of the total space.



**Fig. 9.** Visualization of Athens trucks TD scaling top-k

Note that Fig. 9(b) with  $\text{top}k$  set to less than 20% of the TD cardinality captures almost completely the space, while with less than 4% Fig. 9(c) leads to the same comprehension (not only confining the main two patterns but also the behavior in the non-dense areas). Actually, this result is in accordance to our second experiment where we try to automatically identify the ‘optimal’  $\text{top}k$  by scaling the size of the sampling size and computing the  $SR_{\text{gain}}$  which expresses the gain of including the  $i$ -th trajectory of the TD in the sampling set. Fig. 10(a) clearly depicts that a  $\text{top}k$  value between 35 and 40 (where  $SR_{\text{gain}}$  becomes very low) captures the most representative portions of the TD. Another conclusion is that the increase of dimensionality  $p$  slightly increases the ‘optimal’  $\text{top}k$ , which is expected as the level of detail gets finer.

The second set of experiments is about the execution time of the various steps in our approach. Fig. 10(b) presents the accumulative processing time scaling the dimensionality  $p$ , and setting the cell size to 2.5%, and  $\text{top}k$  to 200 (i.e., we choose the maximum values used in the first experiment). In detail, we compute the processing time for each of the main steps presented in Fig. 4, namely the voting method applied in *ApTra* (line 2), the computation of *SyTra* (line 3), the re-application of the voting scheme to *SyTra* (line 4) and the actual sampling (line 6). The conclusions that can be drawn are: a) the processing time of every step is linear with dimensionality  $p$ ; b) the computation of *SyTra* is extremely fast, negligible to the overall time, though extremely important as it relaxes the initially static dimensionality constraint discovering maximal mobility patterns; c) the processing time of *SyTra* voting is significantly lower than that of *ApTra* (due to merging), while both appear to be affordable (a few seconds are required even without index support). Fig. 10(c) presents



**Fig. 10.** (a) SRgain scaling the sampling set size, (b) and (c) T-Sampling processing time, (d) Clustering accuracy error vs.  $\text{topk}$

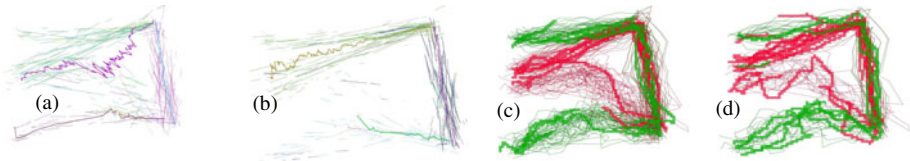
the overall execution time for *T-Sampling*, where a smooth superlinear behavior of the algorithm appears, resulting in a few tens of seconds even for very large dimensionality and  $\text{topk}$  values.

## 5.2 Experimenting with Synthetic Data

The purpose of experimenting with synthetic data is to assess the ability of the proposed methodology to preserve patterns extracted by trajectory data mining algorithms. In our first experiment, we use a state-of-the-art trajectory clustering algorithm [5], and we measure the effect of sampling in clustering. We sampled the  $C_1C_2^{40}, C_1C_2^{40}C_3^{30}, C_1C_2^{40}C_3^{30}C_4^{20}$  datasets including 2, 3, and 4 clusters respectively, scaling the  $\text{topk}$  from 50% to 10% of the cardinality of each dataset (i.e. the reduction after sampling was up to 90%). We measure the clustering accuracy error (i.e. the difference in the clustering accuracy before and after the sampling) for various granularity levels, ranging  $p$  from 10 to 50 (step 10), and *cell size* equal to 0.15%, 0.25%, 0.5%, 0.75% and 1% of total space. As such, for each dataset and for each  $\text{topk}$  we performed  $5 \times 5 = 25$  experiments. Fig. 10(d) illustrates the range of values, the minimum, maximum, and average (i.e., the horizontal in-between crossing segment)

clustering accuracy error for each set of experiments, where it is clear that the error introduced in the clustering patterns is low (around 5%). We would like to note that the maximum error is introduced only in one out of 25 experiments, when  $p = 50$  and  $cell\ size = 0.15\%$ .

In the second experiment, we employ two algorithms that extract the so-called representative or centroid of a trajectory cluster, namely TRACLUS [12] and CenTra [19], respectively. TRACLUS identifies local, more or less linear clusters of segments of trajectories without using the temporal information, while CenTra discovers clusters of whole symbolic trajectories. As such, these two approaches are typical examples of two different clustering techniques having as output different mobility patterns. The idea of the experiment is to evaluate whether the two techniques capture more or less similar mobility patterns when applied before and after sampling. For this purpose, we use the  $C_1C_2$  dataset from which we sample the  $topk = 50\%$  of the cardinality of each dataset. The visualization of the clusters and their mobility patterns in Fig. 11 clearly shows the resemblance of the resulted patterns, before and after sampling. We repeated the experiment several times tuning the parameters of these algorithms and with various sampling sizes and the conclusion remained the same.



**Fig. 11.** TRACLUS [12] (a-b) and CenTra [19] (c-d) results before and after sampling

## 6 Conclusion and Future Work

In this work, we proposed a novel solution to the challenging problem of trajectory sampling, where the challenge is to build a sample by selecting *representatives* among a large set of trajectories in an unsupervised way for general purpose. To the best of our knowledge, there is no related work that addresses this problem apart from explorative, supervised by the user, approaches.

In the future, we plan to add a least enlargement criterion of the merged directed regions to the process of *SyTra* algorithm (as in [1]), so as to tight our implementation with an R-tree like access method for efficiency purposes. Moreover, we plan to evaluate our approach on other types of mobility patterns, such as T-patterns [7] and flock patterns [9], and we will try to adapt it for the purpose of outlier detection [13].

**Acknowledgements.** Research partially supported by the FP7 ICT/FET Project MODAP (Mobility, Data Mining, and Privacy) funded by the European Union. URL: [www.modap.org](http://www.modap.org).

## References

1. Anagnostopoulos, A., Vlachos, M., Hadjieleftheriou, M., Keogh, E., Yu, P.S.: Global distance-based segmentation of trajectories. In: Proc. of KDD (2006)
2. Andrienko, G., Andrienko, N., Wrobel, S.: Visual Analytics Tools for Analysis of Movement Data. *ACM SIGKDD Explorations* 9(2) (2007)
3. Andrienko, G., Andrienko, N., Rinzivillo, S., Nanni, M., Pedreschi, D.: A visual analytics toolkit for cluster-based classification of mobility data. In: Mamoulis, N., Seidl, T., Pedersen, T.B., Torp, K., Assent, I. (eds.) *Advances in Spatial and Temporal Databases. LNCS*, vol. 5644, pp. 432–435. Springer, Heidelberg (2009)
4. Andrienko, G., Andrienko, N., Rinzivillo, S., Nanni, M., Pedreschi, D., Giannotti, F.: Interactive visual clustering of large collections of trajectories. In: Proc of VAST (2009)
5. Gaffney, S., Smyth, P.: Trajectory Clustering with Mixtures of Regression Models. In: Proc. of SIGKDD (1999)
6. Frentzos, E., Gratsias, K., Pelekis, N., Theodoridis, Y.: Algorithms for Nearest Neighbor Search on Moving Object Trajectories. *Geoinformatica* 11, 159–193 (2007)
7. Giannotti, F., Nanni, M., Pedreschi, D., Pinelli, F.: Trajectory Pattern Mining. In: Proc. of SIGKDD (2007)
8. Giannotti, F., Pedreschi, D.: *Mobility, Data Mining and Privacy, Geographic Knowledge Discovery*. Springer, Heidelberg (2008)
9. Gudmundsson, J., Kreveld, M.J., Speckmann, B.: Efficient detection of patterns in 2d trajectories of moving points. *GeoInformatica* 11(2), 195–215 (2007)
10. Kalnis, P., Mamoulis, N., Bakiras, S.: On discovering moving clusters in spatio-temporal data. In: Bauzer Medeiros, C., Egenhofer, M.J., Bertino, E. (eds.) *SSTD 2005. LNCS*, vol. 3633, pp. 364–381. Springer, Heidelberg (2005)
11. Kollios, G., Gunopulos, D., Koudas, N., Berchtold, S.: Efficient biased sampling for approximate clustering and outlier detection in large datasets. *TKDE* 15(5) (2003)
12. Lee, J.-G., Han, J., Whang, K.-Y.: Trajectory clustering: a partition-and-group framework. In: Proc. of SIGMOD (2007)
13. Lee, J.-G., Han, J., Li, X.: Trajectory Outlier Detection: A Partition-and-Detect Framework. In: Proc. of ICDE (2008)
14. Li, Y., Han, J., Yang, J.: Clustering moving objects. In: Proc. of KDD (2004)
15. Nanni, M., Pedreschi, D.: Time-focused clustering of trajectories of moving objects. *Journal of Intelligent Information Systems* 27(3) (2006)
16. Nanopoulos, A., Theodoridis, Y., Manolopoulos, Y.: Indexed-based density biased sampling for clustering applications. *Data and Knowledge Engineering* 57(1), 37–63 (2006)
17. Palmer, R., Faloutsos, C.: Density biased sampling: An improved method for data mining and clustering. In: Proc of SIGMOD (2000)
18. Panagiotakis, C., Pelekis, N., Kopanakis, I.: Trajectory voting and classification based on spatiotemporal similarity in moving object databases. In: Adams, N.M., Robardet, C., Siebes, A., Boulicaut, J.-F. (eds.) *IDA 2009. LNCS*, vol. 5772, pp. 131–142. Springer, Heidelberg (2009)
19. Pelekis, N., Kopanakis, I., Kotsifakos, E., Frentzos, E., Theodoridis, Y.: Clustering Trajectories of Moving Objects in an Uncertain World. In: Proc. of ICDM (2009)
20. Theodoridis, Y., Silva, J.R.O., Nascimento, M.A.: On the Generation of Spatiotemporal Datasets. In: Güting, R.H., Papadias, D., Lochovsky, F.H. (eds.) *SSD 1999. LNCS*, vol. 1651, p. 147. Springer, Heidelberg (1999)
21. Yuan, J., Bo, L., Wang, K., Yu, T.: Adaptive spherical gaussian kernel in sparse bayesian learning framework for nonlinear regression. *Expert Syst. Appl.* 36(2) (2009)

# Fast Extraction of Locally Optimal Patterns Based on Consistent Pattern Function Variations

Frédéric Pennerath

Supélec

2 rue Edouard Belin

F-57070 Metz, France

`frederic.pennerath@supélec.fr`

**Abstract.** This article introduces the problem of searching locally optimal patterns within a set of patterns constrained by some anti-monotonic predicate: given some pattern scoring function, a locally optimal pattern has a maximal (or minimal) score locally among neighboring patterns. Some instances of this problem have produced patterns of interest in the framework of knowledge discovery since locally optimal patterns extracted from datasets are very few, informative and non-redundant compared to other pattern families derived from frequent patterns. This article then introduces the concept of variation consistency to characterize pattern functions and uses this notion to propose GALLOP, an algorithm that outperforms existing algorithms to extract locally optimal itemsets. Finally it shows how GALLOP can generically be applied to two classes of scoring functions useful in binary classification or clustering pattern mining problems.

## 1 Introduction

*Pattern mining* consists in searching in some dataset relevant patterns in relation to some knowledge extraction problem. Formally, a *family of patterns* may be modeled as any partially ordered set  $(\mathcal{P}, \leq_{\mathcal{P}})$  and a *dataset* as a multiset  $\mathcal{D} \subseteq \mathcal{P}$  of patterns representing objects or observations. Then a pattern  $P$  is said to *describe* or *cover* a datum  $d \in \mathcal{D}$  if  $P \leq_{\mathcal{P}} d$ . When objects are described by a set  $\mathcal{I}$  of *items*, patterns are subsets of  $\mathcal{I}$  called *itemsets*, ordered by set inclusion  $\leq_{\mathcal{P}} = \subseteq$ . Whatever their type (itemsets, sequences, graphs, ...), datasets are intended to be analyzed by experts of the application domain in the hope of revealing some new pieces of knowledge. It is thus essential that these patterns are simultaneously characteristics of data, application-relevant, non-redundant, and as few as possible to make the analysis of these patterns practicable.

With respect to these requirements, many pattern mining methods generate too many patterns sharing too much similarity so that in practice, a study of these patterns is an annoying or even impossible task. This is a well-known problem with *frequent patterns* [1], i.e patterns  $P$  such that the proportion of data covered by  $P$  in dataset  $\mathcal{D}$ , called *relative frequency*  $\sigma(P)$ , is not less than some threshold  $\sigma_0 \in [0, 1]$ . But this problem also applies to many other pattern

families like emerging patterns for classification [2], or even condensed representations of frequent patterns like frequent closed patterns [3] or frequent free patterns [4] that are not much fewer than frequent patterns in many applications. A postprocessing step is thus required to select a subset of these patterns. It generally consists in computing for each pattern a non-monotonic score combining frequency with other pattern characteristics like length, classification measures, etc. The list of patterns is then sorted in descending order of score and only the few first hundred patterns of this list (also called top-k patterns [5]) are actually analyzed by experts. However this approach does not address the problem of redundancy: as similar patterns are likely to have similar scores, the top ranked patterns are likely to be considered by experts as clones of the same pattern (see examples of section 2.1 in [6]).

The family of *Most Informative Patterns*, or MIPs, has been introduced in [6] to address previous requirements similarly to pattern teams [7] and other global models like [8,9]. MIPs are defined as patterns maximizing locally some scoring function, that is, patterns whose score is not smaller than scores of neighboring patterns. Neighbors of a pattern  $P$  are defined as the set of *immediate predecessors and successors* of  $P$ , i.e. the set of every pattern  $P'$  such that there exists no other pattern included in between  $P$  and  $P'$ . MIPs are interesting for two reasons: first some scoring function estimates the value of a pattern specific to the application, second the local maximum criterion removes pattern redundancy and drastically reduces the number of selected patterns. The way this criterion removes pattern redundancy corresponds to the way experts are likely to select non-redundant patterns within the list of patterns sorted by descending order of score (see section 2.1 in [6]). The associated data-mining problem then consists in extracting every frequent MIP from a dataset. However the brute-force algorithm to extract frequent MIPs requires much more processing time than the extraction of frequent patterns as i) contrary to pattern frequency, scoring functions are not assumed to have specific properties enabling efficient pruning strategies, ii) while mining algorithms generate every frequent pattern only once, MIP extraction potentially requires to generate every pair of frequent neighboring patterns. A faster and more scalable extraction algorithm has been proposed in [6] but its performance is still two orders of magnitude slower than best frequent pattern mining algorithms like FP-growth [10]. This gap of performance prevents a complete extraction of MIPs for low frequency thresholds.

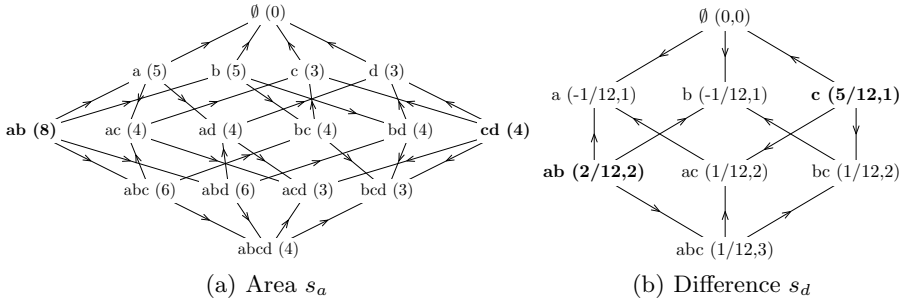
While the MIP model [6] considers some specific class of scoring functions whose properties ensure MIPs are “clustering patterns”, i.e. descriptive of substantial fractions of datasets, the key idea of using the local maximum criterion to remove redundancy may serve other purposes. For this reason, the present article considers the more general problem of extracting *Locally Optimal Patterns* relatively to any pattern scoring function. The main contribution of this paper is the introduction of an algorithm called GALLOP (for Generic ALgorithm to extract Locally Optimal Patterns) that outperforms with typically one order of magnitude prior algorithm of [6] to extract MIPs when patterns are itemsets. Moreover the article shows GALLOP’s performance generalizes to another type



of scoring function suiting binary classification problems. More fundamentally, this article contributes to provide useful concepts to characterize pattern scoring functions in the same way real analysis in mathematics provides useful concepts to characterize real functions. In particular this article introduces concepts of *consistent variation* and *dominance influence*, on which GALLOP’s heuristics are based. The rest of this paper is structured as follows: section 2 formally defines Locally Optimal Patterns, section 3 introduces GALLOP and its heuristics based on both aforementioned concepts, section 4 analyzes results of tests run on reference datasets and section 5 concludes.

## 2 Locally Optimal Patterns

For the sake of generality, no restriction is made at this stage on the type of considered patterns (itemsets, sequences, graphs...). Therefore let a *pattern space*  $\mathcal{P}$  be a set of patterns ordered by some partial ordering relation  $\leq_{\mathcal{P}}$ . A *pattern scoring function* is any function  $s : (\mathcal{P}, \leq_{\mathcal{P}}) \rightarrow (\mathbb{S}, \leq_{\mathbb{S}})$  mapping a pattern  $P$  to a score  $s(P)$ , where scores are elements of any set  $\mathbb{S}$  ordered by some partial or total ordering relation  $\leq_{\mathbb{S}}$ . A simple example of scoring function is the *area function*  $s_a$  that maps an itemset  $P$  to a real number called *area*, that is, the product  $s_a(P) = |P| \cdot \sigma(P) \cdot |\mathcal{D}|$  of pattern length  $|P|$  and absolute frequency  $\sigma(P) \cdot |\mathcal{D}|$  of  $P$  in some dataset. This function is useful in clustering problems as it privileges descriptive patterns representative of large fractions of datasets. This scoring function is illustrated on Fig. 1(a) by the lattice of itemsets made of items  $a, b, c$ , and  $d$ , ordered by inclusion and scored with their area. The



**Fig. 1.** Itemset lattices with score and dominance relation (an arrow  $P_1 \rightarrow P_2$  means pattern  $P_1$  dominates pattern  $P_2$ ) for the area function (a) and the extended difference function (b). Locally optimal patterns appear in bold.

area of pattern  $P$  is computed from frequency of  $P$  in a dataset made of seven objects, whose descriptions are respectively  $a, b, ab, cd, abc, abd$ , and  $abcd$ . For instance, score of pattern  $ac$  is  $s_a(ac) = |\{a, c\}| \times \sigma(ac) \times 7 = 2 \times 2 = 4$  since  $ac$  is a subset of data  $abc$  and  $abcd$ , and thus  $\sigma(ac) = 2/7$ .

Another example of scoring function is the *difference function*  $s_d$  privileging class discriminating patterns extracted from binary classification oriented data. This function maps pattern  $P$  to a real number  $s_d(P) = \sigma_+(P) - \sigma_-(P)$  where  $\sigma_+(P)$  and  $\sigma_-(P)$  respectively denote relative frequencies of  $P$  in datasets of positive and negative examples of some target concept. Length of pattern  $P$  might optionally be appended to  $s_d(P)$ , so that scores are vectors  $(s_d(P), |P|)$  ordered by lexicographic order (i.e. second dimensions of two scores are compared only if first dimensions are found equal). This second dimension is a refinement to privilege  $\leq_{\mathcal{P}}$ -maximal patterns within equivalence classes of patterns covering the same sets of positive and negative examples. Lattice on Fig. 1(b) illustrates this *extended difference function* assuming positive and negative examples are respectively data with and without item  $d$  in the aforementioned dataset. For instance, score of pattern  $ac$  is  $s_d(ac) = (\sigma_+(ac) - \sigma_-(ac), |\{a, c\}|) = (1/3 - 1/4, 2)$ . The length dimension allows to distinguish  $abc$  of length 3 from  $ac$  and  $bc$  of length 2 while all three patterns cover the same positive (i.e.  $abcd$ ) and negative (i.e.  $abc$ ) data and thus have the same frequency difference  $1/3 - 1/4$ .

The general objective that is pursued by the author is to get better insights about how any given pattern function  $s$  “behaves” in the pattern space on which  $s$  is defined. This article provides first concepts and methods to address this question by searching for patterns maximizing locally function  $s$  within their neighboring patterns. Two patterns are hereafter *neighbors* if one is the *immediate successor* of the other. A pattern  $P'$  is an *immediate successor* of a pattern  $P$  (denoted  $P \prec_{\mathcal{P}} P'$ ) if  $P <_{\mathcal{P}} P'$  and no pattern  $P''$  exists such that  $P <_{\mathcal{P}} P'' <_{\mathcal{P}} P'$ . Pattern  $P$  is then an *immediate predecessor* of  $P'$ . Two item-sets  $P_1$  and  $P_2$  are thus neighbors if they differ with one item only, or equivalently, if their edit distance hereafter defined as the cardinality of their symmetric difference  $d(P_1, P_2) = |P_1 \setminus P_2| + |P_2 \setminus P_1|$  is equal to one.

**Definition 1.** *Given a scoring function  $s : \mathcal{P} \rightarrow (\mathbb{S}, \leq_{\mathbb{S}})$ , a pattern  $p_1 \in \mathcal{P}$  is said to dominate pattern  $p_2 \in \mathcal{P}$  if and only if  $p_1$  and  $p_2$  are neighbors and  $s(p_1) >_{\mathbb{S}} s(p_2)$ . A locally optimal pattern or LOP is a pattern that is not dominated by any pattern. Then given an anti-monotonic predicate  $p$  defined over  $\mathcal{P}$ , a pattern is said valid if it satisfies  $p$ . The problem of the extraction of valid locally optimal patterns consists in extracting every optimal pattern relatively to  $s$  that is valid relatively to  $p$ .*

Figure 1 represents dominance by orienting lattice edges: an arc from pattern  $P_1$  to pattern  $P_2$  means  $P_1$  dominates  $P_2$ . Locally optimal patterns, called *optimal* patterns for short, are patterns that are not pointed by any arc like patterns  $ab$  and  $cd$  on Fig. 1(a). The anti-monotonic predicate  $p$  is introduced to control the number of patterns to process as pattern spaces are generally very large or even infinite sets. In pattern mining applications, valid patterns are likely to be frequent patterns. Because  $p$  is a secondary parameter of the problem, it is important that the range of the dominance relation be not limited only to valid patterns but to every neighbor of every valid pattern (including these that are not valid). Otherwise the locally optimal character of valid patterns would depend on the arbitrary choice of predicate  $p$ , which is a negative side effect. For

instance on Fig. 1(a), pattern  $c$  of frequency 3 is not optimal as it is dominated by pattern  $ac$  of frequency 2. However if one arbitrarily sets threshold  $\sigma_0 = 3$ , pattern  $c$  appears not dominated by any other frequent pattern and could be mistakenly believed optimal.

### 3 Locally Optimal Pattern Extraction

LOP extraction algorithms have to proceed unit tasks called *pattern comparisons*, consisting in comparing scores of two neighboring patterns and discarding the dominated neighbor (if any) from the output set. LOP extraction is a problem of higher time and space complexities than a simple enumeration of patterns as used in frequent pattern mining, since the number of pattern comparisons may reach the number of pairs of neighboring patterns (i.e. the number of edges in the diagram of order  $(\mathcal{P}, \leq_{\mathcal{P}})$ ) instead of the number of patterns (i.e. the number of vertices in the order diagram).

#### 3.1 Existing Algorithms

The brute-force extraction of LOPs described in [6] to extract frequent MIPs, consists in an exhaustive depth first order exploration of the pattern space starting from the empty pattern. This approach called *direct extraction* in [6] is clearly not scalable as it requires to memorize every investigated pattern (with its score and a selection flag) and it is also very slow for several reasons: in particular, the method systematically generates every invalid (e.g non-frequent) pattern having at least one valid predecessor. This set, called hereafter the *disjunctive negative border*, is in many applications much larger than the set of valid patterns itself and its generation requires a lot of processing time [6].

A better solution is proposed in [6] based on a level-wise filtering algorithm: in a first step, the method splits valid (e.g frequent) patterns into levels  $(L_n)_{n \geq 0}$  of patterns of size  $n$ , so that a pattern  $P$  in level  $L_n$  is provided to the second step, only if  $P$  is not dominated by any valid neighbor found in levels  $L_{n-1}$  or  $L_{n+1}$ . This way the second step, called *postfiltering*, only has to consider a small number of patterns called *candidates* to compare with their successors in the disjunctive negative border. The level-wise method is considerably faster than the direct extraction as it does not generate any pattern of the disjunctive negative border during the first step and only a small fraction of it during the second step. The method is also more scalable as it only stores one level of frequent patterns in memory at a time.

However, the method systematically compares every pair of valid neighboring patterns. Moreover it requires to split input patterns produced by fastest frequent pattern mining algorithms like FP-Growth into as many files as levels, implying many slow parallel file system accesses. In order to further reduce the gap between times to extract the list of frequent patterns and to extract optimal patterns from them, GALLOP has been designed with two goals: first, GALLOP avoids costly IO access times due to the reordering of many input patterns in

levels by processing sequentially the input pattern file in only one pass, while at the same time reducing memory needs. Second GALLOP reduces the number of useless pattern comparisons that do not discard any pattern from the candidate set. The next two subsections detail how GALLOP addresses each of these goals.

### 3.2 GALLOP’s Pattern Sequential Processing

GALLOP retains the previous idea of a two-step processing to avoid the complete generation of the overlarge disjunctive negative border. Indeed for the full extraction process to be generic, it has been broken down in four successive steps:

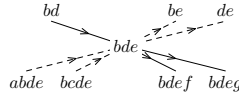
1. The **mining step** first enumerates every valid pattern relatively to some predicate  $p$ , like for instance computing every frequent pattern with its frequency.
2. The **scoring step** sequentially processes valid patterns to score each of them relatively to some function  $s$ .
3. In the **filtering step**, GALLOP extracts candidates from scored patterns, later called *input patterns*.
4. The **postfiltering step** compares scores of candidates with those of their successors in order to determine which candidates are optimal.

As a consequence, GALLOP is a generic algorithm that only compares pattern scores and thus does not depend on  $s$  and  $p$ . Nevertheless GALLOP depends on the scoring order  $(\$, \leq_{\$})$ . In practice the current implementation assumes scores are real vectors ordered by either lexicographic or product ordering relation. As already stated, one objective of GALLOP is to process the very large input pattern file in one single pass while remaining complete. Since the fastest frequent pattern mining algorithms like FP-growth used in the first step mentioned above enumerate itemsets in lexicographic order (assuming some implementation-dependent ordering of items), GALLOP processes input patterns in the same enumeration order thanks to a recursive procedure.

**Definition 2.** A pattern enumeration is formally defined as a linear ordering relation  $\triangleleft$  called precedence relation defined over the set of patterns, so that  $P_1 \triangleleft P_2$  means “ $P_1$  is enumerated before  $P_2$ ”.  $P_1$  then precedes  $P_2$  or conversely  $P_2$  succeeds  $P_1$ . A lexicographic enumeration of itemsets has a precedence relation equal to some lexicographic ordering of itemsets :  $(i_1, \dots, i_n) \triangleleft (i'_1, \dots, i'_n)$  if there exists a subscript  $j \geq 0$  such that for all  $k \leq j$ ,  $i_k = i'_k$  and either  $j = n$  and  $n < n'$  or  $i_{j+1} <_{\mathcal{I}} i'_{j+1}$  where  $<_{\mathcal{I}}$  is an arbitrary linear ordering of items.

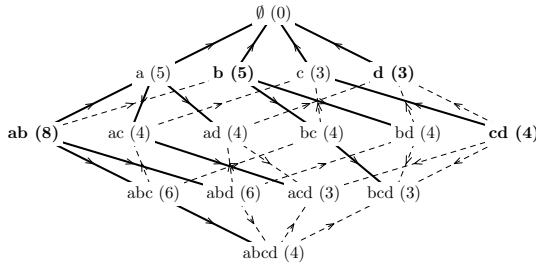
Such enumeration may be efficiently implemented by extending recursively every pattern  $P = (i_1, \dots, i_n)$  to pattern  $P' = (i_1, \dots, i_n, i_{n+1})$  by appending the smallest item  $i_{n+1} >_{\mathcal{I}} i_n$  not already appended to  $P$ . Pattern  $P'$  is one of the *children* of  $P$  and  $P$  is the *parent* of  $P'$ . In such enumeration, two neighboring patterns are said *lineal neighbors* if one pattern is the parent of the other. Otherwise they are said *transverse neighbors*. Thus every pattern  $P$  has up to four classes of neighbors as illustrated on Fig. 2: one lineal predecessor (or parent)

and several transverse successors that all precede  $P$ , and several lineal successors and transverse predecessors that all succeed  $P$ . This distinction between



**Fig. 2.** Lineal and transverse neighbors of itemset  $bde$  wrt lexicographic enumeration of itemsets made of items from  $a$  to  $g$  ordered alphabetically:  $bd$  (parent),  $bdef$  and  $bdeg$  (children) are lineal neighbors.  $be$ ,  $de$ ,  $abde$  and  $bcde$  are transverse neighbors. An arrow from pattern  $P_1$  to pattern  $P_2$  means  $P_1$  precedes  $P_2$ .

lineal and transverse neighbors is important as a sequential processing of patterns enumerated in some lexicographic ordering may compare current pattern  $P$  with its lineal neighbors at almost no computational cost since the score of  $P$  can be passed to recursive processing of its children. In contrast, comparison of  $P$  with a transverse predecessor  $P'$  raises a much more difficult problem as the number of patterns succeeding  $P$  and preceding  $P'$  may be arbitrarily large. However comparisons with transverse neighbors are essential to screen efficiently candidates as later shown in section 4. Figure 3 illustrates this importance: Only



**Fig. 3.** Itemset lattice of Fig. 1(a) with lineal (thick) and transverse (dashed) neighboring. Bold patterns (i.e  $ab$ ,  $b$ ,  $cd$ , and  $d$ ) are not dominated by any lineal neighbor.

comparisons with transverse neighbors are able to eliminate non-optimal candidates  $b$  and  $d$  among the four patterns that are not lineally dominated.

In order to compute efficiently comparisons with transverse neighbors, the sequential process of patterns enumerated in lexicographic order presents an interesting possibility: given some currently processed pattern  $P$ , the algorithm may “postpone” comparisons of  $P$  with its transverse predecessors until these predecessors (which succeed  $P$ ) are in turn processed. This postponing can be implemented by inserting predecessors of  $P$  into a priority queue (i.e. binary heap) of patterns. The sorting order underlying the queue is the same lexicographic order used to enumerate input patterns, so that the top queue element is always the next pattern to process among all queued patterns. Every entry

in the queue carries the postponing pattern  $P$ , its score  $s(P)$  and a flag to remember if  $P$  has already been found dominated. Then since  $P$  is valid and  $p$  is anti-monotonic, every transverse predecessor  $P'$  of  $P$  is valid and will eventually be processed. When  $P'$  becomes the currently processed pattern, GALLOP may learn whether some transverse successor of  $P'$  like  $P$  has previously postponed a comparison with  $P'$ , by checking whether the top of the queue is  $P'$ . In this case, the top entry of the queue is popped,  $P$  and  $P'$  are compared before their flags are updated accordingly. In practice, since the processing of transverse predecessors of  $P$  occur in some deterministic order, only the next occurring of these predecessors is present in the queue. When this predecessor  $P'$  of  $P$  is processed, the next occurring transverse predecessor  $P''$  of  $P$  is computed from  $P'$  and  $P$ 's entry is pushed down to the position of  $P''$  in the queue. This limits the size of the queue to the number of currently postponing patterns. Figure 4 summarizes the general principle of GALLOP's recursive procedure.

```

function filter(Current pattern P, Value V of P)
  (Integer nBT and priority queue Q are global variables)
  while(Q is not empty)
    Let (Pattern P', Value V') be the top of Q
    if(P' ≠ P) break;
    if(V.score ≤g V'.score), V.lop ← false
    else if(V.score >g V'.score), V'.lop ← false
    Be P'' the next predecessor of V'.pattern succeeding P.
    if(P'' is defined)
1      if((P'',V') is to be postponed), move (P'', V') in Q
    else
      if(V'.lop), output candidate (V'.pattern, V'.score)
      pop Q
    end if
  end while
  (Pattern P', Score S, nBT) ← readNextPattern()
  Be value V' with
  V'.pattern ← P', V'.score ← S, V'.lop ← true
  while(nBT = 0)
    if(V.score ≤g V'.score), V.lop ← false
    else if(V.score >g V'.score), V'.lop ← false
    call filter(P',V')
    nBT ← nBT - 1
  end while
2  if((P, V) is to be postponed),
    Be P'' the first transverse predecessor of P succeeding parent of P
    insert (P'',V) into Q
  end if
end

```

Fig. 4. Sketch of GALLOP's recursive procedure

Since postfiltering compares every candidate with its successors, GALLOP must only guarantee two conditions i) every candidate in the output is not dominated by any predecessor ii) every LOP is not discarded from the output. Therefore, a pattern known to be dominated might not be compared with every of its transverse predecessors. This freedom authorizes different strategies appearing in GALLOP's algorithm on lines 1 and 2, where GALLOP decides whether pattern comparisons must be postponed or not. Determining the best strategy is fundamentally a problem of balancing effort between GALLOP and

postfiltering: in other words, the fewer postponed comparisons, the faster GALLOP, but the many more candidates, and finally the slower postfiltering. This optimal strategy must lie between two extreme strategies:

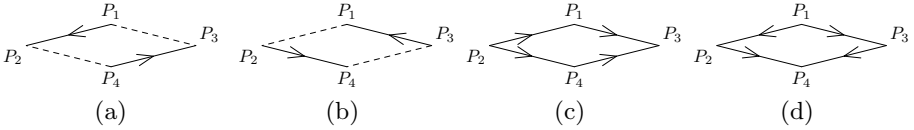
**Maximal effort strategy.** Given current pattern  $P$ , an obvious strategy consists in postponing systematically comparisons of current pattern  $P$  even if  $P$  is already known to be dominated by some pattern preceding  $P$ . Like the level-wise algorithm, this maximal effort strategy guarantees that every pair of valid neighbors has been compared so that GALLOP’s output is reduced to the minimal set of candidates, that is, valid patterns that are not dominated by any other valid pattern. The postfiltering is thus the fastest possible but systematic postponing is likely to require a lot of time and memory.

**Minimal effort strategy.** Conversely minimal effort strategy consists in stopping postponing comparisons with predecessors of current pattern  $P$  as soon as  $P$  appears dominated by some patterns. A pattern dominated by a linear neighbor or a transverse successor is thus never inserted in the queue. This strategy provides the fastest possible version of GALLOP that still guarantees every candidate produced as output is not dominated by any predecessor. However this strategy is “selfish” as its only concern is pattern  $P$ : this strategy does not help to discard transverse predecessors dominated by  $P$ . Consequently this strategy is likely to produce many candidates and to require a long postfiltering.

### 3.3 GALLOP’s Heuristics Based on Variation Consistency

The optimal strategy consists in making only comparisons which discard one of the two compared neighbors from the candidate set, that is, when one pattern is dominated by the other while it was not already known to be dominated by some preceding pattern. Such comparisons are called *useful*, all others are said *useless*. However the optimal strategy is not achievable as the decision that a pattern  $P$  has to be compared with some of its transverse predecessors must be made when  $P$  is currently processed, at a time no transverse predecessors are already known to be dominated. Therefore the processing of  $P$  may only guess, based on some heuristics, which comparisons with  $P$ ’s predecessors are useful and must be postponed in the queue. These heuristics must rely on some expected properties of the scoring function over the pattern space. While scoring functions are not required to have any theoretical property, pattern scores found in practical applications are expected to have some regular distribution, in the same way data points used in regression models are expected to be samples randomly scattered around some piecewise continuous function. The notion of *consistent variation* is hereafter introduced to model regularity existing in variations of pattern functions.

**Definition 3.** A diamond configuration is a set of four patterns  $P_1, P_2, P_3$  and  $P_4$  such that  $P_1 \prec_P P_2 \prec_P P_4$  and  $P_1 \prec_P P_3 \prec_P P_4$  as illustrated on Fig. 5. Given such a diamond configuration and a scoring function  $s$ , variations of  $s$



**Fig. 5.** Diamond configurations: examples of inconsistent (a and b) and consistent (c and d) variations. Arrows give dominance directions.

from  $P_1$  to  $P_2$  and from  $P_3$  to  $P_4$  are inconsistent if  $P_1$  dominates  $P_2$  and  $P_4$  dominates  $P_3$  or if  $P_2$  dominates  $P_1$  and  $P_3$  dominates  $P_4$ .

**Definition 4.** A scoring function has consistent variations within a set  $S$  of patterns if no diamond configuration within  $S$  has some inconsistent variation.

Non-decreasing pattern functions (i.e.  $\forall P_1, \forall P_2, P_1 \leq_{\mathcal{P}} P_2 \Rightarrow s(P_1) \leq_{\mathfrak{S}} s(P_2)$ ) like pattern length, or conversely non-increasing functions like pattern frequency have by definition consistent variations over the whole pattern space. However consistent variation is still statistically true for more complex non-monotonous scoring functions like area or difference scoring functions as shown in Tab. 1. The

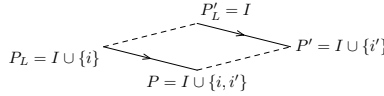
**Table 1.** Variation consistency ratio for different datasets

Dataset	Scoring function	Positive class	Negative class	Thres. $\sigma_0$	Number of diamonds	Consistency Ratio
Mushrooms	area			0.05	132 M	98 %
	diff.	edible	poisonous	0.05	59 M	97 %
Breast-Cancer	area			0.001	3.9 M	85 %
	diff.	cancerous	healthy	0.001	1.6 M	85 %
Vote	area			0.01	10 M	94 %
	diff.	republican	democrat	0.01	4.3 M	95 %
Chess	area			0.5	41 M	97 %
Connect	area			0.8	173 M	98 %

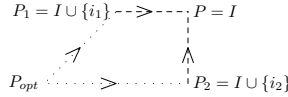
table provides the *consistency ratio* of both scoring functions  $s_a$  and  $s_d$  (when target classes are available) for some UCI datasets. This ratio is computed from a set of frequent patterns: for every possible diamond configuration of frequent patterns, consistency of both possible pairs of variations is tested. The ratio is defined as the number of consistent pairs over the total number of pairs. For every function and dataset (but Breast-Cancer), the ratio is at least 94 %. This observation leads to an alternative strategy called *lazy strategy*.

**Comparison Pruning based on a Lazy Evaluation Strategy.** This strategy consists in making comparisons of current pattern  $P$  with its transverse predecessors only if  $P$  is not dominated by any lineal neighbor. To understand why this strategy is sound, let some current pattern  $P$  be dominated by some lineal neighbor  $P_L$ . Figure 6 illustrates the case  $P_L$  is a predecessor of  $P$ . For every transverse predecessor  $P'$  of  $P$ , let  $i$  be the item such that  $P = P' \cup \{i\}$  and let  $P'_L$  be the itemset such that  $P_L = P'_L \cup \{i\}$ . Then the four patterns builds





**Fig. 6.** Lazy strategy heuristic: if current pattern  $P$  is dominated by a lineal neighbor  $P_L$  (here a predecessor), every transverse predecessor  $P'$  is likely to be dominated by one of its lineal neighbor  $P'_L$ .



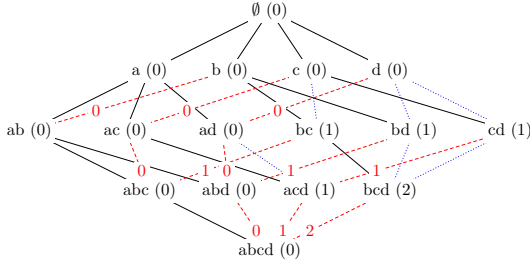
**Fig. 7.** Consistent influence:  $P$  is dominated by two transverse successors  $P_1$  and  $P_2$  because of a common influence of some remote locally optimal pattern  $P_{opt}$ . Dashed and dotted lines represent resp. transverse neighborhood and influence of  $P_{opt}$ .

a diamond where arcs  $P \rightarrow P_L$  and  $P' \rightarrow P'_L$  are parallel and have the same orientation in the order diagram. Assuming variations of the scoring function  $s$  are consistent and since  $P_L$  dominates  $P$ ,  $P'$  is likely to be dominated by  $P'_L$ . As  $P$  is known to be dominated, comparing  $P$  with its transverse predecessors is thus likely to be useless. Implementing the lazy strategy when processing  $P$  is possible by delaying the decision to postpone comparisons with transverse predecessors after processing recursively lineal children of  $P$  (cf line 2 on Fig. 4). At that point,  $P$  is known to be lineally dominated or not.

**More Comparison Pruning based on LOP Dominance Influence.** The concept of variation consistency over current pattern  $P$  and its lineal neighbors leads to define the lazy strategy as an optimized version of the maximal strategy. Similarly the concept of *dominance influence* leads to define an optimized version of the lazy strategy.

**Definition 5.** Given a locally optimal pattern  $P_{opt}$  relatively to some scoring function  $s$  and a set  $\mathcal{E}$  of patterns,  $P_{opt}$  has a dominance influence over  $\mathcal{E}$  if for every pair of neighbors  $\{P, P'\}$  in  $\mathcal{E}$  such that  $P$  is strictly closer to  $P_{opt}$  than  $P'$  (according to the edit distance  $d$  defined in Sect. 2),  $P$  dominates  $P'$ .

The intuition behind this concept is that every local optimal pattern exerts a dominance influence over some surrounding pattern subspace: for example, optimal pattern  $ab$  on Fig. 1(a) exerts a dominance influence over all patterns but  $cd$ ,  $c$ , and  $d$  while optimal pattern  $cd$  only exerts a dominance influence over  $cd$ ,  $c$ ,  $d$ ,  $\emptyset$  and  $bcd$ . This hypothesis is used to prune more useless comparisons in the case current pattern  $P$  is not lineally dominated but is under the dominance influence of some optimal pattern  $P_{opt}$  through a number of transverse successors  $\{P_i, 1 \leq i \leq n\}$  such that  $\forall i, P \prec P_i \subseteq P_{opt}$  and  $P_1 \triangleleft \dots \triangleleft P_n$  as illustrated on Fig. 7. Since  $P$  is not dominated lineally, it is likely that transverse successors  $P_i$  are also not dominated lineally, according to the hypothesis of variation consistency. Therefore the lazy strategy will postpone comparisons of every  $P_i$ .



**Fig. 8.** Shifting technique for potential candidate  $P_{cand} = abcd$ . Red dashed, blue dotted, and black plain edges represent resp. postponed transverse, skipped transverse, and lineal comparisons. Number attached to itemset is its initial shift value, and number attached to edge representing a postponed comparison is the shift value of the edge successor once comparison occurs.

$P$  will thus be compared  $n$  times with each  $P_i$  whereas one comparison with  $P_1$  would have sufficed to discard  $P$  from the candidate set.

The proposed heuristic called *pattern shifting* aims at avoiding the  $n - 1$  useless comparisons with  $P$  by ensuring that for every *potential candidate*  $P_{cand}$  (i.e. a current pattern that has not been found dominated by any neighbor so far, and might potentially have a dominance influence over surrounding patterns), if  $P \subset P_{cand}$ ,  $P$  will be compared with one and only one transverse successor  $P_k$  of  $P$ , such that  $P \subset P_k \subseteq P_{cand}$ . The heuristic expects a single comparison of  $P_k$  will suffice to discard  $P$  according to the dominance influence hypothesis. In practice  $k$  is chosen to be 1, so that pattern  $P_k$  is the first enumerated transverse successor of  $P$  such that  $P_k \subseteq P_{cand}$  and is called *eldest successor*. For instance, if  $P_{cand} = abcd$ , then  $P = ad$  has two transverse successors in the lattice of Fig. 3, that are  $abd$  and  $acd$  and its eldest successor is  $P_1 = abd$ . These eldest successors build a forest structure whose trees are built upward as shown by postponed comparisons (red dashed edges) on Fig. 8. Shifting is an efficient implementation of this forest structure starting from potential candidate  $P_{cand}$ . Given current pattern  $P$ , shifting determines which is the first transverse predecessor  $P'$ , pattern  $P$  should be compared with (i.e should be the eldest successor of). All comparisons with transverse predecessors of  $P$  preceding  $P'$  may thus be ignored. The number of these skipped patterns is called *shift* and initialized to 0. In the optimal case,  $P$  receives a postponed comparison from one unique transverse predecessor which is the eldest successor  $P'$  of  $P$ . The shift of  $P$  is then initialized to the shift of  $P'$ , and every time  $P$  is compared with a new transverse predecessor, its shift is incremented by one. However the shift of  $P$  is sometimes forced to be initialized to 0 when  $P$  appears to be a new potential candidate. This simple algorithm builds upward trees between transverse neighbors starting from potential candidates as illustrated on Fig. 8: assuming  $abcd$  is a potential candidate, its initial shift is set to 0. This shift is incremented when comparing  $abcd$  with  $abd$ ,  $acd$ , and  $bcd$ . Initial shift for  $abd$ ,  $acd$ , and  $bcd$  are

thus resp. 0, 1, and 2. When  $acd$  is processed as the current pattern, its shift is 1 and skips its first transverse predecessor  $ad$ , since  $ad$  is already compared with its eldest successor  $abd$ . When  $bcd$  is processed, its shift value is 2, and since it has only two transverse predecessors  $bd$  and  $cd$ ,  $bcd$  does not postpone any comparison as it is not the eldest successor of any of its transverse predecessors. A consequence of discovering several new potential candidates is that a current pattern  $P$  can be compared with more than one eldest successor, with different interfering shift values. In this case, the most cautious choice is to skip a minimal number of postponed comparisons, by setting the shift of  $P$  to the minimal shift of eldest successors that postponed comparisons with  $P$ .

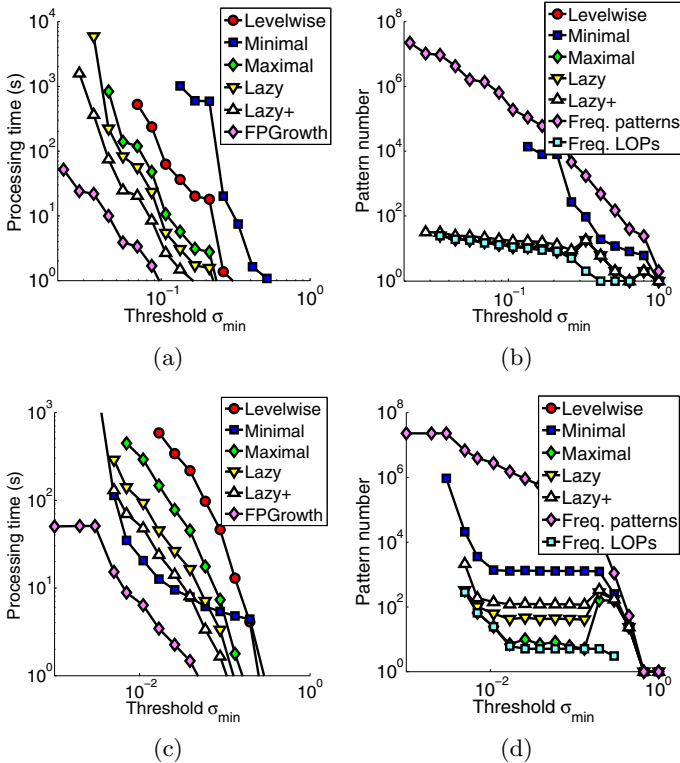
## 4 Tests and Empirical Analysis

Tests on some standard PC (Intel Core2 1.8GHz with 1.5GB RAM) have been performed on reference itemset datasets from the UCI repository in order to compare both processing time and scalability (measured by the maximum number of memorized patterns at a time) of each different algorithm: the prior level-wise algorithm presented in [6] and the four GALLOP’s versions using the maximal, minimal, lazy strategy, and the lazy strategy with shifting (referred as GALLOP-Lazy+). All these algorithms have been used to extract frequent optimal patterns relatively to both area and difference functions. Table 2 summarizes test results. Whatever the dataset and scoring function are, the level-wise algorithm, GALLOP-Maximal, GALLOP-Lazy, and GALLOP-Lazy+ always appear in this order in the list of algorithms sorted in descending order of processing time. GALLOP-Lazy+ outperforms the level-wise processing time with at least one order of magnitude but for the *Connect* dataset. Speed improvements from GALLOP-Maximal to GALLOP-Lazy and from GALLOP-Lazy to GALLOP-Lazy+ attest the soundness of consistent variation and dominance influence heuristics. Since the level-wise algorithm and GALLOP-Maximal compare every pair of frequent neighbors, they always produce the same number of candidates. More surprising is that GALLOP-Lazy, while pruning many more comparisons than GALLOP-Maximal, produces the same number of candidates and thus does not slow down postfiltering: as a consequence, the lazy strategy noticeably reduces processing time and improves scalability of the filtering step without increasing time of the postfiltering step. This is also true about pattern shifting: even if GALLOP-Lazy+ sometimes produces few more candidates than GALLOP-Lazy, the time saved by pruning more comparisons largely balances the small extra time spent in postfiltering. These observations are confirmed for every threshold value as shown on Fig. 9(a) and (b). *Connect* is the only dataset where heuristics do not substantially reduce the total processing time. This is because postfiltering has a time complexity in  $\Theta(|\mathcal{D}| \cdot |C|)$ , proportional to the number of candidates  $|C|$  and to the very large size  $|\mathcal{D}|$  of dataset *Connect*. Since every GALLOP’s version but GALLOP-minimal produces the same 214 candidates and since postfiltering of these candidates is about 500 longer

**Table 2.** Comparison of algorithm performances for different datasets, scoring functions, and frequency thresholds. Every test is described by total processing time (including time for filtering input patterns and postfiltering candidate patterns), ratio of time spent on postfiltering, candidate number and maximal number of memorized patterns. Algorithms are sorted in descending order of processing time. The absence of some algorithm in a test means its processing was too long and aborted.

Dataset	Scoring function	Thres. $\sigma_0$	Frequent patterns	Freq. LOPs	Algorithm	Total time (s)	Postfilt. ratio	LOP candidates	Max. memo. patterns				
Mushrooms 8124 data	area	0.134	110 K	10	Minimal	1010	99 %	14 K	6.7 K				
					Level-wise	36	1 %	11	41 K				
					Maximal	6	5 %	11	40 K				
					Lazy	3	10 %	11	20 K				
					Lazy+	1.5	26 %	17	17 K				
	difference	0.044	4.2 M	19	Maximal	833	0.5 %	19	2 M				
					Lazy	220	0.2 %	19	1 M				
					Lazy+	74	0.6 %	25	0.8 M				
					edible minus poisonous	0.086	328 K	24	Minimal	1950	99 %	27 K	23 K
									Maximal	23	7 %	140	144 K
Lazy	12	13 %	140	72 K									
				Lazy+	6	27 %	140	61 K					
				Lazy	246	2 %	553	1.1 M					
				Lazy+	98	4 %	556	1 M					
Breast-cancer 699 data	area	0.001	297 K	402	Minimal	93	97 %	28 K	15 K				
					Level-wise	76	0.5 %	402	150 K				
					Maximal	13	1 %	402	103 K				
					Lazy	7	3 %	402	50 K				
					Lazy+	3	6 %	430	38 K				
	difference	0.001	147 K	383	Minimal	72	98 %	23 K	9 K				
					Maximal	5	5 %	607	51 K				
					Lazy	3	10 %	607	24 K				
					Lazy+	1.6	17 %	622	18 K				
					cancerous minus healthy	0.001	147 K	383	Minimal	72	98 %	23 K	9 K
Maximal	5	5 %	607	51 K									
Lazy	3	10 %	607	24 K									
				Lazy+	1.6	17 %	622	18 K					
Vote 435 data	area	0.017	1.5 M	6	Level-wise	585	<0.1 %	7	0.5 M				
					Maximal	146	<0.1 %	7	0.7 M				
					Lazy	45	0.1 %	42	0.3 M				
					Lazy+	24	0.4 %	120	0.2 M				
					Minimal	13	36 %	1.3 K	5 K				
	difference	0.005	3.3 M	53	Minimal	1000	95 %	304 K	157 K				
					Maximal	349	0.2 %	2311	1.4 M				
					Lazy	155	0.4 %	2325	0.7 M				
					Lazy+	65	1 %	2415	0.67 M				
					republican minus democrat	0.005	3.3 M	53	Minimal	1000	95 %	304 K	157 K
Maximal	349	0.2 %	2311	1.4 M									
Lazy	155	0.4 %	2325	0.7 M									
				Lazy+	65	1 %	2415	0.67 M					
Chess 3196 data	area	0.585	328 K	1	Minimal	826	99 %	16 K	31 K				
					Level-wise	118	3 %	79	144 K				
					Maximal	26	15 %	79	218 K				
					Lazy	14	27 %	79	110 K				
					Lazy+	11	35 %	79	110 K				
	Connect 67757 data	area	0.907	20 K	0	Minimal	4400	99 %	3300	2.5 K			
						Level-wise	290	98 %	214	10 K			
						Maximal	285	99 %	214	14 K			
						Lazy	284	99 %	214	7 K			
						Lazy+	284	99 %	214	7 K			

than GALLOP running time due to the large dataset, differences of speed between GALLOP’s versions are completely overwhelmed by the same very long postfiltering. The same reasoning explains why GALLOP-Minimal is the slowest algorithm for all tests but one: the poor filtering capability of the algorithm typically produces a hundred to a thousand more candidates than the other methods so that the very long postfiltering overwhelms the very short filtering time. The only exception is the test with the area function and the *Vote* dataset, where GALLOP-Minimal jumps from the last to the first place as show Fig. 9(a) and (c): this surprising result is only possible because *Vote* is a particularly small dataset and that the number of candidates produced by GALLOP-Minimal is relatively small (this is not true anymore for the difference function). However as shown on Fig. 9(c), GALLOP-Minimal only keeps this top position when the number of candidates is kept relatively small, that is, for relatively high frequency threshold.



**Fig. 9.** Processing times for *Mushrooms* (a) and *Vote* (c) using the area function, and candidate numbers bounded by number of frequent patterns and frequent optimal patterns for *Mushrooms* (b) and *Vote* (d). **All axis have a logarithmic scale.** The number of candidates is non-monotonic as a frequent candidate that is dominated by some non-frequent successor  $M$  is removed once the frequent threshold decreases and  $M$  gets frequent.

## 5 Conclusions

This article proposes a generic algorithm GALLOP that outperforms with one order of magnitude previous methods to extract locally optimal itemsets, even if long postfiltering might hide this benefit when datasets are very large and many candidates are generated. More fundamentally this article raises a very general problem that is to find patterns optimizing locally any scoring function within the pattern space. Not only some instances of these patterns show interesting properties in the framework of knowledge discovery but optimal patterns along with notions like variation consistency and dominance influence are believed to be part of a more global research perspective: the development of relevant concepts and tools for representing and studying (scoring) functions defined over ordered sets (of patterns).

## References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: Buneman, P., Jajodia, S. (eds.) Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, pp. 207–216. ACM Press, New York (1993)
2. Dong, G., Li, J.: Efficient mining of emerging patterns: discovering trends and differences. In: KDD '99: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 43–52. ACM, New York (1999)
3. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Efficient mining of association rules using closed itemset lattices. *International Journal of Information Systems* 24(1), 25–46 (1999)
4. Boulicaut, J.F., Bykowski, A., Rigotti, C.: Free-sets: A condensed representation of boolean data for the approximation of frequency queries. *Data Min. Knowl. Discov.* 7(1), 5–22 (2003)
5. Wang, J., Han, J., Lu, Y., Tzvetkov, P.: Tfp: An efficient algorithm for mining top-k frequent closed itemsets. *IEEE Trans. Knowl. Data Eng.* 17(5), 652–664 (2005)
6. Pennerath, F., Napoli, A.: The model of most informative patterns and its application to knowledge extraction from graph databases. In: Buntine, W. L., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS, vol. 5782, pp. 205–220. Springer, Heidelberg (2009)
7. Knobbe, A.J., Ho, E.K.Y.: Pattern teams. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 577–584. Springer, Heidelberg (2006)
8. Siebes, A., Vreeken, J., van Leeuwen, M.: Item sets that compress. In: Ghosh, J., Lambert, D., Skillicorn, D.B., Srivastava, J. (eds.) SDM. SIAM, Philadelphia (2006)
9. Bringmann, B., Zimmermann, A.: One in a million: picking the right patterns. *Knowl. Inf. Syst.* 18(1), 61–81 (2009)
10. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Min. Knowl. Discov.* 8(1), 53–87 (2004)

# Large Margin Learning of Bayesian Classifiers Based on Gaussian Mixture Models\*

Franz Pernkopf and Michael Wohlmayr

Graz University of Technology, Inffeldgasse 16c, A-8010 Graz, Austria  
{pernkopf,michael.wohlmayr}@tugraz.at

**Abstract.** We present a discriminative learning framework for Gaussian mixture models (GMMs) used for classification based on the extended Baum-Welch (EBW) algorithm [1]. We suggest two criteria for discriminative optimization, namely the class conditional likelihood (CL) and the maximization of the margin (MM). In the experiments, we present results for synthetic data, broad phonetic classification, and a remote sensing application. The experiments show that CL-optimized GMMs (CL-GMMs) achieve a lower performance compared to MM-optimized GMMs (MM-GMMs), whereas both discriminative GMMs (DGMMs) perform significantly better than generatively learned GMMs. We also show that the generative discriminatively parameterized GMM classifiers still allow to marginalize over missing features, a case where generative classifiers have an advantage over purely discriminative classifiers such as support vector machines or neural networks.

## 1 Introduction

In statistical learning theory [2], the PAC bound on the expected risk for unseen data depends on the empirical risk on training data and a measure for the generalization ability of the empirical model which is directly related to the Vapnik-Chervonenkis (VC) dimension. One of the most successful discriminative classifiers, namely the support vector machine (SVM) [3], finds a decision boundary which maximizes the margin between samples of distinct classes resulting in good generalization properties of the classifier. In contrast, conventional discriminative training methods relying on the conditional likelihood (CL) optimize only the empirical risk which is suboptimal. Taskar et al. [4] observed that undirected graphical models can be efficiently trained to maximize the margin. More recently, Guo et al. [5] introduced the maximization of the margin to Bayesian networks. Unlike in undirected graphical models, the main difficulty for Bayesian networks is the normalization constraint of the local conditional probabilities. In [5], this constraint is relaxed to obtain a convex optimization

---

\* This work was supported by the Austrian Science Fund (Project number P22488-N23) and (Project number S10604-N13).

problem, whereby conditions on the graph structure are given where the relaxed problem matches the normalized network [6]. In [7], margin optimization has been applied to GMMs, but similar as above, the normalization constraint has been neglected leading to a convex optimization problem. Since then, different margin-based training algorithms have been proposed for HMMs in [8,9] and references therein.

Compared to [5,8], we aim to follow a quite different approach in this paper to maximize the margin in GMM-based classifiers. We keep the sum-to-one constraint which maintains the probabilistic interpretation of the network, e.g. marginalization over missing variables is still possible (as we show in this paper). However, we no longer have a convex optimization problem in general. Convex optimization requires convex loss function, whereas we can also use differentiable non-convex loss functions. Collobert et al. [10] show that the optimization of non-convex loss functions in SVMs can lead to sparse solutions (lower number of support vectors) and accelerated training performance. They conclude that the sacrosanct popularity of convex approaches should not anticipate the exploration of alternative techniques, since they may offer computational advantages. Similar observations are reported in [9].

In this paper, we derive a discriminative training method for GMM-based Bayesian classifiers. The algorithm is based on the EBW parameter re-estimation method [1]. In [11] it is shown that the EBW algorithm resembles the gradient descent algorithm for discriminative GMM optimization using a particular choice of step size in the gradient descent method. Nevertheless, EBW offers an *EM-like* parameter update, whereas the gradient descent method requires additional prudence, e.g. line search or learning rate. We suggest to either optimize the conditional likelihood (CL) or to maximize the margin (MM) [1]. The CL criterion is related to the maximum mutual information (MMI) criterion which is popular in speech processing [12,13]. In [14], EBW has been applied to optimize Gaussian mixture models with respect to CL. However, they neglect to optimize the class prior. In the experiments, we depict the differences of the decision boundary for generatively and discriminatively learned GMMs for classification using synthetic data. Furthermore, we show results for broad phonetic classification [15] and compare discriminative GMM classifiers to SVMs and neural networks (NNs). Moreover, one of the key advantages of generative models over discriminative ones (such as SVMs or NNs) is that it is still possible to marginalize over missing features. We provide empirical results showing that the performance advantage of discriminatively learned GMMs for classification can be maintained for a low number of missing values. This is also shown for a remote sensing application on hyperspectral data.

The paper is organized as follows: In Section 2, we shortly review the Bayesian classifier and generative learning of GMMs, respectively. Additionally, notation is introduced. In Section 3, we derive a discriminative learning method for CL-GMMs based on the EBW algorithm used for classification. Margin-based

---

<sup>1</sup> Both algorithms are implemented in Matlab and can be downloaded at:

[http://www.spsec.tugraz.at/people/franz\\_pernkopf/](http://www.spsec.tugraz.at/people/franz_pernkopf/)



GMM learning is presented in Section 4. We report experimental results on synthetic and real-world data in Section 5. Finally, Section 6 concludes the paper.

## 2 Bayesian Classifier

The Bayesian classifier [16] relies on the Bayes rule to determine the class posterior probability according to

$$p(c|\mathbf{x}^n) = \frac{p(\mathbf{x}^n|c)p(c)}{\sum_{c'=1}^C p(\mathbf{x}^n|c')p(c')}, \quad (1)$$

where  $c \in \{1, \dots, C\}$ , and  $C$  is the number of classes. The posterior probability  $p(c|\mathbf{x}^n)$  models the probability of  $c$  given the feature vector of the  $n^{\text{th}}$  sample  $\mathbf{x}^n$ . We predict the class label using the MAP (maximum posterior) estimate, i.e. the most likely class label  $c^*$  is determined using the class posteriors as

$$c^* = \arg \max_{c \in \{1, \dots, C\}} p(c|\mathbf{x}^n) = \arg \max_{c \in \{1, \dots, C\}} p(\mathbf{x}^n|c)p(c), \quad (2)$$

where the denominator of Eq. (1) can be neglected since it only scales  $p(c|\mathbf{x}^n)$  and does not alter the decision in Eq. (2). This equation is a solution of the Bayesian risk minimization problem with the 0/1-loss function. The term  $p(c)$  is known as class prior distribution. We use GMMs to model the term  $p(\mathbf{x}^n|c)$ , i.e. for each class  $c$  we have a GMM  $p(\mathbf{x}^n|\Theta_c)$ . A Gaussian mixture model  $p(\mathbf{x}^n|\Theta_c)$  is the weighted sum of  $M > 1$  Gaussian components  $\mathcal{N}(\mathbf{x}^n|\theta_c^m)$  in  $\mathbb{R}^d$ ,  $p(\mathbf{x}^n|\Theta_c) = \sum_{m=1}^M \alpha_c^m \mathcal{N}(\mathbf{x}^n|\theta_c^m)$ , where  $\alpha_c^m$  corresponds to the weight of each component  $m \in \{1, \dots, M\}$ . These weights are constrained to be positive  $\alpha_c^m \geq 0$  and  $\sum_{m=1}^M \alpha_c^m = 1$ . The GMM is specified by the set of parameters  $\Theta_c = \{\alpha_c^1, \alpha_c^2, \dots, \alpha_c^M, \theta_c^1, \theta_c^2, \dots, \theta_c^M\}$ , where the Gaussians are specified by the mean vector  $\mu_c^m$  and the covariance matrix  $\Sigma_c^m$ , i.e.  $\theta_c^m = \{\mu_c^m, \Sigma_c^m\}$ . The EM algorithm [16, 17] commonly used for learning GMMs consists of an *expectation* step (E-step) and a *maximization* step (M-step) which are alternately used until the log  $p(\mathcal{X}_c|\Theta_c) = \log \prod_{n=1}^{N_c} p(\mathbf{x}^n|\Theta_c)$  converges to a local optimum, where  $\mathcal{X}_c = \{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^{N_c}\}_c$  are  $N_c$  i.i.d. samples belonging to class  $c$ .  $\mathcal{X}$  contains samples of all classes  $\mathcal{X} = \{\mathcal{X}_1, \dots, \mathcal{X}_C\}$  where  $N$  denotes the size of  $\mathcal{X}$ , i.e.  $N = |\mathcal{X}| = \sum_{c=1}^C N_c$ . The performance of the EM algorithm depends strongly on the choice of the initial parameters.

## 3 Discriminative CL-Based Learning of GMMs in Bayesian Classifiers

Optimizing CL is tightly connected to good classification performance. Hence, we want to learn parameters of the GMM-based Bayesian classifier so that CL

is maximized. Unfortunately, CL does not decompose. The objective function of the conditional log likelihood (CLL) using GMMs in Eq. (II) is

$$\begin{aligned}
 CLL(\mathcal{X}|\Theta) &= \log \prod_{n=1}^N p(c^n|\mathbf{x}^n) = \sum_{n=1}^N \log \frac{p(\mathbf{x}^n|\Theta_{c^n}) \rho_{c^n}}{\sum_{c'=1}^C p(\mathbf{x}^n|\Theta_{c'}) \rho_{c'}} = \\
 &\sum_{n=1}^N \left[ \log \left[ \left( \sum_{m=1}^M \alpha_{c^n}^m \mathcal{N}(\mathbf{x}^n|\theta_{c^n}^m) \right) \rho_{c^n} \right] - \log \sum_{c'=1}^C \left[ \left( \sum_{m=1}^M \alpha_{c'}^m \mathcal{N}(\mathbf{x}^n|\theta_{c'}^m) \right) \rho_{c'} \right] \right],
 \end{aligned} \tag{3}$$

where,  $c^n$  is the class of  $\mathbf{x}^n$ ,  $\rho_{c^n} = p(c^n)$  is the class prior of the  $n^{\text{th}}$  sample,  $0 < \rho_{c^n} < 1$ , and  $\sum_{c=1}^C \rho_{c^n} = 1$ . The set of parameters  $\Theta$  is composed of  $\Theta = \{\Theta_1, \dots, \Theta_C, \rho_1, \dots, \rho_C\}$ .

The EBW algorithm (more details are given in Appendix A) is an iterative procedure which can be used to optimize rational functions [II]. Clearly, the CL criterion in Eq. (3) is a rational function over the discrete model parameters  $\rho_c$  and  $\alpha_c^m$  and the parameter re-estimation equation of the form

$$\theta_i^j \leftarrow \frac{\theta_i^j \left( \frac{\partial CLL(\mathcal{X}|\Theta)}{\partial \theta_i^j} + D \right)}{\sum_l \theta_l^j \left( \frac{\partial CLL(\mathcal{X}|\Theta)}{\partial \theta_l^j} + D \right)}, \tag{4}$$

is used, where  $\theta_i^j \geq 0$ ,  $\sum_i \theta_i^j = 1$ , and  $j$  indicates a particular discrete variable. EBW requires the partial derivative  $\frac{\partial CLL(\mathcal{X}|\Theta)}{\partial \theta_i^j}$  and  $D$ . Both terms are provided in the sequel. Specifically,

$$\frac{\partial CLL(\mathcal{X}|\Theta)}{\partial \rho_c} = \sum_{n=1}^N \left[ \frac{\mathbb{1}_{\{c=c^n\}}}{\rho_c} - \frac{p(\mathbf{x}^n|\Theta_c) \rho_c}{\sum_{c'=1}^C p(\mathbf{x}^n|\Theta_{c'}) \rho_{c'}} \frac{1}{\rho_c} \right] = \frac{1}{\rho_c} \sum_{n=1}^N (\mathbb{1}_{\{c=c^n\}} - w_c^n), \tag{5}$$

where  $w_c^n = p(c|\mathbf{x}^n)$  (same as Eq. (II)) and  $\mathbb{1}_{\{i=j\}}$  is the indicator function (i.e. equals 1 if  $i = j$  and 0 if  $i \neq j$ ).

Further, the derivative for the parameters  $\alpha_c^m$  is

$$\frac{\partial CLL(\mathcal{X}|\Theta)}{\partial \alpha_c^m} = \sum_{n=1}^N \left[ \frac{\gamma_c^{n,m}}{\alpha_c^m} (\mathbb{1}_{\{c=c^n\}} - w_c^n) \right], \tag{6}$$

where

$$\gamma_c^{n,m} = \frac{\alpha_c^m \mathcal{N}(\mathbf{x}^n|\theta_c^m)}{\sum_{m'=1}^M \alpha_c^{m'} \mathcal{N}(\mathbf{x}^n|\theta_c^{m'})}. \tag{7}$$

Considering the derivative in Eq. (6) (similar for Eq. (5)) in the re-estimation Eq. (4) we obtain

$$\alpha_c^m \leftarrow \frac{\sum_{n=1}^N [\gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - w_c^n)] + \alpha_c^m D}{\sum_{m'=1}^M \sum_{n=1}^N [\gamma_c^{n,m'} (\mathbb{1}_{\{c=c^n\}} - w_c^n)] + D}.$$

The derivatives (Eq. (5) and (6)) are sensitive to small parameter values. Meraldo [18] observed that low-valued parameters  $\rho_c$  and  $\alpha_c^m$  in Eq. (5) and (6) may cause a large magnitude of the gradient. Hence, the optimization concentrates on those parameters which are usually unreliable estimates due to lack of data. Therefore, he suggests to focus on modifying better estimated high-valued parameters by using an approximation for the derivative in Eq. (6) (similar for Eq. (5))

$$\frac{\partial CLL(\mathcal{X}|\Theta)}{\partial \alpha_c^m} \approx \frac{\sum_{n=1}^N \gamma_c^{n,m} \mathbb{1}_{\{c=c^n\}}}{\sum_{m'=1}^M \sum_{n=1}^N \gamma_c^{n,m'} \mathbb{1}_{\{c=c^n\}}} - \frac{\sum_{n=1}^N \gamma_c^{n,m} w_c^n}{\sum_{m'=1}^M \sum_{n=1}^N \gamma_c^{n,m'} w_c^n}. \quad (8)$$

EBW has been formulated for discrete probability distributions. Normandin and Morgera [19] introduced a discrete approximation of the Gaussian distribution assuming diagonal covariance matrices. This leads to the re-estimation equation for  $\bar{\boldsymbol{\mu}}_c^m$  and  $\bar{\boldsymbol{\Sigma}}_c^m$  given as

$$\bar{\boldsymbol{\mu}}_c^m \leftarrow \frac{\sum_{n=1}^N [\gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - w_c^n) \mathbf{x}^n] + D \boldsymbol{\mu}_c^m}{\sum_{n=1}^N [\gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - w_c^n)] + D}$$

and

$$\bar{\boldsymbol{\Sigma}}_c^m \leftarrow \frac{\sum_{n=1}^N [\gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - w_c^n) (\mathbf{x}^n)^2] + D (\boldsymbol{\Sigma}_c^m + (\boldsymbol{\mu}_c^m)^2)}{\sum_{n=1}^N [\gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - w_c^n)] + D} - (\bar{\boldsymbol{\mu}}_c^m)^2, \quad (9)$$

where the squares of the vectors  $\mathbf{x}^n$  and  $\boldsymbol{\mu}_c^m$  are element-wise.

The EBW algorithm converges to a local optimum of  $CLL(\mathcal{X}|\Theta)$  providing a sufficiently large value for  $D$ . Setting the constant  $D$  is not trivial. If it is chosen too large then training is slow and if it is too small the update may fail to increase the objective function. In practical implementations heuristics have been suggested [13,14]. We initialize  $D = 1$  and double  $D$  until all variances in Eq. (9) are positive in the re-estimation step. Next, we multiply the obtained  $D$  with a global factor  $F$  (In Section 5.1 we empirically show the dependency of  $F$  on the convergence of EBW.). Value  $D$  is adapted in each iteration of the algorithm. The parameters  $\Theta_c$  for discriminative learning are initialized to the ML estimates of the GMM determined by the EM algorithm (see Section 2). The class prior is set to the normalized class frequency in  $\mathcal{X}$ , i.e.  $\rho_c = \frac{N_c}{N}$ .

## 4 Discriminative Margin-Based Learning of GMMs in Bayesian Classifiers

The multi-class margin [5] of sample  $n$  is

$$d_{\Theta}^n = \min_{c \neq c^n} \frac{p(c^n | \mathbf{x}^n, \Theta)}{p(c | \mathbf{x}^n, \Theta)} = \min_{c \neq c^n} \frac{p(c^n, \mathbf{x}^n | \Theta)}{p(c, \mathbf{x}^n | \Theta)} = \frac{p(c^n, \mathbf{x}^n | \Theta)}{\max_{c \neq c^n} p(c, \mathbf{x}^n | \Theta)}. \quad (10)$$

If  $d_{\Theta}^n > 1$ , then sample  $n$  is correctly classified and vice versa. We replace the max operator by the differentiable approximation  $\max_x f(x) \approx [\sum_x (f(x))^\eta]^\frac{1}{\eta}$ , where  $\eta \geq 1$  and  $f(x)$  is non-negative. In the limit of  $\eta \rightarrow \infty$  the approximation converges to the max operation. Replacing the max with its approximation, we obtain

$$d_{\Theta}^n = \frac{p(c^n, \mathbf{x}^n | \Theta)}{\left[ \sum_{c \neq c^n} (p(c, \mathbf{x}^n | \Theta))^\eta \right]^\frac{1}{\eta}}.$$

Usually, the max margin approach maximizes the margin of the sample with the smallest margin, i.e.  $\min_{n=1, \dots, N} d_{\Theta}^n$  for a separable classification problem [3]. We aim to relax this by introducing a soft margin, i.e. we focus on samples with a  $d_{\Theta}^n$  close to one. Therefore, we consider the *hinge* loss function according to

$$\tilde{D}(\mathcal{X} | \Theta) = \prod_{n=1}^N \min \left[ 2, (d_{\Theta}^n)^\lambda \right]$$

using the margin. Maximizing this function with respect to the parameters  $\Theta$  implicitly means to increase the margin  $d_{\Theta}^n$  whereas the emphasis is on samples with a margin  $(d_{\Theta}^n)^\lambda < 2$ , i.e. samples with a large positive margin have no impact on the optimization. The parameter  $\lambda > 0$  scales the margin and is set by cross-validation. Maximizing  $\tilde{D}(\mathcal{X} | \Theta)$  via EBW or gradient descent is not straight forward due to the discontinuity in the derivative at  $(d_{\Theta}^n)^\lambda = 2$ . Therefore, we propose to use for the *hinge* function  $h(y) = \min[2, y]$  a *smooth hinge* function which enables a smooth transition of the derivative and has a similar shape as  $h(y)$ . We propose the following *smooth hinge* function

$$h(y) = \begin{cases} y + \frac{1}{2}, & \text{if } y \leq 1 \\ 2 - \frac{1}{2}(y - 2)^2, & \text{if } 1 < y < 2 \\ 2, & \text{if } y \geq 2 \end{cases}$$

which requires to divide the data  $\mathcal{X}$  into three partitions depending on  $y = (d_{\Theta}^n)^\lambda$ , i.e.  $\mathcal{X}^1$  contains samples where  $(d_{\Theta}^n)^\lambda \leq 1$ ,  $\mathcal{X}^2$  consists of samples with a margin in the range  $1 < (d_{\Theta}^n)^\lambda < 2$ , and  $\mathcal{X}^3 = \mathcal{X} \setminus \{\mathcal{X}^1 \cup \mathcal{X}^2\}$ . Hence, our objective function for margin maximization is

$$D(\mathcal{X} | \Theta) = \prod_{n=1}^N h((d_{\Theta}^n)^\lambda) = \left\{ \prod_{n \in \mathcal{X}^1} \left( (d_{\Theta}^n)^\lambda + \frac{1}{2} \right) \right\} \left\{ \prod_{n \in \mathcal{X}^2} \left[ 2 - \frac{1}{2} \left( (d_{\Theta}^n)^\lambda - 2 \right)^2 \right] \right\} 2^{|\mathcal{X}^3|}$$

using the smooth hinge function. The  $\lambda$  for scaling the margin is usually selected as fraction number leading to *fractional polynomials* in the numerator and denominator of  $d_{\Theta}^n$ . The growth transform of the EBW algorithm (see [1]) extends to fractional polynomials and we can use the EBW algorithm for maximizing  $D(\mathcal{X} | \Theta)$ . Therefore, the derivative  $\frac{\partial \log D(\mathcal{X} | \Theta)}{\partial \Theta}$  for the re-estimation equation (see Eqn. [4]) of the EBW algorithm is

$$\frac{\partial \log D(\mathcal{X}|\Theta)}{\partial \Theta} = \sum_{n=1}^N s^n \frac{\partial \log d_{\Theta}^n}{\partial \Theta}$$

where  $s^n$  denotes a sample dependent weight given as follows:

$$s^n = \begin{cases} \frac{\lambda(d_{\Theta}^n)^{\lambda}}{(d_{\Theta}^n)^{\lambda} + \frac{1}{2}}, & \text{if } n \in \mathcal{X}^1 \\ \frac{\lambda(2 - (d_{\Theta}^n)^{\lambda})}{2 - \frac{1}{2}(d_{\Theta}^n)^{\lambda}}, & \text{if } n \in \mathcal{X}^2 \\ 0, & \text{if } n \in \mathcal{X}^3 \end{cases}.$$

Introducing GMMs in Eq. [10](#) and using the log gives

$$\log d_{\Theta}^m = \log \left[ \left( \sum_{m=1}^M \alpha_c^m \mathcal{N}(\mathbf{x}^n | \theta_{c^n}^m) \right) \rho_{c^n} \right] - \frac{1}{\eta} \log \sum_{c' \neq c^n} \left[ \left( \sum_{m=1}^M \alpha_{c'}^m \mathcal{N}(\mathbf{x}^n | \theta_{c'}^m) \right) \rho_{c'} \right]^{\eta}.$$

Similar as in Eq. [5](#) (Section [3](#)), the partial derivative of  $\log d_{\Theta}^m$  for the parameters  $\rho_c$  is

$$\frac{\partial \log d_{\Theta}^m}{\partial \rho_c} = \frac{\mathbb{1}_{\{c=c^n\}}}{\rho_c} - \mathbb{1}_{\{c \neq c^n\}} \frac{[p(\mathbf{x}^n | \Theta_c) \rho_c]^{\eta}}{\left[ \sum_{c' \neq c^n} p(\mathbf{x}^n | \Theta_{c'}) \rho_{c'} \right]^{\eta}} \frac{1}{\rho_c} = \frac{1}{\rho_c} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta}),$$

where we introduced

$$r_c^{n,\eta} = \frac{[p(\mathbf{x}^n | \Theta_c) \rho_c]^{\eta}}{\left[ \sum_{c' \neq c^n} p(\mathbf{x}^n | \Theta_{c'}) \rho_{c'} \right]^{\eta}}.$$

Furthermore, the derivative for the parameters  $\alpha_c^m$  is

$$\frac{\partial \log d_{\Theta}^m}{\partial \alpha_c^m} = \frac{\mathcal{N}(\mathbf{x}^n | \theta_c^m)}{\sum_{m'=1}^M \alpha_{c'}^{m'} \mathcal{N}(\mathbf{x}^n | \theta_{c'}^{m'})} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta}) = \frac{\gamma_c^{n,m}}{\alpha_c^m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta}),$$

where  $\gamma_c^{n,m}$  is given in Eq. [7](#). For the Gaussian distributions we use again the discrete approximation proposed in [19](#) assuming diagonal covariance matrices. This leads to the re-estimation equation for  $\bar{\boldsymbol{\mu}}_c^m$  and  $\bar{\boldsymbol{\Sigma}}_c^m$  given as

$$\bar{\boldsymbol{\mu}}_c^m \leftarrow \frac{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta}) \mathbf{x}^n] + D \boldsymbol{\mu}_c^m}{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta})] + D}$$

and

$$\bar{\boldsymbol{\Sigma}}_c^m \leftarrow \frac{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta}) (\mathbf{x}^n)^2] + D (\boldsymbol{\Sigma}_c^m + (\boldsymbol{\mu}_c^m)^2)}{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta})] + D} - (\bar{\boldsymbol{\mu}}_c^m)^2,$$

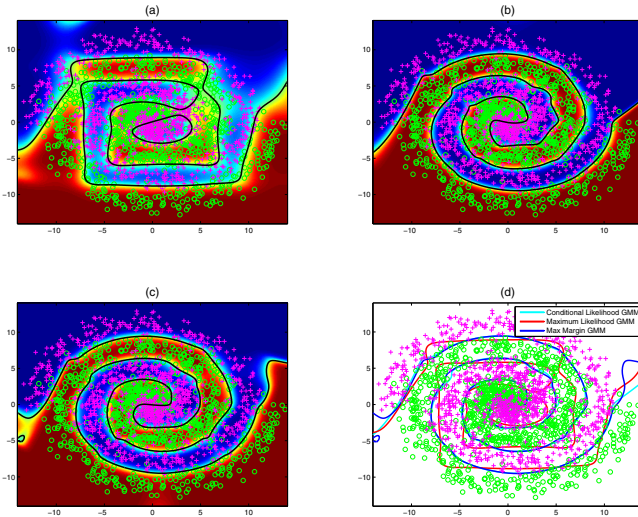
where the squares of the vectors are element-wise. Furthermore, the value  $D$  is determined in a similar manner as in Section 3. The EBW algorithm to discriminatively optimize the margin of GMM-based classifiers is summarized in Algorithm 1. Again, we use a more robust approximation for the derivatives of  $\rho_c$  and  $\alpha_c^m$  as suggested in Section 3.

## 5 Experimental Results

First we show the differences in the decision boundaries of generatively and discriminatively trained GMM-based Bayesian classifiers using synthetic data. Then, we provide classification results for a remote sensing and broad phonetic classification task.

### 5.1 Synthetic Data

We have two classes where each class is represented by a spiral. For class 1, sample  $\mathbf{x} \in \mathbb{R}^2$  is determined according to  $\mathbf{x} = [t \cos(4\pi t) + \epsilon_1 \quad t \sin(4\pi t) + \epsilon_2]^T$ , where  $\epsilon_1$  and  $\epsilon_2$  are independent samples from a zero-mean Gaussian noise process with  $\sigma = 1$ , and  $t$  is sampled from a uniform distribution. Likewise, samples for class 2 are obtained by using  $\mathbf{x} = [-t \cos(4\pi t) + \epsilon_1 \quad -t \sin(4\pi t) + \epsilon_2]^T$ . For each class we draw  $N_c = 5000$  and  $N_c = 1000$  samples for training and testing, respectively. Figure 1 shows various cases of generatively and discriminatively learned GMM-based Bayesian classifiers using  $M = 12$  components per class, i.e. (a) decision boundary of generative GMM, (b) decision boundary of CL-GMM, (c) decision boundary of MM-GMM, and (d) decision boundary of all learning approaches



**Fig. 1.** Synthetic data: (a) generative GMM, (b) CL-GMM, (c) MM-GMM, and (d) decision boundary of all learning approaches

**Input:**  $\mathcal{X} = \{\mathcal{X}_1, \dots, \mathcal{X}_C\}, \eta, \lambda, F$

**Output:**  $\rho_c, \{\alpha_c^m, \mu_c^m, \Sigma_c^m\}_{m=1}^M \quad \forall c \in \{1, \dots, C\}$

**Initialization:** For each  $c$ , train  $\{\alpha_c^m, \mu_c^m, \Sigma_c^m\}_{m=1}^M$  on  $\mathcal{X}_c$ , using the EM-algorithm. Set  $\rho_c$  to class frequency in  $\mathcal{X}$ , i.e.  $\rho_c \leftarrow \frac{|\mathcal{X}_c|}{|\mathcal{X}|}$

**while**  $D(\mathcal{X}|\Theta)$  *not converged* **do**

$$d_{\Theta}^n = \frac{(\sum_{m=1}^M \alpha_c^m \mathcal{N}(\mathbf{x}^n | \theta_c^m)) \rho_c}{\left[ \sum_{c' \neq c^n} \left( \sum_{m=1}^M \alpha_{c'}^m \mathcal{N}(\mathbf{x}^n | \theta_{c'}^m) \right) \rho_{c'} \right]^{\frac{1}{\eta}}} \quad \forall n \in \{1, \dots, N\}$$

Determine:  $\mathcal{X}^1, \mathcal{X}^2, \mathcal{X}^3$  based on  $(d_{\Theta}^n)^{\lambda}$

Determine:  $s^n \quad \forall n \in \{1, \dots, N\}$  based on  $\mathcal{X}^1, \mathcal{X}^2, \mathcal{X}^3$

**E-step:**

**for**  $c \leftarrow 1$  **to**  $C$  **do**

$$\gamma_c^{n,\eta} \leftarrow \frac{[p(\mathbf{x}^n | \Theta_c) \rho_c]^{\eta}}{\left[ \sum_{c' \neq c^n} p(\mathbf{x}^n | \Theta_{c'}) \rho_{c'} \right]^{\eta}} \quad \forall n \in \{1, \dots, N\}$$

$$\partial \rho_c \leftarrow \frac{\sum_{n=1}^N s^n \mathbb{1}_{\{c=c^n\}}}{\sum_{c'=1}^C \sum_{n=1}^N s^n \mathbb{1}_{\{c'=c^n\}}} - \frac{\sum_{n=1}^N s^n \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta}}{\sum_{c'=1}^C \sum_{n=1}^N s^n \mathbb{1}_{\{c' \neq c^n\}} r_{c'}^{n,\eta}}$$

**for**  $m \leftarrow 1$  **to**  $M$  **do**

$$\gamma_c^{n,m} \leftarrow \frac{\alpha_c^m \mathcal{N}(\mathbf{x}^n | \theta_c^m)}{\sum_{m'=1}^M \alpha_{c'}^{m'} \mathcal{N}(\mathbf{x}^n | \theta_{c'}^{m'})} \quad \forall n \in \{1, \dots, N\}$$

$$\partial \alpha_c^m \leftarrow \frac{\sum_{n=1}^N s^n \gamma_c^{n,m} \mathbb{1}_{\{c=c^n\}}}{\sum_{m'=1}^M \sum_{n=1}^N s^n \gamma_{c'}^{n,m'} \mathbb{1}_{\{c=c^n\}}} - \frac{\sum_{n=1}^N s^n \gamma_c^{n,m} \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta}}{\sum_{m'=1}^M \sum_{n=1}^N s^n \gamma_{c'}^{n,m'} \mathbb{1}_{\{c \neq c^n\}} r_{c'}^{n,\eta}}$$

**end**

**end**

**Determine D:**  $D \leftarrow \frac{1}{2}$

**for**  $c \leftarrow 1$  **to**  $C$  **do**

**for**  $m \leftarrow 1$  **to**  $M$  **do**

**repeat**

$D \leftarrow 2D$

$$\bar{\mu}_c^m \leftarrow \frac{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta}) \mathbf{x}^n] + D \mu_c^m}{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta})] + D}$$

$\Sigma_c^m \leftarrow$

$$\frac{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta}) (\mathbf{x}^n)^2] + D (\Sigma_c^m + (\mu_c^m)^2)}{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta})] + D} - (\bar{\mu}_c^m)^2$$

**until** all variances in  $\Sigma_c^m$  positive ;

**end**

**end**

$D \leftarrow DF$

**M-step:**

**for**  $c \leftarrow 1$  **to**  $C$  **do**

$$\bar{\rho}_c \leftarrow \frac{\rho_c (\partial \rho_c + D)}{\sum_{c'=1}^C \rho_{c'} (\partial \rho_{c'} + D)}$$

**for**  $m \leftarrow 1$  **to**  $M$  **do**

$$\bar{\alpha}_c^m \leftarrow \frac{\alpha_c^m (\partial \alpha_c^m + D)}{\sum_{m'=1}^M \alpha_{c'}^{m'} (\partial \alpha_{c'}^{m'} + D)}$$

$$\bar{\mu}_c^m \leftarrow \frac{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta}) \mathbf{x}^n] + D \mu_c^m}{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta})] + D}$$

$$\bar{\Sigma}_c^m \leftarrow \frac{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta}) (\mathbf{x}^n)^2] + D (\Sigma_c^m + (\mu_c^m)^2)}{\sum_{n=1}^N [s^n \gamma_c^{n,m} (\mathbb{1}_{\{c=c^n\}} - \mathbb{1}_{\{c \neq c^n\}} r_c^{n,\eta})] + D} - (\bar{\mu}_c^m)^2$$

$$\mu_c^m \leftarrow \bar{\mu}_c^m$$

**end**

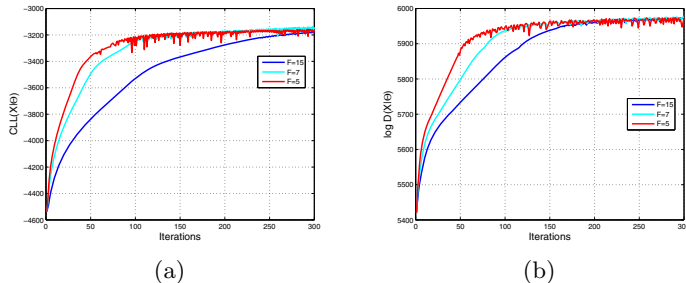
$$\alpha_c^m \leftarrow \bar{\alpha}_c^m \quad \forall m \in \{1, \dots, M\}$$

**end**

$$\rho_c \leftarrow \bar{\rho}_c \quad \forall c \in \{1, \dots, C\}$$

**end**

**Algorithm 1.** Discriminative Margin-based training of GMMs (MM-GMM Algorithm).



**Fig. 2.** Convergence of CL-GMM and MM-GMM depending on  $F$ . The  $x$ -axis denotes the number of iterations. (a)  $CLL(\mathcal{X}|\Theta)$ , (b)  $\log D(\mathcal{X}|\Theta)$ .

**Table 1.** Classification results in [%] on the synthetic training and test data

	GMM	CL-GMM	MM-GMM
Train Data	79.48 $\pm$ 0.40	86.47 $\pm$ 0.34	86.58 $\pm$ 0.34
Test Data	80.05 $\pm$ 0.89	85.80 $\pm$ 0.78	86.05 $\pm$ 0.77

(c) decision boundary of MM-GMM, and (d) shows the decision boundary of all learning approaches. The decision boundary of the DGMM classifiers is smoother and better approximates the original spiral data. Discriminative learning is able to change the decision boundary to improve the classification rate (see Table 1).

Furthermore, we show the evolution of both the conditional log likelihood  $CLL(\mathcal{X}|\Theta)$  and the margin  $\log D(\mathcal{X}|\Theta)$  depending on  $F$  over the iterations of the algorithms (see Figure 2(a) and (b)). As mentioned above, the rate of convergence of EBW strongly depends on the value of  $F$ . Additionally, the performances do not increase at each iteration. One reason is the approximation of the derivative in Eq. (8) as suggested in [18]. In [20], they experimentally observed that this approximation substantially improves convergence, although it is not guaranteed at each iteration.

## 5.2 Broad Phonetic classification

We use the TIMIT speech corpus [21] for broad phonetic classification. Therefore, we employ the standard NIST sets of 462 speakers and 168 speakers for training and testing, respectively. We perform frame-by-frame phone classification. We conduct experiments with only four classes and six classes using 1691462 and 1886792 samples, respectively. Moreover, we perform classification experiments on data of male speakers (Ma), female speakers (Fe), and both genders (Ma+Fe). More details about the experimental setup and the features can be found in [15]. We use the following classifiers:

- GMM: Generatively trained GMM with  $M = 100$  components.
- CL-GMM: Discriminative CL-based trained GMM classifier using  $M = 100$  components.



- MM-GMM: Discriminative margin-based trained GMM classifier using  $M = 100$  components.
- NN-100: Neural network (multi-layered perceptron) with one hidden layer. The number of units in the input and output layer is set to the number of features and the number of classes, respectively. In the hidden layer we use 100 neurons with a hyperbolic tangent sigmoid transfer function. Levenberg-Marquardt backpropagation is used for training and the transfer functions in the output layer are linear.
- SVM-1-0.1: The support vector machine with the radial basis function (RBF) kernel uses two parameters, namely  $C^*$  and  $\sigma$ , where  $C^*$  is the penalty parameter for the errors of the non-separable case and  $\sigma$  is the parameter for the RBF kernel. We set the values for these parameters to  $C^* = 1$  and  $\sigma = 0.1$ .

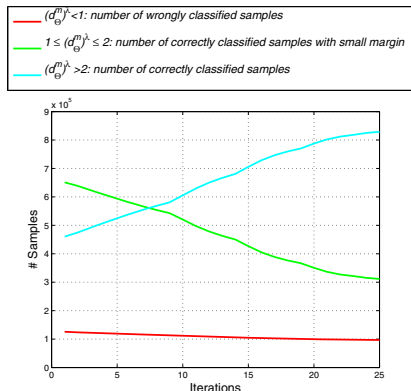
The optimal choice of the parameters (i.e.,  $C^*$ ,  $\sigma$ ), number of neurons in the hidden layer, and transfer functions of the above mentioned classifiers was obtained in each case by cross-validation. The parameters for learning CL-GMM and MM-GMM are initialized to the ML estimates.

The experimental results in Figure 3(a) show that CL-GMMs achieve about the same performance compared to MM-GMMs, whereas both DGMMs perform significantly better than generatively learned GMMs. The classification results of the MM-GMM are  $\approx 0.75\%$  lower compared to NNs and SVMs. The number of parameters for the DGMM is 16404 compared to 202425 and 400442 support vectors of the SVM for the Ma-Fe-4Class and Ma-Fe-6Class data, respectively. Hence, SVMs have roughly  $4 \cdot 10^6$  and  $8 \cdot 10^6$  parameters using the dimensionality of  $d = 20$  for each support vector. This means that DGMM has almost 500 times fewer parameters than the SVM for the Ma-Fe-6Class data. Although, the classification results are slightly worse DGMMs offer advantages compared to the SVM. DGMMs can be directly applied to problems with more than two classes, whereas SVMs are usually limited to binary problems – the multiclass problem is decomposed into binary problems. However, multiclass SVMs have been proposed [22]. For SVMs we have to select  $C^*$  and  $\sigma$ . For MM-GMMs the number of components  $M$  and  $\lambda$  have to be determined. A substantial difference is that the SVMs determine the number of support vectors automatically while in the case of DGMMs the number of components  $M$  is prescribed. Hence, in DGMMs the complexity is controlled manually. DGMMs are an excellent choice when a probabilistic model is required, e.g. marginalizing over the unknown variables is supported. The training time for each iteration of the DGMM scales with  $\mathcal{O}(MN)$ , whereas for the SVM we have  $\mathcal{O}(N^2)$ . Hence, DGMMs have a lower training complexity.

In Figure 3(b), we provide an in-depth analysis of the multi-class margin  $(d_{\Theta}^n)^\lambda$  for Ma-Fe-4 ( $M = 100$ ). The cyan and green colored lines denote the number of correctly classified samples with a margin of  $(d_{\Theta}^n)^\lambda > 2$  (i.e.  $|\mathcal{X}^3|$ ) and  $1 \leq (d_{\Theta}^n)^\lambda \leq 2$  (i.e.  $|\mathcal{X}^2|$ ) over the iterations, respectively. The samples with margin between one and two are still considered during optimization and the algorithm tries to increase the margin above two, i.e the number of those

Data set	Class	Classifier				
		GMM	GMM CL	GMM MM	NN 100	SVM 1-0.1
Ma+Fe	4	90.17 ± 0.06	92.54 ± 0.06	92.30 ± 0.06	92.58 ± 0.06	92.78 ± 0.06
	Ma	90.17 ± 0.08	92.50 ± 0.07	92.31 ± 0.07	92.73 ± 0.07	92.69 ± 0.07
	Fe	90.56 ± 0.11	92.55 ± 0.10	92.63 ± 0.10	92.91 ± 0.10	92.97 ± 0.10
Ma+Fe	6	82.42 ± 0.08	85.81 ± 0.07	85.14 ± 0.07	86.05 ± 0.07	86.26 ± 0.07
	Ma	82.49 ± 0.10	85.66 ± 0.09	85.19 ± 0.09	86.04 ± 0.09	86.16 ± 0.09
	Fe	82.84 ± 0.14	85.74 ± 0.13	85.69 ± 0.13	86.37 ± 0.12	86.65 ± 0.12

(a)



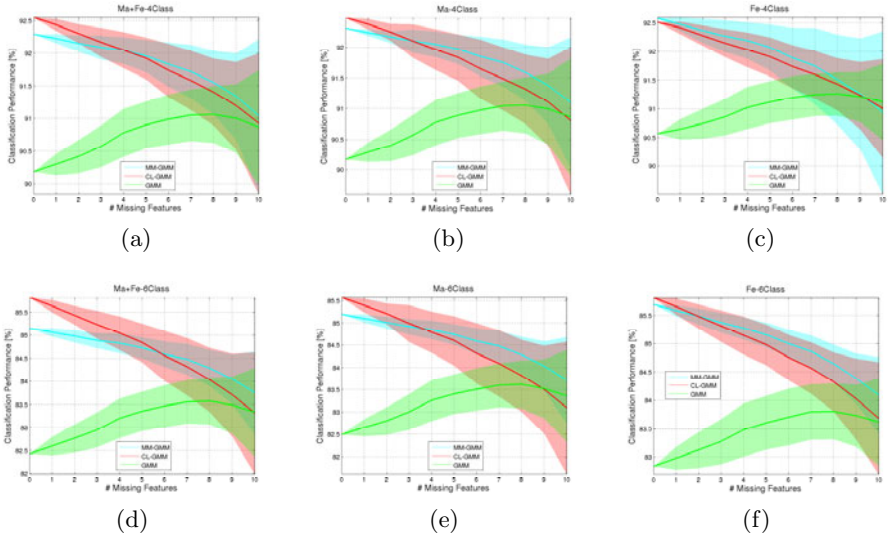
(b)

**Fig. 3.** Broad phonetic classification: (a) Classification accuracy in [%] for 4 and 6 classes with standard deviation. (b) Number of samples in  $\mathcal{X}^1$ ,  $\mathcal{X}^2$ , and  $\mathcal{X}^3$  over the iterations of MM-GMM.

samples decreases over the iterations while the number of samples with  $(d_{\Theta}^n)^{\lambda} > 2$  increases. Additionally, the number of wrongly classified samples (i.e.  $(d_{\Theta}^n)^{\lambda} < 1$ ) decreases (red line).

In the following, we verify that a discriminatively parameterized generative GMM  $p(\mathbf{x}|\Theta_c)$  still offers its advantages in the missing feature case. In particular, the ability to go from  $p(\mathbf{x}|\Theta_c)$  to  $p(\mathbf{x}'|\Theta_c)$  is maintained where  $\mathbf{x}'$  is a subset of the features in  $\mathbf{x}$  and  $\mathbf{x}''$  is the set of missing features, i.e.  $\mathbf{x} \setminus \mathbf{x}'$ . This amounts to performing the marginalization  $p(\mathbf{x}'|\Theta_c) = \int p(\mathbf{x}|\Theta_c) d\mathbf{x}''$ . A discriminative model, however, is inherently conditional and it is not possible in general to simply marginalize away any missing features. This problem is also true for SVMs, logistic regression, and neural networks.

We are particularly interested in a testing context which has arbitrary sets of missing features for each classification sample, unanticipated at training time. In such a case, it is not possible to re-train the model for each potential set of missing features without also memorizing the training set. In Figure 4, we present the classification performance of GMM, CL-GMM, and MM-GMM assuming missing features using the data of TIMIT-4/6. The  $x$ -axis denotes the number of missing features. The curves are the average over 100 classifications of the test data with uniformly at random selected missing features. Standard deviation bars indicate that the resulting differences are significant for a low number of missing features. We use exactly the same missing features for each classifier. We observe that discriminatively parameterized GMM classifiers outperform classical GMMs in the case of a low number of missing features. In case of many missing features classical GMMs seem to be more robust. The rising performance of the generative GMM classifier in case of missing features can be attributed to the phenomenon observed in the feature selection community. There, the reduction of the feature set size may even improve the classification



**Fig. 4.** Classification performance of GMM, CL-GMM, and MM-GMM assuming missing features using data of TIMIT-4/6. The  $x$ -axis denotes the number of missing features and the shaded region corresponds to the standard deviation over 100 classifications. (a) Ma+Fe-4, (b) Ma-4, (c) Fe-4, (d) Ma+Fe-6, (e) Ma-6, (f) Fe-6.

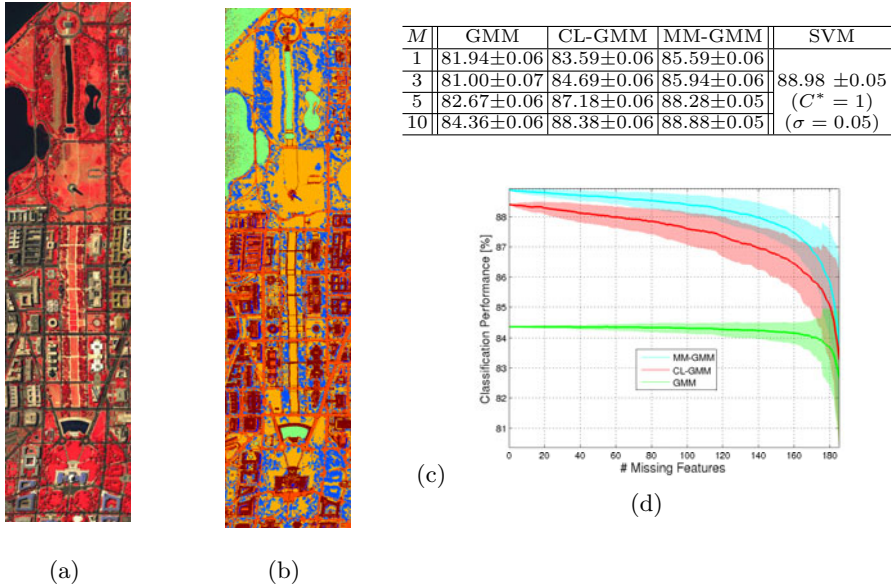
rate by reducing estimation errors associated with finite sample size effects [23]. Generally, this demonstrates, at least empirically, that discriminatively parameterized generative GMMs do not lose their ability to impute missing features.

### 5.3 Remote Sensing

We use a hyperspectral remote sensing image of the Washington, D.C., Mall area containing 191 spectral bands having a spectral width of 5-10 nm.<sup>2</sup> As ground reference a classification performed at Purdue University was used containing 7 classes, namely, roofs, road, grass, trees, trail, water, and shadow.<sup>3</sup> The aerial image using bands 63, 52, and 36 for red, green, and blue colors, respectively, and the reference image are shown in Figure 5(a) and (b). The image contains  $1280 \times 307$  hyperspectral pixels, i.e. 392960 samples. We arbitrarily choose 5000 samples of each class to learn the classifier. This remote sensing application is in particular interesting for our classifiers since spectral bands might be missing or should be neglected due to atmospheric effects, i.e. radiation within the visible range should be neglected in case of clouds or darkness. We use generative GMM as well as discriminatively optimized GMM classifiers, whereas the parameters for discriminative training are initialized to ML estimates. The classification

<sup>2</sup> <http://cobweb.ecn.purdue.edu/~biehl/MultiSpec/hyperspectral.html>

<sup>3</sup> <http://cobweb.ecn.purdue.edu/~landgreb/Hyperspectral.Ex.html>



**Fig. 5.** Washington, D.C., Mall: (a) Spectral bands 63, 52, and 36 are used for pseudo color image. (b) Reference image. (c) Classification results in [%]. (d) Classification results of GMM, CL-GMM, and MM-GMM assuming missing features.

performances for  $M \in \{1, 3, 5, 10\}$  components are shown in Table 5(c). CL-GMM and MM-GMM significantly outperforms the generative GMM classifier whereas best performances are obtained with MM-GMM classifiers. Remarkably, MM-GMMs and SVMs achieve a highly similar performance. The number of parameters for the GMM is roughly 85 times lower than for SVMs (26817 versus 2279394 (i.e. 11934 support vectors,  $N=191$ )).

In Figure 5(d), we report classification results for GMM, CL-GMM, and MM-GMM using  $M = 10$  components assuming missing features. The  $x$ -axis denotes the number of missing features. We average the performances over 100 classifications of the test data with randomly missing features. Standard deviation bars indicate that the resulting differences are significant for a low number of missing features. Discriminatively parameterized GMM classifiers significantly outperform classical GMMs in the case of few missing features.

## 6 Conclusions

We derive two discriminative training methods for GMM-based Bayesian classifiers maximizing either the conditional likelihood or the margin. Both algorithms are based on the extended Baum-Welch (EBW) algorithm. In the experiments we depict the differences of the decision boundary for generatively and discriminatively learned GMMs for classification using synthetic data. Furthermore, we

show results for broad phonetic classification and compare discriminatively optimized GMM classifiers to SVMs and NNs. DGMMs perform slightly worse compared to SVMs in terms of classification rate, however the GMM model uses almost 500 times fewer parameters than the SVM. Additionally, we show that discriminatively optimized GMM classifiers are superior even in the case of missing features. Finally, we compare our classifiers on a hyperspectral remote sensing application which is in particular interesting concerning the missing feature aspect. Margin-based GMMs outperform CL-based GMMs, whereas both significantly outperform generatively optimized GMMs.

## References

1. Gopalakrishnan, O., Kanevsky, D., Nàdas, A., Nahamoo, D.: An inequality for rational functions with applications to some statistical estimation problems. *IEEE Transactions on Information Theory* 37(1), 107–113 (1991)
2. Vapnik, V.: *Statistical learning theory*. Wiley & Sons, Chichester (1998)
3. Schölkopf, B., Smola, A.: *Learning with kernels: Support Vector Machines, regularization, optimization, and beyond*. MIT Press, Cambridge (2001)
4. Taskar, B., Guestrin, C., Koller, D.: Max-margin markov networks. In: *Advances in Neural Information Processing Systems, NIPS* (2003)
5. Guo, Y., Wilkinson, D., Schuurmans, D.: Maximum margin Bayesian networks. In: *International Conference on Uncertainty in Artificial Intelligence, UAI* (2005)
6. Roos, T., Wettig, H., Grünwald, P., Myllymäki, P., Tirri, H.: On discriminative Bayesian network classifiers and logistic regression. *Machine Learning* 59, 267–296 (2005)
7. Sha, F., Saul, L.: Large margin Gaussian mixture modeling for phonetic classification and recognition. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP* (2006)
8. Sha, F., Saul, L.: Comparison of large margin training to other discriminative methods for phonetic recognition by hidden Markov models. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 313–316 (2007)
9. Heigold, G., Deselaers, T., Schlüter, R., Ney, H.: Modified MMI/MPE: A direct evaluation of the margin in speech recognition. In: *International Conference on Machine Learning (ICML)*, pp. 384–391 (2008)
10. Collobert, R., Siz, F., Weston, J., Bottou, L.: Trading convexity for scalability. In: *International Conference on Machine Learning (ICML)*, pp. 201–208 (2006)
11. Schlüter, R., Macherey, W., Müller, B., Ney, H.: Comparison of discriminative training criteria and optimization methods for speech recognition. *Speech Communication* 34, 287–310 (2001)
12. Bahl, L., Brown, P., de Souza, P., Mercer, R.: Maximum Mutual Information estimation of HMM parameters for speech recognition. In: *IEEE Conf. on Acoustics, Speech, and Signal Proc.*, pp. 49–52 (1986)
13. Woodland, P., Povey, D.: Large scale discriminative training of hidden Markov models for speech recognition. *Computer Speech and Language* 16, 25–47 (2002)
14. Klautau, A., Jevtić, N., Orlitsky, A.: Discriminative Gaussian mixture models: A comparison with kernel classifiers. In: *Inter. Conf. on Machine Learning (ICML)*, pp. 353–360 (2003)
15. Pernkopf, F., Van Pham, T., Bilmes, J.: Broad phonetic classification using discriminative Bayesian networks. *Speech Communication* 143(1), 123–138 (2008)

16. Bishop, C.M.: Pattern recognition and machine learning. Springer, Heidelberg (2006)
17. Pernkopf, F., Bouchaffra, D.: Genetic-based EM algorithm for learning Gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27(8), 1344–1348 (2005)
18. Merialdo, B.: Phonetic recognition using hidden Markov models and maximum mutual information training. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 111–114 (1988)
19. Normandin, Y., Morgera, S.: An improved MMIE training algorithm for speaker-independent small vocabulary, continuous speech recognition. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 537–540 (1991)
20. Normandin, Y., Cardin, R., De Mori, R.: High-performance connected digit recognition using maximum mutual information estimation. *IEEE Trans. on Speech and Audio Proc.* 2(2), 299–311 (1994)
21. Lamel, L., Kassel, R., Seneff, S.: Speech database development: Design and analysis of the acoustic-phonetic corpus. In: *DARPA Speech Recognition Workshop*, Report No. SAIC-86/1546 (1986)
22. Crammer, K., Singer, Y.: On the algorithmic interpretation of multiclass kernel-based vector machines. *Journal of Machine Learning Research* 2, 265–292 (2001)
23. Jain, A., Chandrasekaran, B.: Dimensionality and sample size considerations in pattern recognition in practice. *Handbook of Statistics*, vol. 2. North-Holland, Amsterdam (1982)
24. Baum, L., Eagon, J.: An inequality with applications to statistical prediction for functions of Markov processes and to a model of ecology. *Bull. Amer. Math. Soc.* 73, 360–363 (1967)

## Appendix A: EBW Algorithm

In its original form [24], the Baum-Eagon inequality has been formulated for domains of discrete probabilities. Consider a domain  $E$  of discrete probability values  $\Phi = \{\varphi_i^j\}$ , with  $\varphi_i^j \geq 0$ ,  $\sum_i \varphi_i^j = 1$ , and  $j = 1, \dots, J$ . Given a homogeneous polynomial  $Q(\Phi)$  with nonnegative coefficients over the domain  $E$ , the Baum-Eagon inequality offers an iterative method to find local maxima in  $Q$ . It provides a transformation,  $T : E \rightarrow E$ , such that  $Q(T(\Phi)) > Q(\Phi)$ , unless  $T(\Phi) = \Phi$ . This transformation, called *growth transform*, maps from  $\hat{\Phi} \in E$  to  $T(\hat{\Phi}) = \bar{\Phi} \in E$ , where

$$\bar{\varphi}_i^j = \frac{\hat{\varphi}_i^j \frac{\partial Q(\hat{\Phi})}{\partial \varphi_i^j}}{\sum_{i'} \hat{\varphi}_{i'}^j \frac{\partial Q(\hat{\Phi})}{\partial \varphi_{i'}^j}}. \quad (11)$$

For brevity,  $\frac{\partial Q(\hat{\Phi})}{\partial \varphi_i^j}$  denotes the partial derivative  $\frac{\partial Q}{\partial \varphi_i^j}$  evaluated at point  $\hat{\Phi}$ .

In [1], the growth transform is extended<sup>4</sup> to rational functions  $R(\Phi)$  over  $E$ :

$$R(\Phi) = \frac{\text{Num}(\Phi)}{\text{Den}(\Phi)}.$$

---

<sup>4</sup> Additionally, they show that the growth transform in Eq. (11) can be applied to nonhomogeneous polynomials.

This is done by converting  $R(\Phi)$  into a polynomial  $Q_{\hat{\Phi}}(\Phi)$  for a given  $\hat{\Phi}$  such that if  $Q_{\hat{\Phi}}(T(\hat{\Phi})) > Q_{\hat{\Phi}}(\hat{\Phi})$ , then  $R(T(\hat{\Phi})) > R(\hat{\Phi})$ , except  $T(\hat{\Phi}) = \hat{\Phi}$ . The polynomial  $Q_{\hat{\Phi}}(\Phi)$  that fulfills this condition is given in [11] as

$$Q_{\hat{\Phi}}(\Phi) = \text{Num}(\Phi) - R(\hat{\Phi})\text{Den}(\Phi).$$

To see this, first note that  $Q_{\hat{\Phi}}(\hat{\Phi}) = 0$ . Thus, if  $Q_{\hat{\Phi}}(\bar{\Phi}) > Q_{\hat{\Phi}}(\hat{\Phi})$ , then  $\text{Num}(\bar{\Phi}) > R(\hat{\Phi})\text{Den}(\bar{\Phi})$ , and hence  $R(\bar{\Phi}) > R(\hat{\Phi})$ .

Unfortunately, the growth transform can not be applied directly to  $Q_{\hat{\Phi}}(\Phi)$ , as it might have negative coefficients. To ensure nonnegativity, the growth transform is instead applied to

$$S_{\hat{\Phi}}(\Phi) = Q_{\hat{\Phi}}(\Phi) + C(\Phi),$$

where

$$C(\Phi) = \kappa \left( \sum_{j,i} \varphi_i^j + 1 \right)^r$$

has constant value over  $E$ , since  $\sum_i \varphi_i^j = 1$ , and  $r$  denotes the maximal order of  $Q_{\hat{\Phi}}(\Phi)$ . Hence,  $C(\Phi)$  adds a constant  $\kappa$  to every monomial in  $Q_{\hat{\Phi}}(\Phi)$ . This constant  $\kappa$  must be chosen such that  $S_{\hat{\Phi}}(\Phi)$  has nonnegative coefficients for every  $\hat{\Phi}$ . Thus,  $S_{\hat{\Phi}}(\Phi)$  has positive coefficients and still has the same important property as  $Q_{\hat{\Phi}}(\Phi)$ . This polynomial with positive coefficients can now be considered for the growth transform in Eq. (11).

As easily can be verified, the partial derivative of  $S_{\hat{\Phi}}(\Phi)$  can be expressed in terms of  $\frac{\partial \log R(\hat{\Phi})}{\partial \varphi_i^j}$ , according to

$$\frac{\partial S_{\hat{\Phi}}(\hat{\Phi})}{\partial \varphi_i^j} = \text{Num}(\hat{\Phi}) \frac{\partial \log R(\hat{\Phi})}{\partial \varphi_i^j} + D,$$

where  $D = \kappa r(J + 1)^{r-1}$  is the derivative of  $C(\Phi)$ . Plugging this result into Eq. (11), we finally obtain the extended Baum-Welch re-estimation equation for discrete probability distributions of the form

$$\tilde{\varphi}_i^j = \frac{\hat{\varphi}_i^j \left( \frac{\partial \log R(\hat{\Phi})}{\partial \varphi_i^j} + D \right)}{\sum_{i'} \hat{\varphi}_{i'}^j \left( \frac{\partial \log R(\hat{\Phi})}{\partial \varphi_{i'}^j} + D \right)}, \tag{12}$$

where the  $\tilde{\varphi}_i^j$  denotes the updated parameters, and constant  $D$  must be chosen to be sufficiently large.

# Learning with Ensembles of Randomized Trees : New Insights

Vincent Pisetta,  
Pierre-Emmanuel Jouve, and Djamel A. Zighed

Rithme, 59 bd Vivier Merle 69003 Lyon  
Fenics, 59 bd Vivier Merle 69003 Lyon  
ERIC Laboratory, 5 avenue Pierre Mendes-France, 69500 Bron  
vpisetta@rithme.eu,  
pjouve@fenics.com,  
abdelkader.zighed@univ-lyon2.fr

**Abstract.** Ensembles of randomized trees such as Random Forests are among the most popular tools used in machine learning and data mining. Such algorithms work by introducing randomness in the induction of several decision trees before employing a voting scheme to give a prediction for unseen instances. In this paper, randomized trees ensembles are studied in the point of view of the basis functions they induce. We point out a connection with kernel target alignment, a measure of kernel quality, which suggests that randomization is a way to obtain a high alignment, leading to possibly low generalization error. The connection also suggests to post-process ensembles with sophisticated linear separators such as Support Vector Machines (SVM). Interestingly, post-processing gives experimentally better performances than a classical majority voting. We finish by comparing those results to an approximate infinite ensemble classifier very similar to the one introduced by Lin and Li. This methodology also shows strong learning abilities, comparable to ensemble post-processing.

**Keywords:** Ensemble Learning, Kernel Target Alignment, Randomized Trees Ensembles, Infinite Ensembles.

## 1 Introduction

Ensemble methods are among the most popular approaches used in statistical learning. This popularity essentially comes from their simplicity and their efficiency in a large variety of real-world problems. Instead of learning a single classifier, ensemble methods first build several base classifiers, usually via a sequential procedure such as Boosting ([13], [15]), or a parallel strategy using randomization processes such as Bagging [2] or Stochastic Discrimination [21], and in a second phase, use a voting scheme to predict the class of unseen instances.

Because of their impressive performances, understanding the mechanisms of ensemble learning algorithms is one of the main priority in the machine learning community. Several theoretical works have connected the Boosting framework



with the very well known SVMs [30] highlighting the margin's maximization properties of both algorithms (see e.g. [11] [14] [26]). Another popular theoretical framework comes from [15] who pointed out its connection with forward stagewise modelling leading to several improved Boosting strategies.

Ensembles using randomized processes suffer from a lack of well defined theoretical framework. Probably the most well-known result highlighting the benefits of such a strategy is due to Breiman [5] who showed that the performance of majority voting of an ensemble depends on the correlation between members forming the pool and their individual strength. Other notable works concern the study of the consistency of such algorithms [1].

In this paper, we go a step further [5] by analyzing the basis functions induced by an ensemble using a randomized strategy. As pointed out in [19], most ensemble methods can be seen as approaches looking for a linear separator in a space of basis functions induced by the base learners. In this context, analyzing the space of basis functions of an ensemble is of primary importance to better understand its mechanism. We specifically focus on studying the situation where base learners are decision trees. A lot of empirical studies have shown that this class of classifiers is particularly well-suited for ensemble learning (see e.g. [12]).

More precisely, we show a close connection between randomized trees ensembles and Parzen window classifiers. Interestingly, the error of a Parzen window classifier can be bounded from above with a kernel quality measure. This results in a generalization bound for an ensemble of randomized trees and clearly highlights the role of diversity and individual strength on the ensemble performance. Moreover, the connection suggests potential improvements of classical trees ensembles strategies.

Our paper is organized as follows. In section 2, we review some basic elements concerning decision tree induction. We focus on the importance of regularization and we point out a connection between decision trees and Parzen window classifiers. We introduce the notion of kernel target alignment (KTA) [9], a kernel quality measure allowing to bound the error of a Parzen window classifier. Once those base concepts are posed, we will show that an ensemble of randomized trees generates a set of basis functions leading to a kernel which can have a high alignment, depending on the individual strength and correlation between base learners (section 3). Interestingly, the connection shows that increasing the amount of randomization leads to a more regularized classifier.

Based on those results, we present in section 4 two possibilities for improving the performance of a randomized trees ensemble. The first strategy consists in post-processing intensively the committee using powerful linear separators. The second strategy builds an approximate infinite ensemble classifier and is very similar to the one presented in [23]. That is, instead of selecting a set of interesting basis functions as realized by an ensemble, we will fit a regularized linear separator in the (infinite dimensional) space of basis functions induced by all possible decision trees having a fixed number of terminal nodes. Experiments comparing all those approaches are presented in section 5. Finally in section 6 we conclude.

## 2 Single Decision Tree Learning

### 2.1 Decision Tree Induction

We consider the binary classification case specified as follows. We are given access to a set of  $n$  labeled examples  $S = \{(x_1, y_1), \dots, (x_n, y_n)\}$  drawn from some (unknown) distribution  $P$  over  $X \times \{-1, +1\}$ , where  $X$  is a  $d$ -dimensional abstract instance space composed of features  $X_1, \dots, X_d$  taking their values in  $\mathfrak{R}$ . The objective of any classification algorithm is to learn a function  $f : X \rightarrow \{-1, +1\}$  whose generalization error rate  $Pr_{(x,y) \equiv P} [f(x) \neq y]$  is as low as possible. Among the large variety of methods dedicated to this goal, decision trees are very popular thanks to their efficiency, their ability to capture non linear relations between inputs and output and essentially because they are easily interpretable.

A binary decision tree<sup>1</sup> consists in recursively partitioning the input space  $X$  by searching for the transversal cut which optimizes some predefined criterion. The algorithm starts with the root node containing all learning instances and looks for a split of the form  $[X_j \in s_{jm} ; X_j \notin s_{jm}]$  where  $s_{jm} = (l_{jm}; u_{jm}]$  represents a set of possible values of  $X_j$  defined by a lower and upper limit  $l_{jm} < X_j \leq u_{jm}$ . Two new nodes are then added to the tree, one containing instances respecting  $[X_j \in s_{jm}]$  and the other instances respecting  $[X_j \notin s_{jm}]$ . The process is then repeated for each subset (instances in a current node) until a stopping criterion is satisfied. In this context, a decision tree can be seen as a feature mapping  $\Phi : X \rightarrow F$  such that  $F$  is the space represented by the nodes of the tree. Each node represents a basis function  $b_t(x)$  (i.e a dimension of  $F$ ) taking the form of a conjunctive rule [16]:

$$b_t(x) = \prod_{j=1}^d I(X_j \in s_{jm}) \quad (1)$$

where  $I(\cdot)$  is the indicator function of the truth of its argument. We have  $b_t(x) = 1$  if the instance  $x$  belongs to node  $t$  and  $b_t(x) = 0$  elsewhere. Once the decision tree has been constructed, one can make a prediction by fitting a linear function  $f$  in  $F$  :

$$f(x) = \sum_{t=1}^{2T-1} \alpha_t b_t(x) + \alpha_0 \quad (2)$$

where  $T$  is the number of terminal nodes of the tree<sup>2</sup>. Classically a basis function associated to a terminal node has a weight  $\alpha_t$  equal to 1 or  $-1$  depending on the most frequent class of all examples belonging to it, while both  $\alpha_0$  and basis functions associated to non terminal nodes have a weight of 0.

<sup>1</sup> We restrict our framework to binary decision tree. Let us simply note that more complex structures such as n-ary trees can be implicitly constructed using binary trees (see e.g [19]).

<sup>2</sup> A binary decision tree has a total number of node equal to  $2T - 1$  which explains the upper term of the sum.

A lot of different decision trees algorithms have been designed. The main differences between them concern the criterion used to find the “best” split and the strategy employed to stop the induction process. Several experimental studies have shown that the splitting criterion has not a very significant effect on the performance of the decision tree (see e.g [7], [29]). As pointed out in [6], the key of success lies in controlling the tree complexity. Consequently, most efficient decision tree algorithms such as CART [6] or C4.5 [25] particularly focus on tree regularization.

A general method to control model complexity consists in giving a penalty proportional to the model complexity. The goal of the induction process then becomes to find the best tradeoff between performance (small error rate) and complexity, which can be realized by minimizing a criterion of the following form

$$\text{Criterion}(f, \lambda) = L(y, f) + \lambda J(f), \quad \lambda \geq 0 \quad (3)$$

where  $L(y, f)$  is a measure of the classifier error (a loss function) and  $J(f)$  a functional penalty that should be large for complex functions  $f$  [19]. This kind of regularization, widely used in machine learning is known as Tikhonov regularization. In this spirit, Breiman [6] proposed to select  $L(y, f)$  as the error rate and  $J(f)$  as the number of terminal nodes  $T$  of the decision tree. Once a value of  $\lambda$  has been selected, [6] develop a tree of maximum depth using Gini index as splitting criterion and, in a second phase, prune the tree in order to minimize (3) measured on a test set.

## 2.2 Connection with Parzen Window Classifier and Generalization Error

The classical predictive scheme (using only terminal nodes with a weight of +1 or -1) can interestingly be seen as a Parzen window estimator based on the kernel  $K(x, x') = \sum_{t=1}^T b_t(x)b_t(x')$  with  $b_t(x) = 1$  if  $x$  belongs to leaf  $t$  and 0 elsewhere. Note that the basis functions considered here are only those associated to terminal nodes. Here,  $K$  is a very sparse kernel since we have  $K(x, x') = 1$  if  $x$  and  $x'$  are in the same terminal node and 0 if not. The Parzen window estimator consists in labeling  $x$  with  $f(x) = \text{sign}(\sum_{i=1}^n y_i K(x, x_i))$  which is strictly equivalent to predict the most frequent label encountered in the terminal node in which  $x$  falls.

Cristianini et al. [9] have shown that the generalization error  $GE$  of the expected Parzen window estimator  $f(x) = \text{sign}(E_{(x', y')} [y' K(x', x)])$  based on a kernel  $K$  is bounded from above with probability  $1 - \delta$ :

$$GE(f(x)) \leq 1 - \hat{A}(K, y^t y) + \sqrt{\frac{8}{n} \ln \frac{2}{\delta}} \quad (4)$$

where  $\hat{A}(K, y^t y)$  is called (empirical) kernel target alignment (KTA) and is formally defined on a sample  $S$  as :

$$\hat{A}(K, y^t y) = \frac{\langle K, y^t y \rangle_F}{\sqrt{\langle K, K \rangle_F \langle y^t y, y^t y \rangle_F}} = \frac{\sum_{i,s=1}^n y_i y_s K(x_i, x_s)}{n \sqrt{\sum_{i,s=1}^n K(x_i, x_s)^2}} \quad (5)$$

where  $\langle \cdot, \cdot \rangle_F$  is the Frobenius product. KTA was initially designed to reflect the goodness of a kernel and more generally the goodness of a similarity matrix. As we can see from (5), KTA calculates the sum of similarities between objects belonging to the same class and subtracts the sum of similarities between objects belonging to distinct classes. This quantity is then normalized to obtain an indicator varying between  $-1$  and  $1$ .

The connection allows one to use the following regularization scheme. First, we can induce a tree having the purest possible nodes, and then, look for the subtree leading to the kernel that minimizes (4) or equivalently, that maximizes  $\hat{A}(K, y^t y)$  on a test sample. Interestingly, this methodology is very similar to the pruning technique of CART which estimates the expected Parzen window error on a test sample and keep the subtree that minimizes a tradeoff between this error and the tree complexity. In KTA, the complexity of the tree is penalized implicitly because the effect of adding splits is reflected in the sparsity of  $K$  leading to a lower value of  $\langle K, y^t y \rangle_F$ .

### 2.3 Drawbacks of Single Decision Trees

While regularization is an essential feature of the success of decision trees, this is generally not enough to obtain strong learning abilities. The first problem comes from the greedy mechanism used to find a split. In some cases such as the well known XOR problem, it is difficult for the algorithm to find the best split because this implies finding an interaction between two (or more) features in one shot. The second problem is their high variance. It is well-known that small changes in data could lead to drastic changes in the decision tree. This phenomenon comes from their hierarchical structures which implies that an error at split  $k$  will be propagated down to all splits below it [19]. This latter problem is reflected in the basis functions induced by a single tree. If the algorithm does a “strong” mistake at a high level, i.e, at the top levels of the tree, it will be impossible for any regularization strategy based on pruning to obtain a good result.

The main question is how one can overcome these problems. An appealing solution lies in increasing the number of basis functions generated by the algorithm. However, in order to be efficient, this process should induce basis functions in a non-hierarchical manner, else the problem of high variance will remain. Trees ensembles are particularly well-suited to realize such a process. Indeed, they work by generating several decision trees which is equivalent to increase the number of generated basis functions in a non-hierarchical way. In the next section, we will see that randomized trees ensembles allow to increase the kernel target alignment previously introduced. Consequently they improve the classification accuracy compared to a single decision tree and act in the same time as regularizers.

In the same spirit, one can choose to select all basis functions generated by all possible decision trees having a predefined number of terminal nodes. While this seems a priori untractable due to the infinite number of possible basis functions, we will see (section 4) that a solution can nevertheless be found by embedding

the basis functions into an appropriate kernel. However, in this case, we should carefully regularize the classifier which will be realized using a classical Tikhonov regularization as in (3).

### 3 Randomized Trees Ensemble

#### 3.1 Algorithms

One of the most efficient ways to improve the performance of a single decision tree is to construct several distinct decision trees and to aggregate them through a voting scheme. Examples of such procedures are Bagging [2], Random Forests [5], PERT [10] or Extremely Randomized Trees [17]. These algorithms are all particular cases of a more general methodology introducing randomization in the induction process of base learners. The skeleton of these techniques consists in repeating  $M$  times the following steps :

- *Step 1* : Apply a sampling strategy on  $S$  to obtain a new sample  $S_m$
- *Step 2* : Induce a decision tree on  $S_m$  by searching recursively for the best split among a random subset of all possible ones<sup>3</sup>.

Each individual learner predicts a class for an unseen instance  $x$  corresponding to the most frequent class of examples belonging to the terminal node in which  $x$  falls. The prediction associated to the ensemble corresponds to the most frequently voted class among the  $M$  decision trees. The differences between randomized ensembles algorithms comes from the sampling strategy used in step 1 the randomization process chosen to find a split in step 2. In the case of Bagging,  $S$  is sampled iteratively using a bootstrap strategy, while there is no randomization in the split's search. Random Forests work by sampling  $S$  via a bootstrap and look for the best split among a random subset of  $d'$  features ( $d' \leq d$ ). Extremely randomized trees do not use any sampling scheme but look for the best split among  $d'$  cut points randomly chosen on  $d'$  features themselves randomly chosen. Because such procedures lead to classifiers with a low variance, each individual tree is generally fully grown (i.e, until having the purest possible leaves) in order to reduce bias and consequently the generalization error [19].

The use of an ensemble has the effect to increase the number of generated basis functions compared to a single decision tree. Intuitively, this is interesting because it overcomes the problems due to the hierarchical nature of decision trees and to their greedy split's search. However, such a mechanism will work only if the basis functions are enough diverse and individually correlated to the output  $y$ . Indeed, if the trees are identical, the set of generated basis functions will be equivalent to the basis functions generated by a single decision tree leading to an unchanged prediction. The use of randomization procedures is then fully

<sup>3</sup> In [24], the authors define a slightly different way of introducing randomization into decision trees ensembles. Their definition allows to study the “spectrum” of randomization and give interesting insights about the effect of randomization on the performance of decision trees ensembles.

justified. Probably the most well-known theoretical justification is due to [5] who has shown that the generalization error  $GE$  of majority voting of an ensemble is bounded from above :  $GE \leq \bar{\rho}(1 - s^2)/s^2$  where  $\bar{\rho}$  is the average correlation between base classifiers and  $s$  a function of their strength.

### 3.2 Connection with Kernel Target Alignment

We will see that the benefit of using randomized trees ensembles is also reflected through the KTA measured on the kernel induced by the ensemble. Let us assume that each tree of the ensemble is grown until having only pure terminal nodes. Note that this is always possible while there are not two examples with the same initial representation and distinct labels. It is well-known that in classification, the best results are generally obtained by growing each tree until having only pure nodes (see e.g. [5]). Interestingly, in this case, the classical majority voting scheme is equivalent to a Parzen window estimator based on the kernel induced by the ensemble  $K_{ens}(x, x') = M^{-1} \sum_{m=1}^M \sum_{t=1}^{T_m} b_{mt}(x)b_{mt}(x')$ . Here,  $b_{mt}(x)$  represents the basis function associated to the terminal node  $t$  of the tree  $m$ . In this context, the performance of the ensemble should be highly dependent on the KTA of  $K_{ens}$ . The main question is why introducing randomization in the induction of trees could lead to a higher KTA.

$K_{ens}$  can equivalently be written  $K_{ens}(x, x') = M^{-1} \sum_{m=1}^M K_m(x, x')$  where  $K_m(x, x')$  is the kernel induced by the  $m^{th}$  tree in the same manner as in section 2.2. Note that dropping the constant  $M^{-1}$  has no effect on the prediction. In this point of view, we see that  $K_{ens}$  simply consists in summing several base kernels. In [9], the authors have shown that one can benefit from summing two kernels. Indeed, the alignment of the sum of two kernels  $K_1$  and  $K_2$  with the target is given by :

$$\hat{A}(K_1 + K_2, y^t y) = \frac{\|K_1\|_F}{\|K_1 + K_2\|_F} \hat{A}(K_1, y^t y) + \frac{\|K_2\|_F}{\|K_1 + K_2\|_F} \hat{A}(K_2, y^t y)$$

The alignment of the sum will be high if both kernels have a high individual alignment and if their correlation, i.e.  $\hat{A}(K_1, K_2) = \langle K_1, K_2 \rangle_F / \|K_1\|_F \|K_2\|_F$  is low. Indeed, if the kernels are identical, we have  $\hat{A}(K_1 + K_2, y^t y) = \hat{A}(K_1, y^t y) = \hat{A}(K_2, y^t y)$  while if they are different, we have :

$$\frac{\|K_1\|_F}{\|K_1 + K_2\|_F} + \frac{\|K_2\|_F}{\|K_1 + K_2\|_F} > 1$$

leading to a potentially higher overall alignment. Note that if one uses  $M$  kernels, the alignment of sum will be equal to :

$$\hat{A}\left(\sum_{m=1}^M K_m, y^t y\right) = \sum_{m=1}^M \frac{\|K_m\|_F}{\left\|\sum_{m=1}^M K_m\right\|_F} \hat{A}(K_m, y^t y) \quad (6)$$

The effect of randomization will be to decrease a bit the average individual alignment of the kernels in order to decrease their correlation, i.e. to increase

$\|K_m\|_F / \left\| \sum_{m=1}^M K_m \right\|_F$ . In most empirical studies on the relationship between ensemble performance and strength-correlation, the main problem is to measure the diversity [22]. Equation (6) clearly highlights the role of each component and a possible way of measuring them. While randomization aims at playing on diversity and individual strength, its exact role is more complex.

The reason comes from the concentration property of the alignment. As underlined in [9], if the kernel function is selected a priori, that is, if one do not learn the kernel, the value of the alignment measured on a sample  $S$  is highly concentrated around its expected value. As a consequence, building an ensemble of extremely randomized trees as realized by [17] leads to a kernel that would have quite the same alignment on the training sample and any test sample. However, learning too intensively the kernel, i.e, introducing few randomization in the tree induction will result in a larger difference and will be reflected in a lower expected alignment than one could wish to have. The direct implication is that introducing a high level of randomness leads to a more regularized classifier. This also shows that decreasing the amount of randomization in the induction of decision trees will not necessarily result in a higher individual expected alignment of a decision tree. Interestingly, in its experiments, Breiman [5] observed that increasing the number of features to find the best split did not necessarily lead to higher individual strength. The explanation in terms of alignment concentration may give a clue to these results. Experiments showing all those claims are presented in section 5.

## 4 Improved Randomized Trees Ensembles

In this section, we present two possible improvements of an ensembles of randomized trees. Here, we describe the theoretical aspects. Experiments will be presented in section 5.

### 4.1 Post-processing

Globally, randomized trees ensembles can be seen as powerful kernel constructors because they aim at increasing KTA through the introduction of randomization. While the alignment is directly connected to Parzen window estimator, [9] have shown experimentally that maximizing KTA is also a good strategy before employing more complex learners such as SVMs. Because randomized trees ensembles directly act on the kernel target alignment, it seems interesting to post-process them using a more complex learner than a simple Parzen window estimator. That is, instead of simply giving the same weight to all basis functions induced by the ensemble, one can learn “optimal weights” with an appropriate learning strategy. In this case however, we are no more protected against over-fitting because of the lack of links between the new learner and

KTA and should consequently employ a specific regularization. A possible way consists in searching a vector of weights  $\hat{\alpha}$  such that [16]:

$$\{\hat{\alpha}\}_0^{|B|} = \arg \min_{\{\alpha\}_0^{|B|}} \sum_{i=1}^n L \left( y_i, \alpha_0 + \sum_{t=1}^{|B|} \alpha_t b_t(x_i) \right) + \lambda \sum_{t=1}^{|B|} |\alpha_t|^p \quad (7)$$

where  $B$  is the set of basis functions induced by the ensemble of decision trees<sup>4</sup>. Different parameterizations of  $L(\cdot)$  and  $p$  lead to classical statistical learners. For example, choosing  $L(y_i, f(x_i)) = \max(0, 1 - y_i f(x_i))$  (hinge loss), and  $p = 2$  consists in solving the SVM optimization program [30] in the space of basis functions constructed by the tree, while choosing  $L(\cdot)$  as the hinge loss and  $p = 1$  is equivalent to solve the LPBoost problem [11]. The choice of  $L(\cdot)$  is mainly dependent on the type of learning problem we are facing. Typically, in a regression setting,  $L(\cdot)$  is chosen as the square-loss function while in classification, we will tend to choose the hinge loss. The choice of regularization is a harder task since its effect is not yet fully understood. A well known difference is that constraining the coefficients in  $L_1$  norm (i.e,  $p = 1$ ) leads to sparser results than using the  $L_2$  norm, i.e, most  $\alpha_i$  will tend to be equal to 0 [27]. Note that the set of basis functions  $B$  can be chosen to be the set of basis functions associated to terminal nodes or the set of basis functions associated to all nodes (terminal and non terminal).

## 4.2 Generating an Infinite Set of Basis Functions

In case one works with a Tikhonov regularizer, an appealing strategy consists in considering not only basis functions induced by a finite ensemble, but all basis functions that could be induced by any decision tree and let a regularized learner as in (7) finding an optimal solution. The main problem here is that the program (7) will have infinitely many variables and finding a solution seems a priori untractable. However, as we will see, there are some possibilities to overcome this problem.

Consider the optimization problem as stated in (7). Choosing  $p = 2$  and  $L(y, f(x)) = \max(0, 1 - yf(x))$  leads to the well-known SVM optimization problem. Most practical SVM implementations work on the dual formulation of (7)

$$\begin{aligned} \min_{\beta \in \mathbb{R}^n} \quad & \frac{1}{2} \sum_{i=1}^n \sum_{s=1}^n \beta_i \beta_s y_i y_s K(x_i, x_s) - \sum_{i=1}^n \beta_i \quad (8) \\ \text{s.t.} \quad & 0 \leq \beta_i \leq 2/\lambda \\ & \sum_{i=1}^n \beta_i y_i = 0 \end{aligned}$$

<sup>4</sup> As pointed out by a reviewer, the optimization problem presented in (7) can be seen as a particular case of Stacking [31]. The main difference is that instead of using classifiers as new features, we use decision trees' nodes.



$K$  is a kernel function defined as  $K(x, x') = \langle \Phi(x), \Phi(x') \rangle$  where  $\Phi(x)$  is obtained from the feature mapping  $\Phi : X \rightarrow F$  where  $F$  is assumed to be a Hilbert space equipped with the inner product  $\langle \cdot, \cdot \rangle$  [28]. The dual form has the great advantage to enable solving the SVM problem even if  $\Phi$  maps examples into an infinite dimensional space. Indeed, we see from (8) that  $K$  is sufficient to find the optimal solution. This also means that the mapping  $\Phi$  need not to be known if we are sure it exists, which is automatically achieved if the Gram matrix  $\tilde{K}_{i,s} = K(x_i, x_s)$  of dimension  $n \times n$  is always symmetric and semi-positive definite [28].

The use of an SVM makes possible solving (7) for infinitely many variables and consequently, to solve the SVM in the space represented by all possible basis functions induced by any decision tree. To do so, one must find a kernel function which embodies those basis functions. In section 3, we have seen that the kernel corresponding to the basis functions induced by an ensemble of decision trees is equal to the proportion of decision trees letting two examples  $x$  and  $x'$  sharing the same terminal node. The main difference here is that instead of simply counting the number of decision trees letting  $x$  and  $x'$  reaching the same leaf, we must calculate the probability that  $x$  and  $x'$  end in the same terminal node considering all possible decision trees.

Interestingly, Breiman [4] has given an implicit answer to this question. Consider the following assumptions are met : 1) the space of features is entirely bounded, i.e,  $X \subseteq (L_1, R_1) \times (L_2, R_2) \times \dots (L_d, R_d)$  where  $L_j \in \mathfrak{R}$  and  $R_j \in \mathfrak{R}$ ,  $1 \leq j \leq d$ , are respectively the lowest and highest bounds of feature  $X_j$  ; 2) each split of each tree is selected at random, i.e, a feature is first randomly selected, and a split point is then randomly chosen along this feature according to an uniform distribution, 3) the probability measure  $P$  is uniform on the space. Then Breiman [4] showed that if one builds an infinity of decision trees having each  $T$  terminal nodes, the proportion of trees letting  $x$  and  $x'$  sharing the same terminal node is approximately :

$$K_{inf}(x, x') \approx \exp^{-\gamma \|x-x\|_1} \quad (9)$$

where  $\gamma = \log(T)/d$ , where  $d$  the dimensionality of the initial feature space. Interestingly, this is the very well-known Laplacian kernel. A recent and very interesting paper of Lin and Li has pointed a similar result [23]. Their demonstration shows a strict equality between the kernel and the set of basis functions instead of just an approximation as in (9). However, the kernel is derived in a bit different perspective and the relation between the number of terminal nodes and kernel's sharpness (i.e,  $\gamma$ ) is harder to capture. For these reasons, we will keep the framework based on [4]. With an appropriate  $\gamma$ , one could solve the SVM problem using the kernel  $K_{inf}$  to obtain an approximate infinite ensemble classifier working in the (infinite dimensional) space  $H_T$  embedding all basis functions induced by any tree with  $T$  terminal nodes.

Note that the basis functions embedded in  $K_{inf}$  corresponds to basis functions associated to terminal nodes of the trees. It is not evident a priori to choose a good  $\gamma$ . One could proceeds by evaluating the SVM with several values of

$\gamma$  on a test sample or by cross-validation. However, another interesting way of proceeding consists in summing several Laplacian kernels in the following manner

$$K_{Sinf}(x, x') = \sum_{q=2}^Q \exp^{-\gamma_q \|x-x'\|_1} \quad (10)$$

where  $\gamma_q = \log(q)/d$ . It is well-known that summing two kernels embedding respectively  $H_1$  and  $H_2$  leads to a kernel embedding  $H_1 \cup H_2$  (see e.g [23]). As a consequence, solving an SVM program using the kernel  $K_{Sinf}$  in equation (8) leads approximately to search a linear separator in the space represented by all basis functions induced by any tree having a number of leaves from 2 to  $Q$ .

## 5 Experiments

### 5.1 KTA and Random Forests

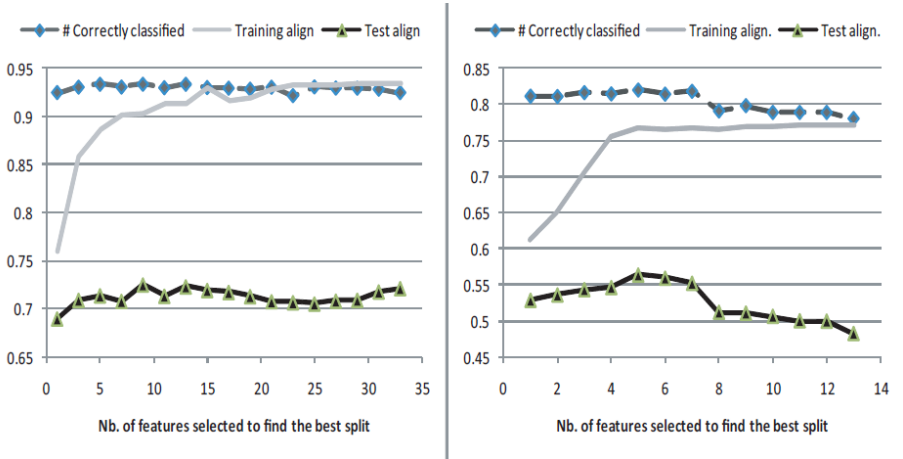
To illustrate the link between KTA and the performance of a randomized trees ensemble, we have run the following experiments on the *Heart* and *Ionosphere* datasets coming from the UCI repository. *Heart* has 13 features and 270 examples, while *Ionosphere* has 351 examples and 34 features. We have randomly splitted each dataset into two subsets of equal size, one used as a training sample and the other as a test sample. One Random Forest of 100 trees [5] has been run for several possible  $d'$  ( $d'$  is the number of features evaluated to find the best split) on the training set. More specifically, we used  $d' = \{1, 3, 5, \dots, 33\}$  for *Ionosphere* and  $d' = \{1, 2, 3, \dots, 13\}$  for *Heart*. Each tree of a Forest was grown until having the purest possible terminal nodes. For each Random Forest, we have calculated its KTA and its error rate on the test sample as well as the KTA on the training sample. Fig. 1 shows the results averaged over 10 runs.

On the *Heart* dataset, we can note an interesting correlation between KTA and the forest's performance. Indeed, the lowest KTA values are met for  $d' > 7$  which corresponds to the poorest performances of the ensemble. The maximal alignment is achieved for  $d' = 5$  which is also the best performance of the ensemble. We clearly see that the higher the level of randomization, the lower the difference between KTA on the training sample and test sample. The results for *Ionosphere* are less conclusive. Indeed, the test error does not vary too much with  $d'$  as well as KTA. We can however note that the higher the level of randomization, the lower the difference between KTA on the learning sample and the test sample as expected.

Of course, more extensive experiments should be carried out to clearly test the relation between KTA and the error rate of a randomized trees ensemble. However, we believe that the framework presented here provides promising perspectives and must be further analyzed.

### 5.2 Comparison of Randomized Trees Ensembles

In this section, we compare the performances of the methodologies presented previously. Our benchmark consists of 10 real-world datasets coming from the



**Fig. 1.** KTA vs. performance of an ensemble. Left : Ionosphere dataset (KTA was multiplied by a factor 1.5 for better lisibility. Right : Heart dataset - KTA was multiplied by a factor 2.

UCI repository, three synthetic datasets (Twonorm, Threenorm and Ringnorm) coming from [3] and three datasets coming from the NIPS 2003 Feature challenge selection [18].

Four classifiers have been tested : a Random Forest of 100 trees [5] (**RF**), the same Random Forest whose basis funtions **associated to terminal nodes** are post-processed according to equation (7) with  $p = 2$  and a hinge loss (**RF-L**), the same Random Forest whose basis funtions **associated to all the nodes** are post-processed according to equation (7) with  $p = 2$  and a hinge loss (**RF-N+L**) and finally an SVM trained with a kernel defined in (10) (**SVM-K**). We have chosen  $p = 2$  and hinge loss for post-processed ensembles in order to have a fair comparison with the SVM trained with (10). The error rates of each method are averaged over 3 trials of 10-fold cross-validation.

All feature elements of all datasets were scaled to  $[0, 1]$  even for RF. For RF, RF-L and RF-N+L, the trees forming the ensemble were grown until having the purest possible leaves. The parameters of all algorithms were searched as to minimize the error estimated by a 5-fold cross-validation on the training set as suggested in [20]. That is, for RF, RF-L and RF-N+L,  $d'$  was chosen within  $\{1, \sqrt{d}, d\}$  which are references in Random Forests [5], and for RF-L, RF-N+L and SVM-K, the regularization parameter  $\lambda$  was searched within the set  $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$ . Finally SVM-K was trained using 3 possible sums of 10 Laplacian kernels. The first sum was chosen to embedd decision trees of depth 1 to 10, the second trees of depth 4 to 13 and the third trees of depth 7 to 16. In binary trees, the relation between depth  $k$  and number of terminal nodes  $T$  is  $T = 2^k$ . Random Forests have been induced using our own implementation, while LIBSVM [8] has been used as SVM soft-margin solver. Results are shown in Table 1.

**Table 1.** Performances of 4 classifiers on several datasets. Results in bold are significantly better than others according to one-sided Student t-test at level 0.05.

Dataset	$n$	$d$	RF	RF-L	RF-N+L	SVM-K
Sonar	208	60	0.846	<b>0.894</b>	<b>0.894</b>	<b>0.905</b>
Breast	569	30	0.963	0.974	0.972	0.975
Ionosphere	351	34	0.940	0.940	0.940	0.945
Pima	768	8	0.754	<b>0.782</b>	<b>0.775</b>	<b>0.772</b>
Musk	476	166	0.917	0.927	0.948	<b>0.979</b>
Heart	270	13	0.807	<b>0.839</b>	<b>0.845</b>	<b>0.856</b>
Vote	435	16	0.960	0.958	0.958	0.940
Australian	690	14	0.801	<b>0.853</b>	<b>0.866</b>	<b>0.858</b>
Spambase	4601	57	0.969	<b>0.982</b>	<b>0.986</b>	<b>0.989</b>
Tic-Tac-Toe	958	9	0.991	1	1	0.996
Twonorm	300	20	0.973	0.971	0.971	0.972
Threernorm	300	20	0.836	0.832	0.832	0.832
Ringnorm	300	20	0.956	<b>0.972</b>	<b>0.972</b>	<b>0.974</b>
Dexter	600	20000	0.905	<b>0.938</b>	<b>0.939</b>	0.897
Arcene	200	10000	0.774	<b>0.836</b>	<b>0.836</b>	<b>0.829</b>
Gisette	7000	5000	0.968	0.978	0.978	0.961

The results show several interesting things. First, post-processing an ensemble of decision trees gives quite systematically lower error rates than non post processed ones. Secondly, RF-L and Rf-N+L share almost indistinguishable performances suggesting that the basis functions induced by terminal nodes are sufficient to learn well.

On both UCI and synthetic datasets, the performances of post-processed ensembles (RF-L and RF-N+L) and SVM-K are very close. Indeed, except on the Musk dataset, there are no statistical differences between both approaches. However, on the high-dimensional datasets coming from the NIPS feature selection challenge, post-processed ensembles tend to outperform SVM-K. This suggests that ensemble of randomized trees are well-suited for high-dimensional data.

Using an infinite ensemble has the advantage of better space covering resulting in a very smooth decision boundary. In the case of a finite ensemble induced by randomized trees strategies, the lack of smoothness seems to be compensated by the search of a subset of interesting basis functions. This can be seen as a feature selection operating directly in the infinite feature space of basis functions induced by all possible decision trees. This feature selection may be an explanation to the strong performance of randomized trees ensembles in very high dimensional data.

The use of a finite ensemble has also the great advantage to give interpretable results. Indeed, each basis function induced by a decision tree can be seen as a rule (see section 2). Post-processing will give a weight to each basis function (i.e, each rule) highlighting the most important ones. Note that one who focus on interpretability should normalize each weight by the support of the corresponding rule (the number of covered examples by the rule) in order to not give too much importance on rules covering a lot of examples (see eg. [16]).

## 6 Conclusion

We analyzed randomized trees ensembles through the basis functions they induce. We pointed out a connection with the kernel target alignment and Parzen window classifiers. The connection can be used to bound the generalization error of an ensemble and gives some insights about the performance of randomized trees ensembles. Experiments realized in this paper showed that it seems to have an empirical relationship between KTA and the ensemble performance. This connection highlights the role of classifiers diversity as well as individual strength. We also showed that increasing the amount of randomization has the effect to better regularize the ensemble. We should however be careful when analyzing the relation between level of randomization and strength-diversity tradeoff. Indeed, increasing the amount of randomization does not necessarily imply increasing the diversity or decreasing the strength and vice-versa. Open questions and interesting future aspects are : 1) how one can find another ensemble strategy acting more intensively on KTA, 2) realizing more experiments to deeper test the relation between KTA and the performance of an ensemble, 3) if it is possible to generalize the KTA framework to other ensemble approaches such as Boosting algorithms.

We have also suggested two possible improvements of classical randomized ensembles strategies. The first one consists in post-processing the ensemble with powerful linear separators. In our experiments, post-processing always led to better performance than a classical majority voting. Another alternative consists in taking into account the set of all possible basis functions induced by any decision tree having a specified number of leaves. This is possible thanks to the use of an appropriate kernel embedding those basis functions. Experimentally, both “improvements” gives very similar results. Possible future works here are : 1) should we benefit of using another post-processing strategy which uses a penalty on the  $L_1$  norm of regressors. The main advantage could lie in the sparsity property of such regularizers leading in more interpretable results. 2) Breiman [4] showed that in a space of  $d$  dimensions, there exists a set of weights  $w_1, \dots, w_\infty$  which applied to all possible decision trees having  $d + 1$  terminal nodes converges to the Bayes rate. Interestingly, the approach presented in section 4, as well as the one of [23] enters completely in this framework and gives perhaps a possible way of finding such weights. A critical step here would be to study the consistency of SVMs when used with Laplacian kernels.

## References

1. Biau, G., Devroye, L., Lugosi, G.: Consistency of random forests and other averaging classifiers. *Journal of Machine Learning Research* 9, 2015–2033 (2008)
2. Breiman, L.: Bagging predictors. *Machine Learning* 24, 123–140 (1996)
3. Breiman, L.: Bias, variance and arcing classifiers (1996), <http://www.sasenterpriseminer.com/documents/arcing.pdf>

4. Breiman, L.: Some infinity theory for predictor ensembles (2000), <http://www.stat.berkeley.edu>
5. Breiman, L.: Random forests. *Machine Learning* 45, 5–32 (2001)
6. Breiman, L., Friedman, J., Olshen, R., Stone, C.: *Classification and Regression Trees*. Wadsworth and Brooks (1984)
7. Buntine, W., Niblett, T.: A further comparison of splitting rules for decision tree induction. *Machine Learning* 8, 75–85 (1992)
8. Chang, C.-C., Lin, C.-J.: Libsvm: A library for support vector machines (2001), Software available at, <http://www.csie.ntu.edu.tw/~cjlin/libsvm>
9. Cristianini, N., Kandola, J., Elisseeff, A., Shawe-Taylor, J.: On kernel-target alignment. In: Holmes, D., Jain, L. (eds.) *Innovations in Machine Learning: Theory and Application*, pp. 205–255 (2006)
10. Cutler, A., Zhao, G.: Pert - perfect random tree ensembles. *Computer Science and Statistics* (2001)
11. Demiriz, A., Bennett, K., Shawe-Taylor, J.: Linear programming boosting via column generation. *Machine Learning* 46, 225–254 (2002)
12. Dietterich, T.: An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning* 40, 139–157 (2000)
13. Freund, Y., Schapire, R.: Experiments with a new boosting algorithm. In: *ICML*, pp. 148–156 (1996)
14. Freund, Y., Schapire, R.: A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence* 14, 771–780 (1999)
15. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. *The Annals of Statistics* 38, 95–118 (2000)
16. Friedman, J., Popescu, B.: Predictive learning via rule ensembles. *The Annals of Applied Statistics* 2, 916–954 (2008)
17. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. *Machine Learning* 63, 3–42 (2006)
18. Guyon, I., Gunn, S., Ben-Hur, A., Dror, G.: Result analysis of the nips 2003 feature selection challenge. In: Saul, L.K., Weiss, Y., Bottou, L. (eds.) *Advances in Neural Information Processing Systems*, vol. 17, pp. 545–552. MIT Press, Cambridge (2005)
19. Hastie, T., Tibshirani, R., Friedman, J.: *The elements of statistical learning: Data Mining, Inference and Prediction*. Springer, Heidelberg (2009)
20. Hsu, C.-W., Chang, C.-C., Lin, C.-J.: A practical guide to support vector classification (2003), <http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>
21. Kleinberg, E.: On the algorithmic implementation of stochastic discrimination. *IEEE Trans. Pattern Anal. Mach. Intell.* 22, 473–490 (2000)
22. Kuncheva, L., Whitaker, C.: Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning* 51, 181–207 (2003)
23. Lin, H.-T., Li, L.: Support vector machinery for infinite ensemble learning. *Journal of Machine Learning Research* 9, 941–973 (2008)
24. Liu, F.-T., Ting, K.-M., Yu, Y., Zhou, Z.-H.: Spectrum of Variable-Random Trees. *Journal of Artificial Intelligence Research* 32, 355–384 (2008)
25. Quinlan, J.: *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco (1993)

26. Ratsch, G., Onoda, T., Muller, K.-R.: Soft margins for adaboost. *Machine Learning* 42, 287–320 (2001)
27. Rosset, S., Zhu, J., Hastie, T.: Boosting as a regularized path to a maximum margin classifier. *Journal of Machine Learning Research* 5, 941–973 (2004)
28. Scholkopf, B., Smola, A.: *Learning with Kernels*. MIT Press, Cambridge (2001)
29. Utgoff, P., Clouse, J.: A kolmogorov-smirnov metric for decision tree induction (1996), <http://www.ics.uci.edu/~mlearn/MLRepository.html>
30. Vapnik, V.: *The Nature of Statistical Learning Theory*. Springer, Heidelberg (1995)
31. Wolpert, D.: Stacked Generalization. *Neural Networks* 5, 241–259 (1992)

# Entropy and Margin Maximization for Structured Output Learning

Patrick Pletscher, Cheng Soon Ong, and Joachim M. Buhmann

Department of Computer Science, ETH Zürich, Switzerland

**Abstract.** We consider the problem of training discriminative structured output predictors, such as conditional random fields (CRFs) and structured support vector machines (SSVMs). A generalized loss function is introduced, which jointly maximizes the entropy and the margin of the solution. The CRF and SSVM emerge as special cases of our framework. The probabilistic interpretation of large margin methods reveals insights about margin and slack rescaling. Furthermore, we derive the corresponding extensions for latent variable models, in which training operates on partially observed outputs. Experimental results for multiclass, linear-chain models and multiple instance learning demonstrate that the generalized loss can improve accuracy of the resulting classifiers.

## 1 Introduction

In structured output prediction, the model predicts a discrete output  $y \in \mathcal{Y}$  given an input  $x \in \mathcal{X}$ . The output domain usually consists of multiple variables, this often renders prediction as a computationally intensive problem. Applications include multiclass and multilabel classification, part-of-speech tagging, and image segmentation. In applications such as part-of-speech tagging or image segmentation, prediction consists of a sequence of tags or a grid of labels, respectively. In this paper we focus on *training* such structured classifiers. The loss function, the key component of training, measures the quality of fit of the model predictions to the training outputs. In the literature, the two most prominent losses are the log-loss and the max-margin loss. The log-loss is used in conditional random fields (CRFs) [1]. The max-margin loss is utilized in structured Support Vector Machines (SSVMs) [2, 3].

Our contributions in this work are as follows. We integrate the concept of a margin and an inverse temperature into CRFs. This leads to a novel family of loss functions for structured output learning. We show that CRF and SSVM are two special cases of this formulation. The dual of this objective sheds new light on the different structured output learning approaches and simplifies their comparison. Furthermore, we show how unobserved (latent) output variables can be integrated into this framework. Finally, we conduct a number of experiments which show that our suggested objective outperforms log-loss and max-margin loss on a number of synthetic and real world data sets.



## 2 Structured Output Learning

Following the setting in [4], we consider a linear prediction rule in a joint input/output space  $\mathcal{H}$ . An input/output mapping  $\phi(x, y) : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{H}$  is specified by domain experts, either explicitly by supplying  $\phi(x, y)$  or implicitly by specifying a graphical model together with the parametrization of its factors. The score of an input/output pair is defined as the inner product of a parameter vector  $w$  and  $\phi(x, y)$ . For a new input  $x$ , the inference method predicts the output  $y^*$  with the largest score

$$y^* = \operatorname{argmax}_{y \in \mathcal{Y}} \langle w, \phi(x, y) \rangle. \quad (1)$$

Depending on  $\mathcal{Y}$  and the structure of  $\phi(x, y)$ , the computational complexity of this maximization ranges from linear complexity in the number of output variables, to NP-hardness.

During training, a data set  $\mathcal{D} = \{(x^{(n)}, y^{(n)})\}_{n=1}^N$  of  $N$  pairs is given. The learning task is to find the parameter  $\hat{w}$  that best predicts the outputs given the inputs. To prevent overfitting, the goodness-of-fit measure is often complemented by a regularizer on  $w$ . Here we use the  $\ell_2$  regularizer and denote it by  $\|\cdot\|_2$ . For a given  $w$ , the loss on the  $n$ -th example is measured by the function  $\ell(w, x^{(n)}, y^{(n)})$ . The regularized risk of a  $w$  for a given dataset  $\mathcal{D}$  amounts to

$$\mathcal{L}_\ell(w; \mathcal{D}, C) = \sum_{n=1}^N \ell(w, x^{(n)}, y^{(n)}) + \frac{C}{2} \|w\|_2^2,$$

where  $C$  is the regularization constant. In training, the empirical risk minimization principle chooses the parameter  $\hat{w}$  with the smallest loss, i.e.,

$$\hat{w} = \operatorname{argmin}_w \mathcal{L}_\ell(w; \mathcal{D}, C). \quad (2)$$

Algorithmic details of this minimization problem are given in Section 6. In the first part of this paper we concentrate on the choice of the loss  $\ell$ .

## 3 Unification of Log-Loss and Max-Margin-Loss

We will now formulate our generalized loss. First, the CRF log-loss is modified through incorporating an *inverse temperature* parameter. The concept of a margin is introduced into this modified loss, resulting in a new family of loss functions. Both the SSVM and the CRF are special cases of this formulation.

In CRFs we consider a log-linear model

$$P(y|x, w) = \frac{1}{Z(x, w)} \exp(\langle w, \phi(x, y) \rangle),$$

with the partition sum

$$Z(x, w) = \sum_{y' \in \mathcal{Y}} \exp(\langle w, \phi(x, y') \rangle).$$

The log-loss can be derived as the negative log-likelihood of the probabilistic conditional model

$$\ell_{LL}(w, x, y) = -\log P(y|x, w) = -\langle w, \phi(x, y) \rangle + \log Z(x, w).$$

Using the log-loss in the regularized training objective in Equation (2) corresponds to maximum-a-posteriori (MAP) parameter estimation, where we assume a Gaussian prior on  $w$ .

The maximum margin principle gives rise to an alternative choice for a structured loss which is employed in the SSVM. The ground-truth output is compared to the output that maximizes the inner product

$$\ell_{MM}(w, x, y) = -\langle w, \phi(x, y) \rangle + \max_{y' \in \mathcal{Y}} [\langle w, \phi(x, y') \rangle + \Delta(y', y)]. \quad (3)$$

Here,  $\Delta(y', y)$  ensures a margin between the ground-truth output  $y$  and an output  $y'$ .  $\Delta(y', y)$  will be discussed in more detail in Section 3.2 and 3.3.

### 3.1 Inverse Temperature

We now introduce a parameter into the log-linear model of the CRF which allows us to control the sharpness of the distribution. For the posterior, we consider the Gibbs distribution with an inverse temperature  $\beta \in \mathbb{R}^+$ :

$$P_\beta(y|x, w) = \frac{1}{Z_\beta(x, w)} \exp(\beta \langle w, \phi(x, y) \rangle), \quad (4)$$

with normalization constant

$$Z_\beta(x, w) = \sum_{y' \in \mathcal{Y}} \exp(\beta \langle w, \phi(x, y') \rangle).$$

For  $\beta = 1$  this reverts to the standard CRF. The inverse temperature  $\beta$  does not have any influence on the MAP prediction for an input  $x$ . However, note that the learning objective is now changed. For reasons that will become clear later on, we choose to scale the per-example loss by  $1/\beta$ . The negative log-loss for an instance  $(x, y)$  thus becomes

$$-\frac{1}{\beta} \log P_\beta(y|x, w) = -\langle w, \phi(x, y) \rangle + \frac{1}{\beta} \log \sum_{y' \in \mathcal{Y}} \exp(\beta \langle w, \phi(x, y') \rangle). \quad (5)$$

Rearranging terms, it can be shown that the introduction of  $\beta$  is equivalent to changing the regularizer in a standard CRF objective to  $C' = C/\beta$  (see supplement 1). Hence without further modification to the loss,  $\beta$  is simply redundant.

### 3.2 Large Margin Learning

A standard CRF considers unbiased output distributions. Motivated by the concept of large margin learning, we bias the conditional distribution of outputs  $y'$ ,

---

<sup>1</sup> Supplement and source code can be obtained from the first author's [website](#).

given the ground-truth output  $y$ , to have a large margin for outputs  $y'$  that are dissimilar. To do so, we assume that a non-negative error term is given:

$$\Delta(y', y) = \begin{cases} 0 & \text{if } y' = y \\ \geq 0 & \text{otherwise.} \end{cases} \quad (6)$$

The error term  $\Delta(y', y)$  specifies a preference on the outputs  $y'$  when compared to the ground-truth output  $y$ . In the coming subsection we will incorporate the margin principle of SVMs into the conditional probabilistic model given in Equation (4). For applications in which the output can be thought of as a labeling, a common choice for the error term is the Hamming distance of the two labelings  $y$  and  $y'$ .

### 3.3 Combining the Posterior and Error Term

The training phase exploits two sources of information:  $\Delta(y', y)$  and  $P_\beta(y'|x, w)$ . In principle, there are many choices for combining the two sources over the same output variable  $y'$ . Here, we specifically discuss two choices corresponding to slack and margin rescaling in SSVM [2].

**Margin rescaling.** For a given ground-truth output  $y$ , the error terms are transformed into conditional probabilities over outputs:

$$P_\beta(y'|y) = \frac{1}{Z_\beta(y)} \exp(\beta\Delta(y', y)), \quad (7)$$

with corresponding partition sum  $Z_\beta(y)$ . For outputs  $y'$  which are very different from the ground-truth  $y$ ,  $P(y'|y)$  is large. In training this is used to make such outputs to be difficult to separate, forcing the classifier to ensure good classification on these outputs. The first option of combining the posterior and error term is by multiplying (4) and (7).

$$P_\beta(y'|y, x, w) \propto P(y'|x, w)P(y'|y)$$

Ensuring normalization of the probability distribution leads to

$$P_\beta(y'|y, x, w) = \frac{1}{Z_\beta(y, x, w)} \exp(\beta\langle w, \phi(x, y') \rangle + \beta\Delta(y', y)), \quad (8)$$

where the partition sum is given by

$$Z_\beta(y, x, w) = \sum_{y'' \in \mathcal{Y}} \exp(\beta\langle w, \phi(x, y'') \rangle + \beta\Delta(y'', y)).$$

Note that the distribution of an output  $y'$  is now conditioned on the true output  $y$ . We do this to ensure good separation of  $y$  to outputs  $y'$  that are unfavourable according to  $\Delta(y', y)$ . In Section 4 we show that combining the two posteriors by means of a product, corresponds to *margin rescaling* in the SSVM case.

For convenience, the error term is absorbed into the feature map by including  $\Delta(y', y)$  as an additional feature:  $\phi_\Delta(x, y', y) = [\phi(x, y')^T, \Delta(y', y)]^T$ . The  $w$  needs to be adjusted accordingly by  $w_\Delta = [w^T, 1]^T$ . The score of the ground-truth output  $y$  remains unchanged by the introduction of the error term, i.e.,  $\langle w, \phi(x, y) \rangle = \langle w_\Delta, \phi_\Delta(x, y, y) \rangle$ , as  $\Delta(y, y) = 0$ .

Under this transformation, the loss of an example  $(x, y)$  is defined as the negative log-likelihood of the conditional probability in Equation (8). As before, rescaling the loss by  $1/\beta$  yields

$$\ell_\beta(w, x, y) = -\langle w_\Delta, \phi_\Delta(x, y, y) \rangle + \frac{1}{\beta} \log \sum_{y' \in \mathcal{Y}} \exp\left(\beta \langle w_\Delta, \phi_\Delta(x, y', y) \rangle\right). \quad (9)$$

In this paper we advocate  $\ell_\beta(w, x, y)$  as a loss for structured outputs, generalizing both CRF and SSVM.

**Slack rescaling.** An alternative option for combining the conditional probability  $P_\beta(y'|x, w)$  with the error term  $\Delta(y', y)$ , corresponds to slack rescaling in the SSVM. Let us define  $g(x, y', y) = \phi(x, y') - \phi(x, y)$ . Then

$$P_{\beta, \text{slack}}(y'|y, x, w) = \frac{1}{Z_{\beta, \text{slack}}(y, x, w)} \exp\left(\beta \left(1 + \langle w, g(x, y', y) \rangle\right)\right)^{\Delta(y', y)},$$

with corresponding partition sum  $Z_{\beta, \text{slack}}(y, x, w)$ . This results in a scaled, negative log likelihood that corresponds to the multiplicative factor in *slack rescaling*.

$$\ell_{\beta, \text{slack}}(w, x, y) = \frac{1}{\beta} \log \sum_{y' \in \mathcal{Y}} \exp\left(\beta \Delta(y', y) \left(1 + \langle w, g(x, y', y) \rangle\right)\right).$$

Note that in this form there is no ground-truth term in front of the sum over all the outputs  $y'$ . Again, the error term corresponds to a modification of the feature map. Thus, we arrive at Equation (9) where  $\phi_\Delta(x, y', y) = \Delta(y', y)[g(x, y', y)^T, 1]^T$  and  $w_\Delta = [w^T, 1]^T$ . The reader should notice the non-linear nature of this combination, which makes slack rescaling more challenging than margin rescaling.

The probabilistic interpretation of margin rescaling is more appealing due to the factorization into two posterior distributions. We will therefore concentrate our analysis on margin rescaling. Nevertheless, most of the findings also hold for slack rescaling.

## 4 Connections to Maximum Entropy and Maximum Margin Learning

In this section we will analyze the implications of our loss in Equation (9). Observe that we recover the standard CRF loss by setting  $\beta = 1$  and using an error term  $\Delta(y', y) = 0 \forall y'$ . We start our analysis by first considering the limit case of  $\beta \rightarrow \infty$  which leads to a probabilistic interpretation of the SSVM. We then derive the dual, which shows a joint regularization by entropy and margin.

#### 4.1 SSVMs as a Limit Case for $\beta \rightarrow \infty$

**Lemma 1.** *The standard max-margin loss used in SSVMs is obtained for the choice  $\beta \rightarrow \infty$ .*

*Proof.* The SSVM is derived as a limit case of  $\ell_\beta(w, x, y)$  for  $\beta \rightarrow \infty$  by adopting the log-sum-exp “trick”, commonly used for stable numerical evaluation of partition sums. The key idea is to factor out the maximum contribution of the partition sum. Denote by  $y^* = \operatorname{argmax}_{y'} \langle w_\Delta, \phi_\Delta(x, y', y) \rangle$  the output with the largest score. Substituting into the second part of the loss yields

$$\begin{aligned} \frac{1}{\beta} \log Z_\beta(y, x, w) &= \langle w_\Delta, \phi_\Delta(x, y^*, y) \rangle \\ &+ \frac{1}{\beta} \log \sum_{y' \in \mathcal{Y}} \exp \left( \beta \left( \langle w_\Delta, \phi_\Delta(x, y', y) \rangle - \langle w_\Delta, \phi_\Delta(x, y^*, y) \rangle \right) \right). \end{aligned}$$

The second term becomes zero when  $\beta \rightarrow \infty$ , as the only terms in the sum that do not vanish, are outputs with exactly the same score as the maximum output  $y^*$ . These terms evaluate to 1. Note that the number of maxima is independent of  $\beta$ . The complete loss for  $\beta \rightarrow \infty$  becomes

$$\ell_\infty(w, x, y) = -\langle w_\Delta, \phi_\Delta(x, y, y) \rangle + \max_{y' \in \mathcal{Y}} \langle w_\Delta, \phi_\Delta(x, y', y) \rangle,$$

which recovers the loss of the SSVM in Equation (3). The presented analysis is a direct consequence of Theorem 8.1 in [5] applied to Equation (8).

A comparison of CRFs and SSVMs reveals two important differences. First, the maximum-margin loss is only affected by the output that minimizes the distance to the ground-truth output. All the other outputs are discarded. Second, the error-term  $\Delta(y', y)$ , which does not exist in CRFs, provides a degree of freedom to specify how much loss a given output  $y'$  should incur given the ground-truth  $y$ .

#### 4.2 Special Case: Binary Classification

To illustrate the new loss, we discuss the special case of binary classification where  $y \in \{-1, +1\}$ . For binary classification, the feature map  $\phi(x, y) = \frac{1}{2}y\phi(x)$ , transforms the loss to

$$\ell_\beta(w, x, y) = \langle w, \phi(x, y) \rangle - \frac{1}{\beta} \log \left( \exp(\beta \langle w, \phi(x, y) \rangle) + \exp(\beta(\langle w, \phi(x, y') \rangle + \Delta)) \right).$$

Where  $y'$  denotes the wrong label  $y' = -y$ . The standard SVM emerges in the limit  $\beta \rightarrow \infty$  and  $\Delta = 1$ . The parameter choice  $\beta = 1$  and  $\Delta = 0$  yields the Logistic Regression (LR) classifier. Different instantiations of this loss are visualized in Fig. 1, including the log-loss and the max-margin loss.

For the special case of binary classification, the influence of the inverse temperature on  $\ell_\beta(x, y, w)$  was in parts discussed in [6]. In our work we focus on classifiers for structured outputs. In this setting the effective number of negative outputs can be exponentially large, which makes the analysis more complex.

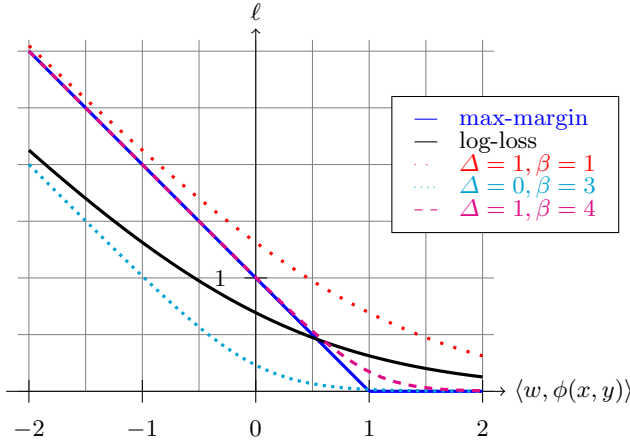


Fig. 1.  $\ell_\beta(w, x, y)$  for different  $\beta$  compared to log-loss and max-margin loss

### 4.3 Regularization by Entropy and Margin

The dual of our new loss can be found by using the method of Lagrange, resulting in Lemma 2. The derivations are similar as in [7], and the details are in the supplement.

**Lemma 2.** *The dual minimization problem corresponding to Equation (2) using our generalized per-example loss  $\ell_\beta(w, x, y)$ , is given by*

$$\begin{aligned} \min_u \quad & \frac{1}{2C} u^T A u - b^T u + \frac{1}{\beta} \sum_{n=1}^N \sum_{y \in \mathcal{Y}} u_{n,y} \log u_{n,y} \\ \text{s.t.} \quad & u_{n,y} \geq 0 \quad \text{and} \quad \sum_{y \in \mathcal{Y}} u_{n,y} = 1 \quad \forall y, n \end{aligned} \quad (10)$$

where  $u_{n,y}$  denotes the dual variable for the output  $y$  in training example  $n$  and  $A$  is given by  $A_{(n_1,y),(n_2,y')} = \langle g_{n_1,y}, g_{n_2,y'} \rangle$ . The difference between two mapped outputs is denoted by  $g_{n,y} = -g(x^{(n)}, y, y^{(n)}) = \phi(x^{(n)}, y^{(n)}) - \phi(x^{(n)}, y)$ . Furthermore, all the possible error terms are collected in a vector  $b$ :  $b_{n,y} = \Delta(y, y^{(n)})$ . A total of  $N \cdot |\mathcal{Y}|$  dual variables are required. The primal and dual variables are related by

$$w = \frac{1}{C} \sum_{n=1}^N \sum_{y \in \mathcal{Y}} u_{n,y} g_{n,y}.$$

The dual in Equation (10) reveals a double regularization of  $\ell_\beta(w, x, y)$  by a margin term and an entropy term. Unsurprisingly, the log-loss and max-margin loss can also be identified as special cases in the dual: if  $b$  is the zero vector, we obtain the dual of the standard CRF, if  $\beta \rightarrow \infty$  the dual of the SSVM.

#### 4.4 The Effect of the Inverse Temperature $\beta$

So far, we argued that in order to reconstruct the log-loss from  $\ell_\beta$ , the parameters  $\beta = 1$  as well as a zero error term  $\Delta(y', y)$  need to be used. However, the dual in Equation (10) shows that it is actually sufficient to only alter the inverse temperature  $\beta$  and the regularization parameter  $C$ , but not the error term itself. For a sufficiently small  $C$  and  $\beta$ , the error term contribution  $-b^T u$  becomes negligible compared to the first and third terms. As a result we identify the CRF dual.

As we have seen,  $\beta$  changes the sharpness of the conditional probability  $P_\beta(y'|y, x, w)$ . For  $\beta \rightarrow 0$  all outputs  $y'$  have an uniform distribution, i.e.,  $P_\beta(y'|y, x, w)$  has an entropy of  $\log(|\mathcal{Y}|)$ . For  $\beta \approx 1$  the distribution behaves as a Gibbs distribution. For large values of  $\beta$  the probability mass concentrates on the outputs with the largest scores. Probabilities on the outputs are in this case not well-defined; the distribution consists of individual, scaled Dirac impulses at the outputs  $y^*$  with maximum scores. These findings are in line with [8], where SVMs are shown to be incapable of estimating conditional probabilities in a multiclass setting.

#### 4.5 Choosing $\beta$

At this point it is natural to ask: “What is the best choice for  $\beta$ ?” Ideally,  $\beta$  is optimized based on the training data. However, looking at the dual in Equation (10), a model order selection question arises. By naively minimizing the loss w.r.t.  $\beta$ , this would always result in choosing  $\beta \rightarrow \infty$ , which is not desired. We thus advocate determining  $\beta$  via cross validation on hold out data.

## 5 Latent Variables

We now turn our attention to structured classifiers for partially observed data. Two training objectives have been suggested for this more challenging setting: The Hidden Conditional Random Field (HCRF) [9], and the Latent Support Vector Machine [10]. Here we show that our formulation also allows for this scenario. Incorporating hidden variables into the output is an important extension of practical relevance: some outputs might for practical reasons be unobservable or one might define a hidden cause that leads to better accuracy of the predictions. Let us denote the observed variables by  $y$  and the hidden, unobserved output variables (latent variables) by  $z \in \mathcal{Z}$ . In HCRFs, the conditional probability of observing  $y$  and  $z$  are modeled using a Gibbs distribution:

$$P_\beta(y, z|x, w) = \frac{1}{Z_\beta(x, w)} \exp\left(\beta \langle w, \phi(x, y, z) \rangle\right).$$

Here, we directly include the inverse temperature  $\beta$ ;  $\beta = 1$  recovers the standard HCRF [9]. The model predicts according to

$$y^* = \operatorname{argmax}_{y \in \mathcal{Y}} \sum_{z \in \mathcal{Z}} P_\beta(y, z|x, w).$$

Comparing this to the fully observed prediction rule in Equation (III), we see that hidden variables are marginalized out. The introduction of the error terms into the Gibbs distribution by multiplying the two posterior distributions yields

$$P_\beta(y', z|y, x, w) = \frac{1}{Z_\beta(y, x, w)} \exp\left(\beta\langle w_\Delta, \phi_\Delta(x, y', z, y) \rangle\right).$$

with  $\phi_\Delta(x, y', z, y) = [\phi(x, y', z)^T, \Delta(y', y)]^T$ . Here it is assumed that  $\Delta(y', y)$  is only dependent on observed output variables. As in the CRF, training of the parameters is performed by minimizing the regularized negative log-likelihood, scaled by  $1/\beta$ . However, for the partially observed case, the hidden variables  $z$  have to be integrated out. This leads to

$$\begin{aligned} \ell_\beta(w, x, y) = & -\frac{1}{\beta} \log \sum_{z \in \mathcal{Z}} \exp\left(\beta\langle w_\Delta, \phi_\Delta(x, y, z, y) \rangle\right) \\ & + \frac{1}{\beta} \log \sum_{\substack{y' \in \mathcal{Y} \\ z' \in \mathcal{Z}}} \exp\left(\beta\langle w_\Delta, \phi_\Delta(x, y', z', y) \rangle\right). \end{aligned}$$

Taking the limit for  $\beta \rightarrow \infty$  and using the log-sum-exp “trick”, the Latent SVM (10) loss emerges:

$$\ell_\infty(w, x, y) = -\max_{z \in \mathcal{Z}} \langle w_\Delta, \phi_\Delta(x, y, z, y) \rangle + \max_{\substack{y' \in \mathcal{Y} \\ z' \in \mathcal{Z}}} \langle w_\Delta, \phi_\Delta(x, y', z', y) \rangle.$$

Again, the Latent SVM can be seen as a probabilistic model, in which all the probability mass is concentrated on the  $y, z$  combination with the largest score. The limit case of the inverse temperature also changes the prediction for new test data to

$$y^* = \operatorname{argmax}_{y \in \mathcal{Y}, z \in \mathcal{Z}} \langle w, \phi(x, y, z) \rangle.$$

Instead of marginalizing the hidden variables out, we now maximize them out. The introduction of latent variables in general turns the empirical risk minimization in Equation (2) into a non-convex optimization problem.

## 6 Algorithmic Issues

So far we have focused on the theoretical comparison of the different losses for structured output prediction. In this section we will discuss issues that are important for an actual implementation.

### 6.1 Minimization of the Objective

Our new loss  $\ell_\beta(w, x, y)$  is both convex (for the completely observed case) and smooth for any inverse temperature except when  $\beta \rightarrow \infty$ , thus standard conjugate-gradient or LBFSG solvers are applicable for the minimization of the



loss. In our implementations we use the LBFGS solver of `minFunc`<sup>2</sup>. This is contrary to the minimization of the standard max-margin objective, where special algorithms for non-differentiable minimization problems are required. For learning with  $\ell_\beta(w, x, y)$ , we are also interested in its derivative w.r.t.  $w$ . For fully observed data and margin rescaling, the gradient takes a form similar to that of standard CRFs:

$$\frac{\partial \ell_\beta(w, x, y)}{\partial w} = -\phi(x, y) + \sum_{y' \in \mathcal{Y}} P_\beta(y'|y, x, w)\phi(x, y').$$

In our implementation we use the gradient information for the efficient minimization of the loss. The LBFGS algorithm computes an approximation to the Hessian of the objective. For small  $\beta$ , this second-order information drastically improves the running time of the training. For large  $\beta$ , the Hessian does not help as the objective becomes essentially piecewise linear.

## 6.2 Efficient Inference in Training

One key step in the optimization of the objective function is the evaluation of the log-partition sum  $Z_\beta(y, x, w)$ , which is generally computationally intractable. There exist cases, like for example a  $\phi(x, y)$  that corresponds to a tree structured graphical model, where the computation of  $Z_\beta(y, x, w)$  can be performed efficiently. The SSVM instead requires computing the maximum violating output  $y^* = \operatorname{argmax}_{y' \in \mathcal{Y}} \langle w_\Delta, \phi_\Delta(x, y') \rangle$ . Both tasks in general are computationally hard, but there exist classes of problems where the maximization is tractable, but not the computation of the partition sum. This is for example the case if submodularity constraints are imposed on the potentials of a general graphical model.

## 7 Related Work

Since [6], there have been various attempts to unify the max-margin and log losses. The connections between SVMs and exponential families have been indicated in [11], and our work makes the link between the log-loss and max-margin loss more explicit through the inverse temperature and also extends to structured classifiers and latent variables. In [12] an algorithm for learning multiclass SVMs in the primal is discussed: The max-margin loss is approximated by a soft-max, which can then be optimized by a conjugate-gradient solver. [13] considers a loss function similar to ours, applied to multiclass SVM.

Two recent papers have appeared which combine the benefits of both the margin idea and the probabilistic model. In [14], a convex combination of log-loss and max-margin loss was proposed. The authors prove Fisher consistency and PAC-Bayes bounds for the resulting classifiers. We conjecture that our model shares many of the advantages of their hybrid model with the additional advantage that

<sup>2</sup> <http://people.cs.ubc.ca/~schmidtm/Software/minFunc.html>

it allows for a probabilistic interpretation. Independently, the softmax-margin was developed in [15]. The proposed loss and ours are very similar in spirit: both introduce the margin concept known from SSVMs also into CRFs. In the application of named-entity recognition which they consider, the margin term shows to improve the accuracy of the classifier. However, the connection between CRFs and SSVMs is not established.

## 8 Experiments

In our experiments we will only consider settings with either a small number of outputs  $|\mathcal{Y}|$ , or where inference can be performed exactly, such as scenarios where the feature map  $\phi(x, y)$  corresponds to a chain structured graphical model.

### 8.1 Multiclass Learning

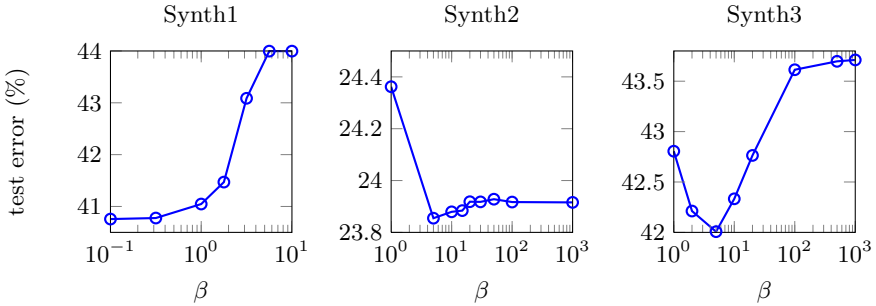
As a first experiment we consider the well-studied multiclass setting in which a data point is assigned to one of  $K$  classes. The feature map  $\phi(x, y)$  as introduced in [16] is used,

$$\phi(x, y) = \begin{bmatrix} \delta_1(y) \cdot x \\ \delta_2(y) \cdot x \\ \vdots \\ \delta_K(y) \cdot x \end{bmatrix}.$$

Here  $\delta_k(y)$  denotes the Kronecker Delta function, which is 1 for  $y = k$  and 0 everywhere else. For all the multiclass experiments, we report the results of the `liblinear`<sup>3</sup> implementation of LR and SVM as baseline classifiers.

**Synthetic data.** We designed three synthetic datasets with the reasoning in Section 4.4 in mind. Each of the datasets shows different characteristics, which can be exploited by the losses. The first dataset, **Synth1**, consists of three classes. Each class is sampled from a Gaussian with means at 0, 1 and 2 and variance 1. We would expect a small  $\beta$  to perform best on this dataset, as the classes overlap to a large extent. The second dataset, **Synth2**, consists of three classes. Each class is sampled from a Gaussian with means at (0, 0), (1, 0.1) and (1, -0.1), each with covariance  $0.25I$ . Here, the prediction error is computed by accounting only 0.1 for a confusion between class 2 and 3, and 1 otherwise. This information is provided to the classifier using the error term. We expect the best results with a large  $\beta$ , as the error term information is crucial. The third dataset, **Synth3**, consists of four Gaussians. Two of which have means (0, 0) and (1, 0), the remaining two have almost indistinguishable means of (0.5, 0.4) and (0.5, 0.6). All classes have a covariance of  $I$ . Again, for the indistinguishable classes we only account an error of 0.1 when confusing them. Here we would expect an intermediate value of  $\beta$  to lead to the best results, as both noise and skewed class importance are present. The training set consists of 2000 examples

<sup>3</sup> <http://www.csie.ntu.edu.tw/~cjlin/liblinear/>



**Fig. 2.** Results on the different synthetic multiclass datasets. Changing the parameter  $\beta$  leads to different test errors.

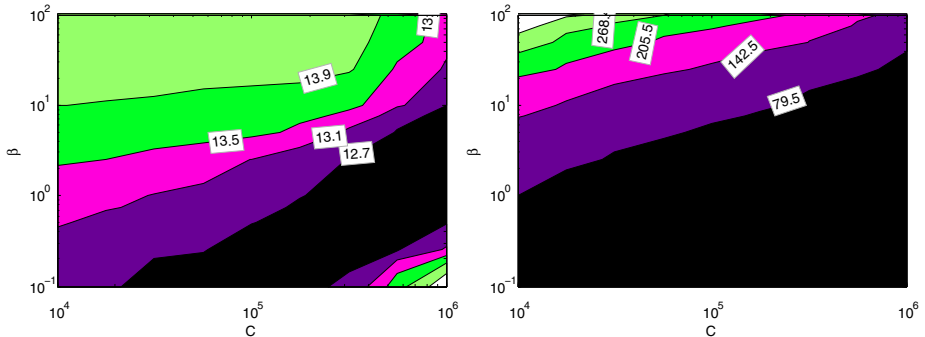
for each class, the test test of 10000 examples for each class. The test error is averaged over 5 random instantiations of the data set. For all classifiers  $C = 1$  is fixed, as there is enough data to prevent overfitting. The results of this experiment are shown in Table 1 and in Fig. 2.

We observe that the inverse temperature can have a substantial influence on the accuracy of the resulting classifier. No value of  $\beta$  is optimal for all three datasets, which is in agreement with the discussion in Section 4.4. The experiment also shows that the limit case of a SVM for  $\beta \rightarrow \infty$  is already achieved for a relatively small  $\beta$ .

**Table 1.** Synthetic multiclass results. The first row corresponds to a LR, the third row to a SVM. The second row is a specific instance of the novel loss. For the `liblinear` SVM we use the 0/1 error term (and not the ones described in the synthetic data generation) and thus inferior results are expected for Synth2 and Synth3.

$\beta$	loss	test error (%)		
	$\Delta(y', y)$	Synth1	Synth2	Synth3
1	no	<b>41.0</b> $\pm$ 0.4	25.8 $\pm$ 0.2	43.9 $\pm$ 0.2
5	yes	44.0 $\pm$ 0.1	<b>24.2</b> $\pm$ 0.1	<b>42.0</b> $\pm$ 0.2
10 <sup>6</sup>	yes	44.0 $\pm$ 0.1	<b>24.2</b> $\pm$ 0.1	43.7 $\pm$ 0.7
<code>liblinear</code> LR		41.7 $\pm$ 0.3	26.5 $\pm$ 0.2	44.5 $\pm$ 0.2
<code>liblinear</code> SVM		44.0 $\pm$ 0.1	31.9 $\pm$ 0.8	50.2 $\pm$ 4.3

**MNIST data.** We consider the MNIST digits dataset, a real world multiclass dataset. For all experiments a 0/1 error term is used. In a first experiment, we analyze the test error and running time on a random subset of the dataset, where for each digit 100 examples are included. The results are visualized in Fig. 3. We observe that for larger  $\beta$  one needs to increase  $C$  in order to get a good prediction error. Furthermore, the running time of the training is substantially smaller for small values of  $\beta$ .



**Fig. 3.** Contour plot showing the test error (left) and the running time (in seconds) of the training (right) for combinations of  $\beta$  and  $C$  on a subset of the MNIST data set

**Table 2.** Results on the MNIST dataset for different instantiations of  $\ell_\beta$

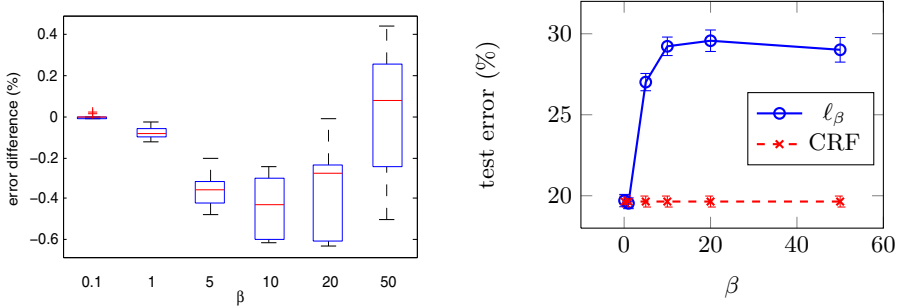
loss	$C$	test error (%)
$\beta = 1$	$10^{5.5}$	7.5
$\beta = 10$	$10^6$	<b>7.1</b>
$\beta = 10^3$	$10^6$	<b>7.1</b>
liblinear LR	$10^6$	8.4
liblinear SVM	$10^6$	<b>7.1</b>

In a second experiment we consider the full MNIST data set. Cross validation is performed for determining the regularization parameter  $C$ . For the full dataset, contrary to the first experiment where only a subset of the dataset is used, we found the max-margin loss to perform best. Using cross validation the model can automatically determine that a large  $\beta$  is beneficial (second row in Table 2).

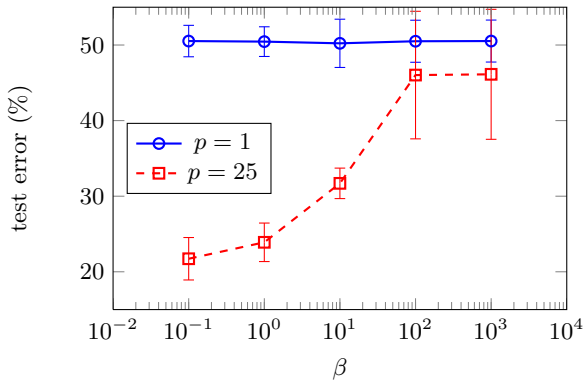
## 8.2 Linear Chain Model

In this experiment we consider the OCR dataset from [3]. Here, the task is to predict the letters of a word from a given sequence of binary images. By exploiting the dependencies between neighboring letters, the accuracy of the classifier can be improved. We use the same folds as in the original publication: The dataset consists of 10 train/test set splits, with each approximately 600 train and 5500 test sequences. We used the Hamming distance as our error term and perform inference in the linear chain model by libDAI [17]. In our experiments we found, both SSVM and CRF match the test error of around 20% (Fig. 4 right) reported in [3]. Varying the parameter  $\beta$  leads to a small, but consistent improvement over log-loss and max-margin loss (Fig. 4 left).

We perform a second experiment on this dataset to evaluate the quality of the probabilities on outputs learned by the model. To do so, we measure the error when predicting using the marginal-posterior-mode (MPM) instead of using



**Fig. 4.** Results on the OCR dataset. Left: Absolute error difference between the standard CRF and  $\ell_\beta$ . Right: Using MPM prediction when including the error term in training deteriorates the accuracy. The dashed line is the test error of the standard CRF. The solid line corresponds to the test error when training with  $\ell_\beta$ .



**Fig. 5.** Results for the synthetic MIL dataset for 400 bags, averaged over 10 random data sets. Depending on the number  $p$  of positive instances, a small  $\beta$  improves the accuracy substantially. The solid line corresponds to a setting where only one instance per bag is positive, the dashed line to 25 positive instances per bag.

the MAP predictor. For an individual variable  $y_i$  of the output  $y$ , the MPM marginalizes out all other variables  $y \setminus y_i$ :

$$y_i^* = \operatorname{argmax}_{y_i} \sum_{y \setminus y_i} P(y|x, w).$$

Using the MPM leads to good accuracy if no error term is included in training, but fails otherwise (Fig. 4 right). This is in agreement with our discussion in Section 4.4 that probabilities on outputs are not well-defined for SSVMs.

### 8.3 Multiple Instance Learning

As a last experiment we consider the problem of learning from multiple instances (MIL). This is a scenario with latent variables in training, as the label of an

individual instance in a bag is not observed; only the label of the whole bag. The model for  $\beta = 1$  and no error terms recovers the MI/LR from [18], for  $\beta \rightarrow \infty$  the model reduces to the MI-SVM [19].

We construct a one-dimensional synthetic dataset which illustrates the deficiencies of the MI-SVM. A positive bag consists of  $p$  positive instances and  $50 - p$  negative ( $0 < p \leq 50$ ), a negative bag contains 50 negative instances. The individual instances are hard to classify: the positive instances are Gaussian distributed with mean 0.6 whereas the negative instances are Gaussian distributed with mean 0, the variance for both classes is 1. Smaller values of  $\beta$  lead to better classification performance, as this corresponds to an averaging over the different instances in a bag, which is a good strategy for large data uncertainty (Fig. 5).

## 9 Conclusions

We have introduced a novel family of losses for structured output learning. The loss is parametrized by an inverse temperature  $\beta$ , which controls the entropy of the posterior distribution on outputs. The dual of the loss shows a double regularization by a margin and an entropy term. The max-margin loss and the log-loss emerge as two special cases of this loss. Additionally, our work also extends to models with hidden variables. We conjecture that different applications require different values of  $\beta$  and validate this claim experimentally on multiclass, linear-chain models and multiple instance learning. Choosing a large  $\beta$ , which corresponds to a large margin setting, while sometimes improving the accuracy, shows to have the severe disadvantage of deteriorating the probability distribution on outputs. The difference between the losses for different values of  $\beta$  is particularly striking in the multiple instance learning experiment.

*Acknowledgments.* We thank Sharon Wulff and Yvonne Moh for proof-reading an early version of this paper. This work was supported in parts by the Swiss National Science Foundation (SNF) under grant number 200021-117946.

## References

- [1] Lafferty, J., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: ICML (2001)
- [2] Tsochantaridis, I., Hofmann, T., Joachims, T., Altun, Y.: Support vector machine learning for interdependent and structured output spaces. In: ICML, p. 104 (2004)
- [3] Taskar, B., Guestrin, C., Koller, D.: Max-margin Markov networks. In: NIPS (2003)
- [4] Bakir, G., Hofmann, T., Schölkopf, B., Smola, A., Taskar, B., Vishwanathan, S.V.N.: Predicting Structured Data. MIT Press, Cambridge (2007)
- [5] Wainwright, M., Jordan, M.: Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning (2008)
- [6] Zhang, T., Oles, F.J.: Text categorization based on regularized linear classification methods. Information Retrieval 4, 5–31 (2000)

- [7] Collins, M., Globerson, A., Koo, T., Carreras, X., Bartlett, P.L.: Exponentiated gradient algorithms for conditional random fields and max-margin Markov networks. *J. Mach. Learn. Res.* 9, 1775–1822 (2008)
- [8] Bartlett, P.L., Tewari, A.: Sparseness vs estimating conditional probabilities: Some asymptotic results. *J. Mach. Learn. Res.* 8, 775–790 (2007)
- [9] Quattoni, A., Wang, S., Morency, L., Collins, M., Darrell, T.: Hidden-state conditional random fields. *PAMI* 29(10), 1848–1852 (2007)
- [10] Yu, C., Joachims, T.: Learning structural SVMs with latent variables. In: *ICML*, pp. 1169–1176 (2009)
- [11] Canu, S., Smola, A.J.: Kernel methods and the exponential family. *Neurocomputing* 69(7-9), 714–720 (2006)
- [12] Chapelle, O., Zien, A.: Semi-supervised classification by low density separation. In: *AISTATS*, pp. 57–64 (2005)
- [13] Zhang, T.: Class-size independent generalization analysis of some discriminative multi-category classification. In: *NIPS*, Cambridge, MA (2005)
- [14] Shi, Q., Reid, M., Caetano, T.: Hybrid model of conditional random field and support vector machine. In: *Workshop at NIPS* (2009)
- [15] Gimpel, K., Smith, N.: Softmax-margin crfs: Training log-linear models with cost functions. In: *HLT*, pp. 733–736 (2010)
- [16] Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass kernel-based vector machines. *J. Mach. Learn. Res.* 2 (2001)
- [17] Mooij, J.: libDAI: A free/open source C++ library for Discrete Approximate Inference (2009)
- [18] Ray, S., Craven, M.: Supervised versus multiple instance learning: An empirical comparison. In: *ICML* (2005)
- [19] Andrews, S., Tsochantaridis, I., Hofmann, T.: Support vector machines for multiple-instance learning. In: *NIPS*, pp. 561–568 (2003)

# Virus Propagation on Time-Varying Networks: Theory and Immunization Algorithms\*

B. Aditya Prakash<sup>1</sup>, Hanghang Tong<sup>1</sup>, Nicholas Valler<sup>2</sup>,  
Michalis Faloutsos<sup>2</sup>, and Christos Faloutsos<sup>1</sup>

<sup>1</sup> Computer Science Department, Carnegie Mellon University  
{badityap, htong, christos}@cs.cmu.edu

<sup>2</sup> Department of Computer Science, University of California - Riverside  
{nvaller, michalis}@cs.ucr.edu

**Abstract.** Given a contact network that changes over time (say, day vs night connectivity), and the SIS (susceptible/infected/susceptible, flu like) virus propagation model, what can we say about its epidemic threshold? That is, can we determine when a small infection will “take-off” and create an epidemic? Consequently then, which nodes should we immunize to prevent an epidemic? This is a very real problem, since, e.g. people have different connections during the day at work, and during the night at home. Static graphs have been studied for a long time, with numerous analytical results. Time-evolving networks are so hard to analyze, that most existing works are simulation studies [5].

Specifically, our contributions in this paper are: (a) we formulate the problem by approximating it by a Non-linear Dynamical system (NLDS), (b) we derive the **first** closed formula for the epidemic threshold of time-varying graphs under the SIS model, and finally (c) we show the usefulness of our threshold by presenting efficient heuristics and evaluate the effectiveness of our methods on synthetic and real data like the MIT reality mining graphs.

## 1 Introduction

The goal of this work is to analytically study the epidemic spread on time-varying graphs. We focus on time-varying graphs that follow an alternating connectivity behavior, which is motivated by the day-night pattern of human behavior. Note that our analysis is not restricted to two graphs: we can have an arbitrary number of alternating graphs. Furthermore, we focus on the SIS model [19], which

---

\* This material is based upon work supported by the Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053, the National Science Foundation under Grants No. CNS-0721736 and CNS-0721889 and a Sprint gift. Any opinions, findings, and conclusions or recommendations in this material are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory, the U.S. Government, the National Science Foundation, or other funding parties. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.



resembles a flu-like virus, where healthy nodes get the virus stochastically from their infected neighbors, and infected nodes get cured with some probability and become susceptible again. The SIS model can be also used in modeling many different types of dynamical processes as well, for example, modeling product penetration in marketing [36].

More specifically, the inputs to our problem are: (a) a set of  $T$  alternating graphs, (b) the infectivity of the virus and the recovery rate ( $\beta, \delta$  for the SIS model), (d)  $k$  number of “vaccinations”. We want to answer two questions (rigorously defined in Section 3):

- Q1. Can we say whether a small infection can “take-off” and create an epidemic under the SIS model (i.e. determine the so-called epidemic threshold)?
- Q2. What is an effective and fast way to vaccinate people to minimize the spread of the virus?

While epidemic spreading on static graphs has been studied extensively (e.g. see [19,23,33,8]), virus propagation on time-varying graphs has received little attention. Moreover, most previous studies on time-varying graphs use only simulations [5]. We review in more detail the previous efforts in Section 2.

We are arguably the first to study virus propagation *analytically* on arbitrary, and *time-varying* graphs. In more detail, the contributions of our work can be summarized in the following points:

1. We formulate the problem, and show that it can be approximated with a *Non-Linear Dynamical System (NLDS)*.
2. We give the **first** closed-formula for the epidemic threshold, involving the first eigenvalue of the so-called *system-matrix* (see Theorem 2). The system-matrix combines the connectivity information (the alternating adjacency matrices) and the characteristics of the virus (infectivity and recovery rate).
3. We show the importance of our threshold by using it to develop and evaluate several immunization policies on real data like the MIT Reality Mining graph.

The rest of the paper is organized as follows: We review related work in Section 2, explain the formal problem definitions in Section 3, and describe the proofs for the threshold and illustrate the theorem in Section 4. We then discuss various immunization policies in Section 5 and present experimental evaluations in Section 6. We discuss and provide additional explanations in Section 7 and finally conclude in Section 8.

## 2 Related Work

In this section, we review the related work, which can be categorized into three parts: epidemic threshold, immunization algorithms and general graph mining.

**Epidemic Thresholds.** The class of epidemiological models that are most widely used are the so-called *homogeneous models* [3,29,2]. A homogeneous model

assumes that every individual has equal contact to others in the population, and that the rate of infection is largely determined by the density of the infected population. Kephart and White [22,23] were among the first to propose epidemiology-based models (hereafter referred to as the KW model) to analyze the propagation of computer viruses. The KW model provides a good approximation of virus propagation in networks where the contact among individuals is sufficiently homogeneous. However, there is overwhelming evidence that real networks (including social networks [10], router and AS networks [12], and Gnutella overlay graphs [35]) deviate from such homogeneity - they follow a power law structure instead.

Pastor-Satorras and Vespignani studied viral propagation for such power-law networks [32,33]. They developed an analytic model for the Barabási-Albert (BA) power-law topology [4]. However, their derivation depends on some assumptions which does not hold for many real networks [25,12]. Pastor-Satorras et al. [33] also proposed an epidemic threshold condition, but this uses the “mean-field” approach, where all graphs with a given degree distribution are considered equal. There is no particular reason why all such graphs should behave similarly in terms of viral propagation. Newman [31] studied the epidemic thresholds for multiple competing viruses on special, *random* graphs.

**Immunization.** Briesemeister et al. [7] focus on immunization of power law graphs. They focus on the random-wiring version (that is, standard preferential attachment), versus the “highly clustered” power law graphs. Their simulation experiments on such synthetic graphs show that such graphs can be more easily defended against viruses, while random-wiring ones are typically overwhelmed, despite identical immunization policies.

Cohen et al. [9] studied the *acquaintance* immunization policy (see Section 5 for a description of this policy), and showed that it is much better than random, for both the SIS as well as the SIR model. For power law graphs (with no rewiring), they also derived formulae for the critical immunization fraction, above which the epidemic is arrested. Madar et al. [27] continued along these lines, mainly focusing on the SIR model for scale-free graphs. They linked the problem to bond percolation, and derived formulae for the effect of several immunization policies, showing that the “acquaintance” immunization policy is the best. Both works were analytical, without studying any real graphs.

Hayashi et al. [18] study the case of a growing network, and they derive analytical formulas for such power law networks (no rewiring). They introduce the SHIR model (Susceptible, Hidden, Infectious, Recovered), to model computers under e-mail virus attack and derive the conditions for extinction under random and under targeted immunization, always for power law graphs with no rewiring.

Thus, none of the earlier related work focus on epidemic thresholds for *arbitrary, real* graphs, with only exceptions of [37,8], and its follow-up paper by Ganesh et al. [13]. However, even these works [37,8,13] assume that the underlying graph is fixed, which is unrealistic in many applications.

**General Graph Mining.** Graph mining is a very active research area in recent years. Representative works include patterns and “laws” discovery e.g., power law distributions [12,26], small world phenomena [30,11], and numerous other regularities. Among them, there is a lot of research interest in studying dynamic processes on large graphs, (a) blogs and propagations [17,24,21,34], (b) information cascades [6,14,16] and (c) marketing and product penetration [36]. These dynamic processes are all closely related to virus propagation.

In sum, to the best of our knowledge, including comprehensive epidemiological texts [2,3] and well-cited surveys [19], we are the **first** to analytically study virus propagation on *arbitrary, real* and *time-varying* graphs.

### 3 Problem Definitions

Table 1 lists the main symbols used in the paper. Following standard notation, we use capital bold letters for matrices (e.g.  $\mathbf{A}$ ), lower-case bold letters for vectors (e.g.  $\mathbf{a}$ ), and calligraphic fonts for sets (e.g.  $\mathcal{S}$ ) and we denote the transpose with a prime (i.e.,  $\mathbf{A}'$  is the transpose of  $\mathbf{A}$ ). In this paper, we focus on un-directed un-weighted graphs which we represent by the adjacency matrix.

Also we deal only with the SIS virus propagation model in the paper. The SIS model is the susceptible/infected/susceptible virus model where  $\beta$  is the probability that an infected node will transmit the infection over a link connected to a neighbor and  $\delta$  is the probability with which an infected node cures itself and becomes susceptible again. Please see [19] for a detailed discussion on SIS and other virus models.

Consider a setting with clearly different behaviors say, day/night, each characterized by a corresponding adjacency matrix ( $\mathbf{A}_1$  for day,  $\mathbf{A}_2$  for night), then what is the epidemic threshold under a SIS virus model? What are the best nodes to immunize to prevent an epidemic as much as possible? More generally, the problems we are tackling can be formally stated as follows:

#### *Problem 1. Epidemic Threshold*

**Given:** (1)  $T$  alternating behaviors, characterized by a set of  $T$  graphs  $\mathcal{A} = \{\mathbf{A}_1, \mathbf{A}_2 \dots \mathbf{A}_T\}$ ; and (2) the SIS model [8] with virus parameters  $\beta$  and  $\delta$ ;  
**Find:** A condition, under which the infection will die out exponentially quickly (regardless of initial condition).

#### *Problem 2. Immunization*

**Given:** (1)  $T$  alternating behaviors, characterized by a set of  $T$  graphs  $\mathcal{A} = \{\mathbf{A}_1, \mathbf{A}_2 \dots, \mathbf{A}_T\}$ ; and (2) the SIS model with virus parameters  $\beta$  and  $\delta$  and (3)  $k$  vaccines;  
**Find:** The best- $k$  nodes for immunization.

We will next solve Problem 1 while we discuss Problem 2 later in Section 5.

**Table 1.** Symbols

Symbol	Definition and Description
$\mathbf{A}, \mathbf{B}, \dots$	matrices (bold upper case)
$\mathbf{A}(i, j)$	element at the $i^{\text{th}}$ row and $j^{\text{th}}$ column of $\mathbf{A}$
$\mathbf{A}(i, :)$	$i^{\text{th}}$ row of matrix $\mathbf{A}$
$\mathbf{A}(:, j)$	$j^{\text{th}}$ column of matrix $\mathbf{A}$
$\mathbf{I}$	standard $n \times n$ identity matrix
$\mathbf{a}, \mathbf{b}, \dots$	column vectors
$\mathcal{I}, \mathcal{J}, \dots$	sets (calligraphic)
$n$	number of nodes in the graphs
$T$	number of different alternating behaviors
$\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_T$	$T$ corresponding size $n \times n$ symmetric alternating adjacency matrices
$\beta$	virus transmission probability in the SIS model
$\delta$	virus death probability in the SIS model
$\lambda_{\mathbf{M}}$	first eigen-value (in absolute value) of a matrix $\mathbf{M}$
$\mathbf{u}_{\mathbf{M}}$	corresponding first eigen-vector (for $\lambda_{\mathbf{M}}$ ) of a matrix $\mathbf{M}$
$p_{i,t}$	probability that node $i$ is infected at time $t$
$\mathbf{p}_t$	$\mathbf{p}_t = (p_{1,t}, p_{2,t}, \dots, p_{n,t})'$
$\mathbf{p}_{2t+1}$	probability of infection vector for odd days
$\mathbf{p}_{2t}$	probability of infection vector for even days
$\eta_t$	the expected number of infected nodes at time $t$

## 4 Epidemic Threshold on Time-Varying Graphs

To simplify discussion, we consider  $T = 2$  in Problem [1](#) with  $\mathcal{A}$  to consist of only two graphs:  $G_1$  with the adjacency matrix  $\mathbf{A}_1$  for the odd time-stamps (the ‘days’) and  $G_2$  with the adjacency matrix  $\mathbf{A}_2$  for the even time-stamps (the ‘nights’). Our proofs and results can be naturally extended to handle any arbitrary sequence of  $T$  graphs.

### 4.1 The NLDS

We first propose to approximate the infection dynamics by a Non-linear dynamical system (NLDS) representing the evolution of the probability of infection vector ( $\mathbf{p}_t$ ) over time. We can compute the probability  $\zeta_t(i)$  that node  $i$  does not receive any infections at time  $t$ . A node  $i$  won’t receive any infection if either any given neighbor is not infected or it is infected but fails to transmit the infection with probability  $1 - \beta$ . Assuming that the neighbors are *independent*, we get:

$$\begin{aligned}
 \zeta_{2t+1}(i) &= \prod_{j \in \mathcal{N}_{\mathcal{E}_1}(i)} (p_{j,2t+1}(1 - \beta) + (1 - p_{j,2t+1})) \\
 &= \prod_{j \in \{1..n\}} (1 - \beta \mathbf{A}_1(i, j) p_{j,2t+1})
 \end{aligned} \tag{1}$$

where  $\mathcal{N}\mathcal{E}_1(i)$  is the set of neighbors of node  $i$  in the graph  $G_1$  with adjacency matrix  $\mathbf{A}_1$ . Similarly, we can write  $\zeta_{2t+2}(i)$  as:

$$\begin{aligned}\zeta_{2t}(i) &= \prod_{j \in \mathcal{N}\mathcal{E}_2(i)} (p_{j,2t}(1 - \beta) + (1 - p_{j,2t})) \\ &= \prod_{j \in \{1..n\}} (1 - \beta \mathbf{A}_2(i, j) p_{j,2t})\end{aligned}\quad (2)$$

So,  $p_{i,2t+1}$  and  $p_{i,2t+2}$  are:

$$\begin{aligned}1 - p_{i,2t+1} &= \delta p_{i,2t} + (1 - p_{i,2t}) \zeta_{2t}(i) \\ \Rightarrow p_{i,2t+1} &= 1 - \delta p_{i,2t} - (1 - p_{i,2t}) \zeta_{2t}(i)\end{aligned}\quad (3)$$

and

$$\begin{aligned}1 - p_{i,2t+2} &= \delta p_{i,2t+1} + (1 - p_{i,2t+1}) \zeta_{2t+1}(i) \\ \Rightarrow p_{i,2t+2} &= 1 - \delta p_{i,2t+1} - (1 - p_{i,2t+1}) \zeta_{2t+1}(i)\end{aligned}\quad (4)$$

Note that we can write our NLDS as:

$$\mathbf{p}_{2t+1} = \mathbf{g}_2(\mathbf{p}_{2t}) \quad (5)$$

$$\mathbf{p}_{2t+2} = \mathbf{g}_1(\mathbf{p}_{2t+1}) \quad (6)$$

where  $\mathbf{g}_1$  and  $\mathbf{g}_2$  are corresponding non-linear functions as defined by Equations [3](#) and [4](#) ( $\mathbf{g}_1$  depends only on  $\mathbf{A}_1$  and  $\mathbf{g}_2$  on  $\mathbf{A}_2$ ).

We have the following theorem about the asymptotic stability of a NLDS at a fixed point:

**Theorem 1. (Asymptotic Stability, e.g. see [\[20\]](#))** *The system given by  $\mathbf{p}_{t+1} = \mathbf{g}(\mathbf{p}_t)$  is asymptotically stable at an equilibrium point  $\mathbf{p}^*$ , if the eigenvalues of the Jacobian  $J = \nabla \mathbf{g}(\mathbf{p}^*)$  are less than 1 in absolute value, where,*

$$J_{k,l} = [\nabla \mathbf{g}(\mathbf{p}^*)]_{k,l} = \left. \frac{\partial p_{k,t+1}}{\partial p_{l,t}} \right|_{\mathbf{p}_t = \mathbf{p}^*}$$

The fixed point of our interest is the  $\mathbf{0}$  vector which is the state when all nodes are susceptible and not infected. We want to then analyze the stability of our NLDS at  $\mathbf{p}_{2t} = \mathbf{p}_{2t+1} = \mathbf{0}$ . From Equations [5](#) and [6](#), we get:

$$\left. \frac{\partial \mathbf{p}_{2t+2}}{\partial \mathbf{p}_{2t+1}} \right|_{\mathbf{p}_{2t+1} = \mathbf{0}} = (1 - \delta) \mathbf{I} + \beta \mathbf{A}_1 = \mathbf{S}_1 \quad (7)$$

$$\left. \frac{\partial \mathbf{p}_{2t+1}}{\partial \mathbf{p}_{2t}} \right|_{\mathbf{p}_{2t} = \mathbf{0}} = (1 - \delta) \mathbf{I} + \beta \mathbf{A}_2 = \mathbf{S}_2 \quad (8)$$

Any eigenvalue  $\lambda_{\mathbf{S}_1}^i$  of  $\mathbf{S}_1$  and  $\lambda_{\mathbf{S}_2}^i$  of  $\mathbf{S}_2$  ( $i = 1, 2, \dots$ ) is related to the corresponding eigenvalue  $\lambda_{\mathbf{A}_1}^i$  of  $\mathbf{A}_1$  and  $\lambda_{\mathbf{A}_2}^i$  of  $\mathbf{A}_2$  as:

$$\lambda_{\mathbf{S}_1}^i = (1 - \delta) + \beta \lambda_{\mathbf{A}_1}^i \quad (9)$$

$$\lambda_{\mathbf{S}_2}^i = (1 - \delta) + \beta \lambda_{\mathbf{A}_2}^i \quad (10)$$

Recall that as  $\mathbf{A}_1$  and  $\mathbf{A}_2$  are symmetric real matrices (the graphs are undirected), from the Perron-Frobenius theorem [28],  $\lambda_{\mathbf{A}_1}$  and  $\lambda_{\mathbf{A}_2}$  are real and positive. So, from Equations 9 and 10  $\lambda_{\mathbf{S}_1}$  and  $\lambda_{\mathbf{S}_2}$  are also real and positive.

### 4.2 The Threshold

We are now in a position to derive the epidemic threshold. First, we have the following lemma:

**Lemma 1.** *If  $\lambda_{\mathbf{S}} < 1$ , then  $\mathbf{p}_{2t}$  dies out exponentially quickly; and  $\mathbf{0}$  is asymptotically stable for  $\mathbf{p}_{2t}$ , where  $\mathbf{S}_1 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_1$ ,  $\mathbf{S}_2 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_2$  and  $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2$ .*

*Proof.* Since  $\mathbf{p}_{2t+2} = g_1(g_2(\mathbf{p}_{2t}))$  (from Equations 5 and 6), we have

$$\begin{aligned} \frac{\partial \mathbf{p}_{2t+2}}{\partial \mathbf{p}_{2t}} \Big|_{\mathbf{p}_{2t}=\mathbf{0}} &= \left( \frac{\partial \mathbf{p}_{2t+2}}{\partial \mathbf{p}_{2t+1}} \times \frac{\partial \mathbf{p}_{2t+1}}{\partial \mathbf{p}_{2t}} \right) \Big|_{\mathbf{p}_{2t}=\mathbf{0}} \\ &= \left( \frac{\partial \mathbf{p}_{2t+2}}{\partial \mathbf{p}_{2t+1}} \Big|_{\mathbf{p}_{2t+1}=\mathbf{0}} \right) \times \left( \frac{\partial \mathbf{p}_{2t+1}}{\partial \mathbf{p}_{2t}} \Big|_{\mathbf{p}_{2t}=\mathbf{0}} \right) \\ &= \mathbf{S}_1 \mathbf{S}_2 = \mathbf{S} \end{aligned} \tag{11}$$

The first equation is due to chain-rule, second equation is because  $\mathbf{p}_{2t} = \mathbf{0}$  implies  $\mathbf{p}_{2t+1} = \mathbf{0}$ ; and the final step is due to Equations 7 and 8.

Therefore, using Theorem 1, we get that if  $\lambda_{\mathbf{S}} < 1$ , we have that  $\mathbf{0}$  is asymptotically stable for  $\mathbf{p}_{2t}$ .

We now prove that  $\mathbf{p}_{2t}$  in fact goes down exponentially to  $\mathbf{0}$  if  $\lambda_{\mathbf{S}} < 1$ . To see this, after linearizing both  $g_1$  and  $g_2$  at  $\mathbf{p}_{2t} = \mathbf{p}_{2t+1} = \mathbf{0}$ , we have

$$\begin{aligned} \mathbf{p}_{2t+2} &\leq \mathbf{S}_1 \mathbf{p}_{2t+1} \\ \mathbf{p}_{2t+1} &\leq \mathbf{S}_2 \mathbf{p}_{2t} \end{aligned} \tag{12}$$

Doing the above recursively, we have

$$\mathbf{p}_{2t} \leq (\mathbf{S}_1 \mathbf{S}_2)^t \mathbf{p}_0 = (\mathbf{S})^t \mathbf{p}_0 \tag{13}$$

Let  $\eta_t$  be the expected number of infected nodes at time  $t$ . Then,

$$\begin{aligned} \eta_{2t} = |\mathbf{p}_{2t}|_1 &\leq |(\mathbf{S})^t \mathbf{p}_0|_1 \\ &\leq |(\mathbf{S})^t|_1 |\mathbf{p}_0|_1 = |(\mathbf{S})^t|_1 \eta_0 \\ &\leq \sqrt{n} |(\mathbf{S})^t|_2 \eta_0 = \sqrt{n} \lambda_{\mathbf{S}}^t \eta_0 \end{aligned} \tag{14}$$

Therefore, if  $\lambda_{\mathbf{S}} < 1$ , we have that  $\eta_{2t}$  goes to zero exponentially fast. □

The above lemma provides the condition for the even time-stamp probability vector to go down exponentially. But, the next lemma shows that this condition is enough to ensure that even the odd time-stamp probability vector to go down exponentially.

**Lemma 2.** *If  $\lambda_{\mathbf{S}} < 1$ , then  $\mathbf{p}_{2t+1}$  dies out exponentially quickly; and  $\mathbf{0}$  is asymptotically stable for  $\mathbf{p}_{2t+1}$ , where  $\mathbf{S}_1 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_1$ ,  $\mathbf{S}_2 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_2$  and  $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2$ .*

*Proof.* Doing the same analysis as in Lemma 1, we can see that the condition for  $\mathbf{p}_{2t+1}$  to be asymptotically stable and die exponentially quickly is:

$$\lambda_{\mathbf{S}_2 \times \mathbf{S}_1} < 1 \tag{15}$$

Now note that as  $\mathbf{S}_1$  and  $\mathbf{S}_2$  are invertible:  $\mathbf{S}_1 \times \mathbf{S}_2 = \mathbf{S}_1 \times \mathbf{S}_2 \times \mathbf{S}_1 \times \mathbf{S}_1^{-1}$ . But this implies that  $\mathbf{S}_2 \times \mathbf{S}_1$  is similar to  $\mathbf{S}_1 \times \mathbf{S}_2$  (matrix  $\mathbf{P}$  is similar to  $\mathbf{Q}$  if  $\mathbf{P} = \mathbf{B}\mathbf{Q}\mathbf{B}^{-1}$ , for some invertible  $\mathbf{B}$ ). We know that similar matrices have the same spectrum [15], thus  $\mathbf{S}_2 \times \mathbf{S}_1$  and  $\mathbf{S}_1 \times \mathbf{S}_2$  have the same eigenvalues. Hence, the condition for exponential die out of  $\mathbf{p}_{2t+1}$  and asymptotic stability is the same as that for  $\mathbf{p}_{2t}$  which is  $\lambda_{\mathbf{S}} < 1$ .  $\square$

Lemma 1 and Lemma 2 imply that this threshold is well-defined in the sense that the probability vector for both the odd and even time-stamps go down exponentially. Thus we can finally conclude the following theorem:

**Theorem 2. (Epidemic Threshold)** *If  $\lambda_{\mathbf{S}} < 1$ , then  $\mathbf{p}_{2t}$  and  $\mathbf{p}_{2t+1}$  die out exponentially quickly; and  $\mathbf{0}$  is asymptotically stable for both  $\mathbf{p}_{2t}$  and  $\mathbf{p}_{2t+1}$ , where  $\mathbf{S}_1 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_1$ ,  $\mathbf{S}_2 = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_2$  and  $\mathbf{S} = \mathbf{S}_1 \times \mathbf{S}_2$ . Similarly for any general  $T$ , the condition is:*

$$\lambda_{\prod_i \mathbf{S}_i} < 1 \tag{16}$$

where  $\forall i \in \{1, 2, \dots, T\}$   $\mathbf{S}_i = (1 - \delta)\mathbf{I} + \beta\mathbf{A}_i$ .

We call  $\mathbf{S}$  as the *system-matrix* of the system; thus, the first eigenvalue of the system-matrix determines whether a given system is below threshold or not.

### 4.3 Salient Points

**Sanity check:** Clearly, when  $T = 1$ , the system is equivalent to a static graph system with  $\mathbf{A}_1$  and virus parameters  $\beta, \delta$ . In this case the threshold is (from Theorem 2)  $\lambda_{(1-\delta)\mathbf{I} + \beta\mathbf{A}_1} < 1 \Rightarrow \beta\lambda_{\mathbf{A}_1}/\delta < 1$  i.e. we recover the known threshold in the static case [8].

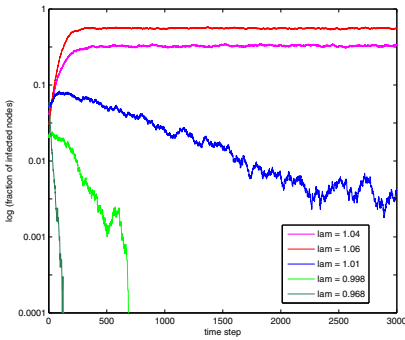
**A conservative condition:** Notice that from Equations 7 and 8 and Theorem 1, for our NLDS to be fully asymptotically stable at  $\mathbf{0}$  (i.e.  $\mathbf{p}_t$  decays monotonically), we need the eigenvalues of both  $\mathbf{S}_1$  and  $\mathbf{S}_2$  be less than 1 in absolute value. Hence,  $\beta\lambda/\delta < 1$  where  $\lambda = \max(\lambda_{\mathbf{A}_1}, \lambda_{\mathbf{A}_2})$  is sufficient for full stability. Intuitively, this argument says that the alternating sequence of graphs can not be worse than static case of having the best-connected graph of the two repeated indefinitely. Let  $\lambda_{\mathbf{A}_1} > \lambda_{\mathbf{A}_2}$ . Consider a sequence of graphs  $\mathcal{S} = \{\mathbf{A}_1, \mathbf{A}_1 \dots\}$  repeating indefinitely instead of our alternating  $\{\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_1, \mathbf{A}_2, \dots\}$  sequence. Clearly, if an infection dies exponentially in  $\mathcal{S}$ , then it will die exponentially in our original alternating sequence as well because  $\lambda_{\mathbf{A}_1} > \lambda_{\mathbf{A}_2}$ . The set  $\mathcal{S}$  is

essentially just the static graph case: hence, if  $\beta\lambda_{A_1}/\delta = \beta\lambda/\delta < 1$ , then  $\mathbf{0}$  is asymptotically stable for  $\mathbf{p}_t$ . The case when  $\lambda_{A_1} < \lambda_{A_2}$  is similar. But this notion of a threshold is too stringent and *conservative*: it can happen that a stronger virus can still lead to a general exponential decrease instead of a strict monotonous decrease. This is because we forced the eigenvalues of *both*  $\mathbf{S}_1$  and  $\mathbf{S}_2$  to be less than 1 in absolute value here, when we can probably get away with less. Theorem 2 precisely formalizes this idea and gives us a more practical condition for a general decreasing trend of every corresponding alternating time-stamp values decaying. We illustrate this further in the experiments.

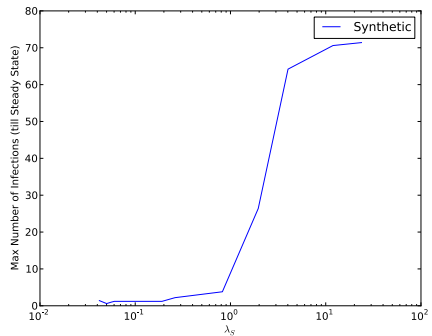
### 4.4 Experiments

Figures 1 and 2 demonstrate our result on a synthetic example and graphs from MIT reality data (more details on the reality mining graphs are in Section 6). In the synthetic example, we have 100 nodes, such that  $G_1$  is a full clique (without self loops) whereas  $G_2$  is a chain. All values are average over several runs of the simulations and the infection is started by infecting 5 nodes. In short, as expected from the theorem, the difference in behavior above, below and at threshold can be distinctly seen in the figures.

Figures 1(A) and 2(A) show the time-plot of number of infections for  $\lambda_S$  values above and below the threshold. While above threshold the infection reaches a steady state way above the starting point, below threshold it decays fast and dies out. In case of Figure 1(A), also note the the difference between the conservative



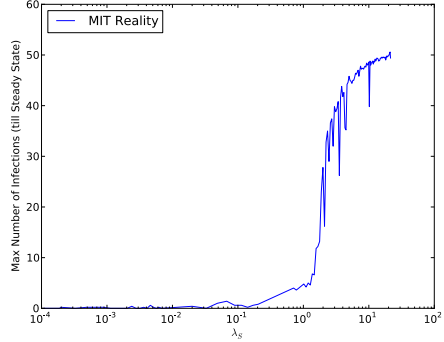
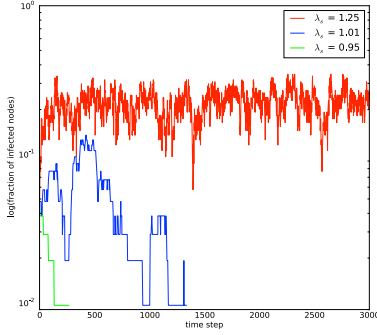
(A) Infected Fraction Time Plot (lin-log)



(B) Max. Infections till steady state vs  $\lambda_S$  (lin-log)

**Fig. 1.** SIS simulations on our synthetic example (all values averages over 20 runs) (A) Fraction of nodes infected vs Time-stamp (lin-log scale). Note the qualitative difference in behavior *under* (green) and *above* (red) the threshold. Also, note that the green line is *below* the threshold but is actually *above* the conservative threshold ( $\beta\lambda/\delta = 1.100$  here). Hence while both  $\mathbf{p}_{2t}$  and  $\mathbf{p}_{2t+1}$  decrease exponentially separately, but  $\mathbf{p}_t$  itself does not monotonously go down. (B) Plot of Max. number of infected nodes till steady state vs  $\lambda_S$  (by varying  $\beta$ ) (lin-log). As predicted by our results, notice that there is a sudden ‘take-off’ and a change of behavior of the curve exactly when  $\lambda_S = 1$ .





(A) Infected Fraction Time Plot (lin-log)

(B) Max. Infections till steady state vs  $\lambda_S$  (lin-log)

**Fig. 2.** SIS simulations on the MIT reality mining graphs (all values averages over 20 runs) (A) Fraction of nodes infected vs Time-stamp (lin-log scale). Note the qualitative difference in behavior *under* (green) and *above* (red) the threshold. (B) Plot of Max. number of infected nodes till steady state vs  $\lambda_S$  (by varying  $\beta$ ) (lin-log). As predicted by our results, notice that there is a sudden ‘take-off’ and a change of behavior of the curve when  $\lambda_S = 1$ .

threshold and our threshold. The green curve is *below* our threshold but *above* the conservative threshold. But again, as predicted from our theorems, clearly while there are dampening oscillations and the infection decays but  $\mathbf{p}_t$  itself does not monotonously go down (and hence the “bumpy” nature of the curve). This exemplifies the practical nature of our threshold and its usefulness as we are more concerned with the general trend of the number of infections curve and not every small ‘bump’ because of the presence of alternating graphs.

Figures 1(B) and 2(B) show a ‘take-off’ plot showing max. number of infections till steady state (intuitively the ‘footprint’) for different values of  $\lambda_S$  (by varying  $\beta$ ). As predicted by our theorem, note the sudden steep change and spike in the size of the footprint when  $\lambda_S = 1$  in both the plots.

## 5 Immunization Algorithms

Given the theoretical results in the previous section, can we exploit them to our advantage to ensure effective immunization (Problem 2)?

### 5.1 Quality Metric

Using our results, we can evaluate the quality of *any* immunization policy. Note that smaller the value of  $\lambda_S$ , lesser are chances of the virus causing any epidemic. Put differently, we want to decrease the  $\lambda_S$  value of the system as much as possible. Thus, the efficacy of any immunization policy should be measured using the amount of “drop” in  $\lambda_S$  it causes and the resulting  $\lambda_S$  after immunization.

## 5.2 Proposed Immunization Policies

We now discuss some new immunization policies for time-varying graphs, partially motivated by known policies used for static graphs. Again, for ease of exposition we focus our attention only on the  $\{\mathbf{A}_1, \mathbf{A}_2\}$  system of Section 3. From the above, it is clear that optimally we should choose that set of  $k$  nodes which result in the smallest  $\lambda_{\mathbf{S}}$  value possible after immunization. This implies that for each set of  $k$  node we test, we need to delete  $k$  rows/columns from both  $\mathbf{A}_1$  and  $\mathbf{A}_2$  to get new matrices  $\mathbf{A}_1^*$  and  $\mathbf{A}_2^*$  and then compute the new  $\lambda_{\mathbf{S}}$  value. The number of  $k$  sets is  $\binom{n}{k}$  and therefore this method is combinatorial in nature and will be intractable even for small graphs. Nevertheless, we call this strategy **Optimal** and show experimental results for this policy too, because this policy will give us the lower-bound on the  $\lambda_{\mathbf{S}}$  that can be achieved after  $k$  immunizations.

We want policies which are practical for large graphs and at the same time be efficient in lowering the  $\lambda_{\mathbf{S}}$  value of the system i.e. they should be close to **Optimal**. Specifically to this effect, we now present several greedy policies as well. As the heuristics are greedy in nature, we only describe how to pick the best *one* node for immunization from a given set of  $G_1$  and  $G_2$  graphs. Our proposed policies are:

**Greedy-DmaxA (Highest degree on  $\mathbf{A}_1$  or  $\mathbf{A}_2$  matrices).** Under this policy, at each step we select the node with the highest degree considering both the  $\mathbf{A}_1$  or  $\mathbf{A}_2$  adjacency matrices. This is motivated by the degree immunization strategy used for static graphs.

**Greedy-DavgA (Highest degree on the “average” matrix).** We select the node with the highest degree in the  $\mathbf{A}_{\text{avg}}$  matrix where  $\mathbf{A}_{\text{avg}} = (\mathbf{A}_1 + \mathbf{A}_2)/2$ .

**Greedy-AavgA (Acquaintance immunization [9] on the average matrix).**

The “acquaintance” immunization policy works by picking a random person, and then immunizing one of its neighbors at random (which will probably be a ‘hub’). We run this policy on the  $\mathbf{A}_{\text{avg}}$  matrix.

**Greedy-S (Greedy on the system-matrix).** This is the greedy strategy of immunizing the node at each step which causes the largest drop in  $\lambda_{\mathbf{S}}$  value. Note that even this can be expensive for large graphs as we have to evaluate the first eigenvalue of  $\mathbf{S}$  after deleting each node to decide which node is the best.

**Optimal.** Finally, this is the optimal through combinatorial strategy mentioned above of finding the best- $k$  set of nodes which decrease  $\lambda_{\mathbf{S}}$  the most.

## 6 Experimental Evaluations

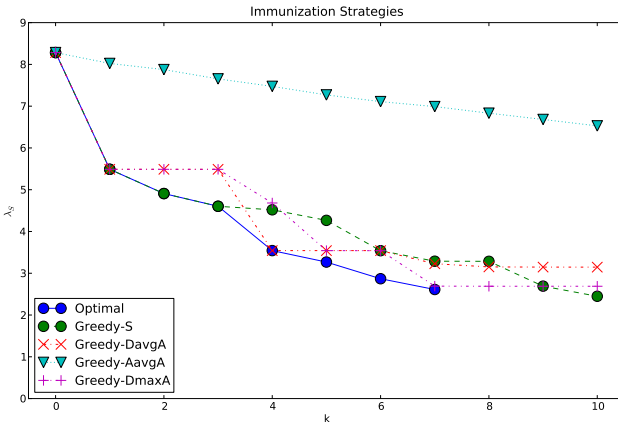
In this section we present experimental results of applying the various immunization policies discussed previously. We have already demonstrated our theoretical threshold results in Section 4.

## 6.1 Experimental Setup

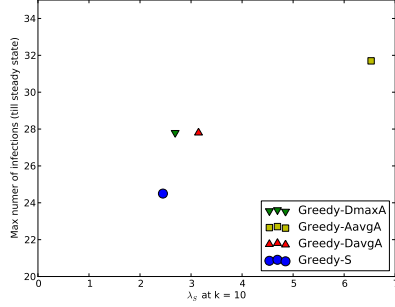
We conducted a series of experiments using the MIT Reality Mining data set [11]. The Reality Mining data consists of 104 mobile devices (cellular phones) monitored over a period of nine months (September 2004 - June 2005). If another participating Bluetooth device was within a range of approximately 5-10 meters, the date and time of the contact and the device’s MAC address were recorded. Bluetooth scans were conducted at 5-minute intervals and aggregated into two 12-hour period adjacency matrices (*day* and *night*). The epidemic simulations were accomplished by alternating the *day* and *night* matrices over the period of simulation. All experiments were run on a 64-bit, quad-core (2.5Ghz each) server running a CentOS linux distribution with shared 72 GB of RAM. Simulations were conducted using a combination of Matlab 7.9 and Python 2.6.

## 6.2 Results

Figure 3 shows the  $\lambda_S$  value after immunizing  $k = 1, 2, \dots, 10$  nodes using each of the policies outlined in Section 5. As **Optimal** and **Greedy-S** require  $\beta$  and  $\delta$  as inputs, we set  $\beta = 0.5, \delta = 0.1$ . Running **Optimal** became prohibitively expensive ( $> 4$  hours on the MIT reality graphs) after  $k = 7$  - hence we don’t show data points for  $k \geq 8$  for **Optimal**. Moving on to the greedy strategies we find that **Greedy-S** performs the best after  $k = 10$  by dropping the  $\lambda_S$  value as aggressively as possible - *equal* to **Optimal** at many places. We find that **Greedy-DavgA** also performs very well. Intuitively this is because the highest degree node in  $\mathbf{A}_{\text{avg}}$  is very well-connected and hence has a huge effect in reducing the  $\mathbf{A}_{\text{avg}}$  value (we discuss more about  $\mathbf{A}_{\text{avg}}$  later in Section 7). At the same time, **Greedy-DmaxA** is comparable to **Greedy-DavgA** as we find the highest degree among both the graphs: so this highest degree will also mostly have the



**Fig. 3.** Experiments on Reality Mining graphs:  $\lambda_S$  vs  $k$  for different immunization policies. Lower is better. **Greedy-DavgA** clearly drops the  $\lambda_S$  value aggressively and is close to the **Greedy-Opt**.



**Fig. 4.** Scatter plot of Max. infections till steady state and  $\lambda_S$  for different immunization policies after  $k = 10$  immunizations. Points closer to the origin are better. All policies perform in accordance to the  $\lambda_S$  values achieved (see Figure 3).

highest mean degree. Finally, **Greedy-AavgA** drops the  $\lambda_S$  value the least among all the policies. Given the first random choice of node, **Greedy-AavgA** can be “trapped” in the neighborhood of a node far from the best node to immunize, and thus can be forced to make a choice based on the limited local information.

Figure 4 demonstrates the effectiveness of our quality metric i.e. the  $\lambda_S$  value for each immunization policy after  $k = 10$  immunizations. It is a scatter plot of Max. infections till steady state and the various  $\lambda_S$  values at the end of the immunizations. So points closer to the origin (minimum footprint and  $\lambda_S$ ) represent better policies. Clearly, **Optimal** should theoretically be the closest to the origin (we don’t show it as it didn’t finish). Also as discussed before, **Greedy-AavgA** is the worst and that is demonstrated by its point. From Figure 3 we can see that **Greedy-S** has the least  $\lambda_S$  value after  $k = 10$ , hence it is closest to the origin and thus has the smallest footprint. Others perform well too, as their final  $\lambda_S$  values were close as well.

To summarize, in our experiments we demonstrated that policies decreasing  $\lambda_S$  the most are the best policies as they result in smaller footprints as well. Also, simple greedy policies were effective and achieved similar  $\lambda_S$  values like expensive combinatorial policies.

## 7 Discussion

We discuss some pertinent issues and give additional explanations in this section.

**Generality of our results:** How can we model more complex situations like ‘unequal duration’ behaviors etc.? Note that the alternation period  $T$  can be longer than 2 and we can have *repetitions* in the set  $\mathcal{A}$  as well e.g. to represent a weekly-style (work day-weekend) alternation we can have  $T = 7$  and  $\mathcal{A} = \{\mathbf{A}_1, \mathbf{A}_1, \dots, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_2\}$ . Similarly, we can model situations like unequal duration of ‘day’ and ‘night’ i.e. unequal duration of matrices  $\mathbf{A}_1$  and  $\mathbf{A}_2$ . Say,

$\mathbf{A}_1$  is present for only 8 hours at work, while  $\mathbf{A}_2$  is present for the remaining 16 hours at home. Then, thinking of an hour as our time-step i.e.  $T = 24$ , the set  $\mathcal{A} = \{\mathbf{A}_1, \dots, \mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_2, \dots\}$ , where  $\mathbf{A}_1$  occurs 8 times in  $\mathcal{A}$  while  $\mathbf{A}_2$  occurs 16 times. All the threshold results carry forward seamlessly in all the above cases.

**Goodness of the  $\mathbf{A}_{\text{avg}}$  matrix:** We saw that Greedy-DavgA gave very good results and was close to Greedy-S and Optimal. This can be explained with the help of the following lemma.

**Lemma 3.**  $(1 - 2\delta)\mathbf{I} + 2\beta\mathbf{A}_{\text{avg}}$  is a first-order approximation of the  $\mathbf{S}$  matrix.

*Proof.* Note that ( $T = 2$ ),

$$\begin{aligned} \mathbf{S} &= \mathbf{S}_1 \times \mathbf{S}_2 \\ &= ((1 - \delta)\mathbf{I} + \beta\mathbf{A}_1) \times ((1 - \delta)\mathbf{I} + \beta\mathbf{A}_2) \\ &= (1 - \delta)^2\mathbf{I} + (1 - \delta)\beta(\mathbf{A}_1 + \mathbf{A}_2) + \beta^2\mathbf{A}_1\mathbf{A}_2 \\ &\approx (1 - 2\delta)\mathbf{I} + \beta(\mathbf{A}_1 + \mathbf{A}_2) = (1 - 2\delta)\mathbf{I} + 2\beta\left(\frac{\mathbf{A}_1 + \mathbf{A}_2}{2}\right) \end{aligned}$$

where we neglected second order terms involving  $\beta$  and  $\delta$ . Thus  $(1 - 2\delta)\mathbf{I} + 2\beta\mathbf{A}_{\text{avg}}$  is a first-order approximation of the  $\mathbf{S}$  matrix.  $\square$

In other words, we can consider the time-varying system to be approximated by a static graph system of the  $\mathbf{A}_{\text{avg}}$  graph adjacency matrix with a virus of the same strength ( $\beta/\delta$  remains the same). The threshold for a static graph with adjacency matrix  $\mathbf{A}$  is  $\beta\lambda_{\mathbf{A}}/\delta$ . So in our static case, we should aim to reduce  $\lambda_{\mathbf{A}_{\text{avg}}}$  as much as possible. Any policy which aims to reduce the  $\lambda_{\mathbf{A}_{\text{avg}}}$  value will approximate our original goal of dropping the  $\lambda_{\mathbf{S}}$  value. Thus, this gives a theoretical justification of why Greedy-DavgA gave good results.

**Temporal Immunization:** In this paper, we concentrated only on *static* immunization policies - policies where once immunized, a node is ‘removed’ from the contact graphs. While this makes sense for biological vaccinations, in a more complex ‘resource’ oriented scenario where each ‘glove’ protects a person only for the time it is worn, a time-varying immunization policy might be more useful. e.g. we may have finite number of gloves to give and we can change the assignment of gloves during day/night. In this case, it may be better to immunize nurses in hospitals during the day by giving them the gloves but during the night, we can decide to give gloves to restaurant waiters or children, because the nurses are now not well-connected in the contact graph. Our threshold results can trivially estimate the impact or any ‘what-if’ scenarios w.r.t. such temporal immunization algorithms.

## 8 Conclusion

In this paper, we analytically studied virus-spreading (specifically the SIS model) on arbitrary, time-varying graphs. Given a set of  $T$  alternating graphs, modeling

e.g. the day/night pattern of human behavior, we ask: (a) what is the epidemic threshold? and (b) what are the best- $k$  nodes to immunize to defend against an epidemic? Our main contributions are:

1. We show how to formulate the problem, namely by approximating it with a *Non-Linear Dynamical System* (NLDS).
2. We give the **first** closed-formula for the threshold, involving the first eigenvalue of the *system-matrix* (see Theorem 2).
3. We use the insight from our threshold to develop and evaluate several immunization policies on real data like MIT reality mining graphs.

Future work can focus on providing bounds for the effectiveness of our immunization heuristics.

## References

1. Albert, R., Jeong, H., Barabási, A.-L.: Diameter of the World-Wide Web. *Nature* 401, 130–131 (1999)
2. Anderson, R.M., May, R.M.: *Infectious diseases of humans: Dynamics and control*. Oxford Press, Oxford (2002)
3. Bailey, N.: *The Mathematical Theory of Infectious Diseases and its Applications*, Griffin, London (1975)
4. Barabási, A.-L., Albert, R.: Emergence of scaling in random networks. *Science* 286, 509–512 (1999)
5. Barrett, C.L., Bisset, K.R., Eubank, S.G., Feng, X., Marathe, M.V.: Episimdemics: an efficient algorithm for simulating the spread of infectious disease over large realistic social networks. In: *ACM/IEEE Conf. on Supercomputing* (2008)
6. Bikhchandani, S., Hirshleifer, D., Welch, I.: A theory of fads, fashion, custom, and cultural change in informational cascades. *Journal of Political Economy* 100(5), 992–1026 (1992)
7. Briesemeister, L., Lincoln, P., Porras, P.: Epidemic profiles and defense of scale-free networks. In: *WORM 2003*, October 27 (2003)
8. Chakrabarti, D., Wang, Y., Wang, C., Leskovec, J., Faloutsos, C.: Epidemic thresholds in real networks. *ACM TISSEC* 10(4) (2008)
9. Cohen, R., Havlin, S., ben Avraham, D.: Efficient immunization strategies for computer networks and populations. *Physical Review Letters* 91(24) (December 2003)
10. Domingos, P., Richardson, M.: Mining the network value of customers. In: *KDD*, pp. 57–66 (2001)
11. Eagle, N., Pentland, A., Lazer, D.: Inferring social network structure using mobile phone data. *Proc. of the National Academy of Sciences* 106(36) (2009)
12. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On power-law relationships of the internet topology. In: *SIGCOMM*, pp. 251–262 (August–September 1999)
13. Ganesh, A., Massoulié, L., Towsley, D.: The effect of network topology on the spread of epidemics. In: *INFOCOM* (2005)
14. Goldenberg, J., Libai, B., Muller, E.: Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters* (2001)
15. Golub, G.H., Van-Loan, C.F.: *Matrix Computations*, 2nd edn. The Johns Hopkins University Press, Baltimore (1989)
16. Granovetter, M.: Threshold models of collective behavior. *Am. Journal of Sociology* 83(6), 1420–1443 (1978)

17. Gruhl, D., Guha, R., Liben-Nowell, D., Tomkins, A.: Information diffusion through blogspace. In: WWW '04 (2004)
18. Hayashi, Y., Minoura, M., Matsukubo, J.: Recoverable prevalence in growing scale-free networks and the effective immunization. arXiv:cond-mat/0305549 v2 (August 6, 2003)
19. Hethcote, H.W.: The mathematics of infectious diseases. *SIAM Review* 42 (2000)
20. Hirsch, M.W., Smale, S.: *Differential Equations, Dynamical Systems and Linear Algebra*. Academic Press, London (1974)
21. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the spread of influence through a social network. In: KDD '03 (2003)
22. Kephart, J.O., White, S.R.: Directed-graph epidemiological models of computer viruses. In: Proceedings of the 1991 IEEE Computer Society Symposium on Research in Security and Privacy, pp. 343–359 (May 1991)
23. Kephart, J.O., White, S.R.: Measuring and modeling computer virus prevalence. In: Proceedings of the 1993 IEEE Computer Society Symposium on Research in Security and Privacy, pp. 2–15 (May 1993)
24. Kumar, R., Novak, J., Raghavan, P., Tomkins, A.: On the bursty evolution of blogspace. In: WWW '03: Proceedings of the 12th International Conference on World Wide Web, pp. 568–576. ACM Press, New York (2003)
25. Kumar, S.R., Raghavan, P., Rajagopalan, S., Tomkins, A.: Trawling the web for emerging cyber-communities. *Computer Networks* 31(11-16), 1481–1493 (1999)
26. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations. In: Proc. of ACM SIGKDD, Chicago, Illinois, USA, pp. 177–187. ACM Press, New York (2005)
27. Madar, N., Kalisky, T., Cohen, R., ben Avraham, D., Havlin, S.: Immunization and epidemic dynamics in complex networks. *Eur. Phys. J. B* 38(2), 269–276 (2004)
28. McCuler, C.R.: The many proofs and applications of perron's theorem. *SIAM Review* 42 (2000)
29. McKendrick, A.G.: Applications of mathematics to medical problems. In: Proceedings of Edin. Math. Society, vol. 14, pp. 98–130 (1926)
30. Milgram, S.: The small-world problem. *Psychology Today* 2, 60–67 (1967)
31. Newman, M.E.J.: Threshold effects for two pathogens spreading on a network. *Phys. Rev. Lett.* (2005)
32. Pastor-Satorras, R., Vespignani, A.: Epidemic dynamics and endemic states in complex networks. *Physical Review E* 63, 066117 (2001)
33. Pastor-Satorras, R., Vespignani, A.: Epidemic dynamics in finite size scale-free networks. *Physical Review E* 65, 035108 (2002)
34. Richardson, M., Domingos, P.: Mining knowledge-sharing sites for viral marketing. In: SIGKDD (2002)
35. Ripeanu, M., Foster, I., Iamnitchi, A.: Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design. *IEEE Internet Computing Journal* 6(1) (2002)
36. Rogers, E.M.: *Diffusion of Innovations*, 5th edn. Free Press, New York (August 2003)
37. Wang, Y., Chakrabarti, D., Wang, C., Faloutsos, C.: Epidemic spreading in real networks: An eigenvalue viewpoint. In: SRDS (2003)

# Adapting Decision DAGs for Multipartite Ranking\*

José Ramón Quevedo, Elena Montañés, Oscar Luaces, and Juan José del Coz

Artificial Intelligence Center, University of Oviedo at Gijón, Spain  
{quevedo,elena,oluaces,juanjo}@aic.uniovi.es

**Abstract.** Multipartite ranking is a special kind of ranking for problems in which classes exhibit an order. Many applications require its use, for instance, granting loans in a bank, reviewing papers in a conference or just grading exercises in an education environment. Several methods have been proposed for this purpose. The simplest ones resort to regression schemes with a pre- and post-process of the classes, what makes them barely useful. Other alternatives make use of class order information or they perform a pairwise classification together with an aggregation function. In this paper we present and discuss two methods based on building a Decision Directed Acyclic Graph (DDAG). Their performance is evaluated over a set of ordinal benchmark data sets according to the C-Index measure. Both yield competitive results with regard to state-of-the-art methods, specially the one based on a probabilistic approach, called PR-DDAG.

## 1 Introduction

Multipartite ranking has been recently named [11] as an extension of the traditional bipartite ranking from the binary to the multiclass case. Bipartite ranking aims to learn a model whose performance is evaluated according to its ability of sorting positive before negative examples. Such ability is commonly assessed in terms of the AUC, which is the area under the Receiver Operating Characteristic (ROC) curve [6]. In fact, multipartite ranking has also been called ordinal ranking, since it relates an ordinal classification and a ranking. Ordinal classification means to perform a classification whose classes display an order. Ordinal ranking goes further and also provides a ranking of the examples within the same class. Obviously, a good multipartite ranker is expected to place examples of the higher classes before examples of the lower ones.

Many applications may take advantage of this special kind of ranking. For instance, a banker commonly classifies customers that ask for a mortgage into classes as high risk, moderate risk, low risk, no risk. Imagine now that a certain number of mortgages are able to grant according to the interests of the bank

---

\* This research has been partially supported by Spanish Ministerio de Ciencia e Innovación (MICINN) grants TIN2007-61273 and TIN2008-06247 and by FICYT, Asturias, Spain, under grant IB09-059-C2.



and that such number falls below the number of customers classified as no risk. Obviously, the bank has to decide within the no risk customers to who give the mortgage. Hence, an order within the examples of each class must be provided. The same demand happens in many other environments, for instance in job vacancies, paper reviews in a conference or waiting list in hospitals according to the urgency degree of the disease.

The main goal of this paper is to explore the use of a Decision Directed Acyclic Graph (DDAG) [24] to address the problem of multipartite ranking. DDAGs have been successfully applied before, but to the best of our knowledge for classification purposes rather than for ranking [3,7,23,24,27]. Unlike other approaches, our proposal makes use of the class order information to lead the build of the graph. We present two different methods that exploit the structure of a DDAG to produce a ranking. The first one follows a crisp approach and produces a global ranking by concatenating a set of consecutive bipartite rankings. The second one is a probabilistic approach where each example is propagated through all possible paths with a cumulative probability attached. The performance of both methods is as good as some state-of-the-art techniques, outperforming them in some situations. In addition, they are computationally more efficient than other algorithms used for the same purpose.

The rest of the paper is organized as follows. Section 2 includes an overview of some related work. In Section 3, the multipartite ranking problem is stated and the main state-of-the-art approaches are described. Then, Section 4 presents the two different DDAG-based methods proposed in this paper to tackle multipartite ranking. In Section 5 results of the experiments over benchmark data sets are described and analyzed. Finally, Section 6 draws some conclusions.

## 2 Related Work

Multipartite ranking is closely related to other fields that have been extensively studied. Ordinal classification, a research field between classification and regression, is one of the closest. In fact, it shares properties with the former that a specific number of classes is stated, and with the latter that such classes are ordered. This is the reason why most of the research is focused on adapting either classification or regression techniques to cope with ordinal classification. However, none of them seem completely adequate, since the former discards class order information, whereas the latter exploits that order relation but making strong assumptions about the distances between classes [19]. Recently, some work has been done to exploit class order information. In [2], authors reduce the problem to the standard two-class problem using both support vector machines and neural networks also providing generalization bounds. Two new support vector approaches for ordinal regression, which optimize multiple thresholds to define parallel discriminant hyperplanes for the ordinal scales are proposed in [4]. Frank and Hall [8] present a simple approach encoding the order information in class attributes allowing the use of any classifier that provides class probability estimates. Also, Herbrich et al. [14] proposed a distribution independent risk formulation of ordinal regression. It allows to derive an uniform convergence bound

whose application configures a large margin algorithm based on a mapping from objects to scalar utility values, thus classifying pairs of objects.

Label ranking is another field also closely related to multipartite ranking, since a required order for the classes is stated. In this framework the goal is to learn a mapping from instances to rankings over a finite number of labels. However, the labels are ranked instead of the instances, as in multipartite ranking. Some research deal with this discipline, as the work in [17], where authors learn a ranking by pairwise comparison or in [1], in which they propose a sophisticated k-NN framework as an alternative to previous binary decomposition techniques.

No so much research can be found about multipartite ranking. In [26], the authors derive distribution-free probabilistic bounds on the expected error of a ranking function learned by a k-partite ranking algorithm. Waegeman et al. [29] generalize the Wilcoxon-Mann-Whitney statistic to more than two ordered categories in order to provide a better evaluation system in ordinal regression. One of the most recent work that copes with the problem itself is [11], in which the use of binary decomposition techniques are explored. On one hand, the authors adapt for this purpose an ordinal classification method [8] that converts the original  $m$  class problem into a  $m - 1$  binary problems. On the other hand, they analyze the use of pairwise classification [10] to induce also a multipartite ranker. The main problem that arises in decomposition approaches is that after such decomposition, an aggregation scheme must be adopted to compose again the original problem. In multipartite ranking combining scoring functions of each model is a better practice than combining the rankings yielded by each model [11].

### 3 Multipartite Ranking

As commented above, multipartite ranking provides a ranking of instances in ordinal classification tasks. Let us include some definitions to formally state the problem.

**Definition 1** *Let be  $\mathcal{L} = \{\ell_1, \dots, \ell_p\}$  a set of classes. Then, a natural order relation denoted by  $\prec$  is defined over  $\mathcal{L}$  such that it holds  $\ell_i \prec \ell_j$  if  $\ell_i$  is semantically below  $\ell_j$ , with  $i, j \in \{1, \dots, p\}$ .*

**Definition 2** *Let be  $\mathcal{L} = \{\ell_1, \dots, \ell_p\}$  a set of classes that satisfy a natural order relation denoted by  $\prec$ . Let also be  $\mathcal{S} = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$  a set of  $m$  instances from an input space  $\mathcal{X}$ , in which each instance  $\mathbf{x}_i$  is labeled as  $\ell_{\mathbf{x}_i} \in \mathcal{L}$ . Then, the goal of multipartite ranking consists of obtaining a ranking function  $f : \mathcal{X} \rightarrow \mathbb{R}$  such that as many as possible  $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$  such that  $\ell_{\mathbf{x}_i} \prec \ell_{\mathbf{x}_j}$  it must satisfy that  $f(\mathbf{x}_i) < f(\mathbf{x}_j)$ .*

Thus,  $f(\cdot)$  must place all the instances classified under the class  $\ell_i$  before any instance classified under the class  $\ell_j$  whenever  $\ell_i \prec \ell_j$ , with  $i, j \in 1, \dots, p$ .

A well-known approach to address this task proposes to convert the original problem into a single binary classification problem. It involves including several

pairwise constraints according to the order relation defined over  $\mathcal{L}$ , each one being a new instance that will feed the binary classifier. Formally, since for all  $i, j \in \{1, \dots, m\}$  such that  $\ell_{\mathbf{x}_i} \prec \ell_{\mathbf{x}_j}$  it must be held that  $f(\mathbf{x}_i) < f(\mathbf{x}_j)$  and assuming that  $f$  is linear, then it holds that  $f(\mathbf{x}_i - \mathbf{x}_j) < 0$ . Analogously,  $f(\mathbf{x}_k - \mathbf{x}_l) > 0$  for all  $k, l \in \{1, \dots, m\}$  such that  $\ell_{\mathbf{x}_l} \prec \ell_{\mathbf{x}_k}$ . Hence,  $\mathcal{S}^+ = \{\mathbf{x}_i - \mathbf{x}_j / \ell_{\mathbf{x}_i} \prec \ell_{\mathbf{x}_j}\}$  is the set of positive examples of the binary classification task and  $\mathcal{S}^- = \{\mathbf{x}_k - \mathbf{x}_l / \ell_{\mathbf{x}_l} \prec \ell_{\mathbf{x}_k}\}$  conforms the negative ones. The main disadvantage of this approach is that the number of instances for such binary classification problem is the order of  $\mathcal{O}(m^2)$ .

Decomposition methods involve several binary learning problems instead of a single one, but with the advantage that such problems are smaller and do not require to increase the number of instances. Some approaches fall into this kind of decomposition previously used for classification [8,10] and recently adapted to multipartite ranking [11]. They differ in the selection of the binary problems they solve.

The Frank and Hall (FH) approach [8] defines  $p - 1$  binary problems, where the  $i$ -th problem consists of obtaining a model  $\mathcal{M}_i$  able to separate the class  $\mathcal{C}_i^+ = \{\ell_1, \dots, \ell_i\}$  from the class  $\mathcal{C}_i^- = \{\ell_{i+1}, \dots, \ell_p\}$ , when  $i$  ranges from 1 to  $p - 1$ . This model provides the probability, denoted by  $P(\ell_i \prec \ell_{\mathbf{x}})$ , of an instance  $\mathbf{x}$  of belonging to a class higher than  $\ell_i$  in the order relation defined over  $\mathcal{L}$ . Such probabilities in turn define one ranking per model. These rankings must be aggregated to obtain a global one. The aggregation function must guarantee that instances of lower classes keep lower in the global ranking and so does the function defined as follows

$$\mathcal{M}(\mathbf{x}) = \sum_{i=1}^{p-1} \mathcal{M}_i(\mathbf{x}) = \sum_{i=1}^{p-1} P(\ell_i \prec \ell_{\mathbf{x}}). \quad (1)$$

The learning by pairwise comparison (LPC) or round robin learning [10] defines  $p(p - 1)/2$  binary problems, where the  $i, j$ -th problem consists of separating the class  $\ell_i$  and  $\ell_j$ , when  $i, j$  range from 1 to  $p$  and  $\ell_i \prec \ell_j$ . This approach is also used in other learning tasks, as in multiclass classification, where the output of each induced model  $\mathcal{M}_{i,j}$  for an example  $\mathbf{x}$  is accounted as a vote for the predicted class  $\ell_i$  or  $\ell_j$ ; then the most voted class is predicted following the MaxWins voting scheme [9]. But, this method is not suitable in multipartite ranking, since it is not able to induce a ranking. In [11], the authors propose two alternative aggregation functions, assuming that binary models yield normalized scores, i.e.  $\mathcal{M}_{i,j}(\mathbf{x}) \in [0, 1]$ , which is the probability that  $\mathbf{x}$  belongs to class  $\ell_j$ . Both sum the predictions in favor of the higher class,

$$\mathcal{M}(\mathbf{x}) = \sum_{1 \leq i < j \leq p} \mathcal{M}_{i,j}(\mathbf{x}), \quad (2)$$

but in the second one those predictions are weighted by the relative frequencies,  $p_i, p_j$  of the classes  $\ell_i, \ell_j$  in the training set,

$$\mathcal{M}(\mathbf{x}) = \sum_{1 \leq i < j \leq p} p_i p_j \mathcal{M}_{i,j}(\mathbf{x}). \quad (3)$$

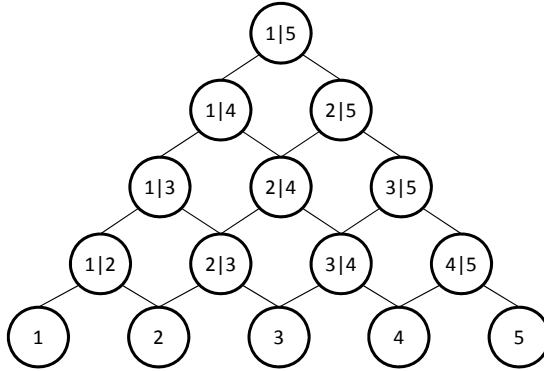


Fig. 1. A Decision Directed Acyclic Graph

## 4 Two DDAG-Based Methods for Multipartite Ranking

In this section we will present and discuss two different methods that are based on a common idea, building a DDAG [24] to cope with the multipartite ranking problem. The first one, called CR-DDAG, produces a global ranking by combining a set of consecutive bipartite rankings. The second one, called PR-DDAG, adds a probabilistic framework based on the structure of a DDAG. Since both methods share the same DDAG structure, we will first describe and motivate it.

A Directed Acyclic Graph (DAG) is a graph whose edges have an orientation and no cycles. A special case is the DDAG presented in [24]. In that paper, the authors describe and analyze a method to solve multiclass classification problems employing a DAG to combine the set of binary classifiers yielded by a decomposition scheme identical to the one used by LPC (see previous section). Thus, DDAG also trains  $p(p-1)/2$  classifiers, one for each pair of classes. However, LPC and DDAG differ in the way they combine the predictions of those classifiers. In the latter, the structure of the DDAG determines such combination.

More in detail, the nodes of a DDAG are arranged in a triangle (see Figure 1) with the single root node at the top, two nodes in the second layer and so on until the final layer of  $p$  leaves. The  $k$ -th node in layer  $l < p$  is connected to the  $k$ -th and  $(k+1)$ -th node in the  $(l+1)$ -th layer. Except for the leaves, each node has an associated model, namely  $\mathcal{M}_{i,j}$ , aimed at separating the classes  $\ell_i$  and  $\ell_j$ , and two successors which will be two leaves when  $i = j - 1$ , or two decision nodes with models  $\mathcal{M}_{i,j-1}$  and  $\mathcal{M}_{i+1,j}$  respectively. The prediction procedure of a DDAG works as follows (see the example in Figure 1). Starting from the root (model  $\mathcal{M}_{1,5}$ ), a DDAG decides at each node, applying model  $\mathcal{M}_{i,j}$ , which of the two classes  $\ell_i$  and  $\ell_j$  is preferred for a certain instance  $\mathbf{x}$ . If the former is the winner, then the instance  $\mathbf{x}$  is evaluated over its left child node (model  $\mathcal{M}_{i,j-1}$  in the example), so class  $\ell_j$  is discarded. Otherwise, the instance  $\mathbf{x}$  is evaluated using its right child node (model  $\mathcal{M}_{i+1,j}$ ), then

class  $\ell_i$  is discarded. The process continues until a leaf is reached whose label is returned for the instance  $\mathbf{x}$ . Thus, an instance is evaluated over  $p - 1$  different models. Notice that a DDAG works as a list in which a class is discarded after each evaluation.

There is no consensus about which classes must be considered first and which ones in last place for each branch of the tree. For instance, in the foundational paper [24] the choice is arbitrary, in [3] a binary decision diagram and Huffman code [30] construction is employed and Jaakkola-Haussler [28] error bound is proposed in [7]. In this paper, we propose an alternative idea. As it can be easily proved, a DDAG predicts incorrectly the true class of an example  $\mathbf{x}$  if one *competent* model in the prediction path followed by  $\mathbf{x}$  fails. As it was defined in other papers [11], a binary model  $\mathcal{M}_{i,j}$  is only competent to classify examples that belong to classes  $\ell_i$  or  $\ell_j$ . Thus, it is quite dangerous to locate at the root a model  $\mathcal{M}_{i,j}$  if the classes  $\ell_i$  and  $\ell_j$  are hard to separate, because that model will evaluate all examples of these classes. Hence, placing such kind of models at lower levels is preferable, since they will classify less examples.

A simple adaptation of this idea to multipartite ranking (and in general to ordinal classification tasks) is based on the intuitive assumption, validated experimentally in [16], that the ordinal structure of the set of classes is also reflected in the topology of the input space. Thus, it seems that the lowest ( $\ell_1$ ) and highest ( $\ell_p$ ) classes of the order defined over  $\mathcal{L}$  are those likely to be separated best. So, model  $\mathcal{M}_{1,p}$  is proposed to be located at the root and, therefore, either the lowest or the highest class is discarded in the first layer. Recursively applying the same idea, our DDAG will place at each node the model between the two extreme classes of the ordered subset of classes that have not been discarded by its ancestor models.

Assuming that the order of the classes is  $\ell_1 \prec \ell_2 \prec \ell_3 \prec \ell_4 \prec \ell_5$ , Figure [1] shows the structure of a DDAG employed for both methods proposed in this paper. In symbols, if  $\mathcal{M}_{i,j}$  corresponds to the  $k$ -th node in layer  $l < p$ , then  $\mathcal{M}_{i,j-1}$  and  $\mathcal{M}_{i+1,j}$  correspond to the  $k$ -th and  $(k + 1)$ -th node in the  $(l + 1)$ -th layer. For instance, in Figure [1] the node 2|4 is the 2-nd node of the 3-rd layer that correspond to the model  $\mathcal{M}_{2,4}$  which separates classes  $\ell_2$  and  $\ell_4$  and it is connected to the 2-nd and 3-rd nodes (respectively 2|3 and 3|4) of the 4-th layer that respectively correspond to models  $\mathcal{M}_{2,3}$  and  $\mathcal{M}_{3,4}$ .

#### 4.1 Consecutive Rankings DDAG (CR-DDAG)

At first sight, it is not trivial how to adapt a DDAGs to obtain a multipartite ranking. Originally, they were designed to deal with multiclass classification [24], which is a quite different task. Our first proposal works under the hypothesis that a multipartite ranking can be broken down into a set of ordered bipartite rankings. For instance, for a problem with 5 classes, i.e.  $\mathcal{L} = \{\ell_1, \dots, \ell_5\}$ , then the whole multipartite ranking can be formed concatenating the consecutive bipartite rankings  $\{1-2, 2-3, 3-4, 4-5\}$ . Notice that these consecutive rankings correspond to the last level before the leaves in the structure of the DDAG described before (see Figure [1]).

This method relies on the assumption that if each bipartite ranker orders well the examples of its two classes, then concatenating those consecutive binary rankings will lead to a good overall ranking. But, first of all a classification procedure is required to decide which bipartite ranker will be used for each instance. Here is when the DDAG structure plays its role. Taking Figure 1 as reference, the proposal consists of employing the remaining models, displayed in the higher levels of the DDAG, to select the bipartite ranker that must be applied. Thus, our DDAG is divided in two parts: the upper levels take charge of classifying examples, and the lower level of ordering them. Hence, a mechanism to merge the consecutive rankings into a global ranking is required.

For that purpose some modifications in DDAG are carried out in order to obtain a CR-DDAG. First, the leaves are ruled out. Secondly, the models of the layer immediately before the leaves, i.e. the set of consecutive bipartite rankers  $(\mathcal{M}_{i,i+1})$ , must yield a value in the interval  $(0, 1)$ . Finally, the final ranking score is computed according to the following expression

$$\mathcal{M}(\mathbf{x}) = i + \mathcal{M}_{i,i+1}(\mathbf{x}), \quad (4)$$

where  $\mathcal{M}_{i,i+1}$  correspond to the model selected by the cascade classification process carried out by the higher nodes of the DDAG. Adding the class index  $i$  to the output of the bipartite ranker guarantees that an instance ranked by  $\mathcal{M}_{i,i+1}$  is placed in the global ranking higher than an instance ranked by  $\mathcal{M}_{j,j+1}$  with  $j < i$ .

The main disadvantage of CR-DDAG is that if a competent model for an instance of class  $\ell_j$  fails during the classification process, then such instance will follow a path ending in node whose model will be  $\mathcal{M}_{i,i+1}$  with  $j \notin \{i, i+1\}$ . Then, if  $j < i$  (respectively  $j > i+1$ ), the instance may be placed much higher (respectively much lower) in the global ranking than it should be. Therefore, CR-DDAG has two potential sources of errors. On one hand, the classification process can choose the incorrect ranking model, and, on the other hand, the bipartite rankers may commit mistakes. The first one is more damaging in the sense that it could produce bigger losses in ranking evaluation measures.

## 4.2 Probabilistic Ranking DDAG (PR-DDAG)

In order to diminish some of the undesirable drawbacks of CR-DDAG, we propose a Probabilistic Ranking Decision Directed Acyclic Graph in which examples are propagated through every edge with a probability attached, so no irrevocable decisions are taken. In fact, PR-DDAG shares some ideas with the LPC approach. First, it exploits the redundancy considering the outcomes of several models to rank an instance. And second, it relies on the assumption that the order of the output space is also reflected on the topology of the input space, in the sense that the prediction of  $\mathcal{M}_{i,k}(\mathbf{x})$  for an instance of class  $\ell_j$  will be higher if  $k \leq j$  and lower if  $j \leq i$ . However, the main difference with LPC is that PR-DDAG computes the posterior probabilities using the structure of the DDAG. The goal is that the contribution of competent models will be higher in

the global ranking, reducing the effect of the so-called *non-competence problem*. In the next section, we will explain deeply this property.

PR-DDAG supposes that every model  $\mathcal{M}_{i,j}(\mathbf{x})$  predicts the probability of  $\mathbf{x}$  of belonging to the positive class ( $\ell_j$ ). Obviously, the probability of being of the negative class ( $\ell_i$ ) will be  $1 - \mathcal{M}_{i,j}(\mathbf{x})$ . These probabilities are successively propagated through the graph from the root to the leaves, in such a way that an instance will reach a node  $i|j$  with a probability  $P_{i,j}(\mathbf{x})$  computed as follows

$$P_{i,j}(\mathbf{x}) = \begin{cases} 1, & i = 1, j = p, \\ P_{i,j+1}(\mathbf{x}) \cdot (1 - \mathcal{M}_{i,j+1}(\mathbf{x})), & i = 1, j < p, \\ P_{i-1,j}(\mathbf{x}) \cdot \mathcal{M}_{i-1,j}(\mathbf{x}), & i > 1, j = p, \\ P_{i-1,j}(\mathbf{x}) \cdot \mathcal{M}_{i-1,j}(\mathbf{x}) + P_{i,j+1}(\mathbf{x}) \cdot (1 - \mathcal{M}_{i,j+1}(\mathbf{x})), & i > 1, j < p. \end{cases} \quad (5)$$

Notice that the method states that  $P_{1,p}(\mathbf{x}) = 1$ , since any instance  $\mathbf{x}$  must arrive to the root node with probability 1. At the end, this propagation process provides a probability distribution of the classes for an instance  $\mathbf{x}$ , that is,  $\{P_{i,i}(\mathbf{x}) : i = 1, \dots, p\}$ .

Finally, in order to produce the global ranking, PR-DDAG employs the aggregation function proposed in [20]:

$$\mathcal{M}(\mathbf{x}) = \sum_{i=1}^p T(i) \cdot P_{i,i}(\mathbf{x}), \quad (6)$$

where  $T(i)$  is some monotone increasing function of the relevance level  $i$ . According to [20], it seems that complex functions do not provide better rankings, hence, the simple  $T(i) = i$ , called *expected relevance*, can be a sensible and stable choice. In this case, the product by the index of the class in the summation enhances the value of the probability as the index of the class increases.

### 4.3 Analyzing the Properties of CR-DDAG and PR-DDAG

Table II shows a summary of the main properties of all approaches described above (FH, LPC, CR-DDAG and PR-DDAG). All methods solve  $p(p-1)/2$  binary problems except FH which solves just  $p-1$ , which is related to the training set used in each binary problem they solve. Particularly, FH employs the whole data set, whereas only instances of two classes are used in the rest of the approaches. All of them take each instance  $(p-1)$  times, and assuming an uniform distribution of the instances through the classes, LPC, CR-DDAG and PR-DDAG only handle  $m/p$  instances in each binary problem against  $m$  that FH uses. Then, taking into account the training size and that a  $\mathcal{O}(m^\alpha)$  binary classifier (typically with  $\alpha > 1$ ) is chosen, FH has a complexity  $\mathcal{O}(pm^\alpha)$ , whereas LPC, CR-DDAG and PR-DDAG reduce it to  $\mathcal{O}(p^{2-\alpha}m^\alpha)$ . Thus, those methods are more efficient than FH whenever a base learner with super-linear complexity is applied. Concerning to the evaluation stage (see again Table II) FH is comparable to CR-DDAG, which only need to compute  $p-1$  evaluations, while LPC and PR-DDAG evaluate all models.

**Table 1.** Binary problems and evaluations required by each method

	FH	LPC	CR-DDAG	PR-DDAG
Binary problems	$p - 1$	$p(p - 1)/2$	$p(p - 1)/2$	$p(p - 1)/2$
Classes in training	All classes	Only two	Only two	Only two
Whole Training size	$(p - 1)m$	$(p - 1)m$	$(p - 1)m$	$(p - 1)m$
Binary Training size	$m$	$m/p$	$m/p$	$m/p$
Binary complexity	$\mathcal{O}(m^\alpha)$	$\mathcal{O}(p^{-\alpha}m^\alpha)$	$\mathcal{O}(p^{-\alpha}m^\alpha)$	$\mathcal{O}(p^{-\alpha}m^\alpha)$
Whole complexity	$\mathcal{O}(pm^\alpha)$	$\mathcal{O}(p^{2-\alpha}m^\alpha)$	$\mathcal{O}(p^{2-\alpha}m^\alpha)$	$\mathcal{O}(p^{2-\alpha}m^\alpha)$
Evaluations	$p - 1$	$p(p - 1)/2$	$p - 1$	$p(p - 1)/2$
Non-competence problem	No	Yes	No	Yes
Redundancy	No	Yes	No	Yes

In classification, the non-competence problem does actually not matter so much for LPC provided all competent models make correct predictions; the same is true for DDAGs. As explained in [11], however, this property is lost for LPC in the case of multipartite ranking. Interestingly, it seems to be maintained for CR-DDAG: as long as all competent binary classifiers make correct decisions, an instance from class  $\ell_i$  must end up either in the bipartite models  $\mathcal{M}_{i-1,i}$  or  $\mathcal{M}_{i,i+1}$ , and these are in turn handled by competent bipartite rankers.

PR-DDAG reduces the influence of non-competent models. Let us explain it with a simple example. Imagine a problem with 3 classes, so the model  $\mathcal{M}_{1,3}$  will be at the root, and the models  $\mathcal{M}_{1,2}$  and  $\mathcal{M}_{2,3}$  at the second level. If we evaluate an instance of class 2, the prediction of  $\mathcal{M}_{1,3}$  will distribute its input probability (1, since it is the root of the DDAG) between its child nodes. Applying recursively Equation (5), the posterior probability for class 2 will be:

$$P_{2,2}(\mathbf{x}) = (1 - \mathcal{M}_{1,3}(\mathbf{x})) \cdot \mathcal{M}_{1,2}(\mathbf{x}) + \mathcal{M}_{1,3}(\mathbf{x}) \cdot (1 - \mathcal{M}_{2,3}(\mathbf{x})).$$

Notice that the sum of the input probabilities of models  $\mathcal{M}_{1,2}$  and  $\mathcal{M}_{2,3}$  is 1. Thus, the role of the non-competent model  $\mathcal{M}_{1,3}$  is to distribute the importance of models  $\mathcal{M}_{1,2}$  and  $\mathcal{M}_{2,3}$  in order to compute that posterior probability. But, since both models are competent to that task, the decision of  $\mathcal{M}_{1,3}$  is not so important. In fact, due to the design of the DDAG, it can be proved that, for any class, the sum of the input probabilities of all its competent models coming from non-competent models is always 1. This means that the role of the non-competent models placed above them on the DDAG is to distribute those input probabilities. At the end, posterior probabilities will depend heavily on competent models.

On the other hand, LPC is the most redundant method. In fact, it is the redundancy obtained from applying such number of models what seems to thwart the misclassification errors produced by the non-competence problem. PR-DDAG also provides some kind of redundancy, due to the evaluations over all competent models, but in a lower degree compared to LPC.



**Table 2.** Properties of the data sets used in the experiments. All of them were taken from the WEKA repository ([http://www.cs.waikato.ac.nz/~ml/weka/index\\_datasets.html](http://www.cs.waikato.ac.nz/~ml/weka/index_datasets.html))

Data set	#examples	#numeric feat.	#nominal feat.	#classes
asbestos	83	1	2	3
balance-scale	625	4	0	3
cmc	1473	2	7	3
pasture	36	21	1	3
post-operative	90	0	8	3
squash (unst.)	52	20	3	3
car	1728	0	6	4
grub-damage	155	8	6	4
nursery	1000	0	9	4
swd	1000	100	0	4
bondrate	57	4	7	5
eucalyptus	736	14	5	5
lev	1000	4	4	5
era	1000	4	4	9
esl	488	4	4	9

## 5 Experiments

In this section we report the results of the experiments performed to evaluate the approaches proposed in this paper to tackle multipartite ranking. This set of experiments was designed to address three main goals. Firstly, CR-DDAG and PR-DDAG were compared with the main state-of-the-art binary decomposition approaches for multipartite ranking, FH and LPC (see Section 3). Secondly, we studied the influence of the learning algorithm used to build each binary classifier on the compared multipartite ranking methods. Finally, since our aim is to predict a ranking, we included a base learner able to optimize the AUC measure in a bipartite ranking task.

Before discussing the experimental results, the next subsection describes the settings used in the experiments: learning algorithms, data sets, procedures to set parameters, and the measure to estimate the scores.

### 5.1 Experimental Settings

Due to the lack of ordinal benchmark data sets, several previous works have used discretized regression data sets. Despite this can be reasonable, here we wanted to study the performance of the different approaches only on truly ordinal data sets. Thus, the experiments were performed over several ordinal data sets whose main properties are shown in Table 2. This group of data sets were previously used in [16].

We compared the five multipartite ranking algorithms described through the paper, namely, FH (Eq. 1), the two different aggregation methods for LPC

approach: the unweighted variant (Eq. 2), called LPCU in the following, and the weighted version LPCW (Eq. 3), and finally the two proposed DDAG-based methods, CR-DDAG (Eq. 4) and PR-DDAG (Eq. 6).

All of them were implemented with three different base learners. First, we employed a binary SVM (*libsvm* implementation [31]) with probabilistic outputs, the second one was the *logistic regression* of [21], and finally, the implementation of  $SVM_{perf}$  presented in [18], setting the target maximization function to be the AUC. In this last case, since its output is not a probability, the algorithm reported in [25] was used to map it into a probability [22].  $SVM_{perf}$  and *libsvm* were used with the linear kernel. In all cases, the regularization parameter  $C$  was established for each binary problem performing a grid search over the interval  $C \in [10^{-2}, \dots, 10^2]$  optimizing the accuracy in the cases of *libsvm* and *logistic regression* and the AUC in the case of  $SVM_{perf}$ . Both, accuracy and AUC, were estimated by means of a 2-fold cross validation repeated 5 times.

The ranking performance of the methods was measured in terms of the C-index, estimated using a 5-fold cross validation. C-index is a metric of concordance [12] commonly used in statistics and equivalent to the pairwise ranking error [14]. It has been recently used in [11] for multipartite ranking as an estimation of the probability that a randomly chosen pair of instances from different classes is ranked correctly. If  $\mathcal{M}$  is a model,  $\mathcal{S}$  the whole training set,  $p$  the number of different classes and  $\mathcal{S}_k$  with  $k = 1, \dots, p$  the instances of  $\mathcal{S}$  of the class  $k$ , then the C-index is defined by

$$C(\mathcal{M}, \mathcal{S}) = \frac{1}{\sum_{i < j} |\mathcal{S}_i| |\mathcal{S}_j|} \sum_{1 \leq i < j \leq p} |\mathcal{S}_i| |\mathcal{S}_j| AUC(\mathcal{M}, \mathcal{S}_i \cup \mathcal{S}_j). \quad (7)$$

This metric considers that each class contribution is proportional to its size. An analogous metric that grants the same importance for all classes is the Jonckheere-Terpstra statistic [15], proposed in [13] as another multiclass extension of the AUC. However, conclusions reported in [11] show little differences between them.

Finally, according to the recommendations exposed in [5], a two-step statistical test procedure is carried out. The first step consists of a Friedman test of the null hypothesis that all rankers have equal performance. Then, in case that this hypothesis is rejected, a Nemenyi test to compare learners in a pairwise way is conducted. The average ranks over all data sets are computed and shown at the last row of every table. The ranks of each data sets are indicated in brackets close to the corresponding C-index. In case of ties, average ranks are shown. Since we are comparing 5 algorithms over 15 data sets, the critical rank differences are 1.58 and 1.42 for significance levels of 5% and 10%, respectively.

## 5.2 Experimental Results

Table 3, Table 4 and Table 5 show the ranking performance in terms of the C-index for all approaches when *libsvm*, *logistic regression* and  $SVM_{perf}$  were respectively adopted as base learners. Analyzing the obtained results using *libsvm*

**Table 3.** C-index for all approaches using *libsvm* as the base learner

	FH	LPCU	LPCW	CR-DDAG	PR-DDAG
asbestos	84.83 (1.00)	75.79 (4.00)	76.48 (2.00)	72.78 (5.00)	76.41 (3.00)
balance-scale	97.91 (4.00)	97.95 (3.00)	97.90 (5.00)	97.99 (2.00)	98.03 (1.00)
cmc	68.59 (1.00)	66.41 (5.00)	66.88 (4.00)	67.24 (3.00)	67.68 (2.00)
pasture	90.40 (4.00)	92.13 (1.50)	92.13 (1.50)	88.67 (5.00)	90.53 (3.00)
post-operative	53.78 (4.00)	50.95 (5.00)	54.48 (2.00)	54.23 (3.00)	57.43 (1.00)
squash (unst.)	89.14 (3.00)	89.59 (1.00)	88.89 (4.50)	88.89 (4.50)	89.41 (2.00)
car	98.59 (2.00)	88.37 (5.00)	98.44 (3.00)	98.92 (1.00)	98.30 (4.00)
grub-damage	73.21 (1.00)	70.67 (2.00)	69.68 (3.00)	68.39 (5.00)	69.13 (4.00)
nursery	98.13 (4.00)	97.03 (5.00)	98.33 (2.00)	98.52 (1.00)	98.30 (3.00)
swd	81.03 (3.00)	81.19 (2.00)	81.01 (4.00)	80.64 (5.00)	81.44 (1.00)
bondrate	66.66 (1.00)	52.76 (5.00)	56.57 (4.00)	61.73 (3.00)	61.98 (2.00)
eucalyptus	93.72 (1.00)	89.47 (5.00)	90.37 (4.00)	93.44 (3.00)	93.63 (2.00)
lev	86.36 (3.00)	86.32 (4.00)	86.46 (1.00)	86.03 (5.00)	86.38 (2.00)
era	73.91 (1.00)	73.81 (3.00)	73.88 (2.00)	72.90 (5.00)	73.66 (4.00)
esl	95.58 (4.00)	95.62 (3.00)	96.16 (1.00)	96.13 (2.00)	95.48 (5.00)
Avg. rank	(2.47)	(3.57)	(2.87)	(3.50)	(2.60)

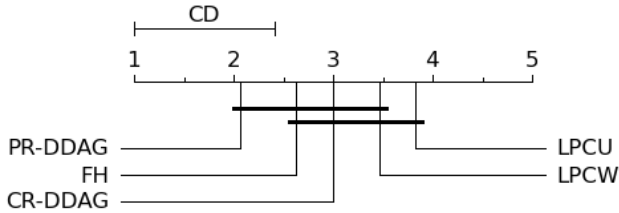
to learn each binary model (see Table 3), we observe that the best method is FH, following by PR-DDAG, LPCW, CR-DDAG and LPCU. However, none of the methods is significantly better using a Nemenyi test. Between our approaches, PR-DDAG wins in 11 out of 15 data sets. The same happens between LPCW and LPCU, the former outperforms the latter 10 times and it is only worse in 4 data sets.

These first results were quite surprising because they contradict in some way the experimental results reported in [11]. In that work, FH was significantly better than LPCW and LPCU. The reasons that can explain these quite opposite conclusions can be: i) we used only truly ordinal data sets, and ii) the base learner was different (in [11] *logistic regression* was used). Motivated for this second reason, we also employed *logistic regression* as the learning algorithm (see Table 4). In this case, the obtained results were similar to those presented in [11]. Now, our approach PR-DDAG obtains better performance, followed by FH. According to a Nemenyi test, PR-DDAG is only significantly better than LPCU (for a significance level of 5%), the worst method in this experiment. The difference between PR-DDAG and LPCW (1.4) is very close to the critical difference (1.42), but it is not significant (see Figure 2). In this case, CR-DDAG obtains better results than both approaches based on LPC. It seems that *logistic regression* is not a good choice as base learner for LPC, since both versions offer the worst performance. However, it keeps the fact that LPCW provides better results than LPCU, although the differences are less remarkable than in case of *libsvm* adopted as base learner.

In the last experiment the goal was to check if these multipartite ranking algorithms can benefit of using a ranking base learner, like  $SVM_{perf}$  optimizing

**Table 4.** C-index for all approaches using *logistic regression* as the base learner

	FH	LPCU	LPCW	CR-DDAG	PR-DDAG
asbestos	82.96 (5.00)	83.50 (2.00)	83.10 (4.00)	83.51 (1.00)	83.46 (3.00)
balance-scale	97.66 (2.00)	97.63 (5.00)	97.65 (3.50)	97.90 (1.00)	97.65 (3.50)
cmc	67.99 (3.00)	67.51 (5.00)	67.98 (4.00)	68.54 (2.00)	68.74 (1.00)
pasture	85.73 (5.00)	86.53 (3.50)	86.53 (3.50)	87.47 (1.00)	86.80 (2.00)
post-operative	45.47 (3.00)	45.52 (2.00)	43.74 (4.00)	42.87 (5.00)	45.66 (1.00)
squash (unst.)	91.26 (2.00)	90.81 (3.00)	90.30 (5.00)	90.64 (4.00)	91.67 (1.00)
car	99.04 (3.00)	86.57 (5.00)	98.94 (4.00)	99.12 (1.00)	99.07 (2.00)
grub-damage	74.16 (1.00)	71.67 (3.00)	71.66 (4.00)	71.04 (5.00)	73.59 (2.00)
nursery	98.58 (1.00)	89.47 (4.00)	88.12 (5.00)	98.57 (2.00)	97.15 (3.00)
swd	81.10 (4.00)	81.14 (3.00)	81.17 (2.00)	81.07 (5.00)	81.49 (1.00)
bondrate	73.07 (3.00)	72.01 (4.00)	71.33 (5.00)	76.85 (1.00)	73.95 (2.00)
eucalyptus	93.99 (2.00)	88.35 (5.00)	89.84 (4.00)	93.76 (3.00)	94.06 (1.00)
lev	86.44 (2.50)	86.41 (4.00)	86.53 (1.00)	86.32 (5.00)	86.44 (2.50)
era	73.90 (2.00)	73.84 (4.00)	73.96 (1.00)	72.83 (5.00)	73.89 (3.00)
esl	96.18 (1.00)	95.39 (5.00)	96.17 (2.00)	95.91 (4.00)	96.08 (3.00)
Avg. rank	(2.63)	(3.83)	(3.47)	(3.00)	(2.07)

**Fig. 2.** Friedman-Nemenyi Test ( $p < 0.1$ ) for all methods using *logistic regression*

the AUC measure (see Table 5). In this case the differences between all the approaches are smaller, only 0.74 between the best (FH) and the worst (LPCU). Now, LPCW and PR-DDAG are almost tied, and the difference between PR-DDAG and CR-DDAG is the smallest of the three experiments. Again, LPCU attains worse results than the other methods. It seems that neither CR-DDAG nor PR-DDAG take advantage of optimizing the AUC. However, it is a good choice for LPCW, since it obtains better position in this case.

Summarizing all these results, we can draw some conclusions. In general, none of the methods is significantly better than the others. In fact, only in one case, an algorithm (PR-DDAG) is significantly better than another (LPCU). However, it seems that FH, PR-DDAG and LPCW perform slightly better than the rest. Both PR-DDAG and FH are quite stable, no matter what base learner is used. On the other hand, the choice of the base learner is quite important for LPC methods. Particularly, in the case of using a *logistic regression*, applying LPC approaches is not the best choice.

**Table 5.** C-index for all approaches using SVM<sub>perf</sub> as the base learner

	FH		LPCU		LPCW		CR-DDAG		PR-DDAG	
asbestos	82.34	(4.00)	82.62	(3.00)	81.97	(5.00)	84.37	(1.00)	83.69	(2.00)
balance-scale	98.06	(1.50)	97.91	(4.50)	97.91	(4.50)	98.06	(1.50)	97.92	(3.00)
cmc	68.15	(2.00)	67.32	(4.00)	67.72	(3.00)	67.10	(5.00)	68.44	(1.00)
pasture	90.40	(4.00)	93.20	(1.50)	93.20	(1.50)	83.07	(5.00)	90.53	(3.00)
post-operative	53.75	(5.00)	55.31	(4.00)	62.81	(1.00)	59.16	(2.00)	56.38	(3.00)
squash (unst.)	87.73	(1.00)	85.58	(4.50)	87.01	(3.00)	87.21	(2.00)	85.58	(4.50)
car	98.86	(3.00)	89.71	(5.00)	98.88	(1.00)	98.83	(4.00)	98.87	(2.00)
grub-damage	72.23	(1.00)	70.72	(3.00)	69.84	(4.00)	64.67	(5.00)	72.03	(2.00)
nursery	98.56	(2.00)	96.90	(5.00)	98.31	(4.00)	98.55	(3.00)	98.60	(1.00)
swd	80.57	(3.00)	81.08	(1.00)	81.03	(2.00)	80.56	(4.00)	80.51	(5.00)
bondrate	69.37	(2.00)	59.95	(5.00)	69.89	(1.00)	68.54	(3.00)	65.77	(4.00)
eucalyptus	93.19	(1.00)	91.11	(4.00)	91.78	(2.00)	88.46	(5.00)	91.46	(3.00)
lev	86.32	(4.00)	86.56	(1.00)	86.27	(5.00)	86.47	(2.00)	86.46	(3.00)
era	73.64	(4.00)	73.78	(2.00)	73.99	(1.00)	72.84	(5.00)	73.73	(3.00)
esl	96.03	(2.00)	95.74	(3.00)	93.90	(5.00)	96.07	(1.00)	95.34	(4.00)
Avg. rank	(2.63)		(3.37)		(2.87)		(3.23)		(2.90)	

Focusing on our proposals, PR-DDAG obtains quite similar results than FH and besides it is computationally more efficient. Despite its appealing idea, CR-DDAG performs worse, but not significantly, than PR-DDAG. One reason for this behavior may be the lack of redundancy CR-DDAG suffers from. Indeed, the performance is affected by the classification stage before the consecutive ranking combination takes place.

Finally, using a binary base learner that optimizes the AUC does not improve the results of decomposition multipartite methods. Following [5], we used the Wilcoxon signed ranks test to compare the performance of the same method using different base learners. Only twice significant differences were found. For LPCU, SVM<sub>perf</sub> is better than *libsvm* at level  $p < 0.05$ , and for LPCW, *logistic regression* is better than SVM<sub>perf</sub> at level  $p < 0.10$ .

## 6 Conclusions

This paper proposes two multipartite ranking approaches that exploits the order information the classes exhibits. Both have as the point of the departure the structure of a Decision Directed Acyclic Graph (DDAG) and include a pairwise decomposition technique. One of them, called Consecutive Ranking DDAG (CR-DDAG) combines a set of consecutive bipartite rankings to obtain a global ranking, but first it performs a classification to decide which bipartite ranker must be applied. The other, called Probabilistic Ranking DDAG (PR-DDAG) includes a probabilistic framework based on propagating probabilities through the graph.

Several experiments over a benchmark ordinal data set were carried out using different base learners, including one algorithm that optimizes the AUC measure. None of the methods outperforms the others, but PR-DDAG exhibits competitive performance, in terms of the C-index measure, with regard to other state-of-the-art algorithms previously proposed in the literature for the same purpose. PR-DDAG also presents interesting theoretical properties, as its computational complexity and its capacity of reducing the non-competence problem.

## References

1. Brinker, K., Hüllermeier, E.: Case-based label ranking. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 566–573. Springer, Heidelberg (2006)
2. Cardoso, J.S., da Costa, J.F.P.: Learning to classify ordinal data: The data replication method. *Journal of Machine Learning Research* 8, 1393–1429 (2007)
3. Chen, P., Liu, S.: An improved dag-svm for multi-class classification. *International Conference on Natural Computation* 1, 460–462 (2009)
4. Chu, W., Sathiya Keerthi, S.: New approaches to support vector ordinal regression. In: De Raedt, L., Wrobel, S. (eds.) *Proceedings of the ICML'05*, vol. 119, pp. 145–152. ACM, New York (2005)
5. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* 7, 1–30 (2006)
6. Fawcett, T.: An introduction to roc analysis. *Pattern Recognition Letters* 27(8), 861–874 (2006)
7. Feng, J., Yang, Y., Fan, J.: Fuzzy multi-class svm classifier based on optimal directed acyclic graph using in similar handwritten chinese characters recognition. In: Wang, J., Liao, X.-F., Yi, Z. (eds.) *ISNN 2005*. LNCS, vol. 3496, pp. 875–880. Springer, Heidelberg (2005)
8. Frank, E., Hall, M.: A simple approach to ordinal classification. In: *EMCL '01: Proceedings of the 12th European Conference on Machine Learning*, London, UK, pp. 145–156. Springer, Heidelberg (2001)
9. Friedman, J.H.: Another approach to polychotomous classification. Technical report, Department of Statistics, Stanford University (1996)
10. Fürnkranz, J.: Round robin classification. *Journal of Machine Learning Research* 2, 721–747 (2002)
11. Fürnkranz, J., Hüllermeier, E., Vanderlooy, S.: Binary decomposition methods for multipartite ranking. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) *ECML PKDD 2009*. LNCS, vol. 5781, pp. 359–374. Springer, Heidelberg (2009)
12. Gonen, M., Heller, G.: Concordance probability and discriminatory power in proportional hazards regression. *Biometrika* 92(4), 965–970 (2005)
13. Hand, D.J., Till, R.J.: A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine Learning* 45(2), 171–186 (2001)
14. Herbrich, R., Graepel, T., Obermayer, K.: Large margin rank boundaries for ordinal regression. In: Smola, Bartlett, Schoelkopf, Schuurmans (eds.) *Advances in Large Margin Classifiers*. MIT Press, Cambridge (2000)
15. Higgins, J.: *Introduction to Modern Nonparametric Statistics*. Duxbury Press, Boston (2004)

16. Hühn, J.C., Hüllermeier, E.: Is an ordinal class structure useful in classifier learning? *Int. J. of Data Mining Modelling and Management* 1(1), 45–67 (2008)
17. Hüllermeier, E., Fürnkranz, J., Cheng, W., Brinker, K.: Label ranking by learning pairwise preferences. *Artificial Intelligence* 172(16–17), 1897–1916 (2008)
18. Joachims, T.: A support vector method for multivariate performance measures. In: *ICML '05: Proceedings of the 22nd International Conference on Machine Learning*, pp. 377–384. ACM, New York (2005)
19. Kramer, S., Widmer, G., Pfahringer, B., de Groeve, M.: Prediction of ordinal classes using regression trees. In: Ohsuga, S., Raś, Z.W. (eds.) *ISMIS 2000. LNCS (LNAI)*, vol. 1932, pp. 426–434. Springer, Heidelberg (2000)
20. Li, P., Burges, C.J.C., Wu, Q.: Mcrank: Learning to rank using multiple classification and gradient boosting. In: Platt, J.C., Koller, D., Singer, Y., Roweis, S.T. (eds.) *NIPS. MIT Press, Cambridge* (2007)
21. Lin, C.-J., Weng, R.C., Sathiyaraj, S.: Trust region newton method for logistic regression. *Journal of Machine Learning Research* 9, 627–650 (2008)
22. Luaces, O., Taboada, F., Albaiceta, G.M., Domínguez, L.A., Enríquez, P., Bahamonde, A.: Predicting the probability of survival in intensive care unit patients from a small number of variables and training examples. *Artificial Intelligence in Medicine* 45(1), 63–76 (2009)
23. Nguyen, C.D., Dung, T.A., Cao, T.H.: Text classification for dag-structured categories. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) *PAKDD 2005. LNCS (LNAI)*, vol. 3518, pp. 290–300. Springer, Heidelberg (2005)
24. Platt, J.C., Cristianini, N., Shawe-taylor, J.: Large margin dags for multiclass classification. In: *Advances in Neural Information Processing Systems*, pp. 547–553. MIT Press, Cambridge (2000)
25. Platt, J.C., Platt, J.C.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: *Advances in Large Margin Classifiers*, pp. 61–74. MIT Press, Cambridge (1999)
26. Rajaram, S., Agarwal, S.: Generalization bounds for  $k$ -partite ranking. In: Agarwal, S., Cortes, C., Herbrich, R. (eds.) *Proceedings of the NIPS 2005 Workshop on Learning to Rank*, pp. 28–23 (2005)
27. Takahashi, F., Abe, S.: Optimizing directed acyclic graph support vector machines. In: *IAPR - TC3 International Workshop on Artificial Neural Networks in Pattern Recognition University of Florence, Italy* (2003)
28. Vapnik, V., Chapelle, O.: Bounds on error expectation for support vector machines. *Neural Computation* 12(9), 2013–2036 (2000)
29. Waegeman, W., De Baets, B., Boullart, L.: Roc analysis in ordinal regression learning. *Pattern Recognition Letters* 29(1), 1–9 (2008)
30. Weiss, M.A.: *Data structures and algorithm analysis in C*, 2nd edn. Addison-Wesley Longman Publishing Co., Inc., Boston (1997)
31. Wu, T.F., Lin, C.J., Weng, R.C.: Probability estimates for multi-class classification by pairwise coupling. *J. of Machine Learning Research* 5, 975–1005 (2004)

# Fast and Scalable Algorithms for Semi-supervised Link Prediction on Static and Dynamic Graphs

Rudy Raymond<sup>1</sup> and Hisashi Kashima<sup>2</sup>

<sup>1</sup> IBM Research – Tokyo  
1623-14 Shimo-tsuruma, Yamato  
Kanagawa 242-8502, Japan  
raymond@jp.ibm.com

<sup>2</sup> Department of Mathematical Informatics  
The University of Tokyo  
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan  
kashima@mist.i.u-tokyo.ac.jp

**Abstract.** Recent years have witnessed a widespread interest on methods using both link structure and node information for link prediction on graphs. One of the state-of-the-art methods is Link Propagation which is a new semi-supervised learning algorithm for link prediction on graphs based on the popularly-studied label propagation by exploiting information on similarities of links and nodes. Despite its efficiency and effectiveness compared to other methods, its applications were still limited due to the computational time and space constraints. In this paper, we propose fast and scalable algorithms for the Link Propagation by introducing efficient procedures to solve large linear equations that appear in the method. In particular, we show how to obtain a compact representation of the solution to the linear equations by using a non-trivial combination of techniques in linear algebra to construct algorithms that are also effective for link prediction on dynamic graphs. These enable us to apply the Link Propagation to large networks with more than 400,000 nodes. Experiments demonstrate that our approximation methods are scalable, fast, and their prediction qualities are comparably competitive.

## 1 Introduction

Many interactions in the real world can be expressed as networks that consist of a set of entities mapped to nodes and a set of links for the relationships between entities. Each entity may have additional information that influences its relationship with other entities. There are many natural examples of such networks, such as a network of webpages where the entities, relationships, and additional information are, respectively, the webpages, URL links, and the textual content of the pages. There are many other networks that exhibit similar properties; ranging from friendships and actions among people in Social Networking Service (SNS) to biological interactions among proteins. In many such networks the total



number of entities is large and the number of relationships per entity is often more than one.

Link mining is one of the most popular research topics that deals with extracting useful insights from such networks. It includes the *link prediction* problem, which is the problem of predicting the existence of unknown links between nodes using information from the known parts of the network. Link prediction has a broad range of applications. In marketing, users' ad-clicking actions might be predicted from the history of their actions and the relationships among their friends (or similar users) and the online ads. In social network analysis, users can be guided to contents of interest based on the friendship relations in their networks. Recent advances in storage technologies have made it possible to collect and store large amounts of information for link prediction. However, this also adds to the complexity of solving the link prediction problem.

Methods for the link prediction problem can be classified into two categories depending on the information used for prediction: link-information [13,14,19,28] and node-information [2,3,16]-based methods. Link-information-based methods such as matrix factorization [18] and link metrics [14] can predict the structures of networks from the observed parts of the networks. However, when only a small parts of the networks are known, the link-information-based methods work poorly. In contrast, node-information-based methods use node features as auxiliary information and can work well even in such cases.

Recently, a novel node-information-based link prediction method, which used a semi-supervised label propagation learning method to predict links (and hence, the method name *Link Propagation*) was proposed by the authors of [10]. Combined with information on node similarities, they devised a method for the Link Propagation by the conjugate gradient method and the *vec-tricks* techniques [24]. They also demonstrated that the qualities of the link prediction by the Link Propagation were competitive to those of other state-of-the-art methods. (See also Figure 1 in Section 3 for the comparison on the effectiveness of the exact Link Propagation by our method with existing state-of-the-art methods). However, like many other methods using node information, it has severe limitations in terms of computational time and space to handle networks with more than thousands of nodes, not to mention the difficulties in coping with link prediction on dynamic networks.

In this paper, we tackle these typical limitations of the node-information-based semi-supervised learning by proposing fast and scalable methods for the Link Propagation. Instead of using the conjugate gradient, our methods utilize the matrix factorization techniques, such as the Cholesky and eigen-decomposition. These enable us to exactly solve linear equations in the Link Propagation with less computational time and space. Moreover, the exact methods open the paths to define an approximate method for the Link Propagation that needs significantly less computational cost while maintaining accuracy. Our methods are also compatible with the *vec-tricks* techniques, which are useful to furtherly reduce the computational time. In addition, our methods can be applied for link prediction on dynamic networks. Recently, there is an emerging interest in the

study of dynamic network models (e.g., see [7]), and accordingly, there were many prior studies for link prediction and proximities on dynamic graphs, such as, in [9,21,22]. However, almost all of them are link-information based ones.

The contributions of this work are:

(1) We propose a new Link Propagation for developing a fast and scalable semi-supervised link prediction method based on the node information. The techniques used in the method also utilize matrix factorization and approximation. Thus, the new Link Propagation method can be considered as a kind of node-information-based link prediction that utilizes fast and scalable techniques in topological-based link prediction.

(2) We show that the new Link Propagation can also be used for efficient link prediction on large dynamic graphs whose edges evolve over time.

(3) We demonstrate experimental results on the effectiveness of the proposed method for link prediction in a large network with 400,000 nodes and more. We show that despite using much less computational time and space, the prediction qualities of the approximated version of the Link Propagation are competitive to (and sometimes are better than) those of the exact ones.

The rest of the paper is organized as follows. Section 2 defines the link prediction problem that we consider in this paper. Section 3 reviews the Link Propagation method first proposed in [10]. In Section 4, the procedures for an exact and approximate Link Propagation methods on static and dynamic (time-evolving) graphs are presented. In Section 5, several experimental results to demonstrate the scalability of the proposed methods and the accuracies of their approximations are shown. Section 6 summarizes the related work, and Section 7 concludes this paper with some discussion and promising future work.

## 2 Link Prediction Problem

The link prediction problem is usually described as a task to predict how likely a link exists between an arbitrary *pair* of nodes in a graph or network.

Let us denote two subsets of nodes of a network by  $X \equiv \{x_1, x_2, \dots, x_M\}$  and  $Y \equiv \{y_1, y_2, \dots, y_N\}$ . Some or all of  $X$  and  $Y$  may be identical depending on the applications. We denote the number of nodes by  $M \equiv |X|$  and  $N \equiv |Y|$ .

Our goal is to predict how likely a link exists for arbitrary node pair  $(x_i, y_j) \in X \times Y$ , which we refer to as the *link strength*. Namely, we want to output a matrix  $F$ , each of whose element  $[F]_{i,j}$  is the link strength between  $x_i$  and  $y_j$ . A large value of link strength indicates high confidence in the existence of the link, and a small value indicates high confidence in the absence of the link.

We define another  $M \times N$  matrix  $F^*$  to represent the observed parts of the links of the network. Each element of  $F^*$  is defined as

$$[F^*]_{i,j} \equiv \begin{cases} 1 & \text{if there exists a link between } (i, j), \\ 0 & \text{otherwise (link status is unknown for } (i, j)). \end{cases}$$

To estimate the link strength for the pairs whose link status is unknown (i.e. for the elements of  $\mathbf{F}^*$  filled with zeros),  $\mathbf{F}^*$  plays the role of the target values given in a training dataset in (positive-and-unlabelled) supervised learning.

As side information for link prediction, we assume that we are also given similarity matrices  $\mathbf{W}_X$  and  $\mathbf{W}_Y$  among the nodes in  $X$  and  $Y$ , respectively. Those matrices are non-negative and symmetric.

In summary, the input and output of the link prediction problem discussed in this paper is defined as follows.

**[Input]:** A matrix  $\mathbf{F}^*$  representing the known parts of the graph, and two symmetric non-negative matrices  $\mathbf{W}_X$  and  $\mathbf{W}_Y$  for two node sets  $X$  and  $Y$ .

**[Output]:** A matrix  $\mathbf{F}$  representing the link strength of all node pairs in  $X \times Y$ .

### 3 Review of the Link Propagation Method

In this section, we review the formulation of the Link Propagation introduced in [10]. The Link Propagation applies the idea of the *label propagation* methods [29,31] to link prediction. The label propagation methods are usually used for predicting the labels of unlabeled nodes by using the *label propagation principle*: “Two nodes that are similar to each other are likely to have the same label”. Modifying the label propagation principle, the Link Propagation states the analogous version of the inference principle as “Two node pairs that are similar to each other are likely to have the same link strength”.

Applying the label propagation principle to pairs of nodes, we obtain the following objective function to minimize (notice that  $\mathbf{vec}(\mathbf{F})$  is the vectorization operation of matrix  $\mathbf{F}$ )

$$J(\mathbf{F}) = \frac{\sigma}{2} \mathbf{vec}(\mathbf{F})^\top \mathbf{L} \mathbf{vec}(\mathbf{F}) + \frac{1}{2} \|\mathbf{vec}(\mathbf{F}) - \mathbf{vec}(\mathbf{F}^*)\|_2^2, \quad (1)$$

where the first term indicates that the two link strength values  $[\mathbf{F}]_{i,j}$  and  $[\mathbf{F}]_{\ell,m}$  for the two pairs should be close to each other if the similarity between the two pairs is large. The second term is the loss function that fits the predictions  $\mathbf{F}$  to their target values  $\mathbf{F}^*$  for the known parts of the network. It also plays a role as a regularization term to prevent the predictions from being too far from zero, and for numerical stability. The  $\sigma > 0$  is a regularization parameter which balances the two terms of Eq. (1). The  $MN \times MN$  matrix  $\mathbf{L}$  is a *Laplacian matrix*. For the Link Propagation, the previous work [10] recommended using the *Kronecker product Laplacian* defined as

$$\mathbf{L} \equiv \mathbf{D}_Y \otimes \mathbf{D}_X - \mathbf{W}_Y \otimes \mathbf{W}_X, \quad (2)$$

or the *Kronecker sum Laplacian* defined as

$$\mathbf{L} \equiv \mathbf{D}_Y \oplus \mathbf{D}_X - \mathbf{W}_Y \oplus \mathbf{W}_X = \mathbf{L}_Y \oplus \mathbf{L}_X, \quad (3)$$

where  $\mathbf{D}_X$  is a diagonal matrix whose diagonal elements are  $[\mathbf{D}_X]_{i,i} := \sum_j [\mathbf{W}_X]_{i,j}$ , and  $\mathbf{L}_X \equiv \mathbf{D}_X - \mathbf{W}_X$  is the Laplacian matrix for  $\mathbf{W}_X$ . The matrices  $\mathbf{D}_Y$  and  $\mathbf{L}_Y$

are defined similarly. Note that the  $\otimes$  operator indicates the Kronecker product and the  $\oplus$  operator indicates the Kronecker sum, whose definitions can be found in [12]. Here are the ideas behind the Kronecker product and the Kronecker sum Laplacians. Let us consider two pairs of nodes  $(x_i, y_j)$  and  $(x_\ell, y_m)$ . The Kronecker product Laplacian indicates that the two pairs are similar to each other if  $x_i$  and  $x_\ell$  are similar to each other as well as  $y_j$  and  $y_m$ . This is basically similar to the pair-wise similarity used in kernel methods [2,3,16]. In contrast, the Kronecker sum Laplacian indicates that the two pairs are similar to each other if  $x_i$  and  $x_\ell$  are identical and  $y_j$  and  $y_m$  are similar to each other, or  $x_i$  and  $x_\ell$  are similar to each other and  $y_j$  and  $y_m$  are identical.

Minimizing Eq. (1) with respect to  $\text{vec}(\mathbf{F})$ , we obtain the system of linear equations

$$(\sigma\mathbf{L} + \mathbf{I}) \text{vec}(\mathbf{F}) = \text{vec}(\mathbf{F}^*). \quad (4)$$

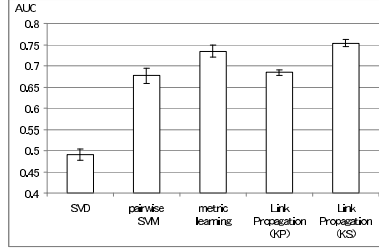
Using the elements of  $\mathbf{F}$  for link prediction, the accuracy of the Link Propagation is shown to be competitive with the other state-of-the-art node-information-based methods such as the pairwise SVM [2,3,16] and the metric learning [23]. Figure 1 shows a comparison of AUC values by several methods on medium-size graphs. They are, SVD as a link-based method (with rank=6), the pairwise SVM (with  $C=1$ ), metric learning (with rank=6), and the Link Propagation method with the Kronecker product (**KP**) similarity and the Kronecker sum (**KS**) similarity (with  $\sigma=0.01$ ) on predicting metabolic networks [26]. The similarity matrices (or kernel matrices) were constructed by taking average of the three given matrices named ‘phylogenetic’, ‘expression’, and ‘localization’, and the Laplacian (or kernel matrices) were normalized. The hyperparameters were tuned appropriately. The AUC values were evaluated by 5-fold cross validation with 10% training data. We can see from the figure that the link-based method (SVD) performs poorly because the training data is sparse, and the Link Propagation (especially KS) achieves the best performance.

The authors of [10] proposed a conjugate gradient-based method for solving Eq. (4). However one unavoidable drawback of the conjugate gradient-based method is its large memory requirements for storing the matrices such as  $W_X$ ,  $F^*$ , and  $\mathbf{F}$  due to the nature of Kronecker operators. The matrix  $\mathbf{F}$  is inherently dense even when the others are sparse, and in many cases, only parts of its elements are needed for the link prediction (e.g., on some small subset of users and ads). In the hereafter, we will develop exact and approximate solutions for the Link Propagation to overcome the limitation and furtherly widen its applications on large-size and dynamic graphs.

## 4 Fast and Scalable Solutions for the Link Propagation

In this section, we propose novel methods for the Link Propagation that require much less memory than those based on the conjugate gradient method. We first show an exact method and then, extending it, present an approximate method for the Link Propagation.

<sup>1</sup> <http://web.kuicr.kyoto-u.ac.jp/supp/yoshi/ismb05/>



**Fig. 1.** The accuracies of the Link Propagation methods (KS & KP) vs. those of other methods

As a matter of convenience in deriving the solution, we use the normalized versions of the Laplacian matrices in Eqs. (2) and (3). The normalized Kronecker product Laplacian matrix is given as

$$\begin{aligned} L &\equiv I - (D_Y \otimes D_X)^{-\frac{1}{2}} (W_Y \otimes W_X) (D_Y \otimes D_X)^{-\frac{1}{2}} \\ &= I - \left( D_Y^{-\frac{1}{2}} W_Y D_Y^{-\frac{1}{2}} \right) \otimes \left( D_X^{-\frac{1}{2}} W_X D_X^{-\frac{1}{2}} \right), \end{aligned} \tag{5}$$

while the normalized Kronecker sum Laplacian matrix is given as

$$\begin{aligned} L &\equiv \left( I - \left( I - D_Y^{-\frac{1}{2}} W_Y D_Y^{-\frac{1}{2}} \right) \oplus \left( I - D_X^{-\frac{1}{2}} W_X D_X^{-\frac{1}{2}} \right) \right) \\ &= 3I - \left( D_Y^{-\frac{1}{2}} W_Y D_Y^{-\frac{1}{2}} \right) \oplus \left( D_X^{-\frac{1}{2}} W_X D_X^{-\frac{1}{2}} \right). \end{aligned} \tag{6}$$

Both of the normalized Laplacians in Eqs. (5) and (6) can be written unifiedly in a general form as

$$L \equiv cI - \left( D_Y^{-\frac{1}{2}} W_Y D_Y^{-\frac{1}{2}} \right) \circledast \left( D_X^{-\frac{1}{2}} W_X D_X^{-\frac{1}{2}} \right).$$

Notice that the operator  $\circledast \in \{\otimes, \oplus\}$  corresponds to the Kronecker product  $\otimes$ , or, the Kronecker sum  $\oplus$  operator. In this paper, we can consider either of them by appropriately setting the value of  $c$  (which is 1 for the Kronecker product, and 3 for the Kronecker sum).

For simplicity, let us write:

$$\tilde{W}_X \equiv D_X^{-\frac{1}{2}} W_X D_X^{-\frac{1}{2}}, \quad \tilde{W}_Y \equiv D_Y^{-\frac{1}{2}} W_Y D_Y^{-\frac{1}{2}}, \tag{7}$$

to obtain the following solution of Eq. (4):

$$\mathbf{vec}(F) = \left( (1 + c\sigma)I - \sigma \tilde{W}_Y \circledast \tilde{W}_X \right)^{-1} \mathbf{vec}(F^*). \tag{8}$$

### 4.1 An Exact Link Propagation

Here we explain an algorithm to compute the exact solution of Eq. (8). The algorithm is the foundation to develop a scalable and faster Link Propagation.

One of the key techniques in this paper is a direct method to efficiently compute the inverse matrix on the right-hand side of Eq. (8) that involves Kronecker operators. In what follows, we will show how to obtain the exact and approximate inverse by exploiting the following well-known theorem (see. e.g., [12]) on the eigenvalues and eigenvectors of the Kronecker sum and product.

**Theorem 1.** *Let the eigenvalues of the matrices  $\tilde{W}_X$  and  $\tilde{W}_Y$  be, respectively,  $\{\lambda_X^{(i)}\}_{i=1}^M$  and  $\{\lambda_Y^{(j)}\}_{j=1}^N$ , with the corresponding eigenvectors are given as, respectively, the column of the matrices  $V_X$  and  $V_Y$ . Then the eigenvectors of the Kronecker product  $\tilde{W}_Y \otimes \tilde{W}_X$  and the Kronecker sum  $\tilde{W}_Y \oplus \tilde{W}_X$  are the same (column vectors of)  $V_Y \otimes V_X$ , while the eigenvalues are  $\{\lambda_X^{(i)} \lambda_Y^{(j)}\}_{(i,j)=(1,1)}^{M,N}$  for the Kronecker product, and  $\{\lambda_X^{(i)} + \lambda_Y^{(j)}\}$  for the Kronecker sum.*

Since the similarity matrices are positive semidefinite matrices, they can be eigendecomposed as follows (those of  $\tilde{W}_Y$  are omitted):

$$\tilde{W}_X = V_X \text{diag} \left( \lambda_X^{(1)}, \lambda_X^{(2)}, \dots, \lambda_X^{(M)} \right) V_X^\top, \quad (9)$$

Now let us define the matrix  $\Lambda$  to be either

$$[\Lambda]_{i,j} \equiv \lambda_X^{(i)} \lambda_Y^{(j)}, \text{ or } [\Lambda]_{i,j} \equiv \lambda_X^{(i)} + \lambda_Y^{(j)}, \quad (10)$$

where the former is for the Kronecker product and the latter is for the Kronecker sum. Then by Theorem 1 we can write the inverse matrix in Eq. (8) as

$$\left( (1 + c\sigma) I - \sigma \tilde{W}_Y \otimes \tilde{W}_X \right)^{-1} = \left( (1 + c\sigma) I - \sigma V \text{diag}(\text{vec}(\Lambda)) V^\top \right)^{-1}, \quad (11)$$

where  $V = V_Y \otimes V_X$ . Since, it holds that  $V V^\top = I$ , Eq. (11) can be transformed into

$$\left( (1 + c\sigma) I - \sigma V \text{diag}(\text{vec}(\Lambda)) V^\top \right)^{-1} = V \left( (1 + c\sigma) I - \sigma \text{diag}(\text{vec}(\Lambda)) \right)^{-1} V^\top.$$

Notice that  $((1 + c\sigma) I - \sigma \text{diag}(\text{vec}(\Lambda)))$  is a diagonal matrix whose inverse can be easily calculated as the following matrix  $D$

$$[D]_{i,j} \equiv (1 + \sigma(c - [\Lambda]_{i,j}))^{-1}. \quad (12)$$

This gives us the the solution of Eq. (8) as  $\text{vec}(F) = V \text{diag}(\text{vec}(D)) V^\top \text{vec}(F^*)$ . We can further simplify this equation as

$$\text{vec}(F) = V \text{diag}(\text{vec}(D)) \text{vec}(V_X^\top F^* V_Y) = \text{vec}(V_X (D * (V_X^\top F^* V_Y)) V_Y^\top),$$

where in the derivation we used the “vec-tricks” techniques [12,24] as

$$(V_Y \otimes V_X) \text{vec}(F^*) = \text{vec}(V_X F^* V_Y^\top), \quad (13)$$

and  $D * A$  is the Hadamard product of matrices  $D$  and  $A$ . Taking out the  $\text{vec}$  operation, we can describe the solution as

$$F = V_X (D * (V_X^\top F^* V_Y)) V_Y^\top. \quad (14)$$

---

**Algorithm 1.** Exact Link Propagation. Input:  $(W_X, W_Y, F^*, \sigma)$ .

---

- 1: Compute the normalized similarity matrices  $\tilde{W}_X$  and  $\tilde{W}_Y$  as Eq. (7)
  - 2: Compute the eigendecomposition of  $\tilde{W}_X$  and  $\tilde{W}_Y$  as in Eq. (9)
  - 3: Compute the elements of the matrix  $D$  as in Eq. (12), where Eq. (10) is used for the Kronecker product or sum
  - 4: Compute the solution from Eq. (14)
- 

The complete procedure to compute the exact solution matrix  $F$  is summarized in Algorithm 1.

Although computing the exact solution of the Link Propagation with Algorithm 1 requires intensive matrix operations, in the *worst-case* analysis it is more efficient than the one with the conjugate gradient in [10]. Algorithm 1 requires only  $O(M^3 + N^3 + M^2N + N^2M + MN|F^*|)$  computational time. In detail, Step 1 needs  $O(M^2 + N^2)$  time, while Step 2 and Step 3 require  $O(M^3 + N^3)$  and  $O(MN)$  time, respectively. Computing all elements of  $F$  in Step 4 takes  $O(M^2N + MN^2 + MN|F^*|)$  time, which means approximately  $N$  or  $M$  operations is required for computing each element of  $F$ . When  $M = N$ , Algorithm 1 only requires  $O(N^3)$  time in contrast to  $O(N^5)$  time of the conjugate gradient method. The space complexity is  $O(N^2)$ , which is the same with the conjugate gradient method<sup>2</sup>. However, the exact solution in Algorithm 1 is potentially more useful because it opens a path to apply rich techniques of linear algebra, such as matrix approximation and factorization, so that the time and space complexities can be reduced while the accuracy levels are maintained. Moreover, it also enables us to design a fast incremental method for the Link Propagation in dynamic graphs without recomputing from scratch. We will briefly explain the techniques in the following sections.

## 4.2 An Approximate Link Propagation

Here we present an approximate solution of the Link Propagation that mitigates the large computational time and space complexities of the exact solution.

For nodes on the order of millions, not only  $F$ , but also storing the similarity matrices in main memory is already prohibitive. Therefore, we need to consider more scalable and faster ways to compute the approximations of the Link Propagation. For this purpose, we can use any matrix approximation technique whose computation process does not require storing all elements of the matrix in the main memory. One of the approximation techniques used in this paper is the *incomplete Cholesky decomposition*. However, note that other matrix approximation techniques, such as those based on singular value decomposition, could also be used. Thus, we can obtain the low-rank approximation of the similarity matrices  $W_X$  and  $W_Y$  as

$$W_X \approx G_X G_X^\top, \quad W_Y \approx G_Y G_Y^\top. \quad (15)$$

---

<sup>2</sup> Without the vec-tricks, the time and space complexities of the conjugate gradient are  $O(N^6)$  and  $O(N^4)$ , respectively.

Here,  $\mathbf{G}_X$  and  $\mathbf{G}_Y$  are  $M \times \bar{M}$  and  $N \times \bar{N}$  matrices, respectively, where  $\bar{M}$  and  $\bar{N}$  are the parameters for the approximation ranks whose values can be set appropriately by the users. The total computational cost of obtaining such matrices by the incomplete Cholesky decomposition is  $O(M\bar{M}^2 + N\bar{N}^2)$ . Moreover, the sum of each row of the approximate matrices can be computed from  $\mathbf{G}_X$  and  $\mathbf{G}_Y$  by  $\mathbf{D}_X = \text{diag}(\mathbf{G}_X \mathbf{G}_X^\top \mathbf{1})$  and  $\mathbf{D}_Y = \text{diag}(\mathbf{G}_Y \mathbf{G}_Y^\top \mathbf{1})$ . Thus, by defining the following matrices:

$$\tilde{\mathbf{G}}_X \equiv \mathbf{D}_X^{-\frac{1}{2}} \mathbf{G}_X, \quad \tilde{\mathbf{G}}_Y \equiv \mathbf{D}_Y^{-\frac{1}{2}} \mathbf{G}_Y, \quad (16)$$

we can write the normalized similarity matrices as

$$\tilde{\mathbf{W}}_X \approx \mathbf{D}_X^{-\frac{1}{2}} \mathbf{G}_X \mathbf{G}_X^\top \mathbf{D}_X^{-\frac{1}{2}} = \tilde{\mathbf{G}}_X \tilde{\mathbf{G}}_X^\top, \quad \tilde{\mathbf{W}}_Y \approx \mathbf{D}_Y^{-\frac{1}{2}} \mathbf{G}_Y \mathbf{G}_Y^\top \mathbf{D}_Y^{-\frac{1}{2}} = \tilde{\mathbf{G}}_Y \tilde{\mathbf{G}}_Y^\top.$$

Notice that since  $\bar{M} \ll M$  and  $\bar{N} \ll N$ , the eigendecomposition of  $\tilde{\mathbf{G}}_X^\top \tilde{\mathbf{G}}_X$  and  $\tilde{\mathbf{G}}_Y^\top \tilde{\mathbf{G}}_Y$  can be performed easily to obtain the following eigenvalues (which are the same as those of the approximate similarity matrices) and eigenvectors satisfying (those of  $\tilde{\mathbf{G}}_Y^\top \tilde{\mathbf{G}}_Y$  are omitted)

$$\tilde{\mathbf{G}}_X^\top \tilde{\mathbf{G}}_X = \bar{\mathbf{U}}_X \text{diag}(\bar{\lambda}_X^{(1)}, \bar{\lambda}_X^{(2)}, \dots, \bar{\lambda}_X^{(\bar{M})}) \bar{\mathbf{U}}_X^\top. \quad (17)$$

Now, the eigenvectors of the approximate similarity matrix  $\tilde{\mathbf{W}}_X$  (and similarly for  $\tilde{\mathbf{W}}_Y$ ) can be computed from the equation

$$\bar{\mathbf{V}}_X \equiv \tilde{\mathbf{G}}_X \bar{\mathbf{U}}_X \text{diag}(\bar{\lambda}_X^{(1)}, \bar{\lambda}_X^{(2)}, \dots, \bar{\lambda}_X^{(\bar{M})})^{-\frac{1}{2}}. \quad (18)$$

Similar to the exact case, by Eqs. (15) and (18), letting  $\bar{\mathbf{V}} = \bar{\mathbf{V}}_Y \otimes \bar{\mathbf{V}}_X$ , we can write the inverse matrix in Eq. (8) as

$$\left( (1 + c\sigma) \mathbf{I} - \sigma \tilde{\mathbf{W}}_Y \otimes \tilde{\mathbf{W}}_X \right)^{-1} = \left( (1 + c\sigma) \mathbf{I} - \sigma \bar{\mathbf{V}} \text{diag}(\text{vec}(\bar{\Lambda})) \bar{\mathbf{V}}^\top \right)^{-1}. \quad (19)$$

Notice that the elements of the matrix  $\bar{\Lambda}$  are appropriately defined as in Eq. (10) for the Kronecker product or sum as in the exact case. However, differing from the exact case, we have  $(\bar{\mathbf{V}}_Y \otimes \bar{\mathbf{V}}_X) (\bar{\mathbf{V}}_Y \otimes \bar{\mathbf{V}}_X)^\top \neq \mathbf{I}$ . Fortunately, by using the Woodbury equation (see, e.g., [4]) and  $\bar{\mathbf{V}}^\top \bar{\mathbf{V}} = \mathbf{I}$ , we can compute the inverse matrix in Eq. (19), as follows.

$$\begin{aligned} & \left( (1 + c\sigma) \mathbf{I} - \sigma \tilde{\mathbf{W}}_Y \otimes \tilde{\mathbf{W}}_X \right)^{-1} \\ &= \frac{\mathbf{I}}{1 + c\sigma} + \frac{\bar{\mathbf{V}}}{(1 + c\sigma)^2} \left( \frac{\text{diag}(\text{vec}(\bar{\Lambda}))^{-1}}{\sigma} - \frac{\mathbf{I}}{1 + c\sigma} \right)^{-1} \bar{\mathbf{V}}^\top. \end{aligned}$$

Defining the matrix  $\bar{\mathbf{D}}$  as

$$[\bar{\mathbf{D}}]_{i,j} \equiv \left( \frac{1}{\sigma[\bar{\Lambda}]_{i,j}} - \frac{1}{1 + c\sigma} \right)^{-1} = \frac{\sigma(1 + c\sigma)[\bar{\Lambda}]_{i,j}}{1 + c\sigma - \sigma[\bar{\Lambda}]_{i,j}}, \quad (20)$$



---

**Algorithm 2.** Approximate Link Propagation. Input:  $(W_X, W_Y, F^*, \sigma)$ .

---

- 1: Compute the low-rank approximation matrices of  $W_X$  and  $W_Y$  as in Eq. (15)
  - 2: Compute the normalized matrices,  $\tilde{G}_X$  and  $\tilde{G}_Y$ , as in Eq. (16)
  - 3: Compute the eigendecomposition of  $\tilde{G}_X^\top \tilde{G}_X$  and  $\tilde{G}_Y^\top \tilde{G}_Y$  as in Eq. (17)
  - 4: Compute the eigenvectors of  $\tilde{G}_X \tilde{G}_X^\top$  and  $\tilde{G}_Y \tilde{G}_Y^\top$  according to Eq. (18)
  - 5: Compute the elements of the matrix  $\bar{D}$  according to Eq. (20), where the values of elements of  $\bar{\Lambda}$  are adjusted according to that in the Kronecker sum or product
  - 6: Using the core solution in Eq. (22), compute the elements of  $F$  according to Eq. (21)
- 

we can compute the approximate solution of the Link Propagation as

$$\mathbf{vec}(F) = \frac{1}{1 + c\sigma} \mathbf{vec}(F^*) + \frac{1}{(1 + c\sigma)^2} \bar{V} \mathbf{diag}(\mathbf{vec}(\bar{D})) \bar{V}^\top \mathbf{vec}(F^*).$$

By using the *vec*-trick techniques and taking-out the *vec* operation, the solution matrix  $F$  is efficiently obtained as

$$F = \frac{1}{1 + c\sigma} F^* + \frac{1}{(1 + c\sigma)^2} \bar{V}_X (\bar{D} * (\bar{V}_X^\top F^* \bar{V}_Y)) \bar{V}_Y^\top. \quad (21)$$

Since  $F$  is a large matrix, storing all of its elements is prohibitively expensive. Instead, we can store its compact representation by computing and storing the smaller *core matrix*

$$\bar{D} * (\bar{V}_X^\top F^* \bar{V}_Y), \quad (22)$$

along with the eigenvectors of the approximate similarity matrices. This information is sufficient to compute the elements of  $F$  on demand. We summarize the procedure to compute the approximate solution of the Link Propagation in Algorithm 2. Two major advantages of Algorithm 2 are its using much less computational space and time, and its capability to compactly represent the elements of  $F$  by storing the core matrix and the eigenvectors of the low-rank similarity matrices whose sizes are mostly linear in the number of nodes (we will show later in the experiments that low-rank approximation matrices are sufficient). That is, its space complexity is only  $O(\bar{M}\bar{N} + \bar{M}\bar{M} + \bar{N}\bar{N})$ , where the first term is due to the core matrix, and the last two terms are due to the eigenvectors. With regards to its computational complexity, we can see that up to Step 5, Algorithm 2 requires  $O(M\bar{M}^2 + N\bar{N}^2 + \bar{M}^3 + \bar{N}^3 + |F^*|\bar{M}\bar{N})$  time: Step 1 and Step 4 need  $O(M\bar{M}^2 + N\bar{N}^2)$  time, Step 2 requires  $O(M\bar{M} + N\bar{N})$  time, and Step 3 takes  $O(M\bar{M}^2 + N\bar{N}^2 + \bar{M}^3 + \bar{N}^3)$  time, while the computation of the core matrix in Step 5 and Step 6 need  $O(\bar{M}\bar{N}|F^*|)$  time. To compute all elements of  $F$  in Step 6, we need  $O(\bar{M}\bar{N}M + \bar{N}MN)$  time, which implies constant computation time per element for small  $\bar{M}$  and  $\bar{N}$ . Indeed, in the experiments we could choose some small values for  $\bar{M}$  and  $\bar{N}$  to obtain satisfactory link prediction results.

### 4.3 Link Prediction on Dynamic Graphs

Here we show another desirable aspect of the exact and approximated solutions of the Link Propagation that enables an efficient adjustment of the link prediction scores to cope with addition and deletion of edges in the networks.

Changes in graphs (which result in the change of values of  $F^*$ ) occur quite frequently, and hence timely updating the link prediction scores is crucial. For example, consider the graph whose node set  $X$  is the subset of users and node set  $Y$  is the subset of ads, where the similarity matrices are computed from the users' profiles, tags, etc. In this setting, the edges represent the users' clicking actions. New clicks can be regarded as addition, while non-clicking actions can be regarded as subtraction of the corresponding elements in  $F^*$ . Changes in the link prediction scores due to the addition and subtraction of elements of  $F^*$  can be computed straightforwardly in our methods.

Let us denote the parts of  $F^*$  that changed as  $\Delta F^*$ , namely, the new matrix is  $\tilde{F}^* = F^* + \Delta F^*$ . It is clear from Eqs. (21) and (14), that the incremental updates of the prediction scores for all elements of  $F$  can be performed by updating the smaller core matrix in Eq. (22) (also Eq. (14)) as

$$\bar{D} * (\bar{V}_X^\top \Delta F^* \bar{V}_Y), \quad (23)$$

that implies the time complexity of the incremental update of the core matrix is  $O(\bar{M}\bar{N}|\Delta F^*|)$ , which is proportional to the number of changes in  $F^*$ . Notice that apart from the approximated similarity matrices, the incremental updates in our methods are exact, i.e., their accuracies are the same with those of the compute-from-scratch methods. The limitation of the incremental update presented here is that it requires the same set of nodes and similarity matrices.

## 5 Experiments

In this section, we present some experimental results for predicting the links between pair of nodes (*pair-wise link prediction*) using exact and approximate solutions of the Link Propagation. The purpose of the experiments is to show that the approximated Link Propagation performs quite well inspite of requiring significantly less time and space than the exact one. We first describe the datasets of the experiments.

### 5.1 Datasets

We tested the exact and approximate Link Propagation on two types of network datasets available from the Web Spam Challenge archive<sup>3</sup>. The summary of the datasets is listed in Tables 1 and 2. The Link Propagation was performed when  $X = Y$ .

The first type of datasets were created from the network of 400,000 web pages in a webgraph (called *Web400K* dataset) whose links correspond to the hyperlinks of webpages. For each webpage, there is a sparse feature vector that contains the frequency of words in the web page. The  $(x, y)$  element of the similarity matrix  $W$  in the experiment is the inner product of the feature vectors of the corresponding webpages  $x$  and  $y$ . From this dataset, we created two datasets each of which contains exactly 1,000 and 3,000 nodes of Web400K (denoted by

<sup>3</sup> <http://webspam.lip6.fr/wiki/pmwiki.php?n=Main.PhaseIITrainingCorpora>

**Table 1.** Datasets from the web graph

	Web1K	Web3K	Web400K
#Nodes	1,000	3,000	400,000
#Links	20,214	38,453	13,068,666

**Table 2.** Datasets from the host graph

	Host1K	Host3K	Host9K
#Nodes	1,000	3,000	9,000
#Links	6,223	156,411	505,809

*Web1K*, and *Web3K*, respectively) that are reachable and closest to the node with id "1" (That is, starting from node 1, we add its neighbors, by prioritizing the node with lower id, to be the members of the dataset and repeat this process until we find exactly  $k \in \{1000, 3000\}$  nodes. If at any time we cannot add a node because the network is disconnected, we add the node with the lowest id that is not in the current dataset, and repeat). We avoid extracting random subgraphs for ease of reproducibility of the results.

The second type of datasets were created from the network of 9,000 hosts in a hostgraph (called *Host9K* dataset) whose links correspond to the existences of linked pages between pages in the hosts. For each host, there is a sparse feature vector that represents a normalized tf-idf vector over the content of its pages. The feature vectors were used to build the similarity matrix similarly as in the webgraph. In our experiments, we again created two small graphs from the hostgraph that consist of, respectively, simply the first 1,000 and 3,000 nodes in the hostgraph. We call the datasets *Host1K* and *Host3K*, respectively. We evaluated the effectiveness and efficiency of the exact and approximate Link Propagation by measuring their time complexities and accuracies. Time complexity means the amount of time for an algorithm to be ready to output a value of prediction and not the amount of time to compute all elements of  $F$  (whose size is very large). The accuracies were evaluated by AUC, area under the ROC curve, which is commonly used in supervised learning. We randomly selected  $\lambda = 10\%$ ,  $75\%$  of the links in the datasets for at least three times for each dataset, and used them as training data, and evaluated averaged AUC values on all other pair of nodes. The standard deviations of AUC values in our experiments were small.

## 5.2 Implementation

In all experiments, we set  $\sigma = 0.001$ , used the Laplacian Sum for all graphs because of the page limit. We evaluated the AUC values and measured the computational time for ten times for datasets other than the Host3K and Web3K graphs (which are just repeated for three times because of the time limitation). All algorithms were implemented in 64-bit Java using Colt<sup>©</sup> packages matrix operations. They were tested on several (virtual) machines running Linux and Microsoft<sup>®</sup> Windows XP<sup>®</sup>. The computation times were measured on instances executed on an IBM IntelliStation<sup>®</sup> running 64-bit Red Hat Enterprise Linux 5 with an Intel<sup>®</sup> Core 2 Quad CPU Q6600@2.40-GHz CPU and 8-GB RAM.

**Table 3.** The AUC scores of link prediction on the webgraph by 10% sampling**Table 4.** The AUC scores of link prediction on the webgraph by 75% sampling

Dataset	Exact	$\bar{M} = 20$	$\bar{M} = 50$	$\bar{M} = 100$	Dataset	Exact	$\bar{M} = 20$	$\bar{M} = 50$	$\bar{M} = 100$
Web1K	0.892	0.854	0.930	<b>0.944</b>	Web1K	0.926	0.858	0.934	<b>0.954</b>
Web3K	0.914	0.769	0.921	<b>0.956</b>	Web3K	0.934	0.771	0.925	<b>0.960</b>
Web400K	NA	0.875	0.876	<b>0.894</b>	Web400K	NA	0.893	0.888	<b>0.908</b>

### 5.3 Experimental Results on Static Graphs

Here we present the accuracies and the computational time of the exact and approximated Link Propagation method for the link prediction on the webgraphs and hostgraphs when some percentage of links are given as training samples.

Tables 3 and 4 show the AUC values by the exact and approximate Link Propagation on webgraphs, when, respectively, 10% and 75% of links were used for training. The tables show a surprising result that at low values of the approximation rank  $\bar{M}$ , the approximate method can sometimes produce better AUC values than the exact method that at  $\bar{M} = 50$  the AUC values of the approximate Link Propagation were better on the Web1K and Web3K graphs. The AUC value of the exact method on the Web400K graph is not available within our limited resources (denoted by *NA* in the tables), but the AUC values of the approximate method were sufficiently high.

Tables 5 and 6 summarize the AUC values on the hostgraphs similarly as in the previous tables. From the tables, we can see that the AUC values of the approximate method increased as the the values of  $\bar{M}$  and the number of sample links increased. However, the AUC values were less than the corresponding AUC values of the exact method (For the same reason, we could not compute the AUC value of the exact method on the Host9K). Unlike what we have observed from the webgraphs, they are never better than those of the exact ones. This might reflect the properties of similarity matrices: the feature vectors of nodes in the webgraphs correspond to pages and tend to be of low rank, while the feature vectors of hosts in the hostgraphs correspond to accumulation of feature vectors of various pages and thus, tend to be of high rank and hard to approximate. The comparison of the computation time for the exact and approximate methods for each graph in the datasets is shown in Figure 2. The approximate method is efficient enough that we could compute the link prediction over the Web400K and Host9K graphs within reasonable time. For example, from the figure we can see that the link prediction on the Web400K graph, whose size is more than one hundred times of that of the Web3K graph can be performed within almost half of the time spent on the Web3K graph to perform the exact Link Propagation. On the Host3K graph, the approximation method can be 50 times faster than the exact one while retaining the accuracy level.

### 5.4 Experimental Results on Dynamic Graphs

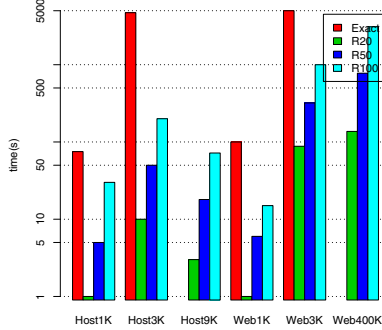
Here we present experimental results for another contribution of our method for the link prediction on dynamic graphs as summarized in Figure 3. We simulated

**Table 5.** The AUC scores of link prediction on the hostgraph by 10% sampling

Dataset	Exact	$\bar{M} = 20$	$\bar{M} = 50$	$\bar{M} = 100$
Host1K	<b>0.829</b>	0.644	0.685	0.711
Host3K	<b>0.910</b>	0.610	0.655	0.686
Host9K	NA	0.629	0.560	<b>0.638</b>

**Table 6.** The AUC scores of link prediction on the hostgraph by 75% sampling

Dataset	Exact	$\bar{M} = 20$	$\bar{M} = 50$	$\bar{M} = 100$
Host1K	<b>0.876</b>	0.649	0.703	0.741
Host3K	<b>0.938</b>	0.611	0.660	0.694
Host9K	NA	0.618	0.570	<b>0.646</b>



**Fig. 2.** Comparison of computational time by the exact method, rank-20 approximation (R20), rank-50 approximation (R50) and rank-100 approximation (R100)

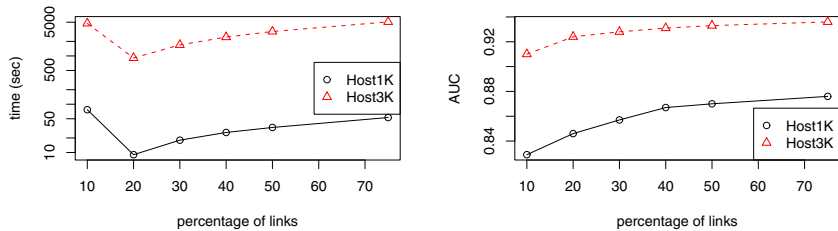
the changes of edges by first performing the Link Propagation with 10% of links of the corresponding graph and then incrementally adding 10%, 20%, 30%, 40%, and 65% of the rest of the links at a time. Notice that the effect of deletion of edges is similar as implied by Eq. (23) and therefore omitted.

The left part of Figure 3 shows the computational time with regards to the number of links used in the training on the Host1K and Host3K graphs. The value at 10% links denotes the computational time used for computing the core matrix, the most dominant part, from scratch, while the rest of the values denote those from incrementally updating the core matrix computed from scratch with 10% of links until sufficient portion of links are added. From the figure we can see that on a large-size graph like the Host3K, incrementally updating the core matrix due to addition (deletion) of 10% links is about 10 times faster than computing it from scratch.

The right part of Figure 3 shows the corresponding AUCs where we can notice that the more samples are available, the higher the prediction qualities are. Thus, even for the exact Link Propagation, we only need computing from scratch once (mainly for reading and decomposing the similarity matrices) and then update the core matrix as needed when edges are added or deleted.

## 6 Related Work

The link prediction problem has been thoroughly studied in the context of predicting biological networks such as protein-protein interaction networks and gene



**Fig. 3.** Computational time (left) and AUCs (right) for the link prediction with incremental addition of links. The initial link prediction was computed with 10% links of the corresponding graph, and the rest was performed with incremental updates.

regulatory networks in the bioinformatics area, and also in the context of link mining [6] in the data mining community.

In bioinformatics, several node-information-based approaches have been proposed, such as an EM-based [11] and dimension-reduction-based approaches [23, 26]. The pair-wise kernel with which we compared our method in Figure 1 was proposed for predicting protein-protein interactions [3]. Interestingly, the same kernel was independently proposed for entity resolution [16] and collaborative filtering [2]. In the data mining community, the link prediction problem is being studied as one of the fundamental tasks for the link mining. There are several methods that utilize structural information only such as link metrics (e.g. [14]). Matrix factorization approaches [13, 19] are also grouped into topological-information-based methods. There are also supervised learning methods using node information as well as topological information such as [8, 15]. Several previous works, e.g. [17, 20], that apply the framework of statistical relational learning to link prediction also exist. A similar framework using the so-called exponential random graph model is used in social network analysis [1]. Recently, sophisticated generative models of networks from a Bayesian perspective have been proposed [5, 27].

The matrix “vec-trick” in Eq. (13) was first proposed by Vishwanathan et al. [24] for accelerating the computation of the graph kernels. The label propagation technique on graphs was also used in [30], where a batch and incremental method for finding a good low-dimensional latent-space embedding of documents utilizing side information from multiple graphs was proposed. The use of Woodbury equation for handling dynamic bipartite graphs can also be found in [22] (and its conference version). Notice that unlike those in [22], our methods can be used beyond pair-wise link prediction.

## 7 Discussion and Concluding Remarks

We show fast and scalable semi-supervised learning algorithms for link prediction on static and dynamic graphs using both link information and node information by devising novel methods for exact and approximate Link Propagation. The applications of our methods to large networks with more than 400,000 nodes

demonstrated that our methods are scalable and their approximation are quite good. The proposed algorithms avoid the use of the conjugate gradient method and directly solve the huge linear equations by (approximating) the matrix inverse, which also allow us to perform efficient leave-one-out estimation [25] for determining the hyper-parameters in the Link Propagation. This will be investigated in the future. Finally, we should also note that although the methods presented in this paper were described for pair-wise link prediction, they can be extended for triplets, quadrlets, etc. This can be done by using higher order tensors. We omit the details due to the space limitation.

## References

1. Anderson, C.J., Wasserman, S., Crouch, B.: A  $p^*$  primer: logit models for social networks. *Social Networks* 21, 37–66 (1999)
2. Basilico, J., Hofmann, T.: Unifying collaborative and content-based filtering. In: *ICML* (2004)
3. Ben-Hur, A., Noble, W.S.: Kernel methods for predicting protein-protein interactions. *Bioinformatics* 21(suppl. 1), i38–i46 (2005)
4. Bishop, C.: *Pattern Recognition and Machine Learning*. Springer, Heidelberg (2006)
5. Chu, W., Sindhvani, V., Ghahramani, Z., Keerthi, S.: Relational learning with Gaussian processes. In: *NIPS* (2007)
6. Getoor, L., Diehl, C.P.: Link mining: a survey. *SIGKDD Explorations* 7(2), 3–12 (2005)
7. Hanneke, S., Xing, E.: Discrete temporal models of social networks. In: Airoidi, E.M., Blei, D.M., Fienberg, S.E., Goldenberg, A., Xing, E.P., Zheng, A.X. (eds.) *ICML 2006*. LNCS, vol. 4503, pp. 115–125. Springer, Heidelberg (2007)
8. Hasan, M.A., Chaoji, V., Salem, S., Zaki, M.: Link prediction using supervised learning. In: *LinkKDD* (2005)
9. Hayashi, K., Hirayama, J., Ishii, S.: Dynamic Exponential Family Matrix Factorization. In: Theeramunkong, T., Kijssirikul, B., Cercone, N., Ho, T.-B. (eds.) *PAKDD 2009*. LNCS, vol. 5476, pp. 452–462. Springer, Heidelberg (2009)
10. Kashima, H., Kato, T., Yamanishi, Y., Sugiyama, M., Tsuda, K.: Link propagation: A fast semi-supervised learning algorithm for link prediction. In: *SDM* (2009)
11. Kato, T., Tsuda, K., Asai, K.: Selective integration of multiple biological data for supervised network inference. *Bioinformatics* 21(10), 2488–2495 (2005)
12. Laub, A.J.: *Matrix Analysis for Scientists and Engineers*. Society for Industrial and Applied Mathematics (2005)
13. Lee, D., Seung, H.: Algorithms for non-negative matrix factorization. In: *NIPS*, pp. 556–562 (2001)
14. Liben-Nowell, D., Kleinberg, J.: The link prediction problem for social networks. In: *CIKM*, pp. 556–559 (2004)
15. O’Madadhain, J., Hutchins, J., Smyth, P.: Prediction and ranking algorithms for event-based network data. *SIGKDD Explorations* 7(2), 23–30 (2005)
16. Oyama, S., Manning, C.D.: Using feature conjunctions across examples for learning pairwise classifiers. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) *ECML 2004*. LNCS (LNAI), vol. 3201, pp. 322–333. Springer, Heidelberg (2004)

17. Popescul, A., Ungar, L.H.: Statistical relational learning for link prediction. In: IJCAI Workshop on Learning Statistical Models from Relational Data (2003)
18. Srebro, N.: Learning with Matrix Factorization. PhD thesis, Massachusetts Institute of Technology (2004)
19. Srebro, N., Rennie, J., Jaakkola, T.: Maximum-margin matrix factorization. In: NIPS, pp. 1329–1336 (2005)
20. Taskar, B., Wong, M., Abbeel, P., Koller, D.: Link prediction in relational data. In: NIPS (2004)
21. Tong, H., Papadimitriou, S., Sun, J., Yu, P.S., Faloutsos, C.: Colibri: fast mining of large static and dynamic graphs. In: KDD, pp. 686–694 (2008)
22. Tong, H., Papadimitriou, S., Yu, P.S., Faloutsos, C.: Fast monitoring proximity and centrality on time-evolving bipartite graphs. *Statistical Analysis and Data Mining* 1(3), 142–156 (2008)
23. Vert, J.-P., Yamanishi, Y.: Supervised graph inference. In: NIPS (2005)
24. Vishwanathan, S.V.N., Borgwardt, K., Schraudolph, N.: Fast computation of graph kernels. In: NIPS (2007)
25. Wu, M., Schölkopf, B.: Transductive classification via local learning regularization. In: AISTATS (2007)
26. Yamanishi, Y., Vert, J.-P., Kanehisa, M.: Supervised enzyme network inference from the integration of genomic data and chemical information. *Bioinformatics* 21, i468–i477 (2005)
27. Yu, K., Chu, W., Yu, S., Tresp, V., Xu, Z.: Stochastic relational models for discriminative link prediction. In: NIPS (2007)
28. Zan Huang, H.C., Li, X.: Link prediction approach to collaborative filtering. In: JCSDL (2005)
29. Zhou, D., Bousquet, O., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: NIPS, pp. 321–328 (2004)
30. Zhou, D., Zhu, S., Yu, K., Song, X., Tseng, B.L., Zha, H., Giles, C.L.: Learning multiple graphs for document recommendations. In: WWW '08, pp. 141–150. ACM, New York (2008)
31. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using Gaussian fields and harmonic functions. In: ICML (2003)



# Modeling Relations and Their Mentions without Labeled Text

Sebastian Riedel, Limin Yao, and Andrew McCallum

University of Massachusetts, Amherst,  
Amherst, MA 01002, U.S.

{riedel,lm,yao,mccallum,lncs}@cs.umass.edu

**Abstract.** Several recent works on relation extraction have been applying the distant supervision paradigm: instead of relying on annotated text to learn how to predict relations, they employ existing knowledge bases (KBs) as source of supervision. Crucially, these approaches are trained based on the assumption that each sentence which mentions the two related entities is an expression of the given relation. Here we argue that this leads to noisy patterns that hurt precision, in particular if the knowledge base is not directly related to the text we are working with. We present a novel approach to distant supervision that can alleviate this problem based on the following two ideas: First, we use a factor graph to explicitly model the decision whether two entities are related, and the decision whether this relation is mentioned in a given sentence; second, we apply constraint-driven semi-supervision to train this model without any knowledge about which sentences express the relations in our training KB. We apply our approach to extract relations from the New York Times corpus and use Freebase as knowledge base. When compared to a state-of-the-art approach for relation extraction under distant supervision, we achieve 31% error reduction.

## 1 Introduction

In relation extraction we often encounter a lack of explicitly annotated text, but an abundance of structured data sources such as company databases or large scale public knowledge bases like Freebase [2]. In the spirit of *distant supervision* [8,19], recent work [18,3] has shown how to exploit such knowledge: they heuristically align the given knowledge base to text and use this alignment to learn a relation extractor. Their approach is based on the following *distant supervision assumption*:

If two entities participate in a relation, **all sentences** that mention these two entities express that relation.

In practice, this allows them to extract features from all the sentence to feed a relation classifier. This approach has helped [18] to extract several thousand relations from Wikipedia at a precision of about 70% using Freebase as supervision source, a knowledge base derived in large parts from Wikipedia info-boxes.

---

<sup>1</sup> Also referred to as *weak* or *self supervision*.

In this work we argue that the distant supervision assumption is too strong and needs to be relaxed, in particular when the training knowledge base is an *external* source of information and not primarily derived from the training text. This is the case, for example, when we want to extract new relations from newswire instead of Wikipedia. This scenario is very relevant in practice—after all, many structured data sources are not derived from the textual data we want to extract new relations from.

When the knowledge base is external, entities may just appear in the same sentence because they are related to the same topic, not necessarily because the sentence is expressing their relations in our training knowledge base. In fact, by manual inspection (see section 2) we find that the distant supervision assumption is violated approximately 13% of the time when aligning Freebase to Wikipedia, but 31% when aligning to the New York Times Corpus [22, 2]

In this paper we employ the following *expressed-at-least-once* assumption and show that it leads to more accurate results:

If two entities participate in a relation, **at least one sentence** that mentions these two entities might express that relation.

Intuitively this statement holds with more certainty, but it also complicates our prediction task. Previously, we could simply take all sentences, aggregate features, and then solve a simple classification task. Now we do not know which sentences express relations, both during testing and training.

To tackle this problem we make two contributions. First, we introduce a novel undirected graphical model that captures both the task of predicting relations between entities, and the task of predicting which sentences express these relations. Our model connects a *relation variable* for two entities with a set of binary *relation mention* variables that indicate whether certain candidate sentences are expressing this relation. Crucially, the relation mention variables are unobserved at training time: we only know that a relation is expressed at least once, but not in which sentences.

Second, we propose to train this graphical model by framing distant supervision as an instance of *constraint-driven semi-supervision* [5, 4, 11, 16]. This type of supervision is applied to settings where some variables are latent. Roughly speaking, here model parameters are optimized to ensure that predictions will satisfy a set of user-defined constraints, as opposed to a set of target labels. In this framework our approach of distant supervision can be implemented by using the expressed-at-least-once constraint at training time.

As learner we choose SampleRank [27], a very efficient method to train parameters of large scale factor graphs. Recent work has shown that it can be naturally extended to the case of constraint-driven learning [24]. We believe that this choice will also be crucial for future work, where we expect our models to make joint relation, entity and coreference decisions across a whole corpus. SampleRank supports this setup because it makes parameter updates early and *within* inference.

---

<sup>2</sup> This is the average over three relation types: *nationality*, *contains* and *place\_of\_birth*.

We apply our model to extract relations from the New York Times corpus using Freebase as the external supervision source. We observe that our model with expressed-at-least-once assumption leads to 91% precision for our top 1000 predictions. When compared to 87% precision for a model based on the distant supervision assumption, this amounts to 31% error reduction.

In the following we will first give some background on relation extraction and distant supervision, then present our factor graph for joint relation type and relation mention identification. We then explain how to use SampleRank to incorporate the expressed-at-least-once assumption, present related work, show our empirical results, and conclude.

## 2 Relation Extraction under Distant Supervision

Relation Extraction is understood here as the task of predicting the relations expressed in natural language text. Consider, for example, the following sentence

**Elevation Partners**, the \$ 1.9 billion private equity group that was *founded* by **Roger McNamee** ...

Here the pair of entity mentions “Elevation Partners” and “Roger McNamee” is a *relation mention candidate* because its context might express a semantic relation between the corresponding pair of entities. A relation extractor takes such a candidate and determines the semantic relation that it might express, if any. In the above case a good extractor predicts the *founded* relation; this implies that the relation mention candidate is indeed a *relation mention*.<sup>3</sup>

In the works of [18,318], relation extraction is understood somewhat differently. Their primary goal is to determine whether a relation between a given pair of entities is expressed *somewhere* in the text, not necessarily where it is expressed. In other words, they care for *relations*, not *relation mentions*.

Focusing on relations instead of relation mentions has several benefits. First, it is very relevant in practice because downstream applications often care for entities and their relations, not for every mention of these. Second, it allows us to aggregate evidence for a relation from several places in the corpus. Finally, it simplifies the machine learning task: while we usually only have a few annotated relation mentions, we often have many existing pairs of related entities of the type we want to extract.

To illustrate the final point, let us consider the work of [18]. Their task is to extract relations of Freebase, a large online and collaborative knowledge base, from Wikipedia text. They tackle it by using the existing relations in Freebase as training data: for each related pair of entities they collect all sentences that mention both entities as input observation  $\mathbf{x}$ , and use their relation type in Freebase as label  $y$ . Together with a set of unrelated pairs of entities as negative instances, they train a classifier to predict relations (but not relation mentions).

---

<sup>3</sup> Note that in this work we focus on *closed* relation extraction where the extractor predicts one of a finite and fixed set of relations.

The approaches of [18,3] assume that each relation mention candidate is indeed a relation mention. Clearly, this assumption can be violated. Let us again consider the *founded* relation between “Roger McNamee” and “Elevation Partners. In an 2007 article of the New York Times we find this relation mention candidate:

**Roger McNamee**, a managing director at **Elevation Partners**, ...

This sentence does not express that Roger McNamee is a founder of Elevation Partners. It may make it more likely, but there are certainly cases where managing directors are not founders. The problem with this observation is that at training time we may learn some positive weight for the feature “<Entity1>, a managing director at <Entity2>”. When testing our model we may see this feature for a director A that is *not* a founder of a company B, and predict a false positive.

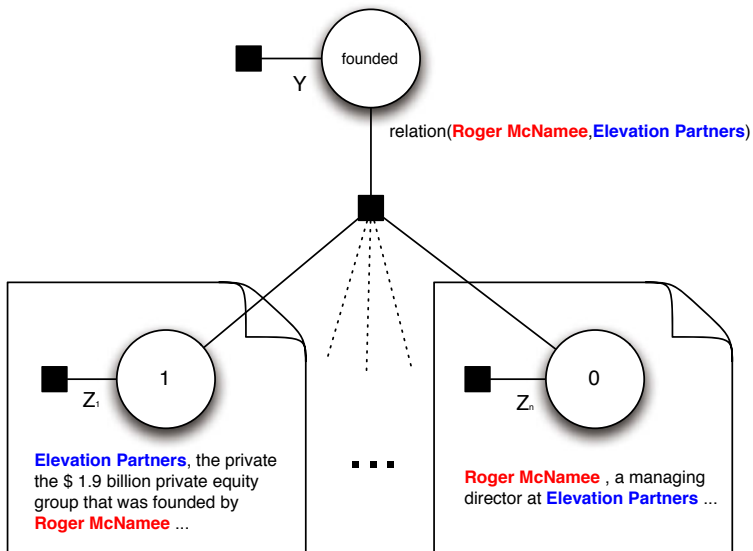
To get a sense of how frequently the distant supervision assumption is violated in practice, we test it for the case of Freebase and two different text corpora: Wikipedia articles and the New York Times corpus. To this end we consider three frequent relation types: *nationality*, *place\_of\_birth*, and *contains*. For each type we sample 100 relation mention candidates from both corpora, and evaluate whether these candidates are or are not expressing the relation in question. Table 1 presents the gathered statistics.

**Table 1.** Percentage of times a related pair of entities is mentioned in the same sentence, but where the sentence does not express the corresponding relation

Relation Type	New York Times	Wikipedia
nationality	38%	20%
place_of_birth	35%	20%
contains	20%	10%

We can see that for NYT data, the probability that a candidate mention is a non-mention is quite high. This is not difficult to understand: Take *nationality* as an example. Since a citizen of a country usually lives in the country, he/she will be involved in events happening in the country. News articles will report such events, and hence naturally mention the country and the person together. However, there is usually no need to express the fact that the person is indeed a citizen of the country—most readers care about the events but not the nationality of their participants.

How does this compare to relation mentions in Wikipedia? Here articles are centered around entities, and express targeted information about them. For example, if the article concerns a person, we expect the article to mention the person’s citizenship. However, unless the person holds a special role (say, a political role) in his country, we do not expect many additional sentences that mention both him and his country. Indeed, when comparing the percentage of non-mentions for *nationality*, we find about twice as many cases of non-mentions in NYT articles than in Wikipedia data.



**Fig. 1.** Factor Graph for joint relation mention prediction and relation type identification. For each pair of entities that are mentioned together in at least one sentence we create one relation variable (the top variable here). For each of the pairs of entity mentions that appear in a sentence we create one relation mention variable, and connect it to the corresponding relation variable. Note that relation variables for different pairs of entities are considered to be independent.

### 3 Model

Our observations in the previous section suggest that we should model both relations and relation mentions in order to avoid the use of noisy patterns. We propose to achieve this using an undirected graphical model with two types of hidden variables. First, for a pair of entities  $S$  (source) and  $D$  (destination) that appears together in at least one sentence, a *relation variable*  $Y$  denotes the relation between them, or NA if there is no such relation. See an example relation variable in figure 1. Second, for each relation mention candidate  $i$ , we define a binary *relation mention variable*  $Z_i$  that is true if and only if mention  $i$  is indeed expressing the relation  $Y$  between the two entities. Two example mention variables can be seen in figure 1.

For each relation mention variable  $Z_i$  we will refer to its two argument entity mentions as (source)  $S_i$  and (destination)  $D_i$ . We will store additional information about the sentence  $Z_i$  appears in, such as the dependency path between  $S_i$  and  $D_i$ , in an observed value  $\mathbf{x}_i$ . This information is aggregated across all mention candidates in the vector  $\mathbf{x}$ . Finally, we will use  $\mathbf{Z}$  to denote the state of all mention candidates, and  $\|\mathbf{z}\|$  to represent the number of active relation mentions for a given assignment  $\mathbf{z}$  of  $\mathbf{Z}$ .

Our conditional probability distribution over these variables is defined as follows:

$$p(Y = y, \mathbf{Z} = \mathbf{z} | \mathbf{x}) \stackrel{\text{def}}{=} \frac{1}{Z_{\mathbf{x}}} \Phi^r(y) \Phi^{\text{join}}(y, \mathbf{z}, \mathbf{x}) \prod_i \Phi^m(z_i, \mathbf{x}_i).$$

Here the factor (template)  $\Phi^r$  assesses the general bias of the model towards a particular relation type  $y$  and is defined as a loglinear potential function  $\Phi^r(y) = \exp(\theta_y^r)$ . The factor  $\Phi^m$  is defined as a function over a relation mention variable and its corresponding observation  $\mathbf{x}_i$ :

$$\Phi^m(z_i, \mathbf{x}_i) \stackrel{\text{def}}{=} \exp\left(\sum_j \theta_j^m \phi_j^m(z_i, \mathbf{x}_i)\right)$$

The feature functions  $\phi_j^m$  are taken to be the binary features presented in [18]. For example, the feature

$$\phi_{101}^m(z_i, \mathbf{x}_i) \stackrel{\text{def}}{=} \begin{cases} 1 & z_i = 1 \wedge S_i, \text{ a managing director of } D_i \in \mathbf{x}_i \\ 0 & \text{otherwise} \end{cases}$$

returns 1 if  $Z_i$  is active and there is a sequence “ $S_i$ , a managing director of  $D_i$ ”, and 0 otherwise.

The factor  $\Phi^{\text{join}}$  links relation variables to their mentions. It is defined as follows:

$$\Phi^{\text{join}}(y, \mathbf{z}, \mathbf{x}) \stackrel{\text{def}}{=} \exp\left(\sum_j \theta_{j,y}^{\text{join}} \phi_j^{\text{join}}(\mathbf{z}, \mathbf{x})\right)$$

Here the feature functions  $\phi_j^{\text{join}}$  are defined in terms of the mention feature functions  $\phi_j^m$ :

$$\phi_j^{\text{join}}(\mathbf{z}, \mathbf{x}) \stackrel{\text{def}}{=} \begin{cases} 1 & \exists i : z_i = 1 \wedge \phi_j^m(z_i, \mathbf{x}_i) = 1 \\ 0 & \text{otherwise} \end{cases}$$

Hence, the feature  $\phi_j^{\text{join}}$  indicates whether feature  $\phi_j^m$  is active for any of the active relation mentions (as indicated by  $z_i = 1$ ). For example,  $\Phi_{101}^{\text{join}}$  fires if “a managing director” appears between the corresponding entity mentions of any the active relation mentions. This is precisely the type of feature used in [18] for the relation classifier. The crucial difference is that here we consider only *active* mention candidates, instead of *all* mention candidates.

To construct this factor graph for each pair of candidate entities, we use FACTORIE [17], a probabilistic programming language that simplifies the construction process, as well as inference and learning.

### 3.1 Inference

There are two types of inference in the model: sampling from the posterior during training (see section 4), and finding the most likely configuration (aka MAP inference). In both settings we employ a block Gibbs sampler [14,13] that randomly picks a relation mention  $i$ , and jointly samples both  $Z_i$  and the corresponding relation variable  $Y$ . Here the sample is drawn conditioned on the state of all remaining relation mention variables. At test time we decrease the temperature of our sampler in order to find an approximation of the MAP solution.

We use block sampling instead of single-variable Gibbs sampling because of the strong correlation between mention variables  $Z_i$  and relation type  $Y$ . Assume, for example, that the current relation type  $Y$  is set to NA, all relation mentions  $Z_i$  are inactive, and we want to sample a new state for the first relation mention  $Z_1$ . In this case a model will give a near zero probability for  $Z_1 = 1$  because it has learned that this assignment is inconsistent with  $Y = \text{NA}$ . Likewise, changing  $Y$  with all  $Z_i$  fixed to be 0 will also receive a very low probability. This may happen even if the model assigns a high probability to the combination of  $Z_1 = 1$  and, say,  $Y = \text{founder}$ . Changing both relation and relation variable in concert overcomes this problem.

## 4 Rank-Based Learning and Distant Supervision

Most learning methods need to calculate the model expectations [15] or the MAP configuration [7] before making an update to the parameters. This step of inference is usually the bottleneck for learning, even when performed approximately.

SampleRank [21,27] is a rank-based learning framework that alleviates this problem by performing parameter updates *within* MCMC inference. Every pair of consecutive samples in the MCMC chain is ranked according to the model and the ground truth, and the parameters are updated when the rankings disagree. This allows the learner to acquire more supervision per instance, and has led to efficient training for models in which inference is expensive and generally intractable [23].

SampleRank considers two ranking functions for an assignment  $\mathbf{y}$ : (1) the probability (model ranking)  $p(\mathbf{y}) = \frac{1}{Z} \exp(\langle \Theta, \phi(\mathbf{y}) \rangle)$ , where  $\phi(\mathbf{y})$  is a feature representation of  $\mathbf{y}$ , and (2) a *truth function*  $\mathcal{F}(\mathbf{y})$  (objective ranking). One such truth function could be a per-entity-pair accuracy with respect to some labeled relations, another could be the F1-measure.

The goal of applying SampleRank is to find parameters that make model ranking and objective ranking as consistent as possible. To achieve this, SampleRank performs the following update at each step of an MCMC chain (see section 3.1). Let  $\mathbf{y}^{i-1}$  be the previous sample, and  $\mathbf{y}^i$  the current sample of the chain,  $\alpha$  be the learning rate, and  $\Delta = \phi(\mathbf{y}^{i-1}) - \phi(\mathbf{y}^i)$ . Then the weights  $\Theta$  are updated as follows:

$$\Theta = \Theta + \begin{cases} \alpha\Delta & \text{if } \frac{p(\mathbf{y}^{i-1})}{p(\mathbf{y}^i)} < 1 \wedge \mathcal{F}(\mathbf{y}^{i-1}) > \mathcal{F}(\mathbf{y}^i) \\ -\alpha\Delta & \text{if } \frac{p(\mathbf{y}^{i-1})}{p(\mathbf{y}^i)} > 1 \wedge \mathcal{F}(\mathbf{y}^{i-1}) < \mathcal{F}(\mathbf{y}^i) \\ 0 & \text{otherwise} \end{cases}.$$

Note that to evaluate the model ratios we do not require to calculate the partition function  $Z$ .

To better illustrate SampleRank, let us consider the factor graph of figure [11](#) and assume we are in the shown state. Our Gibbs Sampler in section [3.1](#) now assigns a new state to relation variable and one of the relation mention variables. For example, it leaves the relation mention variable unchanged but sets the relation variable to *child-of*.

In early stages of training, this proposal may still have a higher probability than the previous state, hence  $\frac{p(\mathbf{y}^{i-1})}{p(\mathbf{y}^i)} < 1$ . However, since we know that Roger McNamee is not a child of Elevation Partners, the previous state has higher truth score than the current state, and hence  $\mathcal{F}(\mathbf{y}^{i-1}) > \mathcal{F}(\mathbf{y}^i)$ . This means that SampleRank will update weights into the direction  $\Delta$  of the feature vector  $\phi(\mathbf{y}^{i-1})$  for the previous state.

In the following we will show how several distant supervision approaches can be incorporated into this framework. In all cases the truth function  $\mathcal{F}(y, \mathbf{z})$  for an assignment of relation and relation mention variables is decomposed into

$$\mathcal{F}(y, \mathbf{z}) = \mathcal{F}_r(y) + \mathcal{F}_m(\mathbf{z}, y_{\text{truth}})$$

where  $\mathcal{F}_r(y)$  only assesses the truth of the relation variable  $y$  and  $\mathcal{F}_m(\mathbf{z}, y_{\text{truth}})$  assesses the truth of the relation mention variables. Here  $y_{\text{truth}}$  is set to be the relation associated with the entity pair of  $y$  in our training knowledge base, or NA if no such relation exists. We will see later why  $\mathcal{F}_m$  needs knowledge of  $y_{\text{truth}}$ .

For all approaches  $\mathcal{F}_r(y)$  is fixed to be

$$\mathcal{F}_r(y) = \begin{cases} 1 & y = y_{\text{truth}} \\ -1 & \text{otherwise} \end{cases}.$$

That is, a match with the true relation increases the truth score by one, otherwise the score is decreased by one.

#### 4.1 Distant Supervision

A distant supervision baseline akin to [18](#) and [3](#) can be easily implemented using SampleRank. In this case we simply consider all variables  $Z_i$  to be fixed:  $Z_i = 0$  if the corresponding relation variable  $y$  is NA, and  $Z_i = 1$  otherwise. Inference then only considers the  $Y$  variables.

#### 4.2 Joint Supervision

We essentially propose two modifications to the original distant supervision approach. First, jointly infer mentions and relations, and second, relax the assumption that each candidate mention is indeed an actual mention. We can easily implement the first modification by using the distant supervision truth function

$$\mathcal{F}_m^{\text{distant}}(\mathbf{z}, y_{\text{truth}}) = \begin{cases} \|\mathbf{z}\| & y_{\text{truth}} \neq \text{NA} \\ -\|\mathbf{z}\| & \text{otherwise} \end{cases}. \quad (1)$$



That is, if the two entities in question are related in our training knowledge base (i.e.,  $y_{\text{truth}} \neq \text{NA}$ ), every active relation mention is encouraged. Otherwise every relation mention is preferred to be inactive.

### 4.3 Expressed-at-Least-Once Supervision

SampleRank allows us to naturally incorporate the expressed-at-least-once assumption we presented in section 4.1. We simply use the following truth function for relation mentions:

$$\mathcal{F}_m^{\text{once}}(\mathbf{z}, y_{\text{truth}}) = \begin{cases} 1 & y_{\text{truth}} \neq \text{NA} \wedge \|\mathbf{z}\| \geq 1 \\ -1 & y_{\text{truth}} \neq \text{NA} \wedge \|\mathbf{z}\| = 0 \\ -\|\mathbf{z}\| & \text{otherwise} \end{cases} \quad (2)$$

That is, if the true relation type  $y_{\text{truth}}$  is not NA, an assignment to the relation mention variables has maximal rank if at least one mention is active. In case the pair of entities is not related according to Freebase, an assignment is discouraged proportional to the amount of active relation mentions.

## 5 Related Work

While much work on relation extraction has focused on fully-supervised approaches [11, 9], we work in the framework of distant supervision. In this context existing work has primarily relied on the distant supervision assumption [18, 3, 8]. [28] use a more sophisticated heuristic to decide which candidates are relation mentions. This heuristic is tailored to extracting infobox attributes from Wikipedia articles. By contrast, our method is designed for relations between entity pairs mentioned in newswire.

Our work is based on constraint-driven semi-supervised learning [6, 16]. Generally, constraint-driven methods are used when (a) only a small set of labelled training instances are available, and (b) there are some (hard or soft) constraints that are known to hold across the unlabelled data. These constraints are then used as additional source of supervision. To our knowledge, constraint-driven learning has not been applied to information extraction under distant supervision.

There are many approaches to train undirected models with latent variables besides SampleRank. An alternative method is the latent perceptron [26]. Here in each iteration the MAP assignment for a set of latent variables is predicted. Then the non-latent label variable is deterministically inferred from the latent variables and compared to the gold label. If labels disagree, weights are updated in a manner similar to the regular perceptron. For us this approach would not directly be helpful for two reasons: First, it is not clear how we could incorporate prior knowledge such as the at-least-once assumption; second, in our case the relation between latent and non-latent label variables is not deterministic.

Contrastive estimation [25] is another related approach. Here parameters of a model with latent variables are learned by maximizing the ratio between the probability of the observed data  $x$  and the sum of probabilities over “sensible but wrong”  $x'$  in a local neighborhood of  $x$ . In some sense, what we do is similar if we consider the relation variables as observed  $x$ , and their negated state  $1 - x$  as local neighborhood. However, we do not maximize probability, but match rankings. Also, our at-least-once constraint would need to be formulated as a (complex) prior over the set of hidden mention variables.

Finally, our approach can be seen as a novel take on Multi-Instance Learning [10]. Here training instances are divided into bags, and we only know that some bags contain at least one positive example while the remaining ones contain only negative examples. Our constraint-driven approach works in the same setting. However, it also allows to include additional constraints that a user may have. For example, we could inject constraints corresponding to the heuristics used in [28]. Moreover, our approach is the first that can discriminatively train general factor graphs. This will be important for future work such as joint coreference, entity type and relation extraction. Here the graphical structure will get more involved.

## 6 Evaluation

Our experiments aim to provide evidence for the following hypothesis: explicitly relaxing the distant supervision assumption in a probabilistic model can lead to substantial increase in precision. To this end we follow [18] and compare it against both the distant and the supervised joint model using a held-out set of relations and documents. Since this type of evaluation suffers from false negatives (some negative relations we extracted may in fact be positive but not in the knowledge base), we also manually evaluate the predicted relations.

Note that for all our models we use exactly the same feature set. However, we choose the best number of training epochs for each model individually. Also note that for training we need negative instances. For this purpose we generally pick 10% of the entity pairs that appear in the same sentence but are not related according to Freebase.

### 6.1 Data

Following [18] we use Freebase as our distant supervision source. Freebase is an online database that stores facts about entities and their relations. We extract all relations from a December 2009 snapshot of Freebase. Four categories of Freebase relations are used: “people”, “business”, “person”, and “location”. These types of relations are chosen because we expect that they appear frequently in the newswire corpus. In total this provided over 3.2 million relation instances of 430 Freebase relation types, and over 1.8 million entities that participate in these relations.

For the choice of text corpus we divert from [18] and use the New York Times corpus [22]. This allows us to evaluate our approach when the distant supervision

source is external. The New York Times data contains over 1.8 million articles written and published by the New York Times between January 1, 1987 and June 19, 2007. Generally, we find that Freebase entities are frequently mentioned in the NYT corpus. For example, for the year 2007 about 700,000 mentions of Freebase entities appear in the corpus. Naturally, we observe a smaller number of cases in which two related entities are mentioned in the same sentence: again for the year 2007 we find about 170,000 such cases.

## 6.2 Preprocessing

In order to find entity mentions in text we first used the Stanford named entity recognizer [12]. The NER tagger segments each document into sentences and classifies each token into four categories: PERSON, ORGANIZATION, LOCATION and NONE. We treat consecutive tokens which share the same category as single entity mention. Then we associate these mentions with Freebase entities. This is achieved by simply performing a string match between entity mention phrase and the canonical names of entities in Freebase.

Next, for each pair of entities participating in a relation of our training KB, we traverse the text corpus and find sentences in which the two entities co-occur. Each pair of entity mentions is considered to be a relation mention candidate. For each such candidate we extract a set of features (see section 3). The types of features are essentially corresponding to the ones used by [18]: we used lexical, Part-Of-Speech (POS), named entity and syntactic features (i.e. features obtained from the dependency parsing tree of a sentence). We applied the openNLP POS tagger<sup>4</sup> to obtain POS tags and used the MaltParser [20] for dependency parsing.

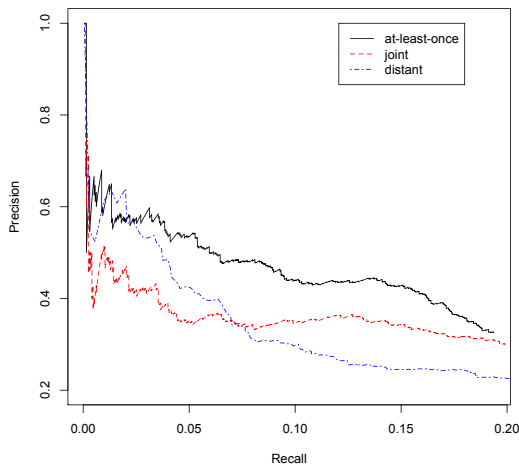
## 6.3 Held-Out Evaluation

Following [18] we divide the Freebase relations into two parts, one for training and one for testing. The former is aligned to the years 2005-2006 of the NYT corpus, the latter to the year 2007. As candidate relation instances we use all pairs of Freebase entities that are at least once mentioned in the same sentence. Note that the amount of Freebase relations mentioned in the training set (4700) and test set (1950) is relatively low due to a smaller overlap between Freebase and the New York Times. Hence we cannot evaluate our models with the same quantity of data as [18].

In figure 2 we compare the precision and recall curve for the baseline distant-supervision model (distant), the supervised joint model (joint) and the distant model with expressed-at-least-once assumption (at-least-once). The curve is constructed by ranking the predicted relation instances using their loglinear score. For the distant supervision baseline this score is first normalized by the number of mentions<sup>5</sup>. We traverse this list from high score to low score, and measure precision and recall at each position.

<sup>4</sup> Available at <http://opennlp.sourceforge.net/>

<sup>5</sup> This yielded the best results for the baseline. We also tried to use conditional probabilities to rank. This led to poor results because SampleRank training has no probabilistic interpretation.



**Fig. 2.** Precision and recall for the held out data and three approaches: distant supervision, joint supervision, and at-least-once supervision

We can see that the model with expressed-at-least-once assumption is consistently outperforming the distant supervision baseline and the supervised joint model. This suggests that the at-least-once model has the best sense of how relations that are already contained in Freebase are expressed in NYT data. However, it does not necessarily mean that it knows best how relations are expressed that are not yet in Freebase. We address this in the next section.

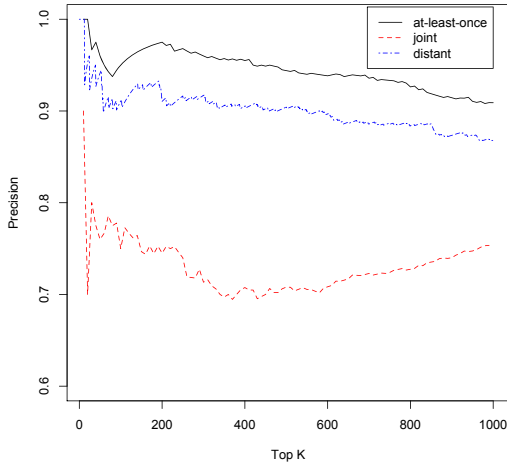
## 6.4 Manual Evaluation

For manual evaluation all Freebase entities and relations are used as training instances. As candidate relation instances we choose those entity pairs which appear together in the NYT test set, but for which least one participating entity is not in Freebase. This means that there is no overlap between the held-out and manual candidates. Then we apply our models to this test set, and asked two annotators to evaluate the top 1000 predicted relation instances.

We cannot calculate recall in this case, since we cannot provide all relation instances expressed in our corpus. Instead we use a “Precision at  $K$ ” metric with respect to the ranked lists we extracted in section 6.3. Figure 3 shows the precisions for values of  $K$  between 0 and 1000.

We first note that the precision is much higher for manual evaluation than for held-out evaluation. This shows that false negatives in Freebase are an issue when doing held-out evaluation. Many of the false positives we predict are in fact true relation instances and just do not appear in Freebase.

For manual evaluation the at-least-once model is still the winner. At  $K = 1000$  we observe a precision of 91% for at-least-once supervision, 87% for distant supervision. This amounts to an error reduction rate of 31%. The sign test shows that the at-least-once model is significantly better than the distant supervision



**Fig. 3.** Precision at  $K$  for manually evaluated predictions

model, with  $p \ll 0.05$ . We also note that despite using the same assumption, the joint model performs much worse than the distant supervision approach in this scenario. Learning a model of relations and mentions is inherently more difficult. Using a wrong assumption will hence more likely hurt performance.

Does the at-least-once model help to fix the type of error discussed in section 2? To find out, we inspect the results of the *founded* relation. When we consider the top 100 instances of this relation for the distant supervision system, we observe a precision of 45%. Compare this to 72% precision for the at-least-once model.

On close inspection, most of the distant supervision errors for the *founded* relation stem from cases where patterns such as “director of” appear. They indicate that the person in question works for the given company. Because in the training set such patterns often appear when a person is a founder, they gain high weights and appear high up in the ranking.

The at-least-once model also makes this type of error, but to a much lesser extent. This is not surprising if we consider that for training instances with only one mention, the at-least-once and distant supervision assumptions are equivalent. Assume that according to Freebase, person A founded company B. If there is only one mention of A and B in the NYT training corpus, it has to be a mention of *founded*, even if the sentence says “director-of”. This leads to a higher weight for “director-of” as *founded* pattern.

## 7 Conclusion

This paper presents a novel approach to extract relations from text without explicit training annotation. Recent approaches assume that every sentence that mentions two related entities expresses the corresponding relation. Motivated

by the observation that this assumption frequently does not hold, in particular when considering external knowledge bases, we propose to relax it. Instead we assume that at least one sentence which mentions two related entities expresses the corresponding relation.

To model this assumption we make two contributions. First, we introduce a novel undirected graphical model that captures both the task of predicting relations between entities, and the task of predicting which sentences express these relations. Second, we propose to train this graphical model by framing distant supervision as an instance of constraint-driven semi-supervision. In particular, we use SampleRank, a discriminative learning algorithm for large factor graphs, and inject the expressed-at-least-once assumption through a truth function.

Empirically this approach improves precision substantially. For the task of extracting 1000 Freebase relation instances from the New York Times, we measure a precision of 91% for at-least-once supervision, and 87% for distant supervision. This amounts to an error reduction rate of 31%.

A crucial aspect of our approach is its extensibility: framed exclusively in terms of factor graphs and truth functions, it is conceptually easy to apply it to larger tasks such as the joint prediction of relations and entity types. In future work we will exploit this aspect and extend our model to jointly perform other relevant tasks for the automatic construction of KBs.

**Acknowledgements.** This work was supported in part by the Center for Intelligent Information Retrieval, in part by SRI International subcontract #27-001338 and ARFL prime contract #FA8750-09-C-0181, in part by The Central Intelligence Agency, the National Security Agency and National Science Foundation under NSF grant #IIS-0326249, and in part by UPenn NSF medium IIS-0803847. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

## References

1. Bellare, K., McCallum, A.: Generalized expectation criteria for bootstrapping extractors using record-text alignment. In: Empirical Methods in Natural Language Processing (EMNLP), pp. 131–140 (2009)
2. Bollacker, K., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: SIGMOD '08: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 1247–1250. ACM, New York (2008)
3. Bunescu, R.C., Mooney, R.J.: Learning to extract relations from the web using minimal supervision. In: Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, ACL' 07 (2007)
4. Chang, M.W., Goldwasser, D., Roth, D., Tu, Y.: Unsupervised constraint driven learning for transliteration discovery. In: NAACL '09: Proceedings of Human Language Technologies: Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 299–307 (2009)

5. Chang, M.W., Ratnoff, L., Rizzolo, N., Roth, D.: Learning and inference with constraints. In: AAAI Conference on Artificial Intelligence, pp. 1513–1518. AAAI Press, Menlo Park (2008)
6. Chang, M.W., Ratnoff, L., Roth, D.: Guiding semi-supervision with constraint-driven learning. In: Annual Meeting of the Association for Computational Linguistics (ACL), pp. 280–287 (2007)
7. Collins, M.: Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '02), vol. 10, pp. 1–8 (2002)
8. Craven, M., Kumlien, J.: Constructing biological knowledge-bases by extracting information from text sources. In: Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology, Germany, pp. 77–86 (1999)
9. Culotta, A., McCallum, A.: Joint deduplication of multiple record types in relational data. In: Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM '05), pp. 257–258. ACM, New York (2005)
10. Dietterich, T., Lathrop, R., Lozano-Pérez, T.: Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence* 89(1-2), 31–71 (1997)
11. Dimitry Zelenko, C.A., Richardella, A.: Kernel methods for relation extraction. *JMLR* 3(6), 1083–1106 (2003)
12. Finkel, J.R., Grenager, T., Manning, C.: Incorporating non-local information into information extraction systems by gibbs sampling. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL' 05), pp. 363–370 (June 2005)
13. Geman, S., Geman, D.: Stochastic relaxation, gibbs distributions, and the bayesian restoration of images, pp. 452–472 (1990)
14. Jensen, C.S., Kong, A., Kjaerulff, U.: Blocking gibbs sampling in very large probabilistic expert systems. *International Journal of Human Computer Studies. Special Issue on Real-World Applications of Uncertain Reasoning* 42, 647–666 (1993)
15. Lafferty, J.D., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: International Conference on Machine Learning, ICML (2001)
16. Mann, G.S., McCallum, A.: Generalized expectation criteria for semi-supervised learning of conditional random fields. In: Annual Meeting of the Association for Computational Linguistics (ACL), pp. 870–878 (2008)
17. McCallum, A., Schultz, K., Singh, S.: Factorie: Probabilistic programming via imperatively defined factor graphs. In: Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C.K.I., Culotta, A. (eds.) *Advances in Neural Information Processing Systems*, vol. 22, pp. 1249–1257 (2009)
18. Mintz, M., Bills, S., Snow, R., Jurafsky, D.: Distant supervision for relation extraction without labeled data. In: Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL' 09), pp. 1003–1011. Association for Computational Linguistics (2009)
19. Morgan, A.A., Hirschman, L., Colosimo, M., Yeh, A.S., Colombe, J.B.: Gene name identification and normalization using a model organism database. *J. of Biomedical Informatics* 37(6), 396–410 (2004)
20. Nivre, J., Hall, J., Nilsson, J.: Memory-based dependency parsing. In: Proceedings of CoNLL, pp. 49–56 (2004)

21. Rohanimanesh, K., Wick, M., McCallum, A.: Inference and learning in large factor graphs with a rank based objective. Tech. Rep. UM-CS-2009-08, University of Massachusetts, Amherst (2009)
22. Sandhaus, E.: The New York Times Annotated Corpus. Linguistic Data Consortium, Philadelphia (2008)
23. Singh, S., Schultz, K., McCallum, A.: Bi-directional joint inference for entity resolution and segmentation using imperatively-defined factor graphs. In: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), pp. 414–429 (2009)
24. Singh, S., Yao, L., Riedel, S., McCallum, A.: Constraint-driven rank-based learning for information extraction. In: North American Chapter of the Association for Computational Linguistics - Human Language Technologies, NAACL HLT (2010)
25. Smith, N.A., Eisner, J.: Contrastive estimation: training log-linear models on unlabeled data. In: ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pp. 354–362. Association for Computational Linguistics, Morristown (2005)
26. Sun, X., Matsuzaki, T., Okanohara, D., Tsujii, J.: Latent variable perceptron algorithm for structured classification. In: IJCAI'09: Proceedings of the 21st International Joint Conference on Artificial Intelligence, pp. 1236–1242. Morgan Kaufmann Publishers Inc., San Francisco (2009)
27. Wick, M., Rohanimanesh, K., Culotta, A., McCallum, A.: Samplerank: Learning preferences from atomic gradients. In: Neural Information Processing Systems (NIPS), Workshop on Advances in Ranking (2009)
28. Wu, F., Weld, D.S.: Autonomously semantifying wikipedia. In: Proceedings of the 16th ACM International Conference on Information and Knowledge Management (CIKM '07), pp. 41–50. ACM Press, New York (2007)



# An Efficient and Scalable Algorithm for Local Bayesian Network Structure Discovery

Sérgio Rodrigues de Morais and Alex Aussem

University of Lyon, F-69000, Lyon, University of Lyon 1,  
LIESP Laboratory, 69622 Villeurbanne, France

**Abstract.** We present an efficient and scalable constraint-based algorithm, called Hybrid Parents and Children (HPC), to learn the parents and children of a target variable in a Bayesian network. Finding those variables is an important first step in many applications including Bayesian network structure learning, dimensionality reduction and feature selection. The algorithm combines ideas from incremental and divide-and-conquer methods in a principled and effective way, while still being sound in the sample limit. Extensive empirical experiments are provided on public synthetic and real-world data sets of various sample sizes. The most noteworthy feature of HPC is its ability to handle large neighborhoods contrary to current CB algorithm proposals. The number of calls to the statistical test, and hence the run-time, is empirically on the order  $O(n^{1.09})$ , where  $n$  is the number of variables, on the five benchmarks that we considered, and  $O(n^{1.21})$  on a real drug design characterized by 138,351 features.

**Keywords:** Bayesian network structure learning, constraint-based methods, feature selection.

## 1 Introduction

This paper presents a novel algorithm which thoroughly exploits Bayesian networks (BN) theory in order to find the variables that are directly associated with a target, that is, those variables that remain probabilistic associated with a target even when conditioning on values of any subset of the other variables in a data set. Those variables are often called among the BN community as the *parents and children* of the target variable. A BN is a probabilistic model formed by a structure and parameters. The structure of a BN is a directed acyclic graph (DAG), whilst its parameters are conditional probability distributions associated with the variables in the model. The graph of a BN itself is an independence map (I-map), which is very useful for many applications, including feature subset selection, dimensionality reduction, and inferring causal relationships from observational data. However, the recent explosion of high dimensional data sets poses a serious challenge to existing BN learning algorithms. A principled solution to this problem is to directly seek a local network around the target without having to find the whole DAG.

Two types of BN structure learning methods have been proposed so far: *constraint-based* (CB) and *score-and-search* methods. Basically, CB learning methods systematically check the data for conditional independence relationships, whilst score-and-search methods make use of a score function for evaluating graphical structures with regard to the data set. While score-and-search methods are efficient for learning the full BN structure (see [18] for instance), the ability to scale up to hundreds of thousands of variables is a key advantage of CB methods over score-and-search methods. Several CB algorithms have been proposed recently [8, 11, 14–16, 20, 21] for local BN structure learning. They construct a local skeleton (i.e., the edges without their orientation) around the target node without having to construct the whole BN first, hence their scalability. Fortunately, the scalability of CB methods does not come at the loss of accuracy; they were recently shown to be among the top-ranking entrants in the WCCI2008 Causation and Prediction Challenge [9]. One of the best prediction accuracy was obtained by [4] using CB methods discussed in [20]. Even the more sophisticated approaches using novel structure-based causal discovery, such as [3], used the standard CB methods, in the first phase of the discovery process, to find a local skeleton.

CB methods can be divided in two classes: *incremental* methods (e.g., IAMB [20], BFMB [8]) and *divide-and-conquer* methods (e.g., MMMB [20]), PCMB [11]). While these algorithms are appropriate for situations where the number of parents and children (PC set) is relatively small, they are plagued by a severe problem: the number of false negatives (missing variables) increases swiftly as the size of the PC set increases<sup>1</sup>. There are mainly two reasons for this. The first is the unreliability of the conditional independence tests as the conditioning sets become large. This well known problem is common to all CB methods and has led several authors to reduce, as much as possible, the size of the conditioning sets with a view to enhancing the *data-efficiency* of their methods [8, 11]. The second reason is that the decisions for a node to enter the candidate PC set are often too severe and conservative. In practice, those problems plague all CB methods for target variables with many adjacent nodes and relatively few instances.

In this paper, we introduce an algorithm called *Hybrid Parents and Children* (HPC) that combines ideas from incremental and divide-and-conquer methods in order to alleviate the problem discussed above. A thorough discussion is provided for explaining why HPC can handle large number of adjacent nodes while still being sound in the sample limit. Extensive empirical experiments are then carried out on public synthetic data sets to assess its accuracy and scalability. Such experiments show that significant improvements in accuracy are obtained. In addition, the empirical number of calls to the independence test (and hence the *effective* complexity) is only  $O(n^{1.09})$  in practice on synthetic data and  $O(n^{1.21})$  on real-world data, where  $n$  is the number of variables. Finally, a proof of HPC's

---

<sup>1</sup> For instance, [4], the winners of the WCCI2008 challenge, outputs only 9 features out of 14 true features with a data set that consists of 500 instances.

correctness under the so called *faithfulness condition* is provided in the final part of this article.

## 2 Preliminaries

Formally, a BN is a tuple  $\langle \mathbb{G}, P \rangle$ , where  $\mathbb{G} = \langle \mathbf{U}, \mathbf{E} \rangle$  is a directed acyclic graph (DAG) whose nodes represent the variables in the domain  $\mathbf{U}$ , and whose edges represent direct probabilistic dependencies between them.  $P$  denotes the joint probability distribution on  $\mathbf{U}$ . The BN structure encodes a set of conditional independence assumptions: that each node  $X_i$  is conditionally independent of all of its non descendants in  $\mathbb{G}$  given its parents  $\mathbf{Pa}_i^{\mathbb{G}}$ . These independence assumptions, in turn, imply many other conditional independence statements, which can be extracted from the network using a simple graphical criterion called d-separation [12].

We denote by  $X \perp_P Y | \mathbf{Z}$  the conditional independence between  $X$  and  $Y$  given the set of variables  $\mathbf{Z}$  where  $P$  is the underlying probability distribution. Note that an exhaustive search of  $\mathbf{Z}$  such that  $X \perp_P Y | \mathbf{Z}$  is a combinatorial problem and can be intractable for high dimension data sets. We use  $X \perp_{\mathbb{G}} Y | \mathbf{Z}$  to denote the assertion that  $X$  is d-separated from  $Y$  given  $\mathbf{Z}$  in  $\mathbb{G}$ . We denote by  $\mathbf{dSep}(X, Y)$ , a set that d-separates  $X$  from  $Y$ . If  $\langle \mathbb{G}, P \rangle$  is a BN,  $X \perp_P Y | \mathbf{Z}$  if  $X \perp_{\mathbb{G}} Y | \mathbf{Z}$ . The converse does not necessarily hold. We say that  $\langle \mathbb{G}, P \rangle$  satisfies the *faithfulness condition* if the d-separations in  $\mathbb{G}$  identify *all and only* the conditional independencies in  $P$ , i.e.,  $X \perp_P Y | \mathbf{Z}$  if and only if  $X \perp_{\mathbb{G}} Y | \mathbf{Z}$ .

An important concept of BN is the Markov blanket of a variable, which is the set of variables that completely shields off this variable from the others. In other words, a Markov blanket  $\mathbf{M}_T$  of  $T$  is any set of variables such that  $T$  is conditionally independent of all the remaining variables given  $\mathbf{M}_T$ . A Markov boundary,  $\mathbf{MB}_T$ , of  $T$  is any Markov blanket such that none of its proper subsets is a Markov blanket of  $T$ . Suppose  $\langle \mathbb{G}, P \rangle$  satisfies the faithfulness condition. Then, for all  $X$ , the set of parents, children of  $X$ , and parents of children (spouses) of  $X$  is the unique Markov boundary of  $X$ . A proof can be found for instance in [10]. We denote by  $\mathbf{PC}_T^{\mathcal{G}}$ , the set of parents and children of  $T$  in  $\mathcal{G}$ , and by  $\mathbf{SP}_T^{\mathcal{G}}$ , the set of spouses of  $T$  in  $\mathcal{G}$ , i.e., the variables that have common children with  $T$ . These sets are unique for all  $\mathcal{G}$ , such that  $\langle \mathcal{G}, P \rangle$  satisfies the faithfulness condition and so we will drop the superscript  $\mathcal{G}$ .

### 2.1 Constraint-Based Structure Learning

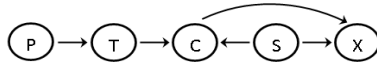
The induction of local BN structures is handled by CB methods through the identification of local neighborhoods. Hence their scalability to very high dimensional data sets. CB methods systematically check the data for conditional independence relationships in order to infer a target's neighborhood. Typically, the algorithms run a  $\chi^2$  independence test when the data set is discrete and a Fisher's Z test when it is continuous in order to decide on dependence or

independence, that is, upon the rejection or acceptance of the null hypothesis of conditional independence. The reliability of a conditional independence test is dependent on the number of instances in the data set and the degree of freedom of the test. A practical consideration regarding the reliability of a conditional independence test is the size of the conditioning set as measured by the number of variables in the set, which in turn determines the number of values that the variables in the set may jointly take. Large conditioning sets produce sparse contingency tables and unreliable tests. This is why it is difficult to learn the neighborhood of a node having a large degree with CB methods. The number of possible configurations of the variables grows exponentially with the size of the conditioning set.

### 3 Pitfalls and Related Work

Both families of CB methods, that is *divide-and-conquer* and *incremental* methods, can be very useful when working in very high dimension data sets because they do not have to search the whole structure of a BN in order to find the local network around a target. Divide-and-conquer methods are based on the identification of the direct neighborhood of a target, that is, its parents and children ( $\mathbf{PC}_T$ ). Of course, divide-and-conquer methods can be iteratively used for finding local networks or the Markov boundary of a target, for instance that is exact what do the algorithms *MMMB* [19] and *PCMB* [11]. Those algorithms identify first the parents and children of  $T$  ( $\mathbf{PC}_T$ ) and then the set of the spouses of  $T$  ( $\mathbf{SP}_T$ ) for forming the Markov boundary of  $T$  ( $\mathbf{MB}_T = \mathbf{PC}_T \cup \mathbf{SP}_T$ ). On the other hand incremental methods incrementally searches the whole Markov boundary of a target ( $\mathbf{MB}_T$ ) without distinguishing the two subsets  $\mathbf{PC}_T$  and  $\mathbf{SP}_T$ . However, incremental methods can also be used for searching  $\mathbf{PC}_T$ . This can be achieved by first finding  $\mathbf{MB}_T$  and then eliminating the variables of  $\mathbf{SP}_T$  ( $\mathbf{PC}_T = \mathbf{MB}_T \setminus \mathbf{SP}_T$ ). This procedure is very simple and can be summarized as follows:  $\forall X \in \mathbf{MB}_T, X \in \mathbf{SP}_T$  (that is,  $X \notin \mathbf{PC}_T$ ) if and only if  $\exists \mathbf{Z} \in (\mathbf{MB}_T \setminus X)$  such that  $X \perp_P T | \mathbf{Z}$ . Nonetheless incremental methods are considered as *data-inefficient* because it often makes use of unnecessary large conditioning sets [11].

Inferring automatically from data the set  $\mathbf{PC}_T$  by the use of independence tests is not as easy as one could think at first sight. The following property serves as the common rule for discarding non-adjacent variables: Let  $\mathbf{U}$  be the set of all the variables in the data set, thus, under the faithfulness assumption,  $X$  and  $Y$  are not adjacent in  $\mathcal{G}$  if and only if  $\exists \mathbf{Z} \in \mathbf{U} \setminus \{X \cup Y\}$  such that  $X \perp Y | \mathbf{Z}$  [10]. An exhaustive search of  $\mathbf{Z}$  is a combinatorial problem and can be intractable for high dimension data sets. To solve this problem parents and children learning algorithms generally search a candidate set for  $\mathbf{PC}_T$  by starting with an empty set  $\mathbf{PC}_T = \emptyset$  and iteratively adding to the current set  $\mathbf{PC}_T$  the best candidate  $X$  such that there exists no set  $\mathbf{Z} \subseteq \mathbf{PC}_T \setminus X$  such that  $X \perp_P T | \mathbf{Z}$ . Let's call this generic procedure *LearnPC*. It can be implemented in several ways. For instance, it is called *GetPCD* in [11] and *MMPC* in [20]. However, this rule actually leads



**Fig. 1.** Toy problem about PC learning:  $\exists \mathbf{Z} \in \mathbf{PC}_T$ , so that,  $X \perp_{\mathcal{G}} T | \mathbf{Z}$

to a superset of  $\mathbf{PC}_T$  as noted in [11, 20]. Consequently, the algorithms must further process  $\mathbf{PC}_T$  in order to search and eliminate false positives.

For sake of illustration, consider the example in Figure 1. Let  $T$  be the target and  $\mathbf{U} = \{P, T, C, S, X\}$ . The set of parents and children is clearly  $\mathbf{PC}_T = \{P, C\}$ . As we can see in Figure 1,  $\exists \mathbf{Z} \subseteq \mathbf{PC}_T \setminus X$  such that  $T \perp_P X | \mathbf{Z}$ . Therefore, to remove  $X$ , the output of  $LearnPC(T)$  must be further processed. Fortunately, owing to Theorem 1 [10], a false positive node  $X$  may be easily identified by testing whether  $T \in LearnPC(X)$ . In fact,  $X \in \mathbf{PC}_T$  if and only if  $[X \in LearnPC(T)] \ \& \ [T \in LearnPC(X)]$ . This is the way divide-and-conquer methods correct the flaw discussed above. Clearly, for the boolean operator to return TRUE, both its operands should be TRUE. In practice, however, the procedure is conservative because makes it harder for a true positive node to enter the set  $\mathbf{PC}_T$ . Consequently, the procedure is highly sensitive to the false negative errors. This problem plagues divide-and-conquer methods for target variables with many adjacent nodes and relatively few instances. Loosely speaking, the larger  $\mathbf{PC}_T$ , the more likely it is to have false negative errors in the output of divide-and-conquer parents and children learning algorithms.

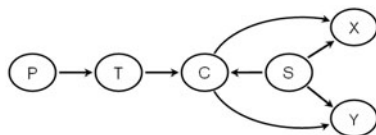
**Theorem 1.** *Let  $\mathcal{G} = (V, E)$  be a DAG and  $X, Y \in \mathbf{U}$ . Then if  $X$  and  $Y$  are  $d$ -separated by some set, they are  $d$ -separated either by the set consisting of the parents of  $X$  or the set consisting of the parents of  $Y$ .*

The conservative procedure applied by divide-and-conquer methods is also a source of errors in the search process when there are some approximate deterministic relationships (see Definition 1) among the variables on the data set. For instance let us consider again Figure 1. Considering that the variables  $S$  and  $C$  are associated through a deterministic relationship, then clearly  $C \perp T | S$ . In this case divide-and-conquer methods will fail in finding the variable  $C$  as a child of the target  $T$  because the output of  $LearnPC(C)$  will contain only the variable  $S$ . Consequently, the necessarily boolean operator used by divide-and-conquer methods will exclude  $C$  from the set  $\mathbf{PC}_T$  even if  $LearnPC(T)$  finds  $C$  as a parent or child of  $T$ . Deterministic relationships are a source of unfaithfulness, but a DAG  $\mathcal{G}$  and a joint probability distribution  $P$  can still be faithful if some *approximate* deterministic relationships (ADR) are present in the data set. ADR can well lead to various errors in the search process as deterministic relationships do because conditional independence tests are highly unreliable in the presence of ADR. The existence of ADR in data is rather frequent. For instance, ADR are often present in survey data owing to hidden redundancies in

the questions [2, 17]. The exclusion of the obligation of applying the conservative boolean operator could help alleviating that problem.

**Definition 1.** *The association between the set of variables  $\mathbf{X}$  and a target  $T$  is an approximate deterministic relationship if and only if the fraction of tuples that violate the deterministic dependency is at most equal to some threshold.*

Divide-and-conquer methods have tried to improve the *data-efficiency* of the learning process by reducing the size of conditioning sets in the independence tests. However, they are not always more *data-efficient* than incremental methods. For instance, Figure 2 shows a case where divide-and-conquer methods are even less *data-efficient* than incremental methods. As one can see in Figure 2 the maximum size of the conditioning set  $\mathbf{Z}$  for a divide-and-conquer  $LearnPC(T)$  will be 4, that is,  $\mathbf{Z} = \{P, C, X, Y\}$ , whilst for an incremental  $LearnPC(T)$  it will be 3, that is,  $\mathbf{Z} = \{P, C, S\}$ .



**Fig. 2.** Toy example where divide-and-conquer algorithms can be less *data-efficient* than incremental algorithms

## 4 The Hybrid Parents and Children Algorithm

In this section, we present the *Hybrid Parents and Children* (HPC) algorithm with the view to alleviate some of the shortcomings discussed previously. HPC combines characteristics from divide-and-conquer, incremental and ensemble methods in order to improve the accuracy of CB algorithms, specially when searching dense networks from data sets with relatively few instances. HPC (Algorithm 1) can be viewed as an ensemble method for combining many weak PC learners in an attempt to produce a stronger PC learner. HPC is based on three subroutines: *Data-Efficient Parents and Children Superset* (DE-PCS), *Data-Efficient Spouses Superset* (DE-SPS), and *Interleaved Incremental Association Parents and Children* (Inter-IAPC), a weak PC learner based on Inter-IAMB [19] that requires little computation. HPC may be thought of as a way to compensate for the large number of false negatives, at the output of the weak PC learner, by performing extra computations.

HPC receives a target node  $T$ , a data set  $\mathcal{D}$  and a set of variables  $\mathbf{U}$  as input and returns an estimation of  $\mathbf{PC}_T$ . It is hybrid in that it combines the benefits of incremental and divide-and-conquer methods. The procedure starts by extracting a superset  $\mathbf{PCS}_T$  of  $\mathbf{PC}_T$  (line 1) and a superset  $\mathbf{SPS}_T$  of  $\mathbf{SP}_T$

(line 2) with a severe restriction on the maximum conditioning size ( $\mathbf{Z} \leq 2$ ) in order to significantly increase the reliability of the tests. A first candidate PC set is then obtained by running the weak PC learner on  $\mathbf{PCS}_T \cup \mathbf{SPS}_T$  (line 3). The key idea is the decentralized search at lines 4-8 that includes, in the candidate PC set, all variables in the superset  $\mathbf{PCS}_T \cup \mathbf{SPS}_T$  that have  $T$  in their vicinity. Note that, in theory,  $X$  is in the output of  $\text{Inter-IAPC}(Y)$  if and only if  $Y$  is in the output of  $\text{Inter-IAPC}(X)$ . However, in practice, this may not always be true, particularly when working in high-dimensional domains with relatively few instances. By loosening the criteria by which two nodes are said adjacent, the effective restrictions on the size of the neighborhood are now far less severe. The decentralized search has significant impact on the accuracy of HPC. It enables the algorithm to handle large neighborhoods while still being correct under the faithfulness condition. The proof of HPC’s correctness is provided in Appendix [A](#).

---

**Algorithm 1.** *HPC*


---

**Require:**  $T$ : target;  $\mathcal{D}$ : data set;  $\mathbf{U}$ : the set of variables  
**Ensure:**  $\mathbf{PC}_T$ : Parents and Children of  $T$

```

1: [ $\mathbf{PCS}_T, \mathbf{dSep}$ ]  $\leftarrow$  DE-PCS( $T, \mathcal{D}$ )
2:  $\mathbf{SPS}_T \leftarrow$  DE-SPS( $T, \mathcal{D}, \mathbf{PCS}_T, \mathbf{dSep}$ )
3:  $\mathbf{PC}_T \leftarrow$  Inter-IAPC( $T, \mathcal{D}(T \cup \mathbf{PCS}_T \cup \mathbf{SPS}_T)$ )
4: for all  $X \in \mathbf{PCS}_T \setminus \mathbf{PC}_T$  do
5:   if  $T \in$  Inter-IAPC( $X, \mathcal{D}(T \cup \mathbf{PCS}_T \cup \mathbf{SPS}_T)$ ) then
6:      $\mathbf{PC}_T \leftarrow \mathbf{PC}_T \cup X$ 
7:   end if
8: end for

```

---

We now discuss the subroutines in more detail.  $\text{Inter-IAPC}$  (Algorithm [2](#)) is a fast incremental method that receives a data set  $\mathcal{D}$  and a target node  $T$  as its input and promptly returns a rough estimation of  $\mathbf{PC}_T$ , hence the term “weak” PC learner.  $\text{Inter-IAPC}$  is an straightforward extension of the algorithm  $\text{Inter-IAMB}$  [\[19\]](#). Notice that neither  $\text{MMPC}$  [\[20\]](#) nor  $\text{GetPC}$  [\[11\]](#) should be used to implement this weak PC learner. The reason is that any break of symmetry of the PC relation in the output of these algorithms is an indication of a false negative member; the decentralized search would not aid in reducing the number of false negative variables at all.  $\text{Inter-IAPC}$  starts with a two-phase approach to infer  $\mathbf{MB}_T$ , that is, the Markov boundary of  $T$ . A growing phase attempts to iteratively add the best candidate variables to  $\mathbf{MB}_T$ , followed by a shrinking phase that attempts to remove as many irrelevant variables as possible, that is, the false negatives in the current set  $\mathbf{MB}_T$ . The function  $\text{dep}(T, X | \mathbf{MB}_T)$  at line 4 returns a statistical estimation of the association between  $T$  and  $X$  given the current set  $\mathbf{MB}_T$ . The shrinking phase is interleaved with the growing phase. Interleaving the two phases allows to eliminate as soon as possible some of the false positives in the current Markov blanket as the algorithm progresses during the Markov boundary search.  $\mathbf{PC}_T$  is obtained by removing the spouses of the

target from the final  $\mathbf{MB}_T$  (lines 14-19). Inter-IAPC is very fast and sound (the proof of soundness is provided in Appendix [A](#)), despite its *data-inefficiency* in practice. The decentralized search in HPC is an attempt to alleviate this problem as discussed earlier.

---

**Algorithm 2.** *Inter-IAPC*


---

**Require:**  $T$ : target;  $D$ : data set;  $\mathbf{U}$ : set of variables;

**Ensure:**  $\mathbf{PC}_T$ : Parents and children of  $T$ ;

```

1:  $\mathbf{MB}_T \leftarrow \emptyset$ 
2: repeat
3:   * Add true positives to  $\mathbf{MB}_T$ 
4:    $Y \leftarrow \operatorname{argmax}_{X \in (\mathbf{U} \setminus \mathbf{MB}_T \setminus T)} \operatorname{dep}(T, X | \mathbf{MB}_T)$ 
5:   if  $T \not\perp Y | \mathbf{MB}_T$  then
6:      $\mathbf{MB}_T \leftarrow \mathbf{MB}_T \cup Y$ 
7:   end if

   * Remove false positives from  $\mathbf{MB}_T$ 
8:   for all  $X \in \mathbf{MB}_T$  do
9:     if  $T \perp X | (\mathbf{MB}_T \setminus X)$  then
10:       $\mathbf{MB}_T \leftarrow \mathbf{MB}_T \setminus X$ 
11:    end if
12:  end for
13: until  $\mathbf{MB}_T$  has not changed

   * Remove spouses of  $T$  from  $\mathbf{MB}_T$ 
14:  $\mathbf{PC}_T \leftarrow \mathbf{MB}_T$ 
15: for all  $X \in \mathbf{MB}_T$  do
16:   if  $\exists \mathbf{Z} \subseteq (\mathbf{MB}_T \setminus X)$  such that  $T \perp X | \mathbf{Z}$  then
17:      $\mathbf{PC}_T \leftarrow \mathbf{PC}_T \setminus X$ 
18:   end if
19: end for

```

---

The subroutines DE-PCS (Algorithm [3](#)) and DE-SPS (Algorithm [4](#)) search a superset of  $\mathbf{PC}_T$  and  $\mathbf{SP}_T$  respectively with a severe restriction on the maximum conditioning size ( $|\mathbf{Z}| \leq 1$  in DE-PCS and  $|\mathbf{Z}| \leq 2$  in DE-SPS) in order to significantly increase the reliability of the tests. The variable filtering has two advantages : i) it allows HPC to scale to hundreds of thousands of variables by restricting the search to a subset of relevant variables, and ii) it eliminates many ADRs that produce many false negative errors in the output of the algorithm, as explained in the last section. DE-SPS works in two steps. First, a growing phase (lines 4-8) adds the variables that are d-separated from the target but still remain associated with the target when conditioned on another variable from  $\mathbf{PCS}_T$ . The shrinking phase (lines 9-16) discards irrelevant variables that are ancestors or descendants of a target’s spouse. Pruning such irrelevant variables speeds up HPC.



---

**Algorithm 3.** *DE-PCS*

---

**Require:**  $T$ : target;  $\mathcal{D}$ : data set;  $\mathbf{U}$ : set of variables;**Ensure:**  $\mathbf{PCS}_T$ : parents and children superset of  $T$ ;  $\mathbf{dSep}$ : d-separating sets;**Phase I:** *Remove  $X$  if  $T \perp X$* 

```

1:  $\mathbf{PCS}_T \leftarrow \mathbf{U} \setminus T$ 
2: for all  $X \in \mathbf{PCS}_T$  do
3:   if  $(T \perp X)$  then
4:      $\mathbf{PCS}_T \leftarrow \mathbf{PCS}_T \setminus X$ 
5:      $\mathbf{dSep}(X) \leftarrow \emptyset$ 
6:   end if
7: end for

```

**Phase II:** *Remove  $X$  if  $T \perp X|Y$* 

```

8: for all  $X \in \mathbf{PCS}_T$  do
9:   for all  $Y \in \mathbf{PCS}_T \setminus X$  do
10:    if  $(T \perp X | Y)$  then
11:       $\mathbf{PCS}_T \leftarrow \mathbf{PCS}_T \setminus X$ 
12:       $\mathbf{dSep}(X) \leftarrow Y$ 
13:      break loop FOR
14:    end if
15:   end for
16: end for

```

---



---

**Algorithm 4.** *DE-SPS*

---

**Require:**  $T$ : target;  $\mathcal{D}$ : data set;  $\mathbf{U}$ : the set of variables;  $\mathbf{PCS}_T$ : parents and children superset of  $T$ ;  $\mathbf{dSep}$ : d-separating sets;**Ensure:**  $\mathbf{SPS}_T$ : Superset of the spouses of  $T$ ;

```

1:  $\mathbf{SPS}_T \leftarrow \emptyset$ 
2: for all  $X \in \mathbf{PCS}_T$  do
3:    $\mathbf{SPS}_T^X \leftarrow \emptyset$ 
4:   for all  $Y \in \mathbf{U} \setminus \{T \cup \mathbf{PCS}_T\}$  do
5:     if  $(T \not\perp Y | \mathbf{dSep}(Y) \cup X)$  then
6:        $\mathbf{SPS}_T^X \leftarrow \mathbf{SPS}_T^X \cup Y$ 
7:     end if
8:   end for
9:   for all  $Y \in \mathbf{SPS}_T^X$  do
10:    for all  $Z \in \mathbf{SPS}_T^X \setminus Y$  do
11:      if  $(T \perp Y | X \cup Z)$  then
12:         $\mathbf{SPS}_T^X \leftarrow \mathbf{SPS}_T^X \setminus Y$ 
13:        break loop FOR
14:      end if
15:    end for
16:   end for
17:    $\mathbf{SPS}_T \leftarrow \mathbf{SPS}_T \cup \mathbf{SPS}_T^X$ 
18: end for

```

---

## 5 Experimental Validation

In this section, we assess the accuracy and the scalability of HPC through several empirical experiments. We first compared HPC with two distinguished CB algorithm proposals that appeared recently in the literature, namely MMPC<sup>2</sup> (Max-Min Parents and Children) [20] and GetPC<sup>3</sup> [11]. Both algorithms are correct under the faithfulness condition and are also scalable to high dimensional data sets. Only the authors' own implementations were used for the empirical experiments. HPC was also compared to Inter-IAPC (Algorithm 2), the weak PC learner. The same critical p-value ( $\alpha = 0.05$ ) was used to make fair comparisons. We also compared HPC with two well known search-and-score global network learning algorithms, namely Greedy Equivalent Search (GES) [6] and SCA (Sparse Candidate Algorithm) [7]. For SCA, we used the implementation available in the Causal Explorer system [1]. The code uses the Bayesian scoring heuristic and an equivalent sample size of 10. The maximum allowed size for the candidate parents' sets  $k$  is set to  $k=10$ . For GES, we used the command-line tool in the WinMine Toolkit<sup>4</sup>. All the data sets used for the empirical experiments presented in this section were sampled from well-known BNs that have been previously used as benchmarks for BN learning algorithms, namely *Asia*, *Alarm*, *Barley*, *Child*, *Genes*, *Insulin*, *Insurance*, *Link*, *Mildew*, *Pigs* and *Carpa* (see [20] for details). Three samples sizes have been considered: 200, 500 and 1500. We do not claim that those data sets resemble real-world problems, however, they make it possible to compare the outputs of the algorithms with the known  $\mathbf{PC}_T$  set of the BNs sampled.

### 5.1 Accuracy

Each CB algorithm was run 10 times on each node of each benchmark. The variables in the output of the algorithms were compared against the true neighbors. As GES and SCA are global network learning algorithms, they were run 10 times on each benchmark. To evaluate the accuracy, we combined precision (i.e., the number of true positives in the output divided by the number of nodes in the output) and recall (i.e., the number of true positives divided the true size of the PC) as  $\sqrt{(1 - \textit{precision})^2 + (1 - \textit{recall})^2}$ , to measure the Euclidean distance from perfect precision and recall, as proposed in [11]. Figure 3 plots the Euclidean distance, averaged over all nodes, as a function of the PC size. The advantage of *HPC* against the other algorithms is clearly noticeable.

### 5.2 Scalability

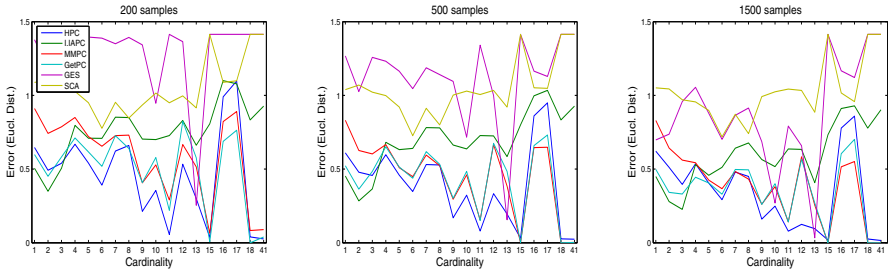
We now provide an empirical evaluation of the scalability of HPC on real and artificial high-dimensional data sets. First, the five benchmarks were replicated

<sup>2</sup> <http://discover1.mc.vanderbilt.edu/discover/public>

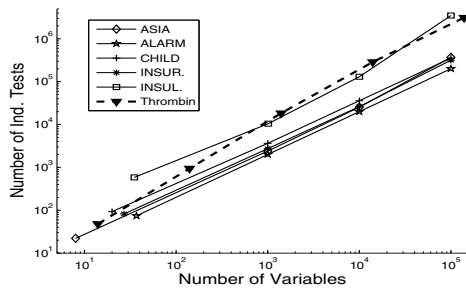
<sup>3</sup> <http://www.ida.liu.se/~jospe>

<sup>4</sup> WinMine Toolkit can be downloaded at

<http://research.microsoft.com/en-us/um/people/dmax/WinMine/Download.html>



**Fig. 3.** Euclidean distance from perfect precision and recall versus the cardinality of the nodes degree. All algorithms were run 10 times on each variable of each of the 11 benchmark.



**Fig. 4.** Number of calls to the conditional independence tests on Thrombin database (dotted line) and on synthetic data sets (plain lines) from 5 benchmarks replicated several times, versus the number of variables.

several times (up to 100 000 variables) to increase artificially the number of variables. Each network is obtained by tiling several copies of the original network. The tiling is performed by maintaining the structural and probabilistic properties of the original network in the tiled network. The learning task is the same as in the previous subsection, but instead of the accuracy, we report in Figure 4 (in plain lines) the average number of conditional independence tests that were conducted for each tiled network (in log-log scale) as a function of the number of variables. The average value is estimated over 100 data sets with only 100 instances for each network size. The number of calls to the statistical test was empirically on the order  $O(n^{1.09})$  where  $n$  is the number of variables, regardless of the size of the neighborhood of the target in the corresponding BN.

The scalability claims were based on taking a basic graph and creating a larger graph by tiling the smaller graph repeatedly. One could claim that the empirical number of  $O(n^{1.09})$  may be optimistic as it hinges on our tiling construction. Running HPC on increasing parts of a real-world database would lend more validity to our statement. To this aim, we considered the Thrombin database which was provided by DuPont Pharmaceuticals for KDD Cup 2001. It is exemplary of a real drug design [5]. The training set contains 1909 instances characterized by

139,351 binary features. Each instance represents a drug compound tested for its ability to bind to a target site on Thrombin, a key receptor in blood clotting. Each compound is labeled with one out of two classes, either it binds or not. The task is again to learn the PC set of the target variable from 1909 given compounds (the learning data). We picked at random 0.01%, 0.1%, 1%, 10% and the full set of variables including the target and used these subsets to learn the target neighborhood. The overall process was repeated 100 times by bootstrap. As done before we measured the average number of conditional independence tests as a function of the number of variables. Here again, the number of calls to the test varies by a multiplicative factor when the size of the neighborhood is varied. The results show that the number of calls to the statistical test is empirically on the order  $O(n^{1.21})$ , that is, slightly superior to the behavior on synthetic data. Nonetheless, the almost-linear time complexity of HPC shows promise for a variety of applications involving hundreds of thousands of variables.

## 6 Conclusion

We discussed a novel scalable algorithm for local BN structure learning, called *Hybrid Parents and Children* (HPC). Extensive simulations have been conducted on public synthetic and real-world data sets of various sample sizes to assess its accuracy and scalability. Significant improvements in accuracy were obtained in all experiments compared to state-of-the-art algorithms. The *effective* complexity is only  $O(n^{1.09})$  in practice on the five benchmarks that we considered and  $O(n^{1.21})$  on the real-world Thrombin database. The moderate time complexity of HPC shows great promise for a variety of applications involving hundreds of thousands of variables. The HPC algorithm was designed for detecting the parents and children of a vertex in a graphical model. Learning the parents and children of a target is a key routine used in constraint-based BN structure learning algorithms. HPC can be applied iteratively to find the Markov boundary of a target for classification purposes, the local or the whole skeleton of the BN although this is not discussed here for the sake of conciseness. Regarding causal inference, HPC assumes no hidden common causes, which is pretty unrealistic. On the other hand, it could easily be modified to detect hidden common causes [3, 13]. These adaptations of HPC are left for future work.

## References

1. Aliferis, C., Tsamardinos, I., Statnikov, A., Brown, L.: Causal explorer: A causal probabilistic network learning toolkit for biomedical discovery. In: Proceedings of the International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences, METMBS, Las Vegas, Nevada, USA, pp. 23–26. CSREA Press (2003)
2. Aussem, A., de Morais, S.R., Corbex, M.: Nasopharyngeal carcinoma data analysis with a novel Bayesian network skeleton learning. In: Bellazzi, R., Abu-Hanna, A., Hunter, J. (eds.) AIME 2007. LNCS (LNAI), vol. 4594, pp. 326–330. Springer, Heidelberg (2007)

3. Brown, L.E., Tsamardinos, I.: A strategy for making predictions under manipulation. In: *JMLR: Workshop and Conference Proceedings*, vol. 3, pp. 35–52 (2008)
4. Cawley, G.: Causal and non-causal feature selection for ridge regression. In: *JMLR: Workshop and Conference Proceedings*, vol. 3 (2008)
5. Cheng, J., Hatzis, C., Hayashi, H., Krogel, M.A., Morishita, S., Page, D., Sese, J.: KDD Cup 2001 Report. In: *ACM SIGKDD Explorations*, pp. 1–18 (2002)
6. Chickering, D.: Learning equivalence classes of bayesian-network structures. *Machine Learning* 2, 445–498 (2002)
7. Friedman, N.L., Nachman, I., Pe’er, D.: Learning bayesian network structure from massive datasets: the “sparse candidate” algorithm. In: Laskey, K.B., Prade, H. (eds.) *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pp. 21–30. Morgan Kaufmann Publishers, San Francisco (1999)
8. Fu, S., Desmarais, M.: Tradeoff analysis of different Markov blanket local learning approaches. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) *PAKDD 2008. LNCS (LNAI)*, vol. 5012, pp. 562–571. Springer, Heidelberg (2008)
9. Guyon, I., Aliferis, C., Cooper, G., Elisseeff, A., Pellet, J.P., Statnikov, P.A.: Design and analysis of the causation and prediction challenge. In: *JMLR: Workshop and Conference Proceedings*, vol. 1, pp. 1–16. MIT Press, Boston (2008)
10. Neapolitan, R.E.: *Learning Bayesian Networks*. Pearson Prentice Hall, Upper Saddle River (2004)
11. Peña, J.M., Nilsson, R., Björkegren, J., Tegnér, J.: Towards scalable and data efficient learning of Markov boundaries. *International Journal of Approximate Reasoning* 45(2), 211–232 (2007)
12. Pearl, J.: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco (1988)
13. Pearl, J.: *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge (2000)
14. Peña, J.: Learning gaussian graphical models of gene networks with false discovery rate control. In: Marchiori, E., Moore, J.H. (eds.) *EvoBIO 2008. LNCS*, vol. 4973, pp. 165–176. Springer, Heidelberg (2008)
15. Rodrigues de Morais, S., Aussem, A.: A novel scalable and data efficient feature subset selection algorithm. In: Daelemans, W., Goethals, B., Morik, K. (eds.) *ECML PKDD 2008, Part II. LNCS (LNAI)*, vol. 5212, pp. 298–312. Springer, Heidelberg (2008)
16. Rodrigues de Morais, S., Aussem, A.: A novel Markov boundary based feature subset selection algorithm. *Neurocomputing* 73, 578–584 (2010)
17. Rodrigues de Morais, S., Aussem, A., Corbex, M.: Handling almost-deterministic relationships in constraint-based Bayesian network discovery: Application to cancer risk factor identification. In: *16th European Symposium on Artificial Neural Networks ESANN’08*, pp. 101–106 (2008)
18. Steck, H.: Learning the Bayesian network structure: Dirichlet prior vs data. In: *Conference on Uncertainty in Artificial Intelligence UAI’08*, pp. 511–518 (2008)
19. Tsamardinos, I., Aliferis, C.F., Statnikov, A.R.: Algorithms for large scale Markov blanket discovery. In: *Florida Artificial Intelligence Research Society Conference FLAIRS’03*, pp. 376–381 (2003)
20. Tsamardinos, I., Brown, L.E., Aliferis, C.F.: The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning* 65(1), 31–78 (2006)
21. Tsamardinos, I., Brown, L.E.: Bounding the false discovery rate in local Bayesian network learning. In: *Proceedings AAAI National Conference on AI AAAI’08*, pp. 1100–1105 (2008)

## A Proof of Correctness

A structure learning algorithm from data is said to be correct (or sound) if it returns the correct DAG pattern (or a DAG in the correct equivalence class) under the assumptions that the independence tests are reliable and that the learning data set is a sample from a distribution  $P$  faithful to a DAG  $\mathcal{G}$ . The (ideal) assumption that the independence tests are reliable means that they decide (in)dependence if and only if the (in)dependence holds in  $P$ . Consequently, a parents and children learning algorithm is said to be correct (or sound) when under the assumptions that the independence tests are reliable and that the learning data set is a sample from a distribution  $P$  faithful to a DAG  $\mathcal{G}$ , the algorithm returns the correct set of parents and children of the target, that is, the target direct neighborhood. Correctness is a desirable asymptotic property though the underlying assumptions may not hold in practice. In general, we would want an edge to mean a direct dependence. Several definitions and intermediate theorems are required before we demonstrate *HPC*'s correctness under faithfulness condition.

**Definition 2.** A Markov blanket  $\mathbf{M}_T$  of  $T$  is any set of variables such that  $T$  is conditionally independent of all the remaining variables given  $\mathbf{M}_T$ . A Markov boundary,  $\mathbf{MB}_T$ , of  $T$  is any Markov blanket such that none of its proper subsets is a Markov blanket of  $T$ .

**Theorem 2.** Suppose  $\langle \mathcal{G}, P \rangle$  satisfies the faithfulness condition. Then for each variable  $X$ , the set of parents, children of  $X$ , and parents of children (spouses) of  $X$  is its unique Markov boundary.

A proof can be found in [10]. Indeed, as  $\mathbf{PCS}_T \cup \mathbf{SPS}_T$  is a subset of  $\mathbf{U}$ , a difficulty arises: a marginal distribution  $P^{\mathbf{V}}$  of  $\mathbf{V} \subset \mathbf{U}$  may not satisfy the faithfulness condition with any DAG even if  $P^{\mathbf{U}}$  does. This is an example of embedded faithfulness, which is defined as follow:

**Definition 3.** Let  $P^{\mathbf{V}}$  be a distribution of the variables in  $\mathbf{V}$  where  $\mathbf{V} \subset \mathbf{U}$  and let  $\mathcal{G} = \langle \mathbf{U}, \mathbf{E} \rangle$  be a DAG.  $\langle \mathcal{G}, P^{\mathbf{V}} \rangle$  satisfies the embedded faithfulness condition if  $\mathcal{G}$  entails all and only the conditional independencies in  $P^{\mathbf{V}}$ , for subsets including only elements of  $\mathbf{V}$ .

We obtain embedded faithfulness by taking the marginal of a faithful distribution as shown by the next theorem:

**Theorem 3.** Let  $P^{\mathbf{U}}$  be a joint probability of the variables in  $\mathbf{U}$  with  $\mathbf{V} \subseteq \mathbf{U}$  and  $\mathcal{G} = \langle \mathbf{U}, \mathbf{E} \rangle$ . If  $\langle \mathcal{G}, P^{\mathbf{U}} \rangle$  satisfies the faithfulness condition and  $P^{\mathbf{V}}$  is the marginal distribution of  $\mathbf{V}$ , then  $\langle \mathcal{G}, P^{\mathbf{V}} \rangle$  satisfies the embedded faithful condition.

The proof can be found in [10]. Note that not every distribution does admit an embedded faithful representation. This property is useful to prove the correctness of *HPC* under the faithfulness condition. Let  $\mathbf{PC}_X^{\mathbf{U}}$  denote the variables  $Y \in \mathbf{U}$

so that there is no set  $\mathbf{Z} \subseteq \mathbf{U} \setminus \{X, Y\}$  such that  $X \perp_P Y | \mathbf{Z}$ . If  $\langle \mathcal{G}, P^{\mathbf{U}} \rangle$  satisfies the faithfulness condition,  $\mathbf{PC}_X^{\mathbf{U}}$  are the parents and children of  $X$  in  $\mathbf{U}$ . In any case,  $\mathbf{PC}_X^{\mathbf{U}}$  is the unique set of the variables  $Y_i$  that remain dependent on  $X$  conditioned on any set  $\mathbf{Z} \subseteq \mathbf{U} \setminus \{X, Y_i\}$ .

**Theorem 4.** *Let  $\mathbf{U}$  be a set of random variables and  $\mathcal{G} = \langle \mathbf{U}, \mathbf{E} \rangle$  be a DAG. If  $\langle \mathcal{G}, P^{\mathbf{U}} \rangle$  satisfies the faithfulness condition, then every target  $T$  admits a unique Markov boundary  $\mathbf{MB}_T^{\mathbf{U}}$ . Moreover, for all  $\mathbf{V}$  such that  $\mathbf{MB}_T^{\mathbf{U}} \subseteq \mathbf{V} \subseteq \mathbf{U}$ ,  $T$  admits a unique Markov boundary over  $\mathbf{V}$  and  $\mathbf{MB}_T^{\mathbf{V}} = \mathbf{MB}_T^{\mathbf{U}}$ .*

*Proof:* If  $\mathbf{MB}_T^{\mathbf{U}}$  is the Markov boundary of  $T$  in  $\mathbf{U}$ , then  $T$  is independent of all variable  $Y \in (\mathbf{V} \setminus (\mathbf{MB}_T^{\mathbf{U}} \cup T))$  conditionally on  $\mathbf{MB}_T^{\mathbf{U}}$ , then  $\mathbf{MB}_T^{\mathbf{U}}$  is a Markov blanket in  $\mathbf{V}$ . Moreover, none of the proper subsets of  $\mathbf{MB}_T^{\mathbf{U}}$  is a Markov blanket of  $T$  in  $\mathbf{V}$ , so  $\mathbf{MB}_T^{\mathbf{U}}$  is also a Markov boundary of  $T$  in  $\mathbf{V}$ . So if it is not the unique MB for  $T$  in  $\mathbf{V}$  there exists some other set  $\mathbf{S}_T$  not equal to  $\mathbf{MB}_T^{\mathbf{U}}$ , which is a MB of  $T$  in  $\mathbf{V}$ . Since  $\mathbf{MB}_T^{\mathbf{U}} \neq \mathbf{S}_T$  and  $\mathbf{MB}_T^{\mathbf{U}}$  cannot be a subset of  $\mathbf{S}_T$ , there is some  $X \in \mathbf{MB}_T^{\mathbf{U}}$  such that  $X \notin \mathbf{S}_T$ . Since  $\mathbf{S}_T$  is a MB for  $T$ , we would have  $T \perp_P X | \mathbf{S}_T$ . If  $X$  is a parent or child of  $T$ , we would not have  $T \perp_{\mathcal{G}} X | \mathbf{S}_T$  which means we would have a conditional independence that is not entailed by d-separation in  $\mathcal{G}$ , which contradicts the faithfulness condition. If  $X$  is a parent of a child of  $T$  in  $\mathcal{G}$ , let  $Y$  be their common child in  $\mathbf{U}$ . If  $Y \in \mathbf{S}_T$  we again would not have  $T \perp_{\mathcal{G}} X | \mathbf{S}_T$ . If  $Y \notin \mathbf{S}_T$  we would have  $T \perp_P Y | \mathbf{S}_T$  because  $\mathbf{S}_T$  is a MB of  $T$  in  $\mathbf{V}$  but we do not have  $T \perp_{\mathcal{G}} Y | \mathbf{S}_T$  because  $T$  is a parent of  $Y$  in  $\mathcal{G}$ . So again we would have a conditional independence which is not a d-separation in  $\mathcal{G}$ . This proves that there can not be such set  $\mathbf{S}_T$ .  $\square$

**Theorem 5.** *Let  $\mathbf{U}$  be a set of random variables and  $T$  a target variable. Let  $\mathcal{G} = \langle \mathbf{U}, \mathbf{E} \rangle$  be a DAG such that  $\langle \mathcal{G}, P^{\mathbf{U}} \rangle$  satisfies the faithfulness condition. Let  $\mathbf{V}$  be such that  $\mathbf{MB}_T^{\mathbf{U}} \subseteq \mathbf{V} \subseteq \mathbf{U}$  then,  $\mathbf{PC}_T^{\mathbf{V}} = \mathbf{PC}_T^{\mathbf{U}}$ .*

*Proof:* Clearly  $\mathbf{PC}_T^{\mathbf{U}} \subseteq \mathbf{PC}_T^{\mathbf{V}}$  as  $\mathbf{MB}_T^{\mathbf{U}} \subseteq \mathbf{V} \subseteq \mathbf{U}$ . If  $X \in \mathbf{PC}_T^{\mathbf{V}}$  and  $X \notin \mathbf{PC}_T^{\mathbf{U}}$ ,  $\exists \mathbf{Z} \subseteq \mathbf{MB}_T^{\mathbf{U}} \setminus X$  such that  $T \perp_P X | \mathbf{Z}$  because all non adjacent nodes may be d-separated in  $\mathcal{G}$  by a subset of its Markov boundary. As  $\mathbf{MB}_T^{\mathbf{U}} = \mathbf{MB}_T^{\mathbf{V}}$  owing to Theorem 4, so  $X$  and  $T$  can be d-separated in  $\mathbf{V} \setminus \{X, T\}$ . Therefore,  $X$  cannot be adjacent to  $T$  in  $\mathbf{V}$ .  $\square$

**Theorem 6.** *Let  $\mathbf{U}$  be a set of random variables and  $T$  a target variable. Let  $\mathcal{G} = \langle \mathbf{U}, \mathbf{E} \rangle$  be a DAG such that  $\langle \mathcal{G}, P^{\mathbf{U}} \rangle$  satisfies the faithfulness condition. Let  $\mathbf{V}$  be such that  $\mathbf{MB}_T^{\mathbf{U}} \subseteq \mathbf{V} \subseteq \mathbf{U}$ . Under the assumption that the independence tests are reliable, *Inter-IAPC*( $T, \mathcal{D}, \mathbf{V}$ ) returns  $\mathbf{PC}_T^{\mathbf{U}}$ . Moreover, let  $X \in \mathbf{V} \setminus T$ , then  $\forall T \in \mathbf{V}$ ,  $T$  is in the output of *Inter-IAPC*( $X, \mathcal{D}, \mathbf{V}$ ) iff  $X \in \mathbf{PC}_T^{\mathbf{U}}$ .*

*Proof:* We prove first that *Inter-IAPC*( $T, \mathcal{D}, \mathbf{V}$ ) returns  $\mathbf{PC}_T^{\mathbf{U}}$ . In lines 1-13, *Inter-IAPC* seeks a minimal set  $\mathbf{S}_T \subseteq \mathbf{V} \setminus T$  that renders  $\mathbf{V} \setminus \mathbf{S}_T$  independent of  $T$  conditionally on  $\mathbf{S}_T$ . This set is unique owing to Theorem 4, therefore  $\mathbf{S}_T = \mathbf{MB}_T^{\mathbf{V}} = \mathbf{MB}_T^{\mathbf{U}}$ . In the backward phase, *Inter-IAPC* removes the variables  $X \in \mathbf{MB}_T^{\mathbf{V}}$  such that  $\exists \mathbf{Z} \subseteq (\mathbf{MB}_T^{\mathbf{V}} \setminus X)$  for which  $T \perp X | \mathbf{Z}$ . These variables are the spouses of  $T$  in  $\mathcal{G}$ , so *Inter-IAPC*( $T, \mathcal{D}, \mathbf{V}$ ) returns  $\mathbf{PC}_T^{\mathbf{U}}$ . Now, if  $X \notin \mathbf{PC}_T^{\mathbf{U}}$

then  $X \notin \mathbf{PC}_T^{\mathbf{V}}$  owing to Theorem 5. So there is a set  $\mathbf{Z} \subseteq \mathbf{V} \setminus \{X, Y\}$  such that  $T \perp X \mid \mathbf{Z}$ . Therefore,  $X$  cannot be in the output of  $\text{Inter-IAPC}(T, \mathcal{D}, \mathbf{V})$ , nor  $T$  can be in the output of  $\text{Inter-IAPC}(X, \mathcal{D}, \mathbf{V})$ .  $\square$

**Theorem 7.** *Under the assumptions that the independence tests are reliable and that the data set is a sample from a probability distribution  $P^U$  faithful to a DAG  $\mathcal{G}$ , then  $\text{HPC}(T, \mathcal{D}, \mathbf{U})$  returns  $\mathbf{PC}_T^{\mathbf{U}}$ .*

*Proof.* Let  $\mathbf{V} = (\mathbf{PCS} \cup \mathbf{SPS})$ , then  $\mathbf{V}$  is a superset of  $\mathbf{MB}_T^{\mathbf{U}}$ . Based on what is stated by Theorem 4 we know that  $\mathbf{MB}_T^{\mathbf{V}} = \mathbf{MB}_T^{\mathbf{U}}$ . If  $T$  is in the output of  $\text{Inter-IAPC}(X, \mathbf{V}, \mathcal{D})$  then  $X$  should be in the output of  $\text{Inter-IAPC}(T, \mathbf{V}, \mathcal{D})$  owing to Theorem 6. So  $\text{HPC}$  returns  $\mathbf{PC}_T^{\mathbf{U}}$ .  $\square$



# Selecting Information Diffusion Models over Social Networks for Behavioral Analysis

Kazumi Saito<sup>1</sup>, Masahiro Kimura<sup>2</sup>, Kouzou Ohara<sup>3</sup>, and Hiroshi Motoda<sup>4</sup>

<sup>1</sup> School of Administration and Informatics, University of Shizuoka  
52-1 Yada, Suruga-ku, Shizuoka 422-8526, Japan  
k-saito@u-shizuoka-ken.ac.jp

<sup>2</sup> Department of Electronics and Informatics, Ryukoku University  
Otsu 520-2194, Japan  
kimura@rins.ryukoku.ac.jp

<sup>3</sup> Department of Integrated Information Technology, Aoyama Gakuin University  
Kanagawa 229-8558, Japan  
ohara@it.aoyama.ac.jp

<sup>4</sup> Institute of Scientific and Industrial Research, Osaka University  
8-1 Mihogaoka, Ibaraki, Osaka 567-0047, Japan  
motoda@ar.sanken.osaka-u.ac.jp

**Abstract.** We investigate how well different information diffusion models can explain observation data by learning their parameters and discuss which model is better suited to which topic. We use two models (AsIC, AsLT), each of which is an extension of the well known Independent Cascade (IC) and Linear Threshold (LT) models and incorporates asynchronous time delay. The model parameters are learned by maximizing the likelihood of observation, and the model selection is performed by choosing the one with better predictive accuracy. We first show by using four real networks that the proposed learning algorithm correctly learns the model parameters both accurately and stably, and the proposed selection method identifies the correct diffusion model from which the data are generated. We next apply these methods to behavioral analysis of topic propagation using the real blog propagation data, and show that although the relative propagation speed of topics that are derived from the learned parameter values is rather insensitive to the model selected, there is a clear indication as to which topic better follows which model. The correspondence between the topic and the model selected is well interpretable.

## 1 Introduction

The growth of Internet has enabled to form various kinds of large-scale social networks, through which a variety of information including innovation, hot topics and even malicious rumors can be propagated in the form of so-called "word-of-mouth" communications. Social networks are now recognized as an important medium for the spread of information, and a considerable number of studies have been made [1–5]. Widely used information diffusion models in these studies are the *independent cascade (IC)* [6–8] and the *linear threshold (LT)* [9, 10]. They have been used to solve such problems as the *influence maximization problem* [7, 11].

These two models focus on different information diffusion aspects. The IC model is sender-centered and each active node *independently* influences its inactive neighbors with given diffusion probabilities. The LT model is receiver-centered and a node is influenced by its active neighbors if their total weight exceeds the threshold for the node. Which model is more appropriate depends on the situation and selecting the appropriate one is not easy. First of all, we need to know how different model behaves differently and how well or badly explain the observation data. Both models have parameters that need be specified in advance: diffusion probabilities for the IC model, and weights for the LT model. However, their true values are not known in practice. This poses yet another problem of estimating them from a set of information diffusion results that are observed as time-sequences of influenced (activated) nodes.

This falls in a well defined parameter estimation problem in machine learning framework. Given a generative model with some parameters and the observed data, it is possible to calculate the likelihood that the data are generated and the parameters can be estimated by maximizing the likelihood. This approach has a thorough theoretical background. In general, the way the parameters are estimated depends on how the generative model is given. To the best of our knowledge, we are the first to follow this line of research. We addressed this problem for the IC model [12] and its variant that incorporates asynchronous time delay (referred to as the AsIC model) [13]. Gruhl et.al. also challenged the same problem of estimating the parameters and proposed an EM-like algorithm, but they did not formalize the likelihood and it is not clear what is being optimized in deriving the parameter update formulas. Goyal et.al attacked this problem from a different angle [14]. They employed a variant of the LT model and estimated the parameter values by four different methods, all of which are directly computed from the frequency of the events in the observed data. Their approach is efficient, but it is more likely ad hoc and lacks in theoretical evidence. Bakshy et.al [15] addressed the problem of diffusion of user-created content (asset) and used the maximum likelihood method to estimate the rate of asset adoption. However, they only modeled the rate of adoption and did not consider the diffusion model itself. Their focus is on data analysis.

In this paper, we first propose a method of learning the parameter values of a variant of the LT model that incorporates asynchronous time delay, similarly to the AsIC model, under the maximum likelihood framework. We refer to this diffusion model as the AsLT model. The model is similar to the one used in [14] but different in that we explicitly model the delay of node activation after the activation condition has been satisfied. Next we propose a method of model selection based on the predictive accuracy, using the two models: AsIC and AsLT.

It is indispensable to be able to cope with asynchronous time delay to do realistic analyses of information diffusion because, in the real world, information propagates along the continuous time axis, and time-delays can occur during the propagation asynchronously. In fact, the time stamps of the observed data are not equally spaced. Thus, the proposed learning method has to estimate not only the weight parameters but also the time-delay parameters from the observed data. Incorporating time-delay makes the time-sequence observation data structural, which makes the analyses of diffusion process difficult because there is no way of knowing which node has activated which other node from the observation data sequence. Knowing the optimal parameter values does

not mean that the observation follows the model. We have to decide which model better explains the observation. We solve this problem by comparing the predictive accuracy of each model. We use a variant of hold-out method applied to a set of sequential data, which is similar to the leave-one-out method applied to a multiple time sequence data. Extensive experiments have been performed to evaluate the effectiveness of the proposed method using both artificially generated data and real observation data. Experiments that used artificial data using four real network structures showed that the method can correctly 1) learn the parameters and 2) select the model by which the data have been generated. Experiments that used real diffusion data of topic propagation showed that 1) both AsIC and AsLT models well capture the global characteristics of topic propagations but 2) the predictive accuracy of each model is different for each topic and some topics have clear indication as to which model each better follows.

## 2 Information Diffusion Models

We first present the *asynchronous independent cascade (AsIC) model* introduced in [13], and then define the *asynchronous linear threshold (AsLT) model*. We mathematically model the spread of information through a directed network  $G = (V, E)$  without self-links, where  $V$  and  $E \subset V \times V$  stand for the sets of all the nodes and links, respectively. For each node  $v$  in the network  $G$ , we denote  $F(v)$  as a set of child nodes of  $v$ , i.e.,  $F(v) = \{w; (v, w) \in E\}$ . Similarly, we denote  $B(v)$  as a set of parent nodes of  $v$ , i.e.,  $B(v) = \{u; (u, v) \in E\}$ . We call nodes *active* if they have been influenced with the information. In the following models, we assume that nodes can switch their states only from inactive to active, but not the other way around, and that, given an initial active node set  $S$ , only the nodes in  $S$  are active at an initial time.

### 2.1 Asynchronous Independent Cascade Model

We first recall the definition of the IC model according to [7], and then introduce the AsIC model. In the IC model, we specify a real value  $\kappa_{u,v}$  with  $0 < \kappa_{u,v} < 1$  for each link  $(u, v)$  in advance. Here  $\kappa_{u,v}$  is referred to as the *diffusion probability* through link  $(u, v)$ . The diffusion process unfolds in discrete time-steps  $t \geq 0$ , and proceeds from a given initial active set  $S$  in the following way. When a node  $u$  becomes active at time-step  $t$ , it is given a single chance to activate each currently inactive child node  $v$ , and succeeds with probability  $\kappa_{u,v}$ . If  $u$  succeeds, then  $v$  will become active at time-step  $t + 1$ . If multiple parent nodes of  $v$  become active at time-step  $t$ , then their activation attempts are sequenced in an arbitrary order, but all performed at time-step  $t$ . Whether or not  $u$  succeeds, it cannot make any further attempts to activate  $v$  in subsequent rounds. The process terminates if no more activations are possible.

In the AsIC model, we specify real values  $r_{u,v}$  with  $r_{u,v} > 0$  in advance for each link  $(u, v) \in E$  in addition to  $\kappa_{u,v}$ , where  $r_{u,v}$  is referred to as the *time-delay parameter* through link  $(u, v)$ . The diffusion process unfolds in continuous-time  $t$ , and proceeds from a given initial active set  $S$  in the following way. Suppose that a node  $u$  becomes active at time  $t$ . Then,  $u$  is given a single chance to activate each currently inactive child node  $v$ . We choose a delay-time  $\delta$  from the exponential distribution with parameter

$r_{u,v}$ .<sup>1</sup> If  $v$  has not been activated before time  $t + \delta$ , then  $u$  attempts to activate  $v$ , and succeeds with probability  $\kappa_{u,v}$ . If  $u$  succeeds, then  $v$  will become active at time  $t + \delta$ . Under the continuous time framework, it is unlikely that  $v$  is activated simultaneously by its multiple parent nodes exactly at time  $t + \delta$ . So we ignore this possibility. The process terminates if no more activations are possible.

## 2.2 Asynchronous Linear Threshold Model

Similarly to the above, we first define the LT model. In this model, for every node  $v \in V$ , we specify a *weight* ( $\omega_{u,v} > 0$ ) from its parent node  $u$  in advance such that  $\sum_{u \in B(v)} \omega_{u,v} \leq 1$ . The diffusion process from a given initial active set  $S$  proceeds according to the following randomized rule. First, for any node  $v \in V$ , a *threshold*  $\theta_v$  is chosen uniformly at random from the interval  $[0, 1]$ . At time-step  $t$ , an inactive node  $v$  is influenced by each of its active parent nodes,  $u$ , according to weight  $\omega_{u,v}$ . If the total weight from active parent nodes of  $v$  is no less than  $\theta_v$ , that is,  $\sum_{u \in B_t(v)} \omega_{u,v} \geq \theta_v$ , then  $v$  will become active at time-step  $t + 1$ . Here,  $B_t(v)$  stands for the set of all the parent nodes of  $v$  that are active at time-step  $t$ . The process terminates if no more activations are possible.

Next, we define the AsLT model. In the AsLT model, in addition to the weight set  $\{\omega_{u,v}\}$ , we specify real values  $r_v$  with  $r_v > 0$  in advance for each node  $v \in V$ . We refer to  $r_v$  as the *time-delay parameter* on node  $v$ . Note that  $r_v$  depends only on  $v$  unlike  $r_{u,v}$  of the AsIC model, which means that it is the node  $v$ 's decision when to receive the information once the activation condition has been satisfied.<sup>2</sup> The diffusion process unfolds in continuous-time  $t$ , and proceeds from a given initial active set  $S$  in the following way. Suppose that the total weight from active parent nodes of  $v$  became no less than  $\theta_v$  at time  $t$  for the first time. Then,  $v$  will become active at time  $t + \delta$ , where we choose a delay-time  $\delta$  from the exponential distribution with parameter  $r_v$ . Further, note that even if some other non-active parent nodes of  $v$  has become active during the time period between  $t$  and  $t + \delta$ , the activation time of  $v$ ,  $t + \delta$ , still remains the same. The other diffusion mechanisms are the same as the LT model.

## 3 Learning Algorithms

We define the time-delay parameter vector  $\mathbf{r}$  and the diffusion parameter vector  $\boldsymbol{\kappa}$  by  $\mathbf{r} = (r_{u,v})_{(u,v) \in E}$  and  $\boldsymbol{\kappa} = (\kappa_{u,v})_{(u,v) \in E}$  for the AsIC model and the parameter vectors  $\boldsymbol{\omega}$  and  $\mathbf{r}$  by  $\boldsymbol{\omega} = (\omega_{u,v})_{(u,v) \in E}$  and  $\mathbf{r} = (r_v)_{v \in V}$  for the AsLT model. We next consider an observed data set of  $M$  independent information diffusion results,  $\{D_m; m = 1, \dots, M\}$ . Here, each  $D_m$  is a set of pairs of active nodes and their activation times in the  $m$ th diffusion result,  $D_m = \{(u, t_{m,u}), (v, t_{m,v}), \dots\}$ . For each  $D_m$ , we denote the observed initial time by  $t_m = \min\{t_{m,v}; (v, t_{m,v}) \in D_m\}$ , and the observed final time by  $T_m \geq \max\{t_{m,v}; (v, t_{m,v}) \in D_m\}$ . Note that  $T_m$  is not necessarily equal to the final activation time. Hereafter, we express our observation data by  $\mathcal{D}_M = \{(D_m, T_m); m = 1, \dots, M\}$ . For any  $t \in [t_m, T_m]$ , we set  $C_m(t) = \{v; (v, t_{m,v}) \in D_m, t_{m,v} < t\}$ . Namely,  $C_m(t)$  is the set

<sup>1</sup> Similar formulation can be derived for other distributions such as power-law and Weibull.

<sup>2</sup> It is also possible to adopt the same edge time-delay model as in the AsIC model, in which case, for example,  $r_v$  in Equation (2) in Section 3 is replaced with  $r_{u,v}$ .

of active nodes before time  $t$  in the  $m$ th diffusion result. For convenience sake, we use  $C_m$  as referring to the set of all the active nodes in the  $m$ th diffusion result. Moreover, we define a set of non-active nodes with at least one active parent node for each by  $\partial C_m = \{v; (u, v) \in E, u \in C_m, v \notin C_m\}$ . For each node  $v \in C_m \cup \partial C_m$ , we define the following subset of parent nodes, each of which has a chance to activate  $v$ .

$$\mathcal{B}_{m,v} = \begin{cases} B(v) \cap C_m(t_{m,v}) & \text{if } v \in C_m(t_{m,v}), \\ B(v) \cap C_m & \text{if } v \in \partial C_m. \end{cases}$$

Note that the underlying model behind the observed data is not available in reality. Thus, we investigate how the model affects the information diffusion results, and consider selecting a model which better explains the given observed data from the candidates, i.e., AsIC and AsLT models. To this end, we first have to estimate the values of  $\mathbf{r}$  and  $\kappa$  for the AsIC model, and the values of  $\mathbf{r}$  and  $\omega$  for the AsLT model for the given  $\mathcal{D}_M$ . For the former, we adopt the method proposed in [13], which is only briefly explained here. For the latter, we propose a novel method of estimating those values.

### 3.1 Learning Parameters of AsIC Model

To estimate the values of  $\mathbf{r}$  and  $\kappa$  from  $\mathcal{D}_M$  for the AsIC model, We derived the following likelihood function  $\mathcal{L}(\mathbf{r}, \kappa; \mathcal{D}_M)$  to use as the objective function [13],

$$\mathcal{L}(\mathbf{r}, \kappa; \mathcal{D}_M) = \prod_{m=1}^M \prod_{v \in C_m} \left( h_{m,v} \prod_{w \in F(v) \setminus C_m} g_{m,v,w} \right), \quad (1)$$

where  $h_{m,v}$  is the probability density that the node  $v$  such that  $v \in D_m$  with  $t_{m,v} > 0$  for the  $m$ th diffusion result is activated at time  $t_{m,v}$ , and  $g_{m,v,w}$  is the probability that a node  $w$  is not activated by a node  $v$  within the observed time period  $[t_m, T_m]$  when there is a link  $(v, w) \in E$  and  $v \in C_m$  for the  $m$ th diffusion result. Then, we derived an iterative algorithm to stably obtain the values of  $\mathbf{r}$  and  $\kappa$  that maximize equation (1). Please refer to [13] for more details. Hereafter, we refer to this method as the *AsIC model based method*.

### 3.2 Learning Parameters of AsLT Model

**Likelihood function.** To estimate the values of  $\mathbf{r}$  and  $\omega$  from  $\mathcal{D}_M$  for the AsLT model, we first derive the likelihood function  $\mathcal{L}(\mathbf{r}, \omega; \mathcal{D}_M)$  with respect to  $\mathbf{r}$  and  $\omega$  in a rigorous way to use as the objective function. For the sake of technical convenience, we introduce a slack weight  $\omega_{v,v}$  for each node  $v \in V$  such that  $\omega_{v,v} + \sum_{u \in B(v)} \omega_{u,v} = 1$ . Here note that we can regard each weight  $\omega_{*,v}$  as a multinomial probability since a threshold  $\theta_v$  is chosen uniformly at random from the interval  $[0, 1]$  for each node  $v$ .

Suppose that a node  $v$  became active at time  $t_{m,v}$  for the  $m$ th result. Then, we know that the total weight from active parent nodes of  $v$  became no less than  $\theta_v$  at the time when one of them,  $u \in \mathcal{B}_{m,v}$ , became first active. However, in case of  $|\mathcal{B}_{m,v}| > 1$ , there is no way of exactly knowing the actual nodes due to the asynchronous time-delay. Suppose that a node  $v$  was actually activated when a node  $\zeta \in \mathcal{B}_{m,v}$  became activated. Then

$\theta_v$  is between  $\sum_{u \in B(v) \cap C_m(t_{m,\xi})} \omega_{u,v}$  and  $\omega_{\xi,v} + \sum_{u \in B(v) \cap C_m(t_{m,\xi})} \omega_{u,v}$ . Namely, the probability that  $\theta_v$  is chosen from this range is  $\omega_{\xi,v}$ . Here note that such events with respect to different active parent nodes are mutually disjoint. Thus, the probability density that the node  $v$  is activated at time  $t_{m,v}$ , denoted by  $h_{m,v}$ , can be expressed as

$$h_{m,v} = \sum_{u \in \mathcal{B}_{m,v}} \omega_{u,v} r_v \exp(-r_v(t_{m,v} - t_{m,u})). \quad (2)$$

Here we define  $h_{m,v} = 1$  if  $t_{m,v} = t_m$ .

Next, we consider any node  $w \in V$  belonging to  $\partial C_m = \{w; (v, w) \in E \wedge v \in C_m(T_m) \wedge w \notin C_m(T_m)\}$  for the  $m$ th result. Let  $g_{m,v}$  denote the probability that the node  $v$  is not activated within the observed time period  $[t_m, T_m]$ . We can calculate  $g_{m,v}$  as

$$\begin{aligned} g_{m,v} &= 1 - \sum_{u \in \mathcal{B}_{m,v}} \omega_{u,v} \int_{t_{m,u}}^{T_m} r_v \exp(-r_v(t - t_{m,u})) dt = 1 - \sum_{u \in \mathcal{B}_{m,v}} \omega_{u,v} (1 - \exp(-r_v(T_m - t_{m,u}))) \\ &= \omega_{v,v} + \sum_{u \in B(v) \setminus \mathcal{B}_{m,v}} \omega_{u,v} + \sum_{u \in \mathcal{B}_{m,v}} \omega_{u,v} \exp(-r_v(T_m - t_{m,u})). \end{aligned} \quad (3)$$

Therefore, by using Equations (2) and (3), and the independence properties, we can define the likelihood function  $\mathcal{L}(\mathbf{r}, \omega; \mathcal{D}_M)$  with respect to  $\mathbf{r}$  and  $\omega$  by

$$\mathcal{L}(\mathbf{r}, \omega; \mathcal{D}_M) = \prod_{m=1}^M \left( \prod_{v \in C_m} h_{m,v} \right) \left( \prod_{v \in \partial C_m} g_{m,v} \right). \quad (4)$$

Thus, our problem is to obtain the time-delay parameter vector  $\mathbf{r}$  and the diffusion parameter vector  $\omega$ , which together maximize Equation (4).

**Learning Algorithm.** For the above learning problem, we can derive an estimation method based on the Expectation-Maximization algorithm in order to stably obtain its solutions. Hereafter, we refer to this proposed method as the *AsLT model based method*. By the following formulas, we define  $\phi_{m,u,v}$  for each  $v \in C_m$  and  $u \in \mathcal{B}_{m,v}$ ,  $\varphi_{m,u,v}$  for each  $v \in \partial C_m$  and  $u \in \{v\} \cup B(v) \setminus \mathcal{B}_{m,v}$ , and  $\psi_{m,u,v}$  for each  $v \in \partial C_m$  and  $u \in \mathcal{B}_{m,v}$ , respectively.

$$\begin{aligned} \phi_{m,u,v} &= \omega_{u,v} r_v \exp(-r_v(t_{m,v} - t_{m,u})) / h_{m,v}, & \varphi_{m,u,v} &= \omega_{u,v} / g_{m,v}, \\ \psi_{m,u,v} &= \omega_{u,v} \exp(-r_v(T_m - t_{m,u})) / g_{m,v}. \end{aligned}$$

Let  $\bar{\mathbf{r}} = (\bar{r}_v)$  and  $\bar{\omega} = (\bar{\omega}_{u,v})$  be the current estimates of  $\mathbf{r}$  and  $\omega$ , respectively. Similarly, let  $\bar{\phi}_{m,u,v}$ ,  $\bar{\varphi}_{m,u,v}$ , and  $\bar{\psi}_{m,u,v}$  denote the values of  $\phi_{m,u,v}$ ,  $\varphi_{m,u,v}$ , and  $\psi_{m,u,v}$  calculated by using  $\bar{\mathbf{r}}$  and  $\bar{\omega}$ , respectively.

From equations (2), (3), (4), we can transform  $\mathcal{L}(\mathbf{r}, \omega; \mathcal{D}_M)$  as follows:

$$\log \mathcal{L}(\mathbf{r}, \omega; \mathcal{D}_M) = Q(\mathbf{r}, \omega; \bar{\mathbf{r}}, \bar{\omega}) - H(\mathbf{r}, \omega; \bar{\mathbf{r}}, \bar{\omega}), \quad (5)$$

where  $Q(\mathbf{r}, \omega; \bar{\mathbf{r}}, \bar{\omega})$  is defined by

$$Q(\mathbf{r}, \omega; \bar{\mathbf{r}}, \bar{\omega}) = \sum_{m=1}^M \left( \sum_{v \in C_m} Q_{m,v}^{(1)} + \sum_{v \in \partial C_m} Q_{m,v}^{(2)} \right), \quad (6)$$

$$\begin{aligned}
 Q_{m,v}^{(1)} &= \sum_{u \in \mathcal{B}_{m,v}} \bar{\phi}_{m,u,v} \log(\omega_{u,v} r_v \exp(-r_v(t_{m,v} - t_{m,u}))) \\
 Q_{m,v}^{(2)} &= \sum_{u \in \{v\} \cup B(v) \setminus \mathcal{B}_{m,v}} \bar{\varphi}_{m,u,v} \log(\omega_{u,v}) + \sum_{u \in \mathcal{B}_{m,v}} \bar{\psi}_{m,u,v} \log(\omega_{u,v} \exp(-r_v(T_m - t_{m,u}))).
 \end{aligned}$$

It is easy to see that  $Q(\mathbf{r}, \omega; \bar{\mathbf{r}}, \bar{\omega})$  is convex with respect to  $\mathbf{r}$  and  $\omega$ , and  $H(\mathbf{r}, \kappa; \bar{\mathbf{r}}, \bar{\omega})$  is defined by

$$\begin{aligned}
 H(\mathbf{r}, \omega; \bar{\mathbf{r}}, \bar{\omega}) &= \sum_{m=1}^M \left( \sum_{v \in C_m} H_{m,v}^{(1)} + \sum_{v \in \partial C_m} H_{m,v}^{(2)} \right), \tag{7} \\
 H_{m,v}^{(1)} &= \sum_{u \in \mathcal{B}_{m,v}} \bar{\phi}_{m,u,v} \log(\phi_{m,u,v}), \\
 H_{m,v}^{(2)} &= \sum_{u \in \{v\} \cup B(v) \setminus C_m} \bar{\varphi}_{m,u,v} \log(\varphi_{m,u,v}) + \sum_{u \in \mathcal{B}_{m,v}} \bar{\psi}_{m,u,v} \log(\psi_{m,u,v}).
 \end{aligned}$$

Since  $H(\mathbf{r}, \omega; \bar{\mathbf{r}}, \bar{\omega})$  is maximized at  $\mathbf{r} = \bar{\mathbf{r}}$  and  $\omega = \bar{\omega}$  from equation (7), we can increase the value of  $\mathcal{L}(\mathbf{r}, \omega; \mathcal{D}_M)$  by maximizing  $Q(\mathbf{r}, \omega; \bar{\mathbf{r}}, \bar{\omega})$  (see equation (5)).

Thus, we obtain the following update formulas of our estimation method as the solution which maximizes  $Q(\mathbf{r}, \omega; \bar{\mathbf{r}}, \bar{\omega})$  with respect to  $\mathbf{r}$ :

$$r_v = \left( \sum_{m \in \mathcal{M}_v^{(1)}} \sum_{u \in \mathcal{B}_{m,v}} \bar{\phi}_{m,u,v} \right) \times \left( \sum_{m \in \mathcal{M}_v^{(1)}} \sum_{u \in \mathcal{B}_{m,v}} \bar{\phi}_{m,u,v} (t_{m,v} - t_{m,u}) + \sum_{m \in \mathcal{M}_v^{(2)}} \sum_{u \in \mathcal{B}_{m,v}} \bar{\psi}_{m,u,v} (T_m - t_{m,u}) \right)^{-1}$$

where  $\mathcal{M}_v^{(1)}$  and  $\mathcal{M}_v^{(2)}$  are defined by

$$\mathcal{M}_v^{(1)} = \{m \in \{1, \dots, M\}; v \in C_m\}, \quad \mathcal{M}_v^{(2)} = \{m \in \{1, \dots, M\}; v \in \partial C_m\}.$$

As for  $\omega$ , we have to take the constraints  $\omega_{v,v} + \sum_{u \in B(v)} \omega_{u,v} = 1$  into account for each  $v$ , which can easily be made using the Lagrange multipliers method, and we obtain the following update formulas of our estimation method:

$$\omega_{u,v} \propto \sum_{m \in \mathcal{M}_{u,v}^{(1)}} \bar{\phi}_{m,u,v} + \sum_{m \in \mathcal{M}_{u,v}^{(2)}} \bar{\varphi}_{m,u,v} + \sum_{m \in \mathcal{M}_{u,v}^{(3)}} \bar{\psi}_{m,u,v}, \quad \omega_{v,v} \propto \sum_{m \in \mathcal{M}_v^{(2)}} \bar{\varphi}_{m,v,v}$$

where  $\mathcal{M}_{u,v}^{(1)}$ ,  $\mathcal{M}_{u,v}^{(2)}$  and  $\mathcal{M}_{u,v}^{(3)}$  are defined by

$$\begin{aligned}
 \mathcal{M}_{u,v}^{(1)} &= \{m \in \{1, \dots, M\}; v \in C_m, u \in \mathcal{B}_{m,v}\}, \\
 \mathcal{M}_{u,v}^{(2)} &= \{m \in \{1, \dots, M\}; v \in \partial C_m, u \in B(v) \setminus \mathcal{B}_{m,v}\}, \\
 \mathcal{M}_{u,v}^{(3)} &= \{m \in \{1, \dots, M\}; v \in \partial C_m, u \in \mathcal{B}_{m,v}\}.
 \end{aligned}$$

The actual values are obtained after normalization. Recall that we can regard our estimation method as a kind of the EM algorithm. It should be noted that each time the iteration proceeds the value of the likelihood function never decreases and the iterative algorithm is guaranteed to converge.

### 3.3 Model Selection

Next, we describe our model selection method. We select the model based on predictive accuracy. Here, note that we cannot use an information theoretic criterion such as AIC (Akaike Information Criterion) or MDL (Minimum Description Length) because we need to select one from models with completely different probability distributions. Moreover, for both models, it is quite difficult to efficiently calculate the exact activation probability of each node more than two information diffusion cascading steps ahead. In order to avoid these difficulties, we propose a method based on a hold-out strategy, which attempts to predict the activation probabilities at one step later.

For simplicity, we assume that for each  $D_m$ , the initial observation time  $t_m$  is zero, i.e.,  $t_m = 0$  for  $m = 1, \dots, M$ . Then, we introduce a set of observation periods

$$\mathcal{I} = \{[0, \tau_n]; n = 1, \dots, N\},$$

where  $N$  is the number of observation data we want to predict sequentially and each  $\tau_n$  has the following property: There exists some  $(v, t_{m,v}) \in D_m$  such that  $0 < \tau_n < t_{m,v}$ . Let  $D_{m;\tau_n}$  denote the observation data in the period  $[0, \tau_n)$  for the  $m$ th diffusion result, i.e.,

$$D_{m;\tau_n} = \{(v, t_{m,v}) \in D_m; t_{m,v} < \tau_n\}.$$

We also set  $\mathcal{D}_{M;\tau_n} = \{(D_{m;\tau_n}, \tau_n); m = 1, \dots, M\}$ . Let  $\boldsymbol{\theta}$  denote the set of parameters for either the AsIC or the AsLT models, i.e.,  $\boldsymbol{\theta} = (\mathbf{r}, \boldsymbol{\kappa})$  or  $\boldsymbol{\theta} = (\mathbf{r}, \boldsymbol{\omega})$ . We can estimate the values of  $\boldsymbol{\theta}$  from the observation data  $\mathcal{D}_{M;\tau_n}$  by using the learning algorithms in Sections 3.1 and 3.2. Let  $\widehat{\boldsymbol{\theta}}_{\tau_n}$  denote the estimated values of  $\boldsymbol{\theta}$ . Then, we can calculate the activation probability  $q_{\tau_n}(v, t)$  of node  $v$  at time  $t (\geq \tau_n)$  using  $\widehat{\boldsymbol{\theta}}_{\tau_n}$ .

For each  $\tau_n$ , we select the node  $v(\tau_n)$  and the time  $t_{m(\tau_n),v(\tau_n)}$  by

$$t_{m(\tau_n),v(\tau_n)} = \min \left\{ t_{m,v}; (v, t_{m,v}) \in \bigcup_{m=1}^M (D_m \setminus D_{m;\tau_n}) \right\}.$$

Note that  $v(\tau_n)$  is the first active node in  $t \geq \tau_n$ . We evaluate the predictive performance for the node  $v(\tau_n)$  at time  $t_{m(\tau_n),v(\tau_n)}$ . Approximating the empirical distribution by

$$p_{\tau_n}(v, t) = \delta_{v,v(\tau_n)} \delta(t - t_{m(\tau_n),v(\tau_n)})$$

with respect to  $(v(\tau_n), t_{m(\tau_n),v(\tau_n)})$ , we employ the Kullback-Leibler (KL) divergence

$$KL(p_{\tau_n} \| q_{\tau_n}) = - \sum_{v \in V} \int_{\tau_n}^{\infty} p_{\tau_n}(v, t) \log \frac{q_{\tau_n}(v, t)}{p_{\tau_n}(v, t)} dt,$$

where  $\delta_{v,w}$  and  $\delta(t)$  stand for Kronecker's delta and Dirac's delta function, respectively. Then, we can easily show

$$KL(p_{\tau_n} \| q_{\tau_n}) = - \log h_{m(\tau_n),v(\tau_n)}. \quad (8)$$

By averaging the above KL divergence with respect to  $\mathcal{I}$ , we propose the following model selection criterion  $\mathcal{E}$  (see Equation (8)):

$$\mathcal{E}(\mathcal{X}; D_1 \cup \dots \cup D_M) = - \frac{1}{N} \sum_{n=1}^N \log h_{m(\tau_n),v(\tau_n)}, \quad (9)$$



where  $\mathcal{X}$  expresses the information diffusion model (i.e., the AsIC or the AsLT models). In our experiments, we adopted

$$\mathcal{I} = \{[0, t_{m,v}); (v, t_{m,v}) \in D_1 \cup \dots \cup D_M, t_{m,v} \geq \tau_0\},$$

where  $\tau_0$  is the median time of all the observed activation time points.

### 3.4 Behavioral Analysis

Thus far, we assumed that  $\theta$  can vary with respect to nodes and links but is independent of the topic of information diffused. However, they may be sensitive to the topic. We follow [13] and place a constraint that  $\theta$  depends only on topics but not on nodes and links of the network  $G$ , and assign a different  $m$  to a different topic. Therefore, we set  $r_{m,u,v} = r_m$  and  $\kappa_{m,u,v} = \kappa_m$  for any link  $(u, v) \in E$  in case of the AsIC model and  $r_{m,v} = r_m$  and  $\omega_{m,u,v} = q_m |B(v)|^{-1}$  for any node  $v \in V$  and link  $(u, v) \in E$  in case of the AsLT model. Here note that  $0 < q_m < 1$  and  $\omega_{v,v} = 1 - q_m$ . Without this constraint, we only have one piece of observation for each  $(m, u, v)$  and there is no way to learn  $\theta$ .

Using each pair of the estimated parameters,  $(r_m, q_m)$  for the AsLT model and  $(r_m, \kappa_m)$  for the AsIC model, we can discuss which model is more appropriate for each topic, and analyze the behavior of people with respect to the topics of information by simply plotting them as a point in 2-dimensional space.

## 4 Performance Evaluation by Artificial Data

Our goal here is to evaluate the parameter learning and model selection methods to see how accurately it can detect the true model that generated the data, using topological structure of four large real networks. Here, we assumed the true model by which the data are generated to be either AsLT or AsIC.

### 4.1 Data Sets

We employed four datasets of large real networks (all bidirectionally connected). The first one is a traceback network of Japanese blogs used in [8] and has 12,047 nodes and 79,920 directed links (the blog network). The second one is a network of people derived from the “list of people” within Japanese Wikipedia, also used in [8], and has 9,481 nodes and 245,044 directed links (the Wikipedia network). The third one is a network derived from the Enron Email Dataset [16] by extracting the senders and the recipients and linking those that had bidirectional communications. It has 4,254 nodes and 44,314 directed links (the Enron network). The fourth one is a coauthorship network used in [17] and has 12,357 nodes and 38,896 directed links (the coauthorship network).

Here, according to [13], we assumed the simplest case where the parameter values are uniform across all links and nodes, i.e.,  $\omega_{u,v} = q |B(v)|^{-1}$ ,  $r_v = r$  for AsLT, and  $r_{u,v} = r$ ,  $\kappa_{u,v} = \kappa$  for AsIC. Under this assumption there is no need for the observation sequence data to pass through every link or node at least once. This drastically reduces the amount of data necessary to learn the parameters. Then, our task is to estimate the values of these parameters from data. The true value of  $q$  was set to 0.9 for every network to achieve reasonably long diffusion results, and the true value of  $r$  was set to 1.0.

**Table 1.** Parameter estimation error of the learning method for four networks

Network		Blog	Wiki	Enron	Coauthor
$\mathcal{D}_M(AsLT)$	$r$	0.248	0.253	0.200	0.244
	$q$	0.080	0.078	0.077	0.089
$\mathcal{D}_M(AsIC)$	$r$	0.114	0.026	0.029	0.167
	$\kappa$	0.020	0.013	0.002	0.054

**Table 2.** Accuracy of the model selection method for four networks

Network	Blog	Wiki	Enron	Coauthor
$\mathcal{D}_M(AsLT)$	79 (28.2)	86 (54.0)	99 (47.7)	76 (19.0)
$\mathcal{D}_M(AsIC)$	92 (370.2)	100 (920.8)	100 (1500.6)	93 (383.5)

According to [7], we set  $\kappa$  to a value smaller than  $1/\bar{d}$ , where  $\bar{d}$  is the mean out-degree of a network. Thus, the true value of  $\kappa$  was set to 0.2 for the coauthorship network, 0.1 for the blog and Enron networks, and 0.02 for the Wikipedia network. Using these values, two sets of data were generated for each network, one for the true AsLT model and the other for the true AsIC model, denoted by  $\mathcal{D}_M(AsLT)$  and  $\mathcal{D}_M(AsIC)$ , respectively. For each of these, sequences of data were generated, each starting from a randomly selected initial active node and having at least 10 nodes. In our experiments, we set  $M = 100$  and evaluated our model selection method in the framework of behavioral analysis. Parameter updating is terminated when either the iteration number reaches its maximum (set to 100) or the following condition is first satisfied:  $|r(s+1) - r(s)| + |q(s+1) - q(s)| \leq 10^{-6}$  for AsLT,  $|r(s+1) - r(s)| + |\kappa(s+1) - \kappa(s)| \leq 10^{-6}$  for AsIC. In most of the cases, the latter inequality is satisfied in less than 100 iterations. The converged values are rather insensitive to the initial values, and we confirmed that the parameter updating algorithm stably converges to the correct values. In actual computation, the learned values for  $\tau_n$  is used as the initial values for  $\tau_{n+1}$  for efficiency purpose.

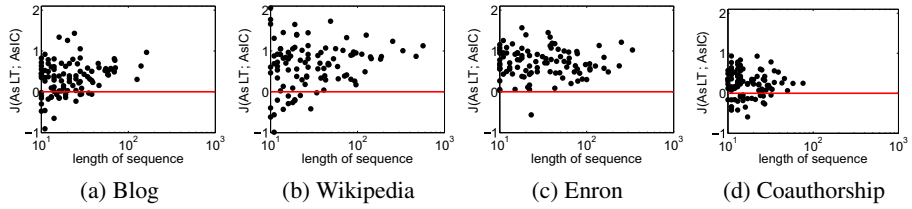
## 4.2 Learning Results

Table 1 shows the error in the estimated parameters for four networks by the proposed learning method. In this evaluation we treated each sequence as a separate observation and learned the parameters from each, repeated this  $M (=100)$  times and took the average. More specifically, the parameters of AsLT were estimated from  $\mathcal{D}_M(AsLT)$ , and those of AsIC from  $\mathcal{D}_M(AsIC)$ . Even though each pair of the parameters for individual models was estimated by using only one sequence data, we can see that the estimated values were reasonably close to the true one. This confirms that our proposed learning methods work well. The results indicate that the estimation performance on AsIC is substantially better than that on AsLT. We consider that this performance difference is attributed to the average sequence length, as discussed later.

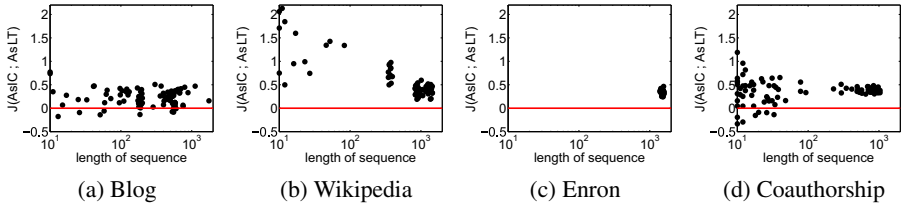
## 4.3 Model Selection Results

The average KL divergence given by equation (9) is the measure for the goodness of the model  $X$ , given the data  $D_m$ . The smaller its value is, the better the model explains the data in terms of predictability. Thus, we can estimate the true model from which  $D_m$  is generated to be AsLT if  $\mathcal{E}(AsLT; D_m) < \mathcal{E}(AsIC; D_m)$ , and vice versa.

Table 2 summarizes the number of sequences for which the model selection method correctly identified the true model. The number within the parentheses is the average length of the sequences in each dataset. From these results, we can say that the proposed



**Fig. 1.** Relation between the length of sequence and the the accuracy of model selection for  $\mathcal{D}_M(AsLT)$



**Fig. 2.** Relation between the length of sequence and the the accuracy of model selection for  $\mathcal{D}_M(AsIC)$

method achieved a good accuracy, 90.6% on average. Especially, for the Enron network, its estimation was almost perfect. To analyze the performance of the proposed method more deeply, we investigated the relation between the length of sequence and the model selection result. It is shown in Fig. 1 for  $\mathcal{D}_M(AsLT)$ , where the horizontal axis denotes the length of sequence in each dataset and the vertical axis is the difference of the average KL divergence defined by  $J(AsLT; AsIC) = \mathcal{E}(AsIC; D_m) - \mathcal{E}(AsLT; D_m)$ . Thus,  $J(AsLT; AsIC) > 0$  means that the proposed method correctly estimated the true model for the dataset  $D_m(AsLT)$  because it means  $\mathcal{E}(AsLT; D_m)$  is smaller than  $\mathcal{E}(AsIC; D_m)$ . From these figures, we can see that there is a correlation between the length of sequence and the estimation accuracy, and that the misselection occurs only in short sequences for every network. We notice that the overall accuracy becomes 95.5% when considering only the sequences that contain no less than 20 nodes. This means that the proposed model selection method is highly reliable for a long sequence and its accuracy could asymptotically approach to 100% as the sequence gets longer. Figure 2 is the results for  $\mathcal{D}_M(AsIC)$ , where  $J(AsIC; AsLT) = \mathcal{E}(AsLT; D_m) - \mathcal{E}(AsIC; D_m)$ . The results are better than for  $\mathcal{D}_M(AsLT)$ . In particular, Wikipedia and Blog networks have no misselection. We note that the plots are shifted to the right for all networks, meaning that the data sequences are longer for  $\mathcal{D}_M(AsIC)$  than for  $\mathcal{D}_M(AsLT)$ . The better accuracy is attributed to this.

## 5 Behavioral Analysis of Real World Blog Data

We analyzed the behavior of topics in a real world blog data. Here, again, we assumed the true model behind the data to be either AsLT or AsIC. Then, we first applied our

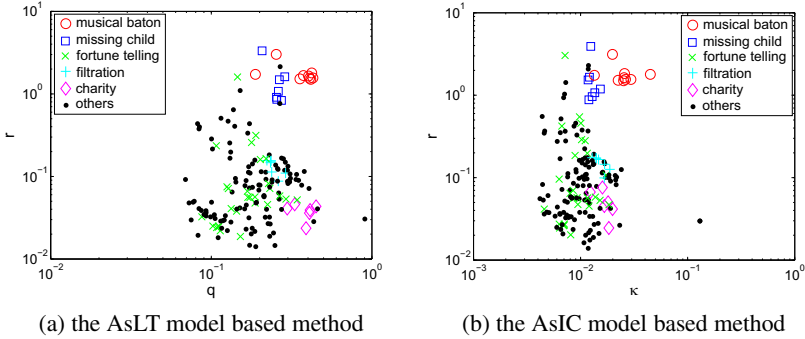


Fig. 3. Results for the Doblog database

learning method to behavioral analysis based on the method described in Section 3.4, assuming two possibilities, i.e. the true model being either AsLT or AsIC for all the topics, and investigated how each topic spreads throughout the network by comparing the learned parameter values. Next, we estimated the true model of each data sequence for each topic by applying the model selection method described in Section 3.3.

## 5.1 Data Sets

We used the real blogroll network used in [13], which was generated from the database of a blog-hosting service in Japan called *Doblog* [3]. In the network, bloggers are connected to each other and we assume that topics propagate from blogger  $x$  to another blogger  $y$  when there is a blogroll link from  $y$  to  $x$ . In addition, according to [18], it is assumed that a topic is represented as a URL which can be tracked down from blog to blog. We used the propagation sequences of 172 URLs for this analysis, each of which has at least 10 time steps. Please refer to [13] for more details.

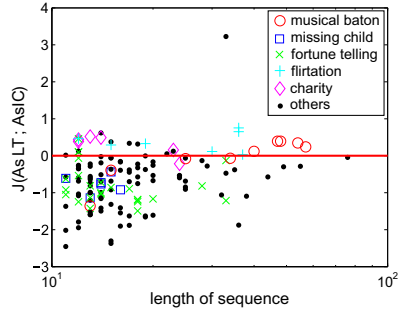
## 5.2 Behavioral Analysis

We ran the experiments for each identified URL and obtained the parameters  $q$  and  $r$  for the AsLT model based method and  $\kappa$  and  $r$  for the AsIC model based method. Figures 3a and 3b are the plots of the results for the major URLs (topics) by the AsLT and AsIC methods, respectively. The horizontal axis is the diffusion parameter  $q$  for the AsLT method and  $\kappa$  for the AsIC method, while the vertical axis is the delay parameter  $r$  for both. The latter axis is normalized such that  $r = 1$  corresponds to a delay of one day, meaning  $r = 0.1$  corresponds to a delay of 10 days. In these figures, we used five kinds of markers other than dots, to represent five different typical URLs: the circle (○) stands for a URL that corresponds to the musical baton which is a kind of telephone game on the Internet (the musical baton), the square (◻) for a URL that corresponds to articles about a missing child (the missing child), the cross (×) for a URL that corresponds to articles about fortune telling (the fortune telling), the diamond (◊) for a URL of a certain charity site (the charity), and the plus (+) for a URL of a site for flirtatious

<sup>3</sup> Doblog(<http://www.doblog.com/>), provided by NTT Data Corp. and Hotto Link, Inc.

**Table 3.** Results of model selection for the Doblog dataset

Topic	Total	AsLT	AsIC
Musical baton	9	5	4
Missing child	7	0	7
Fortune telling	28	4	24
Charity	6	5	1
Flirtation	7	7	0
Others	115	11	104



**Fig. 4.** The relation between the KL difference and sequence length for the Doblog database

tendency test (the flirtation). All the other topics are denoted by dots ( $\cdot$ ), which means they are a mixture of many topics.

The results indicates that in general both the AsLT and AsIC models capture reasonably well the characteristic properties of topics in a similar way. For example, it captures the urgency of the missing child, which propagates quickly. Musical baton which actually became the latest craze on the Internet also propagates quickly. In contrast non-emergency topics such as the flirtation and the charity propagate very slowly. Unfortunately, this highlights the people’s low interest level of the charity activity in the real world. We further note that the dependency of topics on the parameter  $r$  is almost the same for both AsLT and AsIC, but that on the parameters  $q$  and  $\kappa$  is slightly different, e.g., relative difference of musical baton, missing child and charity. Although  $q$  and  $\kappa$  are different parameters but both are the measures that represent how easily the diffusion takes place. We showed in [13] that the influential nodes are very sensitive to the model used and this can be attributed to the differences of these parameter values.

### 5.3 Model Selection

In the analysis of previous subsection, we assumed that each topic follows the same diffusion model. However, in reality this is not true and each topic should propagate following more closely to either one of the AsLT and AsIC models. Thus, in this subsection, we attempt to estimate the underlying behavior model of each topic by applying the model selection method to individual sequence as described in section 4. Namely, we regard that each observation consists of only one observed data sequence, i.e.,  $\mathcal{D}_1$ , and calculate its KL divergences by equation (9) for the both models, and compare the goodness.

Table 3 and Fig. 4 summarize the results. From these results, we can see that most of the diffusion behaviors on this blog network follows the AsIC model. It is interesting to note that the model estimated for the musical baton is not identical to that for the missing child although their diffusion patterns are very similar in the previous analysis. The missing child strictly follows the AsIC model. This is attributed to its greater urgency. On the other, musical baton seems to follow more closely to AsLT. This is because the longer sequence results in a better accuracy and the models selected in longer sequences

are all AsLT in Fig. 4 although the numbers are almost tie (4 vs. 5) in Table 3. This can be interpreted that people follow their friends in this game. Likewise, it is easy to imagine that one would align oneself with the opinions of those around when requested to raise funds. This explains that charity follows AsLT. The flirtation clearly follows AsLT. This is probably because the information of this kind of play site easily diffuses within close friends. Note that there exists one dot at near the top center in Fig. 4, showing the greatest tendency to follow AsLT. This dot represents a typical circle site that distributes one's original news article on personal events.

## 6 Discussion

We now have ways to compare the diffusion process with respect to two models (the AsLT model and the AsIC model) for the same observed dataset. Being able to learn the parameters of these models enable us to analyze the diffusion process more precisely. Comparing the results bring us deeper insights into the relation between models and information diffusion processes.

We note that the formulation in Sections 2 and 3 allows the parameters to depend on links and nodes, but the analysis we showed in Section 4 is for the simplest case where the parameters are uniform across the whole network. Actually, if all the parameters are node and link dependent, the number of the parameters becomes so huge and it is not practical (almost impossible) to estimate them accurately because the amount of observation data needed is prohibitively huge and there is always a problem of overfitting. However, this can be alleviated. In a more realistic setting we can divide  $E$  into subsets  $E_1, E_2, \dots, E_L$  and assign the same value for each parameter within each subset. For example, we may divide the nodes into two groups: those that strongly influence others and those not, or we may divide the nodes into another two groups: those that are easily influenced by others and those not. If there is some background knowledge about the node grouping, our method can make the best use of it. Obtaining such background knowledge is also an important research topic in the knowledge discovery from social networks.

The discussion above is also related to the use of the data for model selection in Section 5 in which we used each sequence separately to learn the model parameter values and select the model rather than using them altogether for the same topic and obtaining a single set of parameter values. The results in Section 5 show that the model parameters thus obtained for each sequence are very similar to each other for the same topic. This in turn justifies the use of the same parameter values for multiple sequence observation data (the way we formulated in Section 3.3).

As we mentioned in Section 5.2 but did not show in this paper due to the space limitation, the ranking results that involve detailed probabilistic simulation is very sensitive to the underlying model which is assumed to generate the observed data. In other words, it is very important to select an appropriate model for the analysis of information diffusion from which the data has been generated if the node characteristics are the main objective of analysis, e.g. such problems as the influence maximization problem [7, 11], a problem at a more detailed level. However, it is also true that the parameters for the topics that actually propagated quickly/slowly in observation converged to the values that enable them to propagate quickly/slowly on the model, regardless of the model chosen. Namely, we can say that the difference of models does not have much influence on the

relative difference of topic propagation which indeed strongly depends on topic itself. Both models are well defined and can explain this property at this level of abstraction. Nevertheless, the model selection is very important if we want to characterize how each topic propagates through the network.

Finally, the proposed learning method is efficient and the runtime is not an issue. The convergence is fast and it can handle networks of millions of nodes because the complexity depends directly on the data size, not the number of nodes. In particular, the complexity of learning from a single sequence is proportional to the number of active nodes, their average degree, and the EM iteration number.

## 7 Conclusion

We considered the problem of analyzing information diffusion process in a social network using two kinds of information diffusion models, incorporating asynchronous time delay, the AsLT model and the AsIC model, and investigated how the results differ according to the model used. To this end, we proposed novel methods of 1) learning the parameters of the AsLT model from the observed data (the method for learning the parameters of the AsIC model has already been reported), and 2) selecting models that better explains the observation. We experimentally confirmed that the learning method converges to the correct values very stably and the model selection method can correctly identifies the diffusion models by which the observed data is generated based on extensive simulations on four real world datasets. We further applied the methods to the real blog data and analyzed the behavior of topic propagation. The relative propagation speed of topics, i.e. how far/near and how fast/slow each topic propagates, that are derived from the learned parameter values is rather insensitive to the model selected, but the model selection algorithm clearly identifies the difference of model goodness for each topic. We found that many of the topics follow the AsIC models in general, but some specific topics have clear interpretations for them being better modeled by either one of the two, and these interpretations are consistent with the model selection results. There are numerous factors that affect the information diffusion process, and there can be a number of different models. Model selection is a big challenge in social network analysis and this work is the first step towards this goal.

## Acknowledgments

This work was partly supported by Asian Office of Aerospace Research and Development, Air Force Office of Scientific Research under Grant No. AOARD-10-4053, and JSPS Grant-in-Aid for Scientific Research (C) (No. 20500147).

## References

1. Newman, M.E.J., Forrest, S., Balthrop, J.: Email networks and the spread of computer viruses. *Physical Review E* 66, 035101 (2002)
2. Newman, M.E.J.: The structure and function of complex networks. *SIAM Review* 45, 167–256 (2003)

3. Gruhl, D., Guha, R., Liben-Nowell, D., Tomkins, A.: Information diffusion through blogspace. *SIGKDD Explorations* 6, 43–52 (2004)
4. Domingos, P.: Mining social networks for viral marketing. *IEEE Intelligent Systems* 20, 80–82 (2005)
5. Leskovec, J., Adamic, L.A., Huberman, B.A.: The dynamics of viral marketing. In: *Proceedings of the 7th ACM Conference on Electronic Commerce (EC'06)*, pp. 228–237 (2006)
6. Goldenberg, J., Libai, B., Muller, E.: Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters* 12, 211–223 (2001)
7. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the spread of influence through a social network. In: *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003)*, pp. 137–146 (2003)
8. Kimura, M., Saito, K., Motoda, H.: Blocking links to minimize contamination spread in a social network. *ACM Transactions on Knowledge Discovery from Data* 3, 9:1–9:23 (2009)
9. Watts, D.J.: A simple model of global cascades on random networks. *Proceedings of National Academy of Science, USA* 99, 5766–5771 (2002)
10. Watts, D.J., Dodds, P.S.: Influence, networks, and public opinion formation. *Journal of Consumer Research* 34, 441–458 (2007)
11. Kimura, M., Saito, K., Nakano, R.: Extracting influential nodes for information diffusion on a social network. In: *Proceedings of the 22nd AAAI Conference on Artificial Intelligence (AAAI-07)*, pp. 1371–1376 (2007)
12. Saito, K., Kimura, M., Nakano, R., Motoda, H.: Finding influential nodes in a social network from information diffusion data. In: *Proceedings of the International Workshop on Social Computing and Behavioral Modeling (SBP'09)*, pp. 138–145 (2009)
13. Saito, K., Kimura, M., Ohara, K., Motoda, H.: Learning continuous-time information diffusion model for social behavioral data analysis. In: Zhou, Z.-H., Washio, T. (eds.) *ACML 2009*. LNCS, vol. 5828, pp. 322–337. Springer, Heidelberg (2009)
14. Goyal, A., Bonchi, F., Lakshmanan, L.V.S.: Learning influence probabilities in social networks. In: *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, pp. 241–250 (2010)
15. Bakshy, E., Karrer, B., Adamic, L.A.: Social influence and the diffusion of user-created content. In: *Proceedings of the Tenth ACM Conference on Electronic Commerce*, pp. 325–334 (2009)
16. Klimt, B., Yang, Y.: The enron corpus: A new dataset for email classification research. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) *ECML 2004*. LNCS (LNAI), vol. 3201, pp. 217–226. Springer, Heidelberg (2004)
17. Palla, G., Derényi, I., Farkas, I., Vicsek, T.: Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435, 814–818 (2005)
18. Adar, E., Adamic, L.A.: Tracking information epidemics in blogspace. In: *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*, pp. 207–214 (2005)



# Learning Sparse Gaussian Markov Networks Using a Greedy Coordinate Ascent Approach

Katya Scheinberg<sup>1</sup> and Irina Rish<sup>2</sup>

<sup>1</sup> IEOR Department, Columbia University, New York, NY

<sup>2</sup> IBM T.J. Watson Research Center, Yorktown Heights, NY

**Abstract.** In this paper, we introduce a simple but efficient greedy algorithm, called *SINCO*, for the Sparse INverse COvariance selection problem, which is equivalent to learning a sparse Gaussian Markov Network, and empirically investigate the structure-recovery properties of the algorithm. Our approach is based on a coordinate ascent method which naturally preserves the sparsity of the network structure. We show that SINCO is often comparable to, and, in various cases, outperforms commonly used approaches such as *glasso* [7] and COVSEL [1], in terms of both structure-reconstruction error (particularly, false positive error) and computational time. Moreover, our method has the advantage of being easily parallelizable. Finally, we show that SINCO's greedy nature allows reproduction of the regularization path behavior by applying the method to one (sufficiently small) instance of the regularization parameter  $\lambda$  only; thus, SINCO can obtain a desired number of network links directly, without having to tune the  $\lambda$  parameter. We evaluate our method empirically on various simulated networks and real-life data from biological and neuroimaging applications.

## 1 Introduction

In many practical applications of statistical learning the objective is not simply to construct an accurate predictive model but rather to discover meaningful interactions among the variables. For example, in applications such as reverse-engineering of gene networks, discovery of functional brain connectivity patterns, or analysis of social interactions, the main focus is on reconstructing the network structure representing dependencies among multiple variables, such as genes, brain areas, or individuals. Probabilistic graphical models, such as Markov networks, provide a statistical tool that captures such variable interactions explicitly in a form of a graph.

Herein, we focus on learning sparse Markov Networks over Gaussian random variables (also called Gaussian Markov Random Fields, or GMRFs), which is equivalent to reconstructing the inverse covariance (concentration, or precision) matrix  $C$ , assuming the data are centered to have zero mean. Following the parsimony principle, our objective is to choose the simplest model, i.e. the sparsest network (matrix) that adequately explains the data. This sparsity requirement not only improves the interpretability of the model, but also serves as a regularizer that helps to avoid overfitting.

The sparse inverse covariance selection problem, first introduced in [4], is to find the maximum-likelihood model with a constraint on the number of parameters

(i.e., small  $l_0$ -norm of  $C$ ). In general, this is an intractable combinatorial problem. Early approaches used greedy forward or backward search that required  $O(p^2)$  maximum-likelihood-estimation (MLE) fits for different models in order to add (delete) an edge [8], where  $p$  is the number of variables. This approach does not scale well with the number of variables<sup>1</sup>; moreover, the existence of MLE for  $C$  is not even guaranteed when the number of variables exceeds the number of observations [3].

Recently, however, an alternative approximation approach to the above problem was suggested in [21] that replaces the intractable  $l_0$  constraint with its  $l_1$ -relaxation, known to enforce sparsity, and yields a convex optimization problem that can be solved efficiently. A variety of algorithms for solving this problem were proposed in the past few years [21][7][14][5][15][9]<sup>2</sup>.

In this paper, we introduce a very simple algorithm for solving the above  $l_1$ -regularized maximum-likelihood problem, and provide a convergence proof. Our algorithm, called SINCO (for Sparse INverse COvariance), solves the primal problem, unlike most of its predecessors that focus on the dual (e.g. COVSEL [1], *glasso* [7], as well as [5]). SINCO uses coordinate ascent, in a greedy manner, optimizing one diagonal or two symmetric off-diagonal elements of  $C$  at each step, unlike, for example, COVSEL or *glasso* which optimize one row (column) of the dual matrix. Thus, SINCO naturally preserves the sparsity of the solution and tends to avoid introducing unnecessary (small) nonzero elements, which appears to be beneficial when the “ground-truth” structure is sufficiently sparse.

Note that, although the current state-of-art algorithms for the above problem are converging to the same optimal solution in the limit, the near-optimal solutions obtained after any fixed number of iterations can be different structure-wise, even though they reach similar precision in the objective function reconstruction. Indeed, it is well-known that similar likelihoods can be obtained by two distributions with quite different structures due to multiple (sufficiently) weak links. As to the  $l_1$ -norm regularization, although it often tends to enforce solution sparsity, it is still only an approximation to  $l_0$  (i.e. a sparse solution may have same  $l_1$ -norm as a much denser one). Adding  $l_1$ -norm penalty is only guaranteed to recover the “ground-truth” model under certain condition on the data (that are not always satisfied in practice) and for certain asymptotic growth regimes of the regularization parameter, with growing number of samples  $n$  and dimensions  $p$  (with unknown constant). So the optimal solution, as well as near-solutions at given precision, could possibly include false positives, and one optimization method can potentially choose sparser near-solutions (at same precision) than another method.

Thus, especially in case of sufficiently sparse ground-truth models, a method such as SINCO may be preferable, since it is more “cautious” about adding nonzero elements than its competitors (i.e., it adds at most two nonzero elements at a time, which are also providing the maximum improvement in the objective function - i.e., the method selects, in a sense, the “most important” edges first). Indeed, as demonstrated by our empirical results, SINCO has a better capability of reducing the false-positive error rate (while

<sup>1</sup> E.g., [11] reported difficulties running “the forward selection MLE for more than thirty nodes in the graph”.

<sup>2</sup> Moreover, recent extensions of this approach impose additional structure on the graph, allowing, for example, to learn blockwise-sparse models [5][15][10].

maintaining a similar true positive rate) when compared to *glasso*, a commonly used method that we choose as a baseline here (together with the similar but less efficient *COVSEL* method), since it is the only other method that maintains the initial sparsity of solution in a controlled manner.

Another property of SINCO is that evaluating each candidate edge can be performed very efficiently, in constant time, by solving a quadratic equation. In terms of the overall computational time, while *glasso* is comparable to, or faster than SINCO for a relatively small number of variables  $p$ , SINCO appears to have much better scaling when  $p$  increases (e.g., gets closer to 1000 variables), and can significantly outperform *glasso* (and, of course, *COVSEL*). Moreover, SINCO has the advantage of being easily parallelizable due to the nature of its greedy steps. While we are not claiming SINCO's computational superiority to all state-of-art methods in the *sequential setting* (it is known that the recently proposed projected gradient [5] and smooth optimization [9] methods outperform *glasso* which is comparable to SINCO), we must underscore that straightforward massive parallelization appears to be SINCO's unique property, as none of its competitors seem to be parallelizable, at least not in such an easy way. In particular, *glasso* solves a sequence of Lasso problems, each of which is solved using sequential coordinate descent, which does not gain from parallelization. The gradient-based methods of [5] and [9] require an eigenvalue factorization or a matrix inverse. These operations, while parallelizable, do not scale as efficiently as simple arithmetic operations involved in SINCO's computations.

Next, we investigate empirically the "path-building" property of SINCO. Note that the structure reconstruction accuracy is known to be quite sensitive to the choice of the regularization parameter  $\lambda$ , and the problem of selecting the "best" value of this parameter in practical settings remains open. (As mentioned before, recent theoretical work has focused mainly on asymptotic consistency results [11,12,11,18,13].) Thus, we explore SINCO vs *glasso* behavior in several regimes of  $\lambda$ . What we observe is that SINCO's greedy approach introduces "important" nonzero elements in a manner similar to the path-construction process based on sequentially reducing the value of  $\lambda$ . SINCO can reproduce the regularization path behavior without actually varying the value of the regularization parameter, following instead the "greedy solution path", i.e. sequentially introducing non-zero elements. We observe such behavior on both synthetic problems and real-life biological networks, such as E.coli transcriptional network from the DREAM-07 challenge [16]. This behavior is somewhat similar to LARS [6] for Lasso, however, unlike LARS, SINCO updates the coordinates which provide the best optimal function value improvement, rather than the largest gradient component.

Finally, experiments on real-life brain imaging (fMRI) data demonstrate that SINCO reconstructs Markov Networks that achieve the same or better classification accuracy than its competitors while using much smaller fraction of edges (non-zero entries of the inverse-covariance matrix). In summary, the advantages of SINCO include (1) simplicity, (2) natural tendency to preserve sparsity (beneficial on sufficiently sparse problems), (3) efficiency and a relatively straightforward massive parallelization, as well as (4) an interesting property associated with its solution path.

## 2 Problem Formulation

We consider a multivariate Gaussian probability density function over a set of  $p$  random variables  $X = \{X_1, \dots, X_p\}$  with the covariance matrix  $\Sigma$  and zero mean. A Markov network (also called a Markov Random Field, or MRF) represents the conditional independence structure of a joint distribution, where a missing edge  $(i, j)$  implies conditional independence between  $X_i$  and  $X_j$  given all remaining variables [8]. In Gaussian MRFs, missing edges correspond to zero entries in the inverse covariance (concentration) matrix  $C = \Sigma^{-1}$ , and vice versa [8]. Thus, learning the structure of a Gaussian MRF is equivalent to recovering the zero-pattern of the corresponding inverse-covariance matrix. Note that the straightforward approach of just taking the inverse of the empirical covariance matrix  $A = \frac{1}{n} \sum_{i=1}^n \mathbf{x}_i^T \mathbf{x}_i$ , where  $\mathbf{x}_i$  is the  $i$ -th sample,  $i = 1, \dots, n$  (i.e., the inverse of the maximum-likelihood estimate of the covariance matrix  $\Sigma$ ), does not produce the desired result. Indeed, even if the inverse exists (which is not necessarily the case when  $p \gg n$ ), it does not typically contain any elements that are exactly zero. Therefore, an explicit sparsity-enforcing constraint needs to be added to the maximum-likelihood formulation.

A common approach to enforcing sparsity of  $C$  is to include as a penalty the (vector)  $l_1$ -norm of  $C$ , which is equivalent to imposing Laplace priors on the elements of  $C$  in the maximum-likelihood framework [21][7][15][15] (see [21] for the derivation details). The standard approach assumes that all entries of  $C$  follow the same Laplace distribution with a common parameter  $\lambda$ , i.e.  $p(C_{ij}) = \frac{\lambda_{ij}}{2} e^{-\lambda_{ij}|C_{ij}|}$ , yielding the following penalized log-likelihood maximization problem [21][17].

$$\max_{C \succ 0} \frac{n}{2} [\ln \det(C) - \text{tr}(AC)] - \lambda \|C\|_1. \quad (1)$$

Herein, we make a more general assumption about  $p(C)$ , allowing different elements of  $C$  to have different parameters  $\lambda_{ij}$  (as, for example, in [5]). Hence we consider the following formulation

$$\max_{C \succ 0} \frac{n}{2} [\ln \det(C) - \text{tr}(AC)] - \|C\|_S. \quad (2)$$

Here by  $\|C\|_S$  we denote the sum of absolute values of the elements of the matrix  $S \cdot C$ , where  $\cdot$  denotes the element-wise product. For example, if  $S$  is a product of  $\rho = \frac{n}{2}\lambda$  and the matrix of all ones, then the problem reduces to the standard problem in the eq.

**[1]** The dual of this problem is

$$\max_{W \succ 0} \left\{ \frac{n}{2} \ln \det(W) - np/2 : \text{s.t. } -S \leq \frac{n}{2}(W - A) \leq S \right\}, \quad (3)$$

where the inequalities involving matrices  $W$ ,  $A$  and  $S$  are element-wise. The optimality conditions for this pair of primal and dual problems imply that  $W = C^{-1}$  and that  $(n/2)W_{ij} - A_{ij} = S_{ij}$  if  $C_{ij} > 0$  and  $(n/2)W_{ij} - A_{ij} = -S_{ij}$  if  $C_{ij} < 0$ .

### 3 The SINCO Method

#### 3.1 Relation to Prior Art

Problem (2) is a special case of a semidefinite programming problem (SDP) [20], which can be solved in polynomial time by interior point methods (IPM). However, each iteration requires  $O(p^6)$  time and  $O(p^4)$  space, which is very costly. Another reason why using IPMs is less desirable for the structure recovery problem is that the sparsity pattern is recovered only in the limit, i.e., the solution does not typically include exact zeros, and thus numerical inaccuracy can potentially interfere with the structure recovery.

As an alternative to IPMs, several more efficient approaches were developed recently for problem (2). Most of those approaches are primarily focused on solving the dual problem in (3). For example, [1] and [7] apply the block-coordinate descent method to the dual formulation, [9] uses a first-order optimal gradient ascent approach, and [5] uses a projected gradient approach.

Herein, we propose a novel algorithm, called SINCO, for Sparse INverse COvariance problem. SINCO solves the primal problem directly and uses coordinate ascent, which naturally preserves the sparsity of the solution. Unlike, for example, COVSEL and *glasso* that optimize one row (and the corresponding symmetric column) of the dual matrix is at each step, SINCO only optimizes one diagonal or two (symmetric) off-diagonal entries of the matrix  $C$  at each step. The advantage of our approach is that the solution to each subproblem is available in closed form as a root of a quadratic equation. Computation at each step requires a constant number of arithmetic operations, independent of  $p$ . Hence, in  $O(p^2)$  operations a potential step can be computed for *all pairs* of symmetric elements (i.e., for all pairs  $(i, j)$ ). Then the step which provides the *best* function value improvement can be chosen, which is the essence of the greedy nature of our approach. Once the step is taken, the update of the gradient information requires  $O(p^2)$  operations. Hence, overall, each iteration takes  $O(p^2)$  operations. As we will see later, each step is also suitable for massive parallelization.

In comparison, *glasso* and COVSEL require solving a quadratic programming problem when updating a row (column)<sup>3</sup>, and its theoretical and empirical complexity varies depending on the method used, but always exceeds  $O(p^2)$ : it is  $O(p^4)$  for COVSEL and  $O(p^3)$  for *glasso*. Also, the methods of [14] and [5] iterate through the columns and require  $O(p^4)$  and  $O(p^3)$  time per iteration, respectively (see [5] for detailed discussion). Note, however, that the overall number of iterations can be potentially lower than in the case of SINCO, since the above methods update each row (column) at once. We will mainly focus on comparing our method with *glasso* as a representative state-of-the-art technique, particularly since it is the only other method that maintains the initial sparsity of the solution in a controlled manner. (In some cases, when we only compare the accuracy of the solution (Section 4.4), we perform experiments with COVSEL, a similar but less efficient implementation of the same approach as *glasso*).

---

<sup>3</sup> COVSEL solves the subproblems via an interior point approach (as second order cone quadratic problems (SOCP)), while *glasso* poses the subproblem as a dual of the Lasso problem [17], which is solved by coordinate descent method.

As we will show in our numerical experiments, SINCO, in a serial mode, is comparable to or faster than *glasso*, which is orders of magnitude faster than COVSEL [7]. Also, SINCO often reaches lower false-positive error than *glasso* since it introduces nonzero elements greedily. Perhaps the most interesting consequence of SINCO’s greedy nature is that it reproduces the regularization path behavior while using only one value of the regularization parameter  $\lambda$  (see Section 4.1). Another important feature of SINCO is the ability to efficiently utilize warm starts in various modes. For instance, it is easy to compute a range of solutions for various values of  $\lambda$ , which defines matrix  $S$ .

### 3.2 Algorithm Description

The main idea of the method is the following: at each iteration, the matrix  $C'$  or the matrix  $C''$  is updated by changing one element on the diagonal or two symmetric off-diagonal elements. This implies that the updated  $C$  can be written at  $C + \theta(e_i e_j^T + e_j e_i^T)$ , where  $i$  and  $j$  are the indices corresponding to the elements that are being changed. We can therefore rewrite the objective function of the problem (2) as a function of  $\theta$  (denoted  $f(\theta)$  below). The key observation is that, given the matrix  $W = C^{-1}$ , the exact line search that optimizes  $f(\theta)$  along the direction  $e_i e_j^T + e_j e_i^T$  reduces to a solution of a quadratic equation. Hence each such line search takes a constant number of operations. Moreover, given the starting objective value, the new function value on each step can be computed in a constant number of steps. This means that we can perform such line search for all  $(i, j)$  pairs in  $O(p^2)$  time, which is linear in the number of unknown variables  $C_{ij}$ . We then can choose the step that gives the best improvement in the value of the objective function. After the step is chosen, the dual matrix  $W = C^{-1}$  and, hence, the objective function gradient, are updated in  $O(p^2)$  operations<sup>4</sup>.

We now present the method. First, we can reformulate the problem (2) as:

$$\begin{aligned} \max_{C', C''} \quad & \frac{n}{2} [\ln \det(C' - C'') - \text{tr}(A(C' - C''))] - \|C' - C''\|_S, \\ \text{s. t.} \quad & C' \geq 0, C'' \geq 0, C' - C'' \succ 0 \end{aligned}$$

Note that  $\|C' - C''\|_S = \text{tr}(S(C' + C''))$  if  $C'$  and  $C''$  have non-overlapping nonzero structure.

For a fixed pair  $(i, j)$ , we consider the update of  $C'$  given by  $C'(\theta) = C' + \theta(e_i e_j^T + e_j e_i^T)$ , such that  $C' \geq 0$ . Then we can write the objective as the function of  $\theta$ :

$$f'(\theta) = \frac{n}{2} (\ln \det(C + \theta e_i e_j^T + \theta e_j e_i^T) - \text{tr}(A(C + \theta e_i e_j^T + \theta e_j e_i^T))) - \|C + \theta e_i e_j^T + \theta e_j e_i^T\|_S$$

Similarly, if we consider the update of the form  $C''(\theta) = C'' + \theta(e_i e_j^T + e_j e_i^T)$  such that  $C'' > 0$ , the objective function becomes

$$f''(\theta) = \frac{n}{2} (\ln \det(C - \theta e_i e_j^T - \theta e_j e_i^T) - \text{tr}(A(C - \theta e_i e_j^T - \theta e_j e_i^T))) - \|C - \theta e_i e_j^T - \theta e_j e_i^T\|_S$$

<sup>4</sup> Note that there is no need to enforce the posdef constraint explicitly, as  $\ln \det(C)$  goes to negative infinity when  $C$  approaches singularity. At each step, we maximize the objective along the direction of increase until the local maximum is reached. Hence, it is impossible for the method to move past the point where the objective is negative infinity.

The method we propose works as follows:

### Algorithm 1

0. Initialize  $C' = I$ ,  $C'' = 0$ ,  $W = I$
1. Form the gradient  $G' = \frac{n}{2}(W - A) - S$  and  $G'' = -S - \frac{n}{2}(W + A)$
2. For each pair  $(i, j)$  such that
  - (i)  $G'_{ij} > 0$ ,  $C''_{ij} = 0$ , compute the maximum of  $f'(\theta)$  for  $\theta > 0$ .
  - (ii)  $G'_{ij} < 0$ ,  $C'_{ij} > 0$ , compute the maximum of  $f'(\theta)$  for  $\theta < 0$  subject to  $C' \geq 0$ .
  - (iii)  $G''_{ij} > 0$ ,  $C'_{ij} = 0$ , compute the maximum of  $f''(\theta)$  for  $\theta > 0$ .
  - (iv)  $G''_{ij} < 0$ ,  $C''_{ij} > 0$ , compute the maximum of  $f''(\theta)$  for  $\theta < 0$  subject to  $C'' \geq 0$ .
3. Choose the step which provides the maximum function improvement.  
If relative function improvement is below tolerance, then Exit.
4. Update  $W^{-1}$  and the function value and repeat.

The inverse  $W$ , then, is updated, according to the Sherman-Morrison-Woodbury formula  $(X + ab^T)^{-1} = X^{-1} - X^{-1}a(1 + b^T X^{-1}a)^{-1}b^T X^{-1}$  in  $O(p^2)$  operations. The following theorem is the result of the analysis presented in Appendix.

**Theorem 1.** *The steps of SINCO algorithm are well-defined (that is, the quadratic equation always yields the maximum of  $f(C)$  along the chosen direction). The algorithm converges to the unique optimal solution of (2).*

Note that the algorithm lends itself readily to massive parallelization. Indeed, at each iteration of the algorithm the step computation for each  $(i, j)$  pair can be parallelized and the procedure that updates  $W$  involves simply adding to each element of  $W$  a function that involves only two rows of  $W$  (see Appendix for details). Hence the updates can be also done in parallel and in very large scale cases the matrix  $W$  can also be stored in a distributed manner. The same is true for the storage of matrices  $A$  and  $S$  (assuming that  $S$  needs to be stored, that is not all elements of  $S$  are the same), while the best way to store  $C'$  and  $C''$  matrices may be in sparse form.

## 4 Empirical Evaluation

In order to test structure-reconstruction accuracy, we first performed experiments on several types of synthetic problems. Note that, unlike prediction of an observed variable, structure reconstruction accuracy is harder to test on “real” data since (1) the “true” structure may not be available and (2) known links in “real” networks (e.g., known gene networks) may not necessarily correspond to links in the underlying Markov net. We generated uniform-random, as well as semi-realistic, structured “scale-free” networks, that follow a power-law degree distribution; such networks are known to model well various biological, ecological, social, and other real-life networks [2]. The scale-free (SF) networks were generated using the preferential attachment (Barabasi-Albert) model [2, 5].

<sup>5</sup> We used the open-source Matlab code available at

<http://www.mathworks.com/matlabcentral/fileexchange/11947>

We generated networks with various density, measured by the % of non-zero off-diagonal entries. For each density level, we generated the networks over  $p$  variables, that defined the structure of the “ground-truth” inverse covariance matrix, and for each of them, we generate matrices with random covariances corresponding to the non-diagonal non-zero entries (while maintaining positive definiteness of the resulting covariance matrix). We then sampled  $n$  instances, with the value of  $n$  depending on the experiment, from the corresponding multivariate Gaussian distribution over  $p$  variables.

We also experimented with several real-life datasets, including (a) microarray data for the genome-scale transcriptional network of E.coli (a DREAM-2007 challenge [16]), and (b) the brain activity data from a set of fMRI experiments described in [12].

We used  $\epsilon = 10^{-6}$  threshold on the improvement in the objective function as a stopping criterion.

#### 4.1 Regularization Path

One of the main challenges in sparse inverse covariance selection is the proper choice of the weight matrix  $S$  in (2). Typically the matrix  $S$  is chosen to be a multiple of the matrix of all ones. The multiplying coefficient is denoted by  $\lambda$  and is called the “regularization parameter”. Hence the norm  $\|C\|_S$  in (2) reduces to  $\lambda\|C\|_1$  (in the vector-norm sense) as in (11). Clearly, for large values of  $\lambda$  as  $\lambda \rightarrow \infty$  the solution to (2) is likely to be very sparse and eventually diagonal, which means that no structure recovery is achieved. On the other hand, if  $\lambda$  is small as  $\lambda \rightarrow 0$ , the solution  $C$  is likely to be dense and eventually approach  $A^{-1}$ , and, again, no structure recovery occurs. Hence exploration of a regularization path is an integral part of the sparse inverse covariance selection.

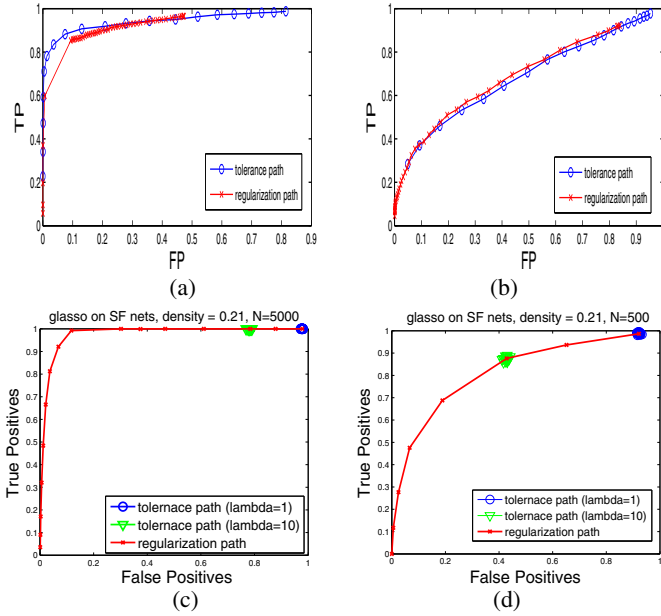
The SINCO method is very well-suited for the efficient regularization path computation, since it directly exploits warm starts. When  $\lambda$  is relatively large, a very sparse solution can be obtained quickly. This solution can be used as a warm start to the problem with a smaller value of  $\lambda$  and, if the new value of  $\lambda$  is not much smaller than the previous value, then the new solution is typically obtained in just a few iterations, because the new solution has only a few extra nonzero elements. Warm starts can also be used to initiate different subproblems for the leave-one-out validation approach, where the structure learning is performed on  $n$  subsets of the data (one sample being left out each time), so that the stability of the solution can be evaluated. Since each leave-one-out subproblem differs from another one by a rank-two update of matrix  $A$ , and since the resulting nonzero pattern is expected to be not very different, the solution to one subproblem can be an efficient warm start for another subproblem.

Typically the output of the regularization path is evaluated via the ROC curves showing the trade-off between the number of true positive (TP) element recovered and the number of false positive (FP) elements. Producing better curves (where the number of TPs rises fast relative to FPs) is usually an objective of any method that does not focus on specific  $\lambda$  selection. An interesting property of SINCO is that it introduces nonzero entries to the matrix  $C$  as it progresses. Hence, if we use looser tolerance and stop the algorithm early, then we will observe fewer nonzero entries, hence a sparse solution for any specific value of  $\lambda$ . What we observe, as seen in Figure 11, is that if we apply

<sup>6</sup> For more details, see the StarPlus website

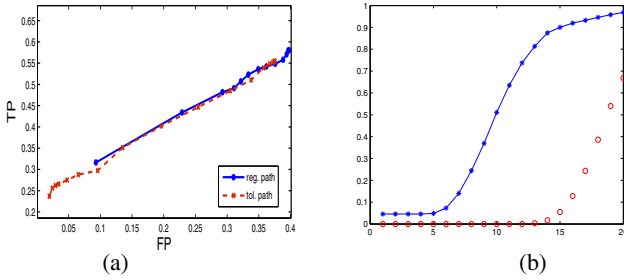
<http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-81/www/>





**Fig. 1. Scale-free networks:** SINCO and *glasso* paths when varying tolerance (tolerance path - blue 'o') and  $\lambda$  (lambda, or regularization, path - red 'x'). (a) and (b) show SINCO paths, on problems with (a)  $p = 100, n = 5000$  and (b)  $p = 100, n = 500$ , respectively; (c) and (d) show *glasso* paths on the same problems

SINCO to problem (2) with ever tighter tolerance (equivalent to observing the path of intermediate solutions) then the ROC curves obtained from the tolerance solution path match the ROC curves obtained from the regularization path. Here we show examples of the matching ROC curves for various networks with which we experimented. We use a randomly generated structured (scale-free) network that is 21% dense and a randomly generated unstructured network, 3% dense (due to space restriction, we only show the results for scale-free networks; random unstructured networks produce very similar results). We use  $p = 100$  and two instances:  $n = 500$  and  $n = 5000$ . We applied SINCO to one instance of problem (2) with  $\lambda = 0.01$  (very small regularization) with a range of stopping tolerances from  $10^{-4}$  to  $10^{-7}$ . The ROC curve of that path is presented by a line with "o"s. We also applied SINCO with fixed tolerance of  $10^{-6}$  to a range of  $\lambda$  values from 300 to 0.01. The corresponding ROC curves are denoted by lines with "x"s. We can see that the ROC curve of the regularization path for the given range of values of  $\lambda$  is somewhat less steep than that of the tolerance path, but the curves are still very close in the area where they overlap. For baseline we also present the ROC curve of the regularization path computed by *glasso*, which is very similar to the SINCO's ROC curves. Note that changing tolerance does not have the same effect on *glasso* as it does on SINCO. The number of TP and FP does not change noticeably with increasing tolerance. This is due to the fact that the algorithm in *glasso* updates a whole row and a column of  $C$  at each iteration while it cycles through the rows and columns, rather than selecting the updates in a greedy manner.



**Fig. 2.** SINCO’s tolerance path vs regularization path: (a) two path for the E.coli subnetwork; (b) comparing positives on scale-free networks

Figure 2a shows a very similar behavior when comparing the two paths for the DREAM-07 challenge problem of E.coli transcriptional network reconstruction, for  $n = 300$  microarray samples and a subset of  $p = 133$  transcription factors that form a connected component in the graph.

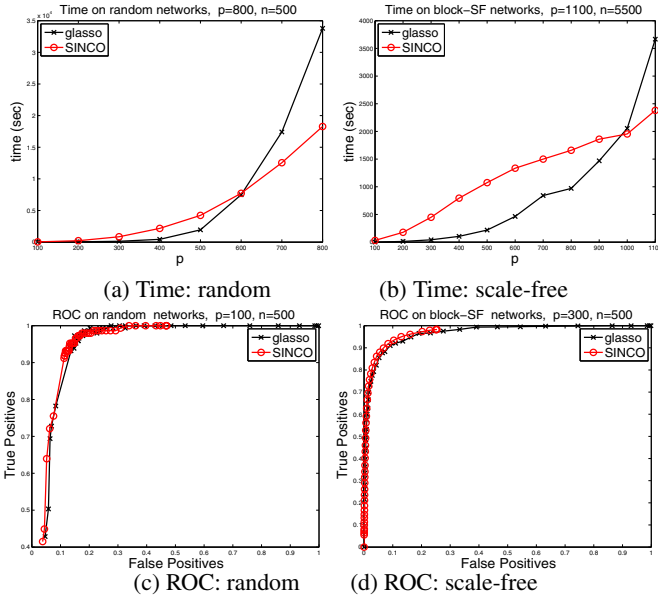
Note, however, that it is not always the case that SINCO’s solution path (tolerance path) is actually the same as the regularization path. In Figure 2b, the lower curve shows the percentage of the positives (nonzero entries in  $C$ ) in solutions from SINCO’s tolerance path which are *not* present in the solution on the regularization path. The higher curve represents the percentage of true positives; the  $x$  axis of the figure represent the points along the tolerance and regularization paths, which are matched to each other. We observe that the SINCO and the regularization paths largely coincide until the TP reach its maximum and further nonzeros are in the FP category and hence, in a way, are random noise.

Our observations imply that SINCO can be used to greedily select the elements of graphical model until the desired trade-off between FPs and TPs or the desired number of nonzero elements is achieved or the allocated CPU time is exhausted. In the limit SINCO solves the same problem as *lasso* and hence the limit number of the true and false positives is dictated by the choice of  $\lambda$ . But since the real goal is to recover the true nonzero structure of the covariance matrix, it is not necessary to solve problem (2) accurately. For the purpose of recovering a good TP/FP ratio one can apply the SINCO method, without the adjustments to  $\lambda$ .

We should note that computing the regularization path presented in our experiments is typically more efficient in terms of CPU time than computing the tolerance path; the largest computational cost lies in computing the tail of the path for smaller tolerances. On the other hand, the tolerance path appears to be more precise and exhaustive, in terms of possible FP/TP tradeoffs. It is also important to note that the entire tolerance path is automatically produced as a sequence of iterates produced by SINCO, while the regularization path can only be computed as a sequence of solutions for a given set of values of  $\lambda$ .

## 4.2 Empirical Complexity

Here we will discuss the empirical dependence of the runtime of the SINCO algorithm on the choice of stopping tolerance and the problem size  $p$ . We also investigate the effect

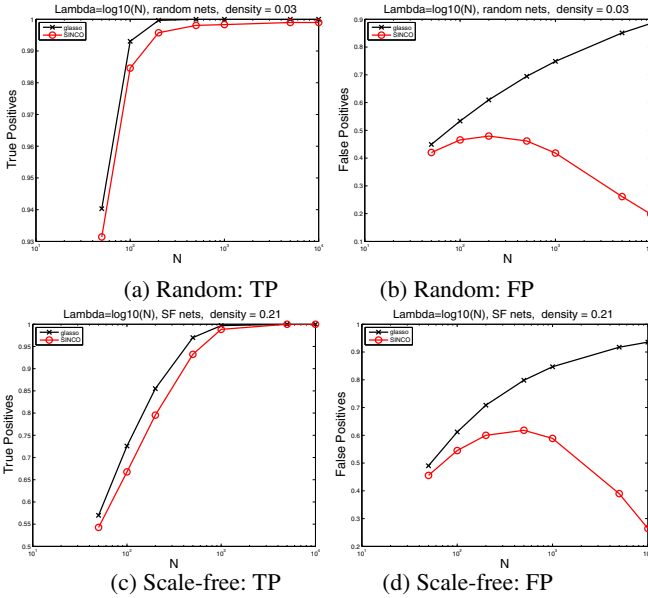


**Fig. 3.** CPU time: SINCO vs *glasso* on (a) random networks ( $n = 500$ , fixed range of  $\lambda$ ) and (b) scale-free networks (density 21%,  $n$  and  $\lambda$  scaled by the same factor with  $p$ ,  $n = 500$  for  $p = 100$ ). ROC curves: SINCO vs *glasso* on (c) random networks ( $p = 100$ ,  $n = 500$ , fixed range of  $\lambda$ ) and (d) scale-free networks ( $p = 300$ ,  $n = 1500$ , density 21%,  $n$  and  $\lambda$  scaled by the same factor with  $p$ , starting with  $n = 500$  for  $p = 100$ ).

increasing  $p$  has on the results produced by SINCO and *glasso*. Both methods were executed on Intel Core 2Duo T7700 processor (2.40GHz); note, however, that *glasso* is based on well-tuned Fortran code with an R interface, while SINCO a straight-forward C++ implementation of the algorithm in Section 3 with Matlab interface.

We consider the situation when  $p$  increases. If together with  $p$  the number of nonzeros in the true inverse covariance also increases, then to obtain a comparable problem we need to increase  $n$  accordingly. Increasing  $n$ , in turn, affects the contribution of  $\lambda$ , since the problem scaling changes. Here we chose to consider the following two simple settings, where we can account for these effects. In the first setting, we increase  $p$  while keeping the number of the off-diagonal nonzero elements in the randomly generated unstructured network constant (around 300). We do not, therefore, increase  $n$  or  $\lambda$ . The CPU time necessary to compute the entire path for  $\lambda$  ranging from 300 to 0.01 is plotted for  $p = 100, 200, 300, 500, 800$  and 1000 in Figure 3a. In the second case, we generated block-diagonal matrices of sizes  $p = 100, 200, 300, 500, 800, 1000$ , with  $100 \times 100$  diagonal blocks, each of which equals the inverse covariance matrix of a 21%-dense structured (scale-free) network from the previous section. Since the number of nonzero elements grows linearly with  $p$ , we increased  $n$  and the appropriate range of  $\lambda$  linearly as well. The CPU time for this case is shown in the last plot of Figure 3b.

The first two plots in Figure 3 shows that the CPU time (in seconds) for SINCO scales up more slowly than that of *glasso*, with increasing number of variables,  $p$ . The



**Fig. 4.** SINCO and *glasso* accuracy with growing  $n$ : (a) and (b) show the results averaged over 20 random networks ( $p = 100$ , density 3%), (c) and (d) show similar results averaged over 25 scale-free networks ( $p = 100$ , density 21%)

reason for the difference in scaling rates is evident in the ROC curves shown in Figures 3c and 3d, which demonstrate that, for similarly high true-positive rate, *glasso* tends to have much higher false-positive rate than SINCO, thus producing a less sparse solution, overall.

### 4.3 Asymptotic Behavior with Increasing $\lambda$

Finally, we investigate the behavior of SINCO for a fixed value of  $p$  as  $n$  grows. In this setting, we expect to obtain larger TP values and smaller FP error with increasing  $n$ . The consistency result in [21] suggests that for our formulation, to obtain an accurate asymptotic structure recovery, we should pick  $\lambda$  that grows with  $n$ , but so that its growth is slower than  $\sqrt{n}$ .

Here we use  $\lambda = \log_{10}(n)$ . We again apply our algorithm and *glasso* to the 21%-dense scale-free networks with  $p = 100$ . In Figure 4 we show the how the value of TP and FP returned by the two algorithms changes with growing  $n$  (note that  $\lambda$  is kept fixed for each value of  $n$ ). We observe that SINCO achieves in the limit nearly 0% false-positive error and nearly 100% true-positive rate, while *glasso*'s FP error grows with increasing  $n$ . This result is, again, a consequence of the greedy approach utilized by SINCO.

### 4.4 Application to fMRI Analysis

Here we describe the results of applying SINCO to a real-life data, where the “ground truth” network structure was not available; thus, we could not measure the structure

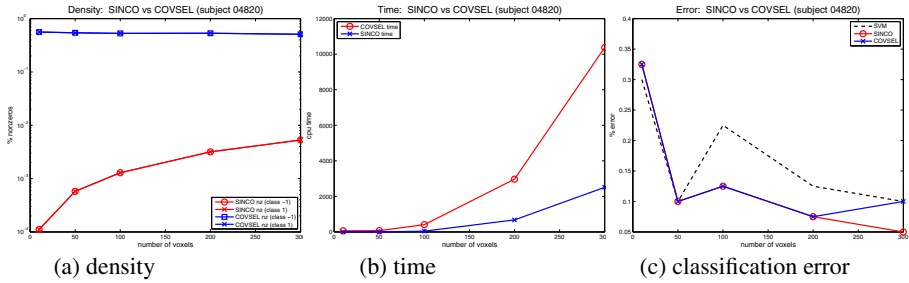


Fig. 5. SINCO vs. COVSEL on fMRI data

reconstruction accuracy, and instead evaluated the prediction accuracy of the resulting Markov networks. In this section, we compare SINCO with COVSEL [11] rather than *glasso*, since COVSEL is the other available Matlab implementation solving the same dual problem as *glasso* (as opposed to SINCO solving the primal one), and, although SINCO was shown to be slower than *glasso* [7], the objective here was rather to compare the prediction accuracy of the two approaches and the density of solutions.

We used fMRI data for the mind-state prediction problem described in [12]. The data consists of a series of trials in which the subject is being shown either a picture (+1) or a sentence (-1). Our dataset consists of 1700 to 2200 features, dependent on a particular subject, and 40 samples, where half of the samples correspond to the picture stimulus (+1) and the remaining half correspond to sentence stimulus (-1). (One sample corresponds to the averaged fMRI image over 6 scans sequentially taken while a subject is presented with a particular stimulus). We used leave-one-out cross-validation, and report average results over 40 cross-validation folds, each corresponding to one sample left out for testing, and the remaining 39 used for training.

For each class  $Y = \{-1, 1\}$ , we learn a sparse Markov Net model that provides us with an estimate of the Gaussian conditional density  $p(\mathbf{x}|y)$ , where  $\mathbf{x}$  is the feature (voxel) vector; on the test data, we choose the most-likely class label  $\arg \max_y p(\mathbf{x}|y)P(y)$  for each unlabeled test sample  $\mathbf{x}$ .

Figure 5 show the results of comparing SINCO versus COVSEL for one of the subjects in the above study (similar results were obtained for two more subjects). We observe that SINCO produces classifiers that are equally (or more) accurate than those produced by COVSEL (Figure 5c), but is much faster (Figure 5b) and uses much sparser Markov Net models (Figure 5a), which suggests that COVSEL (and hence *glasso*, since they are different implementations of the same approach) learns many links that are not essential for the discriminative ability of the classifier [8].

<sup>7</sup> For more details, see the StarPlus website

<http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-81/www/>

<sup>8</sup> It is also interesting to note that both Markov Net classifiers are competitive with, and often more accurate than the (linear) SVM classifier (Figure 5c). Herein, we used the SVM code by A. Schwaighofer available at

<http://ida.first.fraunhofer.de/~anton/software.html>

## 5 Discussion

We propose a novel approach, called SINCO, for solving the sparse inverse-covariance selection problem, which is equivalent to learning the structure (and parameters) of a Gaussian MRF. Our method is very simple; it uses greedy coordinate ascent, efficiently performing each evaluation step in a constant time, by solving a quadratic equation. We also provide a convergence proof. The method we present has two major advantages: (1) natural tendency to preserve the sparsity of solution, leading to better true-positive vs. false-positive error rate trade-off, especially on sparse problems, and (2) potential for a straightforward massive parallelization that could provide a significant  $O(p^2)$  speedup at each iteration. Also, our method has interesting (and useful) property of replicating the regularization path *behavior* (although not necessarily replicating the actual regularization path) by applying the method to one (sufficiently small) instance of the regularization parameter  $\lambda$  only. Thus, a desired number of network links can be obtained directly from the greedy solution path, without having to tune the  $\lambda$  parameter. SINCO properties are evaluated on a range of randomly generated problems, as well as on two real-life applications including gene-network reconstruction and neuroimaging. A important direction for future work is a more detailed investigation of the near-solution space of the sparse inverse covariance problem considered herein, and a better characterization of the relation between the objective function near its optimum and the variance in the structure of potential solutions.

## Appendix

Herein we present the derivation of the SINCO algorithm.

As mentioned in Section [3.2](#), the maximum of the one-dimensional function in Step 3 of SINCO is available in closed form. Indeed, consider the step  $\bar{C}' = C' + \theta(e_i e_j^T + e_j e_i^T)$ . Let us assume that  $\theta > 0$  and that  $C''_{ij} = 0$ , which implies that we can write the step as  $\bar{C} = C + \theta(e_i e_j^T + e_j e_i^T)$ , since the  $(i, j)$  and  $(j, i)$  elements do not become zero for any such step.

The inverse  $W$ , then, is updated, according to the Sherman-Morrison-Woodbury formula  $(X + ab^T)^{-1} = X^{-1} - X^{-1}a(1 + b^T X^{-1}a)^{-1}b^T X^{-1}$ , as follows:

$$\bar{W} = W - \theta(\kappa_1 W_i W_j^T + \kappa_2 W_i W_i^T + \kappa_3 W_j W_j^T + \kappa_1 W_j W_i^T),$$

$$\kappa_1 = -(1 + \theta W_{ij})/\kappa, \quad \kappa_2 = \theta W_{jj}/\kappa, \quad \kappa_3 = \theta W_{ii}/\kappa,$$

$$\kappa = \theta^2(W_{ii} * W_{jj} - W_{ij}^2) - 1 - 2\theta W_{ij}.$$

Let us now compute the objective function as the function of  $\theta$ :

$$f(\theta) = \frac{n}{2}(\ln \det(C + \theta e_i e_j^T + \theta e_j e_i^T)) - \text{tr}(A(C + \theta e_i e_j^T + \theta e_j e_i^T)) - \|C + \theta e_i e_j^T + \theta e_j e_i^T\|_S.$$

We use the following property of the determinant:

$$\det(X + ab^T) = \det(X)(1 + b^T X^{-1}a)$$

and the Scherman-Morrisson-Woodbury formula. We have

$$\begin{aligned} \det(C + \theta e_i e_j^T + \theta e_j e_i^T) &= \det(C + \theta e_j e_i^T)(1 + \theta e_j^T (C + \theta e_j e_i^T)^{-1} e_i) = \\ \det(C)(1 + \theta e_i^T C^{-1} e_j)(1 + \theta e_j^T C^{-1} e_i - \theta^2 e_j^T C^{-1} e_j (1 + \theta e_i^T C^{-1} e_j)^{-1} e_i^T C^{-1} e_i) &= \\ \det(C)(1 + 2\theta e_i^T C^{-1} e_j + (\theta e_j^T C^{-1} e_i)^2 - \theta^2 e_i^T C^{-1} e_i e_j^T C^{-1} e_j). \end{aligned}$$

Given the dual solution  $W = C^{-1}$ , and recalling that  $W$  and  $A$  are symmetric, but  $S$  is not necessarily so, we can write the above as

$$\det(C + \theta e_i e_j^T + \theta e_j e_i^T) = \det(C)(1 + 2\theta W_{ij} + \theta^2(W_{ij}^2 - W_{ii}W_{jj})).$$

Then the change in the objective function is

$$f(\theta) - f = \frac{n}{2}(\ln(1 + 2\theta W_{ij} + \theta^2(W_{ij}^2 - W_{ii}W_{jj})) - 2A_{ij}\theta) - S_{ij}\theta - S_{ji}\theta,$$

the last term being derived from the fact that  $C_{ij} + \theta$  and  $C_{ji} + \theta$  remain positive. Let us now consider the derivative of the objective function with respect to  $\theta$

$$\frac{df(\theta)}{d\theta} = \frac{nW_{ij} + n\theta(W_{ij}^2 - W_{ii}W_{jj})}{\theta^2(W_{ij}^2 - W_{ii}W_{jj}) + 1 + 2\theta W_{ij}} - nA_{ij} - S_{ij} - S_{ji}.$$

To find the maximum of  $f(\theta)$  we need to find  $\theta > 0$  for which  $\frac{df(\theta)}{d\theta} = 0$ . Letting  $a$  denote  $W_{ii}W_{jj} - W_{ij}^2$ , this condition can be written as:

$$nW_{ij} - nA_{ij} - S_{ij} - S_{ji} - (na + 2W_{ij}(nA_{ij} + S_{ij} + S_{ji})\theta + a(nA_{ij} + S_{ij} + S_{ji})\theta^2) = 0.$$

To find the value of  $\theta$  for which the derivative of the objective function equals zero we need to solve the above quadratic equation

$$ab\theta^2 - (na + 2W_{ij}b)\theta + nW_{ij} - b = 0, \tag{4}$$

where  $a = W_{ii}W_{jj} - W_{ij}^2$  and  $b = nA_{ij} + S_{ij} + S_{ji}$ . Notice that  $a$  is always nonnegative, because matrix  $W$  is positive definite, and it equals zero only when  $i = j$ . We know that at  $\theta = 0$   $\frac{df(\theta)}{d\theta} > 0$ . Let us investigate what happens when  $\theta$  grows. The discriminant of the quadratic equation is

$$\begin{aligned} D &= (na + 2W_{ij}b)^2 - 4ab(nW_{ij} - b) = (na)^2 + 4nW_{ij}ab + 4W_{ij}^2b^2 - 4abnW_{ij} + 4ab^2 \\ &= (na)^2 + 4b^2W_{ii}W_{jj} > 0, \end{aligned}$$

hence the quadratic equation always has a solution. At  $\theta = 0$  the quadratic function equals

$$nW_{ij} - nA_{ij} - S_{ij} - S_{ji} = G'_{ij} + G'_{ji} > 0.$$

Now let us again consider the derivative  $\frac{df(\theta)}{d\theta}$ . At  $\theta = 0$  we know that the derivative is positive. We also know that the denominator

$$\theta^2(W_{ij}^2 - W_{ii}W_{jj}) + 1 + 2\theta W_{ij} = (1 + \theta W_{ij})^2 - \theta^2 W_{ii}W_{jj}$$

is positive when  $\theta = 0$  and is equal to zero when  $\theta = \theta_{max} = 1/(\sqrt{W_{ii}W_{jj}} - W_{ij}) > 0$ . The function  $f(\theta)$  approaches negative infinity when  $\theta \rightarrow \theta_{max}$ , hence so does  $\frac{df(\theta)}{d\theta}$ . This implies that  $\frac{df(\theta)}{d\theta}$  has to reach the value zero for some  $\theta \in (0, \theta_{max})$ . Hence the quadratic equation (4) has one positive solution in this interval. This solution gives us the maximum of  $f(\theta)$  and hence the length of the step along the direction  $e_i e_j^T + e_j e_i^T$ .

The objective function value is easy to update using the formula

$$\det(C' - C'' + \theta(e_i e_j^T + e_j e_i^T)) = \det(C' - C'')(1 + 2\theta W_{ij} - \theta^2 a).$$

Let us consider the negative step along the direction  $e_i e_j^T + e_j e_i^T$  when  $C'_{ij} > 0$ . The derivations are exactly as above, except for we are now looking for solution  $\theta < 0$ . As discussed above, the term under the logarithm

$$\theta^2(W_{ij}^2 - W_{ii}W_{jj}) + 1 + 2\theta W_{ij} = (1 + \theta W_{ij})^2 - \theta^2 W_{ii}W_{jj}$$

is positive when  $\theta = 0$  and is also equal to zero when  $\theta = \theta_{min} = -1/(\sqrt{W_{ii}W_{jj}} + W_{ij}) < 0$ . The derivative of  $f(\theta)$  at  $\theta = 0$  is negative, this time (which is why we are considering a negative step, in the first place), which means that there exists a  $\theta \in (\theta_{min}, 0)$  for which this derivative is zero, hence the quadratic equation (4) has a negative solution. This negative solution  $\theta_- < 0$  determines the length of the step in the direction  $-e_i e_j^T - e_j e_i^T$ . It is important to note that the length of the step cannot exceed the value  $C'_{ij}$ , hence the final step length is computed as  $\max(\theta_-, -C'_{ij})$ .

The other two possible steps listed in Step 3 can be analyzed analogously, the main difference being the sign before the terms  $nA_{ij}$ ,  $S_{ij}$  and  $S_{ji}$  in the case of the step that updates  $C''$ .

Each step can be computed by a constant number of arithmetic operations, hence to find the step that provides the largest function value improvement it takes  $O(p^2)$  operations - the same amount of work (up to a constant) that it takes to update  $W$  and the gradient after one iteration. Hence the overall per-iteration complexity is  $O(p^2)$ . Moreover, this algorithms lends itself readily to massive parallelization, as discussed earlier in Section 3.2.

The convergence of the method follows from the convergence of a block-coordinate descent method on a strictly convex objective function. The only constraints are box constraints (nonnegativity) and they do not hinder the convergence. In fact we can view our method as a special case of the row by row (RBR) method for SDP described in [19]. In the case of SINCO we extensively use the fact that each coordinate descent step is cheap and, unlike the RBR algorithm, we select the next step based on the best function value improvement. On the other hand, we maintain the inverse matrix  $W$ , which RBR



method does not. However, none of these differences prevent the convergence result for RBR in [19] to apply to our method. Hence the convergence to the optimal solution holds for SINCO.

## References

1. Banerjee, O., El Ghaoui, L., d'Aspremont, A.: Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Journal of Machine Learning Research* 9, 485–516 (2008)
2. Barabasi, A.-L., Albert, R.: Emergence of scaling in random networks. *Science* 286, 509–512 (1999)
3. Buhl, S.L.: On the existence of maximum likelihood estimators for graphical gaussian models. *Scandinavian Journal of Statistics* 20(3), 263–270 (1993)
4. Dempster, A.P.: Covariance selection. *Biometrics* 28(1), 157–175 (1972)
5. Duchi, J., Gould, S., Koller, D.: Projected subgradient methods for learning sparse gaussians. In: *Proc. of UAI-08* (2008)
6. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least angle regression. *Ann. Statist.* 32(1), 407–499 (2004)
7. Friedman, J., Hastie, T., Tibshirani, R.: Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* (2007)
8. Lauritzen, S.: *Graphical Models*. Oxford University Press, Oxford (1996)
9. Lu, Z.: Smooth optimization approach for sparse covariance selection. *SIAM Journal on Optimization* 19(4), 1807–1827 (2009)
10. Marlin, B., Murphy, K.: Sparse gaussian graphical models with unknown block structure. In: *Proc. of ICML-09* (2009)
11. Meinshausen, N., Bühlmann, P.: High dimensional graphs and variable selection with the Lasso. *Annals of Statistics* 34(3), 1436–1462 (2006)
12. 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, 145–175 (2004)
13. Ravikumar, P., Wainwright, M., Raskutti, G., Yu, B.: Model selection in Gaussian graphical models: High-dimensional consistency of  $\ell_1$ -regularized MLE. In: *NIPS-08* (2008)
14. Rothman, A., Bickel, P., Levina, E., Zhu, J.: Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics* (2), 494–515 (2008)
15. Schmidt, M., van den Berg, E., Friedlander, M., Murphy, K.: Optimizing costly functions with simple constraints: A limited-memory projected quasi-newton algorithm. In: *Proc. of AISTATS-09* (2009)
16. Stolovitzky, G., Prill, R.J., Califano, A.: Lessons from the dream2 challenges. *Annals of the New York Academy of Sciences* (1158), 159–195 (2009)
17. Tibshirani, R.: Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society, Series B* 58(1), 267–288 (1996)
18. Wainwright, M., Ravikumar, P., Lafferty, J.: High-Dimensional Graphical Model Selection Using  $\ell_1$ -Regularized Logistic Regression. In: *NIPS 19*, pp. 1465–1472 (2007)
19. Wen, Z., Goldfarb, D., Ma, S., Scheinberg, K.: Row by row methods for semidefinite programming. Technical report, Department of IEOR, Columbia University (2009)
20. Wolkowicz, H., Saigal, R., Vanenberghe, L. (eds.): *Handbook of Semidefinite Programming*. Kluwer Academic Publishers, Dordrecht (2000)
21. Yuan, M., Lin, Y.: Model Selection and Estimation in the Gaussian Graphical Model. *Biometrika* 94(1), 19–35 (2007)

# Online Structural Graph Clustering Using Frequent Subgraph Mining

Madeleine Seeland, Tobias Girschick, Fabian Buchwald, and Stefan Kramer

Technische Universität München,  
Institut für Informatik/I12,  
Boltzmannstr. 3, 85748 Garching b. München, Germany  
{madeleine.seeland,tobias.girschick,  
fabian.buchwald,stefan.kramer}@in.tum.de

**Abstract.** The goal of graph clustering is to partition objects in a graph database into different clusters based on various criteria such as vertex connectivity, neighborhood similarity or the size of the maximum common subgraph. This can serve to structure the graph space and to improve the understanding of the data. In this paper, we present a novel method for structural graph clustering, i.e. graph clustering without generating features or decomposing graphs into parts. In contrast to many related approaches, the method does not rely on computationally expensive maximum common subgraph (MCS) operations or variants thereof, but on frequent subgraph mining. More specifically, our problem formulation takes advantage of the frequent subgraph miner gSpan (that performs well on many practical problems) without effectively generating thousands of subgraphs in the process. In the proposed clustering approach, clusters encompass all graphs that share a sufficiently large common subgraph. The size of the common subgraph of a graph in a cluster has to take at least a user-specified fraction of its overall size. The new algorithm works in an online mode (processing one structure after the other) and produces overlapping (non-disjoint) and non-exhaustive clusters. In a series of experiments, we evaluated the effectiveness and efficiency of the structural clustering algorithm on various real world data sets of molecular graphs.

## 1 Introduction

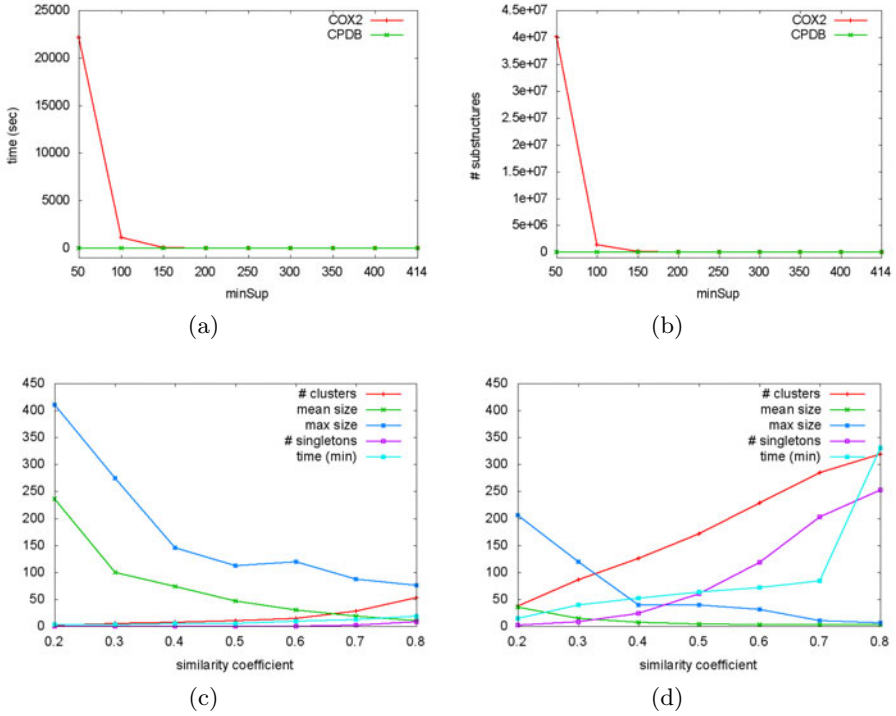
Mining graph data has attracted a lot of attention in the past ten years [1–3]. One family of methods is concerned with mining subgraph patterns in graph databases [1, 2]. The criteria for interestingness are often based on the support of a pattern in the graph database, e.g., requiring a minimum and/or maximum frequency, closedness, freeness or class-correlation. However, in all of these cases, the structural diversity of graph databases, i.e. the existence of groups of similar or dissimilar graphs, is not explicitly taken into account or revealed by the algorithm. Vice versa, the structural composition and existence of groups of similar graphs has a serious impact on the output and runtime performance of

pattern mining algorithms. To gain insights into the structural characteristics of graph data sets, we developed a graph clustering algorithm that discovers groups of structurally similar and dissimilar graphs. The algorithm can be practically useful for a variety of purposes: benchmarking other graph mining algorithms, descriptor calculation (e.g., for QSAR studies), computing local models for classification or regression (e.g., one per cluster) and calculation of the so-called applicability domain of models.

To illustrate the impact of structural diversity on graph mining results, we consider two data sets of molecular graphs of the same size and with approximately the same number of atoms per molecule. The first data set, the COX2 data set (<http://pubs.acs.org/doi/suppl/10.1021/ci034143r>), contains 414 compounds, which possess a relatively high structural homogeneity. The second data set is a subset of the CPDB data set (<http://potency.berkeley.edu/>) that matches the COX2 data both in the number of structures and the number of atoms per structure. The results of a typical graph mining representative, gSpan, and the results of the graph clustering algorithm presented in this paper are shown in Figure 1. In the upper part of the figure, we see the huge difference in the runtime and the number of discovered patterns. For structurally homogeneous data (COX2), the number of patterns and runtime explodes, whereas for structurally heterogeneous data (CPDB) the algorithm behaves as expected. The reason for this difference in performance becomes evident in the graph clustering results in the lower part of Figure 1. As can be seen, there is a small number of large clusters in COX2 and a large number of small clusters in CPDB (for each value of a parameter that is varied on the x-axis). This indicates a high degree of structural homogeneity in COX2 and a low degree in CPDB, and also hints at the usefulness of graph clustering to make the characteristics of a graph database explicit.

The graph clustering algorithm presented in this paper operates directly on the graphs, i.e. it does not require the computation of features or the decomposition into subgraphs. It works online (processing one graph after the other) and creates a non-disjoint and non-exhaustive clustering: graphs are allowed to belong to several clusters or no cluster at all. One important component of the algorithm is a variant of gSpan to determine cluster membership. Thus, the proposed graph clustering approach is based on a practically fast graph mining algorithm and not on typically time-consuming maximum common subgraph (MCS) operations [4]. In contrast to another graph clustering approach based on graph pattern mining [5], the (often quite numerous) frequent subgraphs are just by-products, and not part of the output of the algorithm: the actual output consists just of the clustered graphs sharing a common scaffold.

The remainder of the paper is organized as follows. In Section 2 the methodology of our structure-based clustering algorithm is introduced. Section 3 presents a description of the data sets and experiments as well as an interpretation of the results. In Section 4 related work is discussed before Section 5 gives a conclusion and an outlook to future work.



**Fig. 1.** (Above) (a) Runtime behavior and (b) number of subgraphs for gSpan on COX2 and CPDB. (Below) Results of structural clustering on (c) COX2 and (d) CPDB.

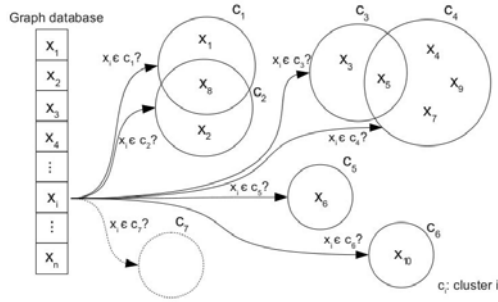
## 2 Structural Clustering

The following section presents the structural clustering algorithm that can be used to cluster graph instances based on structural similarity. Starting with some definitions from graph theory, we present the problem definition and the algorithm in detail.

### 2.1 Notation and Definitions

In the following, all graphs are assumed to be labeled, undirected graphs. To be more precise, a graph and its subgraphs are defined as follows: A labeled *graph* is represented as a 4-tuple  $g = (V, E, \alpha, \beta)$ , where  $V$  is a set of vertices and  $E \subseteq V \times V$  is a set of edges representing connections between all or some of the vertices in  $V$ .  $\alpha : V \rightarrow L$  is a mapping that assigns labels to the vertices, and  $\beta : V \times V \rightarrow L$  is a mapping that assigns labels to the edges. Given two labeled graphs  $g = (V, E, \alpha, \beta)$  and  $g' = (V', E', \alpha', \beta')$ ,  $g'$  is a *subgraph* of  $g$ , ( $g' \subseteq g$ ) if:

- $V' \subseteq V$
- $E' \subseteq E$



**Fig. 2.** Schematic overview of the cluster membership assignment for instance  $x_i$ . Graph instances are represented by  $x_1, \dots, x_n$ , clusters by  $C_1, \dots, C_k$ .

- $\forall x \in V' : \alpha'(x) = \alpha(x)$
- $\forall (x, y) \in V' \times V' : \beta'((x, y)) = \beta((x, y))$

Given two arbitrary labeled graphs  $g_1 = (V_1, E_1, \alpha_1, \beta_1)$  and  $g_2 = (V_2, E_2, \alpha_2, \beta_2)$ , a *common subgraph* of  $g_1$  and  $g_2$ ,  $cs(g_1, g_2)$ , is a graph  $g = (V, E, \alpha, \beta)$  such that there exists a *subgraph isomorphism* from  $g$  to  $g_1$  and from  $g$  to  $g_2$ . This can be generalized to sets of graphs. The set of common subgraphs of a set of graphs  $\{g_1, \dots, g_n\}$  is then denoted by  $cs(\{g_1, \dots, g_n\})$ . Moreover, given two graphs  $g_1$  and  $g_2$ , a graph  $g$  is called a *maximum common subgraph* of  $g_1$  and  $g_2$  if  $g$  is a common subgraph of  $g_1$  and  $g_2$  and there exists no other common subgraph of  $g_1$  and  $g_2$  that has more vertices than  $g$ . Finally, we define the size of a graph as the number of its vertices, i.e.  $|V|$ .

### 2.2 Problem Definition

Structural clustering is the problem of finding groups of graphs sharing some structural similarity. Instances with similar graph structures are expected to be in the same cluster provided that the common subgraphs match to a satisfactory extent. Only connected subgraphs are considered as common subgraphs. The similarity between graphs is defined with respect to some user-defined size threshold. The threshold is set such that the common subgraphs shared among a query graph and all cluster instances make up a specific proportion of the size of each graph. A graph is assigned to a cluster provided that there exists at least one such common subgraph whose size is equal or bigger than the threshold. In this way, an object can simultaneously belong to multiple clusters (overlapping clustering) if the size of at least one common subgraph with these clusters is equal or bigger than the threshold. If an object does not share a common subgraph with any cluster that meets the threshold, this object is not included in any cluster (non-exhaustive clustering). A graphical overview is shown in Figure 2. For one graph after the other, it is decided whether it belongs to an existing cluster or whether a new cluster is created.

Formally, we frame the problem of structural clustering as follows. Given a set of graph objects  $X = \{x_1, \dots, x_n\}$ , we need to assign them into clusters which may overlap with each other. In clustering these objects, one objective is considered: to maximize the average number of objects contained in a cluster, such that at any time for each cluster  $C$  there exists at least one common subgraph that makes up a specific proportion,  $\theta$ , of the size of each cluster member. Considering the state of a cluster  $C = \{x_1, \dots, x_m\}$ <sup>1</sup> at any point in time, the criterion can formally be defined as:

$$\exists s \in cs(\{x_1, \dots, x_m\}) \forall x_i \in C : |s| \geq \theta |x_i| \quad (1)$$

where  $s$  is a subgraph and  $\theta \in [0, 1]$  is a user-defined similarity coefficient. According to this goal, a minimum threshold for the size of the common subgraphs shared by the query graph  $x_{m+1}$  and the instances in cluster  $C$  can be defined as

$$minSize = \theta \max(|x_{max}|, |x_{m+1}|), \quad (2)$$

where  $\theta \in [0, 1]$  and  $x_{max}$  is the largest graph instance in the cluster. To obtain meaningful and interpretable results, the minimum size of a graph considered for cluster membership is further constrained by a *minGraphSize* threshold. Only graphs whose size is greater than *minGraphSize* are considered for clustering. Thus, the identification of the general cluster scaffold will not be impeded by the presence of a few graph structures whose scaffold is much smaller than the one the majority of the cluster members share. This will be especially useful in real-world applications that often contain small fragments (see the minimum size column in Table 1).

### 2.3 Algorithm

The clustering algorithm works as follows. Let *minGraphSize* be the minimum threshold for the graph size and *minSize* be the minimum threshold for the size of the common subgraphs specified by the user and defined in Equation 2. In the first step, an initial cluster is created containing the first graph object that is larger than *minGraphSize*. In the following steps, each instance is compared against all existing clusters. In case the query instance meets the *minGraphSize* threshold and shares at least one common subgraph with one or more clusters that meets the cluster criterion in Equation 2, the instance is added to the respective cluster. Unlike many traditional clustering algorithms, an object is allowed to belong to no cluster, since it is possible that an object is not similar to any cluster. Thus, in this case, a new singleton cluster is created containing the query instance. The proposed clustering algorithm has two main advantages over many clustering algorithms. First, the algorithm works in an online mode, since it does not keep all the examples in memory at the same time, but processes them one by one in a single pass. Second, in contrast to many clustering algorithms which assume that the number of clusters is known beforehand, our algorithm does not require the specification of the number of clusters a priori.

<sup>1</sup> In slight abuse of notation, we use the same indices as above.

---

**Algorithm 1.** Structural Clustering

---

```

1: // graph[] - array of n graphs to be clustered
2: //  $\theta$  - similarity coefficient ( $\theta \in [0, 1]$ )
3: // minGraphSize - minimum graph size
4: procedure SC(graph[],  $\theta$ , minGraphSize)
5:   // Initialize the clustering with graph[0]
6:   clusters[]  $\leftarrow \infty$ 
7:   // loop over all graphs
8:   for ( $j \leftarrow 0, n$ ) do
9:     hasCluster  $\leftarrow$  false
10:    if (graph[ $j$ ]  $\geq$  minGraphSize) then
11:      // compare graph against all existing clusters
12:      for all  $c \in$  clusters[] do
13:        minSize  $\leftarrow \theta \cdot \max(\text{size}(\text{graph}[j]), \text{size}(c.\text{max}))$ 
14:        // check for cluster exclusion criteria defined in Equation 3 and 4
15:        if (3) || (4) then
16:          continue
17:        else
18:          minSup  $\leftarrow c.\text{size} + 1$ 
19:          // add graph[ $j$ ] to cluster  $c$  if gSpan finds at least one
20:          // common subgraph that meets the minSize threshold
21:          if gSpan'(graph[ $j$ ]  $\cup$   $c.\text{graphs}$ , minSup, minSize) then
22:             $c[\text{last} + 1] \leftarrow \text{graph}[j]$ 
23:            hasCluster  $\leftarrow$  true
24:          end if
25:        end if
26:      end for
27:      // create new cluster if the graph was not clustered
28:      if (hasCluster = false) then
29:        clusters[ $\text{last} + 1$ ]  $\leftarrow$  newCluster(graph[ $j$ ])
30:      end if
31:    end if
32:  end for
33: end procedure

```

---

The pseudocode for the structural clustering algorithm is shown in Algorithm 1.

For computing common subgraphs, we use a modified version of the graph mining algorithm gSpan [2] that mines frequent subgraphs in a database of graphs satisfying a given minimum frequency constraint. In this paper, we require a minimum support threshold of  $\text{minSup} = 100\%$  in a set of graphs, i.e. all common subgraphs have to be embedded in all cluster members. For our experiments with molecular graph data, we use gSpan', an optimization of the gSpan algorithm for mining molecular databases (<http://wwwkramer.in.tum.de/projects/gSpan.tgz>). Since we do not need to know all common subgraphs of a set of graphs, but rather only want to find out if there exists at least one common subgraph that meets the minimum size threshold defined in Equation 2, it is possible to terminate search once a solution

is found. Due to the structural asymmetry of the search tree (all descendants of a subgraph are generated before its right siblings are extended), it is thus possible to modify gSpan' such that the procedure exits immediately when a common subgraph is found that satisfies the minimum size threshold defined in Equation 2. In this way, a substantial improvement in runtime performance can be achieved. In the pseudocode, this modification of gSpan' is called gSpan". We introduced a special label for edges in cyclic graphs to ensure that cyclic graph structures are not subdivided any further. Moreover, we introduced the following two cluster exclusion criteria to avoid unnecessary calls to the gSpan" algorithm:

$$|x_{m+1}| > |x_{max}| \wedge minSize > |x_{min}| \quad (3)$$

$$|x_{m+1}| < |x_{min}| \wedge minSize > |x_{m+1}|, \quad (4)$$

where  $x_{min}$  is the smallest graph in cluster  $C$  and  $x_{m+1}$  and  $x_{max}$  are defined as above. Due to these exclusion criteria, graph instances which cannot fulfill the minimum subgraph size threshold are eliminated from further consideration. The first criterion (3) excludes too large query instances that would break up an existing cluster while the second one (4) excludes too small query instances. In case at least one of the two exclusion criteria is met, we omit the computation of the common subgraphs and continue with the next cluster comparison.

In summary, three factors contribute to the practically favorable performance of the approach: First, the use of a gSpan variant to compute a sufficiently large common subgraph, which is known to be effective on graphs of low density. Second, the possibility to terminate search as soon as such a subgraph is found. Third, the cluster exclusion criteria to avoid unnecessary runs of gSpan".

### 3 Experiments

To evaluate the the effectiveness and efficiency of the new structure-based clustering approach introduced in Section 2.3, we conducted several experiments on eight publicly available data sets of molecular graphs (Table I). In this section, we describe the experimental set-up and the results.

**Baseline Comparison with Fingerprint Clustering** The structure-based clustering algorithm was compared with a baseline clustering algorithm based on fingerprint similarity. The goal of this experiment is to determine if our algorithm is able to increase cluster homogeneity as compared to fingerprint clustering. Fingerprint-based similarities can be calculated extremely fast and have been found to perform reasonably well in practice. For the fingerprint calculation of the molecular graph data, the chemical fingerprints in Chemaxon's JChem Java package are used. The Tanimoto coefficient is used as similarity measure between fingerprints, since these fingerprints are equivalent to Daylight fingerprints (<http://www.daylight.com/dayhtml/doc/theory/theory.finger.html>) which were shown to work well in combination with the Tanimoto coefficient [4, 6].



**Table 1.** Overview of the data sets used for assessing the structural clustering method

Short-hand	$n$	min size	mean size	max size
<i>CPDBAS_MOUSE</i> <sup>a</sup>	444	2	13	64
<i>CPDBAS_RAT</i> <sup>a</sup>	580	2	14	90
<i>CYP</i> <sup>b</sup>	700	1	24	86
<i>NCIanti – HIV</i> <sup>c</sup>	36255	3	25	139
<i>SACA</i> <sup>d</sup>	107	5	27	79
<i>EPAFHM</i> <sup>e</sup>	580	2	10	55
<i>FDAMDD</i> <sup>f</sup>	1216	3	23	90
<i>RepDose</i> <sup>g</sup>	590	2	10	88

<sup>a</sup> [http://epa.gov/NCCT/dsstox/sdf\\_cpdbas.html](http://epa.gov/NCCT/dsstox/sdf_cpdbas.html) <sup>b</sup> <http://pubs.acs.org/doi/suppl/10.1021/ci0500536>

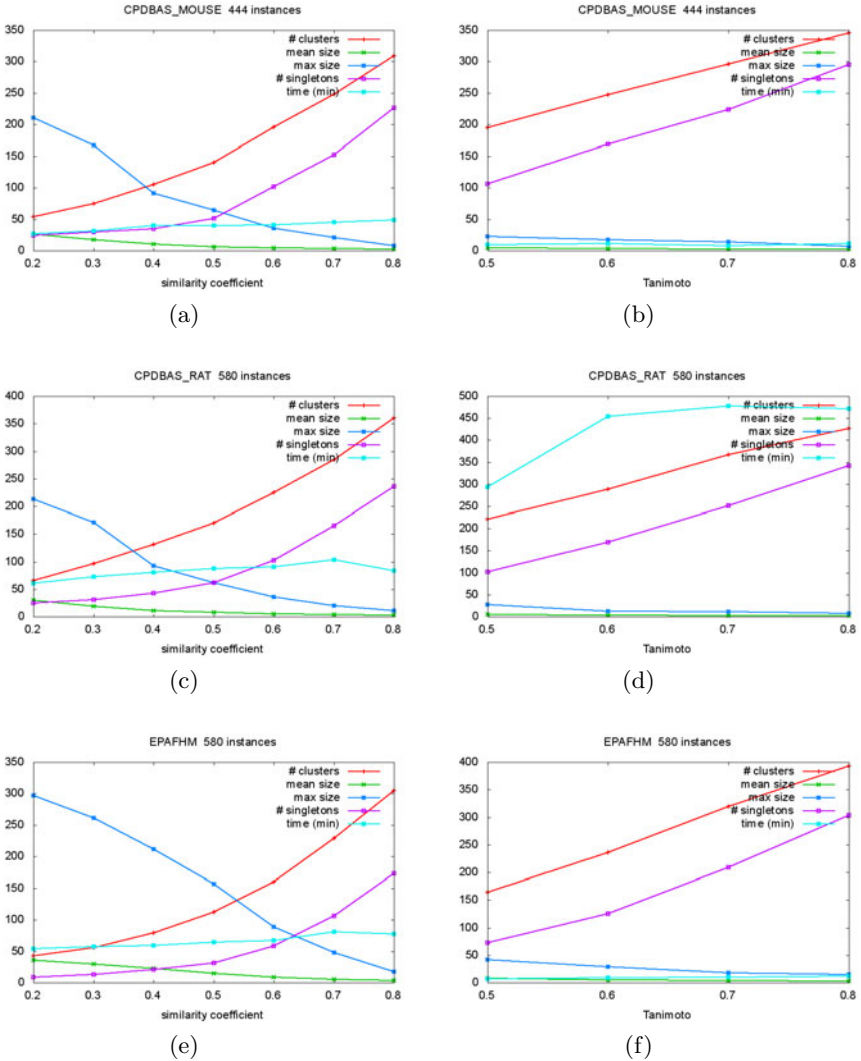
<sup>c</sup> [http://dtp.nci.nih.gov/docs/aids/aids\\_data.html](http://dtp.nci.nih.gov/docs/aids/aids_data.html) <sup>d</sup> <http://dtp.nci.nih.gov/docs/cancer/>

[searches/standard\\_mechanism.html](searches/standard_mechanism.html) <sup>e</sup> [http://epa.gov/NCCT/dsstox/sdf\\_epafhm.html](http://epa.gov/NCCT/dsstox/sdf_epafhm.html)

<sup>f</sup> [http://epa.gov/NCCT/dsstox/sdf\\_fdamdd.html](http://epa.gov/NCCT/dsstox/sdf_fdamdd.html) <sup>g</sup> <http://www.fraunhofer-repdose.de/>

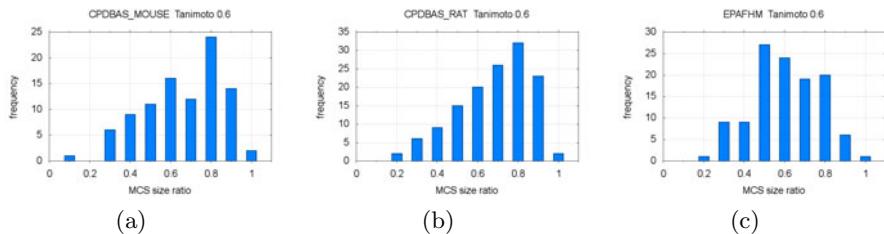
The fingerprint-based clustering (FP clustering) works as follows. Iteratively, each molecular graph is compared against all yet existing clusters. In case the query graph meets a predefined minimum graph size threshold, *minGraphSize*, and exceeds the minimum accepted Tanimoto similarity coefficient compared to each graph in the cluster, the query graph is added to the respective cluster; otherwise a new singleton cluster is created containing the query graph. As the FP does not provide a measure for the size of the shared subgraph between the FP cluster members, the common cluster scaffold is obtained by calculating the MCS common to all members in order to get a comparison metric for the proportion of the common subgraph. Note that this step can be omitted in our structural clustering approach, since the defined similarity coefficient provides a measure for the proportion of the common subgraph. In case a graph  $i$  is added to a cluster, the MCS between  $i$  and the current MCS of the cluster  $j$  is calculated. This MCS is iteratively reduced in size as it is compared to the new cluster members that may not share the entire subgraph. For the MCS calculation the maximum common edge subgraph algorithm was used which is implemented in Chemaxon’s JChem java package.

Our structural clustering approach was compared with the baseline FP clustering method on the data sets in Table 1. Due to space limitations, we present the results on three representative data sets, CPDBAS\_MOUSE, CPDBAS\_RAT and EPAFHM. In all experiments, we performed structural clustering for  $\theta \in [0.2, 0.8]$ . For FP clustering we used a Tanimoto coefficient value in the range of  $[0.4, 0.8]$ . Due to the different input parameters of both clustering approaches, it is not obvious how to compare the clustering results. However, the clustering statistics in Figure 3 suggest a correlation between the results from structural clustering for a similarity coefficient value of  $x$  ( $x \in [0, 1]$ ) and the results from FP clustering for a Tanimoto similarity value of  $y = x - 0.1$  ( $y \in [0, 1]$ ), due to similar clustering results in terms of the number of clusters, the number of singletons and the mean and maximum size of the clusters. Thus, we compare the clustering results from both algorithms with respect to this heuristic. Figure



**Fig. 3.** Results of structural clustering (a), (c), (e) vs. fingerprint clustering (b), (d), (f) on CPDBAS\_MOUSE, CPDBAS\_RAT and EPAFHM

**4** shows the histogram of the share of the MCS of the largest cluster instance for all non-singleton FP clusters for a sample Tanimoto coefficient value of 0.6. The results indicate that the MCS size proportions are, in many cases, below the according structural similarity coefficient  $\theta$ , which serves as a lower bound on the MCS size proportion. In contrast, each cluster obtained by structural clustering contains at least one common subgraph whose share of each cluster member is equal or bigger than  $\theta$ . The results suggest that, in comparison to FP clustering,



**Fig. 4.** Histogram of the share of the MCS of the largest cluster instance for fingerprint clustering on (a) CPDBAS\_MOUSE, (b) CPDBAS\_RAT and (c) EPAFHM using a Tanimoto coefficient value of 0.6.

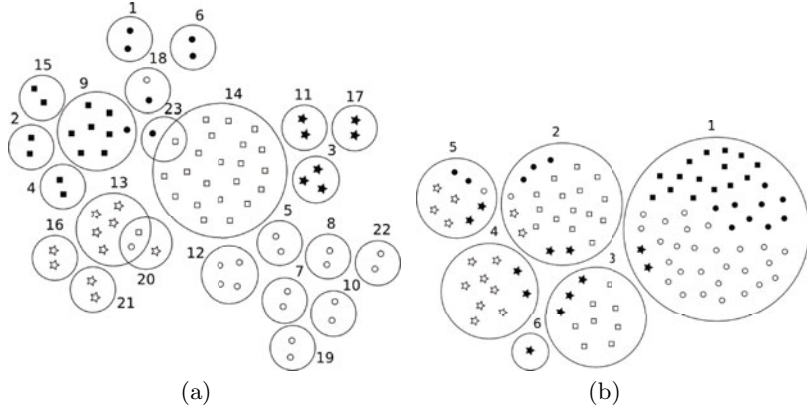
structural clustering provides a superior clustering with reduced heterogeneity in the individual clusters in the overall clustering.

**Qualitative Analysis of Structure-Based Clustering.** Cluster analysis was performed on the data set of 107 standard anti-cancer agents (SACA) whose class labels corresponding to their mechanisms of action have been clearly classified [7, 8]. The purpose of the experiment was to test if the clusters obtained by structural clustering were in good agreement with the known SACA class labels. As an external measure for clustering validation we used the Rand index to quantify the agreement of the clustering results with the SACA classes. Larger values for the Rand index correspond to better agreement between the clustering results and the SACA classes, with 1.0 indicating perfect concordance. Table 2 shows the Rand index values for different similarity coefficient values. Structural clustering clearly shows the peak point of the Rand index at  $\theta = 0.6$ . In the following, we present the clustering results for  $\theta = 0.6$  partitioning the 107 agents into 52 clusters. 23 of these clusters have at least two members, while the final 29 clusters consist of a single graph. Figure 5(a) gives a representation of the structural clusters with at least two instances in a hypothetical (non-Euclidean) two-dimensional (descriptor) space, where large circles represent clusters and dots, rectangles and stars denote cluster members according to the SACA classes. The results indicate that the clusters tend to be associated with certain SACA classes. Across different values for  $\theta$  we observed that with a higher similarity coefficient a finer but cleaner grouping of the structures at the cost of generating a larger number of smaller clusters is achieved. The graphs in each class are more cleanly discriminated from other graphs in the data set. Moreover, the clustering produces less overlapping clusters with internally higher structural similarity. In summary, structural clustering is capable of effectively grouping the 107 agents. Graphs instances from the same cluster not only share common subgraphs but are also strongly associated with specific SACA classes of mechanisms of action.

**Graph Clustering Comparison Method.** Our method was compared with a graph-based clustering based on variational Dirichlet process (DP) mixture models and frequent subgraph mining by Tsuda and Kurihara [5]. This clustering approach addresses the problem of learning a DP mixture model in the

**Table 2.** Number of clusters and Rand index values for structural clustering on SACA

$\theta$	0.02	0.05	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
# Clusters	6	7	11	22	32	39	48	52	60	66	82
Rand Index	0.408	0.436	0.515	0.765	0.827	0.854	0.863	0.869	0.866	0.848	0.833

**Fig. 5.** Results of (a) structural clustering for  $\theta = 0.6$  and (b) DP Clustering for  $\alpha = 0.1$  and  $m = 1000$  on the SACA data set. The different symbols for the cluster instances represent the six SACA classes.

high dimensional feature space of graph data. We investigated if the approach is also able to rediscover the known structure classes in the SACA database. In this experiment, we varied the number of features  $m$  from 50 to 5000 and set  $\alpha = 0.01, 0.1, 1, 10$ . Table 3 shows the experimental results for  $\alpha = 0.1$ . The results for  $\alpha = 0.01, 1, 10$  were similar. We observed that the number of clusters increases along with the number of features for  $m \leq 500$ ; for  $m > 500$  the number of clusters decreases significantly. Compared to structural clustering, DP clustering produces less clusters. In order to make the results more comparable to the results of our method, we varied the user-specified parameters. Nonetheless, it is impossible to parameterize the DP clustering method to obtain more than seven clusters. Figure 5(b) presents the clustering results for  $m = 1000$ , since Tsuda reported a good behavior of the algorithm for this value. Moreover, additional features can reveal detailed structure of the data. However, this advantage presents a disadvantage at the same time, since graph clusters with thousands of features are difficult to interpret. The DP clustering results indicate that the method is not able to discriminate the known structure classes in the SACA data set very well. In contrast to the results of structural clustering presented in Section 3, the DP clusters are, in many cases, associated with different structure classes, indicated by lower values of the Rand index (Table 3).

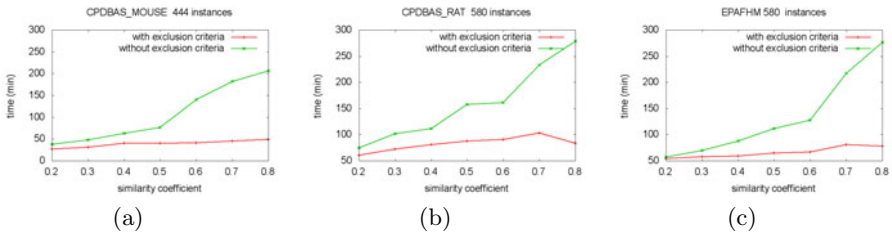
**Table 3.** Number of clusters and size of the DFS code tree for DP clustering on the SACA data set with  $\alpha = 0.1$ 

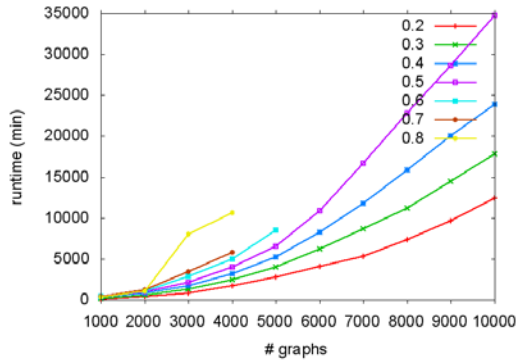
# Features	50	100	500	1000	5000
# Clusters	6	7	7	6	2
Rand Index	0.639	0.572	0.761	0.747	0.364

**Cluster Stability.** The outcome of the proposed structure-based clustering approach is dependent on the order in which the objects in the data set are processed. In this experiment, we studied the impact of the order of objects in the data sets by assessing the stability of the resulting clusters. We performed several experiments on data sets that are based on different permutations of a data set. The Tanimoto similarity coefficient was used as a cluster-wise measure of cluster stability, which defines the similarity between two clusters in terms of the intersection of common instances compared to the union of common instances. Our results suggest that structural clustering is sensitive to the particular order of the data sets. However, the obtained clusters are stable with respect to data permutations, since approximately 85% of the clusters of size  $\geq 2$  of a clustering yield a Tanimoto similarity value of 1 compared to the most similar cluster in a reference clustering. Taking also singletons into account, the similarity of the clusters rises to 94%.

**Performance with/without Cluster Exclusion Criteria.** We investigated the impact of the cluster exclusion criteria defined in Equation 3 and 4 on the performance of the structure-based clustering algorithm. Therefore, we ran the algorithm on the data sets in Table 1 with and without the exclusion criteria. Figure 6 shows the results of the experiment on three representative data sets, i.e. CPDBAS\_MOUSE, CPDBAS\_RAT and EPAFHM. The results indicate that a significant performance improvement can be achieved with the application of the cluster exclusion criteria.

**Scalability Experiments.** To study the scalability, we performed experiments on ten data sets from the NCI anti-HIV database that consist of  $x$  graphs ( $x \in [1000, 10000]$ ) with a similarity coefficient  $\theta \in [0.2, 0.8]$ . As it can be seen in Figure 7, the structure-based clustering algorithm scales favorably as the size of

**Fig. 6.** Runtime performance of structure-based clustering with and without clustering exclusion criteria on (a) CPDBAS\_MOUSE, (b) CPDBAS\_RAT and (c) EPAFHM



**Fig. 7.** Runtime performance of the structure-based clustering approach on ten data sets from the NCI anti-HIV database consisting of  $x$  graphs ( $x \in [1000, 10000]$ )

the data set increases. However, for  $0.6 \leq \theta \leq 0.8$ , the algorithm did not respond within a certain timeout period for a data set size larger than 4000 / 5000 objects. The overall results suggest that, depending on reasonable parameter settings, our clustering approach can handle data sets of at least 10,000 graphs.

## 4 Related Work

Graph clustering has been extensively investigated over the past few years. Basically, there exist two complementary approaches to graph clustering [9]. The simpler and more established one is to calculate a vectorial representation of the graphs and use standard similarity or distance measures in combination with standard clustering algorithms. The feature vector can be composed of properties of the graph and / or of subgraph occurrences [10]. Yoshida *et al.* [11] describe a graph clustering method based on structural similarity of fragments (connected subgraphs are considered) in graph-structured data. The approach is experimentally evaluated on synthetic data only and does not consider edge and node labels. The second approach is to use the structure of the graphs directly. Tsuda and Kudo [3] proposed an EM-based method for clustering graphs based on weighted frequent pattern mining. Their method is fully probabilistic, adopting a binomial mixture model defined on a very high dimensional vector indicating the presence or absence of all possible patterns. However, the number of clusters has to be specified a priori, and the model selection procedure is not discussed in their paper. Bunke *et al.* [12] proposed a new graph clustering algorithm, which is an extension of Kohonen's well-known Self-Organizing Map (SOM) algorithm [13] into the domain of graphs. The approach is experimentally evaluated on the graph representations of capital letters that are composed of straight line segments. Chen and Hu [14] introduced a non-exhaustive clustering algorithm that allows for clusters to be overlapping by modifying the traditional k-medoid algorithm. This implies the drawback that the number of clusters has

to be known beforehand. In a comparison of an MCS-based clustering with a fingerprint-based clustering, Raymond *et al.* [15] report that no obvious advantage results from the use of the more sophisticated, graph-based similarity measures. They draw the conclusion, that although the results obtained from the use of graph-based similarities are different from fingerprint-based similarities, there is no evidence to suggest that one approach is consistently better than the other.

Summing up, all MCS-based clustering approaches appear to suffer from the NP-hardness of the MCS computation. While no running times are reported and no implementations appear to be publicly available, the graph data sets tested in related papers typically contain less than 500 graphs [4, 15]. Our choice to modify a frequent graph miner instead is partly motivated by the observation by Bunke *et al.* [16] that for dense graphs association graph methods are preferable, whereas for sparse graphs (as the molecular structures occurring in our application domains) methods enumerating frequent subgraphs are to be preferred.

Related to our clustering approach, Aggarwal *et al.* [17] propose a structural clustering method for clustering XML data. It employs a projection based structural approach and uses a set of frequent substructures as the representative with respect to an intermediate cluster of XML documents. Paths extracted from XML documents are used as a document representation. To mine frequent closed sequences, the sequential pattern mining algorithm BIDE [18] was revised in order to terminate search once a sequence reaches a specified size. Our work is most closely related to the graph clustering approach by Tsuda and Kurihara [5]. They presented a graph clustering approach based on frequent pattern mining that addresses the problem of learning a DP mixture model in the high dimensional feature space of graph data. To keep the feature search space small, an effective tree pruning condition was designed. Although our clustering approach is similarly based on frequent subgraph mining there are several important differences. First, there are differences in the output that makes our clustering results easier to interpret. Despite the proposed feature selection method to obtain a reduced feature set, DP clustering still outputs quite numerous frequent subgraphs which make the graph clusters difficult to interpret. In contrast, our method actually outputs just the clustered graphs sharing a common cluster scaffold. Second, we provide an effective online algorithm that allows for overlapping and non-exhaustive clustering. Non-exhaustive clustering may be more robust in case the set of objects to be clustered contains outliers. The rationale of overlapping clustering is that in some applications it is not appropriate for an object to belong to only one cluster.

## 5 Conclusion and Future Work

We presented a new online algorithm for clustering graph objects in terms of structural similarity. Structural graph clustering can offer interesting new insights into the composition of graph data sets. Moreover, it can be practically useful to benchmark other graph mining algorithms, to derive new substructural

descriptors, to compute local models for classifying graphs, and to calculate the applicability domain of models. Several experiments were designed to evaluate the effectiveness and efficiency of our approach on various real world data sets of molecular graphs. First of all, the results indicate that the clustering method is able to rediscover known structure classes in the NCI standard anti-cancer agents. Moreover, a baseline comparison with a fingerprint-based clustering was presented. The results demonstrate that the structural clustering approach yields larger and more representative cluster scaffolds compared to FP based clustering, thus reducing the heterogeneity in the clusters obtained by fingerprint clustering. To show the importance of the cluster exclusion criteria defined in Equation 3 and 4, we evaluated the performance of the structural clustering approach with and without these criteria. Finally, to investigate how well the algorithm scales regarding running time, we performed extensive experiments with 10,000 compounds selected from the NCI aids anti-viral screen data. In summary, our results suggest that this overlapping, non-exhaustive structural clustering approach generates interpretable clusterings in acceptable time. Further work, from an application point of view, includes the following: First, it would be interesting to investigate the effects of preprocessing steps, e.g., down-weighting longer chains (acyclic substructures) or reduced graph representations (transforming cycles, in chemical terms: rings, into special nodes). Second, the algorithm could be extended easily to take into account the physico-chemical properties of whole molecules. Technically, this would mean that only graphs within a certain distance with respect to such global graph properties are added to a cluster.

## Acknowledgements

This work was partially supported by the German Federal Ministry for the Environment, Nature Conservation and Nuclear Safety and the EU FP7 project (HEALTH-F5-2008-200787) OpenTox (<http://www.opentox.org>).

## References

1. Inokuchi, A., Washio, T., Motoda, H.: An APriori-based algorithm for mining frequent substructures from graph data. In: PKDD '00: Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery, pp. 13–23 (2000)
2. Yan, X., Han, J.: gSpan: Graph-based substructure pattern mining. In: Proceedings of the 2002 IEEE International Conference on Data Mining, pp. 721–724 (2002)
3. Tsuda, K., Kudo, T.: Clustering Graphs by Weighted Substructure Mining. In: Cohen, W.W., Moore, A. (eds.) ICML 2006, pp. 953–960. ACM Press, New York (2006)
4. Stahl, M., Mauser, H.: Database clustering with a combination of fingerprint and maximum common substructure methods. *J. Chem. Inf. Model.* 45, 542–548 (2005)
5. Tsuda, K., Kurihara, K.: Graph mining with variational Dirichlet process mixture models. In: Proceedings of the 8th SIAM International Conference on Data Mining, pp. 432–442 (2008)



6. Martin, Y.C., Kofron, J.L., Traphagen, L.M.: Do structurally similar molecules have similar biological activity? *J. Med. Chem.* 45, 4350–4358 (2002)
7. Weinstein, J., Kohn, K., Grever, M., Viswanadhan, V.: Neural computing in cancer drug development: Predicting mechanism of action. *Science* 258, 447–451 (1992)
8. Koutsoukos, A.D., Rubinstein, L.V., Faraggi, D., Simon, R.M., Kalyandrug, S., Weinstein, J.N., Kohn, K.W., Paull, K.D.: Discrimination techniques applied to the NCI in vitro anti-tumour drug screen: predicting biochemical mechanism of action. *Stat. Med.* 13, 719–730 (1994)
9. Raymond, J.W., Willett, P.: Effectiveness of graph-based and fingerprint-based similarity measures for virtual screening of 2D chemical structure databases. *J. Comput. Aided. Mol. Des.* 16(1), 59–71 (2002)
10. McGregor, M.J., Pallai, P.V.: Clustering of large databases of compounds: Using the MDL "keys" as structural descriptors. *J. Chem. Inform. Comput. Sci.* 37(3), 443–448 (1997)
11. Yoshida, T., Shoda, R., Motoda, H.: Graph clustering based on structural similarity of fragments. In: Jantke, K.P., et al. (eds.) *Federation over the Web. LNCS (LNAI)*, vol. 3847, pp. 97–114. Springer, Heidelberg (2006)
12. Günter, S., Bunke, H.: Validation indices for graph clustering. *Pattern Recogn. Lett.* 24(8), 1107–1113 (2003)
13. Kohonen, T.: *Self-organizing maps*. Springer, Heidelberg (1997)
14. Chen, Y.L., Hu, H.L.: An overlapping cluster algorithm to provide non-exhaustive clustering. *Eur. J. Oper. Res.* 173(3), 762–780 (2006)
15. Raymond, J.W., Blankley, C.J., Willett, P.: Comparison of chemical clustering methods using graph- and fingerprint-based similarity measures. *J. Mol. Graph. Model.* 21(5), 421–433 (2003)
16. Bunke, H., Foggia, P., Guidobaldi, C., Sansone, C., Vento, M.: A comparison of algorithms for maximum common subgraph on randomly connected graphs. In: Caelli, T.M., Amin, A., Duin, R.P.W., Kamel, M.S., de Ridder, D. (eds.) *SPR 2002 and SSPR 2002. LNCS*, vol. 2396, pp. 123–132. Springer, Heidelberg (2002)
17. Aggarwal, C.C., Ta, N., Wang, J., Feng, J., Zaki, M.: XProj: a framework for projected structural clustering of XML documents. In: *KDD '07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 46–55. ACM, New York (2007)
18. Wang, K., Han, J.: Bide: Efficient mining of frequent closed sequences. In: *International Conference on Data Engineering* (2004)

# Large-Scale Support Vector Learning with Structural Kernels

Aliaksei Severyn and Alessandro Moschitti

Department of Computer Science and Engineering  
University of Trento  
Via Sommarive 14, 38100 POVO (TN) - Italy  
{severyn,moschitti}@disi.unitn.it

**Abstract.** In this paper, we present an extensive study of the cutting-plane algorithm (CPA) applied to structural kernels for advanced text classification on large datasets. In particular, we carry out a comprehensive experimentation on two interesting natural language tasks, e.g. predicate argument extraction and question answering. Our results show that (i) CPA applied to train a non-linear model with different tree kernels fully matches the accuracy of the conventional SVM algorithm while being ten times faster; (ii) by using smaller sampling sizes to approximate subgradients in CPA we can trade off accuracy for speed, yet the optimal parameters and kernels found remain optimal for the exact SVM. These results open numerous research perspectives, e.g. in natural language processing, as they show that complex structural kernels can be efficiently used in real-world applications. For example, for the first time, we could carry out extensive tests of several tree kernels on millions of training instances. As a direct benefit, we could experiment with a variant of the partial tree kernel, which we also propose in this paper.

**Keywords:** Structural Kernels; Support Vector Machines; Natural Language Processing.

## 1 Introduction

In many computer science areas such as Natural Language Processing (NLP), Bioinformatics, Data Mining and Information Retrieval, structural kernels have been widely used to capture rich syntactic information, e.g. [3,12,29,30]. In particular, in NLP, tree kernel functions can capture a representation of syntactic parse trees, which is considerably more effective than the one provided by straightforward bag-of-words models. This often results in higher accuracy, e.g. [16,18,31,11]. Unfortunately, the use of kernels forces us to solve SVM optimization problem in the dual space, which, in case of the very large datasets, makes SVM learning prohibitively expensive.

Recently, cutting plane approaches have been proposed to achieve a great speed-up in the learning of linear models on large datasets, but they still fail to provide the same training performance on non-linear learning tasks. Indeed, the

number of kernel evaluations scales quadratically with the number of examples. To overcome this bottleneck, Yu and Joachims [28] developed two approximate cutting plane algorithms (CPAs). However, their experiments were carried out using only Gaussian kernels on unstructured data.

In this paper, we study models that combine the speed of CPAs with the efficiency of SVM-light-TK, which encodes state of the art structural kernels [17,18] in SVM-light [6]. More specifically, by the means of extensive experimentation we examine the applicability of CPAs to well-known structural kernels, i.e. subtree (ST), subset tree (SST), and partial tree (PT) kernels, which provide a good sample of the efficient tree kernel technology currently available. To obtain more general results we considered two advanced and very different text categorization tasks for which syntactic information is essential: (i) Semantic Role Labeling (SRL) or predicate argument extraction [20,15] whose associated dataset contains several millions of examples; and (ii) Question Classification (QC), e.g. [13,26] from question answering domain. Some of the distinct properties of the aforementioned datasets (compared to the previous works) are:

- Features are structured and embed syntactic information: from their relationship, the classifier derives higher semantics.
- The number of features is huge (millions) and they tend to be very sparse.
- The space is discrete, the a-priori weights are very skewed, and large fragments receive exponentially lower scores than small fragments.
- There is high redundancy and inter-dependency between features.

Finally, we defined a novel kernel, unlexicalized PTK (uPTK), which is a variant of PTK. It excludes single nodes from the feature space so that structural features are better emphasized.

Our findings reveal that:

- As the theory suggests, CPAs can be successfully applied to structural spaces. Indeed, we show that CPA is at least 10 times as fast as the exact version while giving the same classification accuracy. For example, training a conventional SVM solver with tree kernels on 1 million examples requires more than seven days, while CPAs match the same accuracy in just a few hours.
- By decreasing the sampling size used in the approximation of the cutting planes, we can trade off accuracy to a small degree for even faster training time. For example, learning a model that is only 1.0 percentage point apart from the exact model reduces the training time by a factor of 50.
- Using a sample size of only 100 instances for CPAs, which takes just a couple of minutes of learning on one million of examples, we can correctly estimate the best kernel, its hyper-parameters, and the trade-off parameter. The identified set of optimal parameters can be used by more computationally expensive and accurate models (CPAs with larger sample sizes or SVM-light).
- Thanks to the gained speed-up we could efficiently compare different kernels on the large SRL datasets and establish an absolute rank, where uPTK proves to be more effective than PTK.

Our results open up new perspectives for the application of structural kernels in various NLP tasks. The experiments that could not be carried out until now on full subsets of very large corpora, e.g. training SVM-based SRL classifier such as in [22,15] using either polynomial or tree kernels on the full dataset; training SVM re-rankers for syntactic parsing [3,25] using tree kernels on the available data; and training SVM question classifiers using tree kernels and other features on large subsets of Yahoo! Answers dataset. The aforementioned motivations encouraged us to release the new Tree Kernel toolkit to the public<sup>1</sup>.

In the remainder of this paper, Section 2 reviews the related work, while Section 3 gives careful theoretical treatment to the workings of the cutting plane approach. Section 4 introduces well-known tree kernels along with the new uPTK. The experimental analysis (Section 5) describes our experiments on SRL and QC datasets and also discusses how approximate cutting plane algorithms could be used to drive parameter selection for the exact SVM solver. Finally, in Section 6, we draw conclusions and provide directions for the further research.

## 2 Related Work

Since the introduction of SVMs, a number of fast algorithms that can efficiently train non-linear SVMs have been proposed: for example, decomposition method along with working set selection [6]. Decomposition methods work directly in the dual space and perform well on moderately large datasets but their performance degrades when the number of training examples reaches a level of millions of examples.

Recently, a number of efficient algorithms using cutting planes to train conventional SVM classifiers have been proposed. For example,  $SVM^{perf}$  [7] is based on a cutting plane algorithm and exhibits linear computational complexity in the number of examples when linear kernels are used. To improve the convergence rate of the underlying cutting plane algorithm, Franc and Sonnenburg [5] developed the optimized cutting plane algorithm (OCAS) that achieves speed-up factor of 29 over  $SVM^{perf}$ . Alternatively, another promising approach based on stochastic gradient descent and projection steps called Pegasos [24] has shown promising performance for linear kernels in binary classification tasks.

While the aforementioned algorithms deliver state of the art performance with respect to accuracy and training time, they scale well only when linear kernels are used. To overcome this bottleneck, an idea to use approximate cutting planes with random sampling was employed by Yu and Joachims [28]. At each iteration step an exact cutting plane is replaced by its approximation that is built from a small subset of examples sampled from the training dataset.

Another approximate technique to speed up the computation time of tree kernels by comparing only a sparse subset of relevant subtrees is discussed in [23].

Among the basis pursuit approaches that find a set of basis vectors to construct sparse kernel SVMs are works by Keerthi [9] and Joachims [8]. The former uses a greedy technique to select vectors, while the latter extracts basis vectors

---

<sup>1</sup> Available at <http://projects.disi.unitn.it/iKernels>

as a part of the cutting-plane optimization process. Not only the trained sparse model speeds up the classification time, but it also improves the training complexity. Even though the achieved scaling behavior is roughly linear, the method in [8] is limited to the use of only Gaussian kernels.

Following a different line of research, a number of methods that exploit low-rank approximation of a kernel matrix have been proposed in [4,27]. However, the experiments were carried out only on fairly small datasets (thousands of examples).

The approach we choose to employ in this paper is different in a sense that it is generally applicable to the learning of any non-linear discriminant function, structural kernels in particular, and can efficiently handle datasets with hundreds of thousands examples.

### 3 Cutting Plane Algorithm for Structural Kernels

In this section, we illustrate the cutting plane approach. Previous works focus on structural SVMs whereas the topic of this paper is binary classification, thus we present a re-elaborated version of the algorithm tailored for the binary classification task starting off with the derivation of the dual formulation. This results in an easier discussion of the sampling approach to compute the approximate cutting planes.

#### 3.1 Cutting-Plane Algorithm (Dual)

Below is an equivalent formulation of an SVM optimization problem, known as a 1-slack reformulation [7], used to derive the cutting-plane algorithm:

$$\begin{aligned} & \underset{\mathbf{w}, \xi \geq 0}{\text{minimize}} && \frac{1}{2} \|\mathbf{w}\|^2 + C\xi \\ & \text{subject to} && \forall \mathbf{c} \in \{0, 1\}^n : \\ & && \frac{1}{n} \mathbf{w} \cdot \sum_{i=1}^n c_i y_i \mathbf{x}_i \geq \frac{1}{n} \sum_{i=1}^n c_i - \xi, \end{aligned} \quad (1)$$

where each constraint can be viewed as an average sum of a subset of constraints of the form:  $y_i(\mathbf{w} \cdot \mathbf{x}_i) \geq 1 - \xi_i$  that are selected by a vector  $\mathbf{c} = (c_1, \dots, c_n) \in \{0, 1\}^n$ . While the total number of constraints is  $2^n$ , there is only a single slack variable  $\xi$  shared across all the constraints. For more details on the 1-slack formulation, an interested reader should refer to [7].

To derive the dual formulation, the Lagrangian of the primal problem (1) is computed as:

$$L_P = \frac{1}{2} \|\mathbf{w}\|^2 + C\xi - \sum_{j=1}^{|S|} \alpha_j \left( \frac{1}{n} \sum_{k=1}^n c_{kj} (y_k \mathbf{w} \cdot \mathbf{x}_k - 1) + \xi \right), \quad (2)$$

where  $S$  denotes the set of constraints in (1) and  $|S| = 2^n$ ,  $c_{kj}$  denotes the  $k^{\text{th}}$  component of a vector  $\mathbf{c}_j$  that corresponds to the  $j^{\text{th}}$  constraint in (1), and  $\alpha_j \geq 0$  are the Lagrange multipliers.

Using the fact that both gradients of  $L_P$  with respect to  $w$  and  $\xi$  vanish:

$$\begin{aligned} \frac{\delta L_P}{\delta \mathbf{w}} &= \mathbf{w} - \sum_{j=1}^{|S|} \alpha_j \left( \frac{1}{n} \sum_{k=1}^n c_{kj} y_k \mathbf{x}_k \right) = 0 \\ \frac{\delta L_P}{\delta \xi} &= \frac{1}{n} \sum_{j=1}^{|S|} \alpha_j - C = 0, \end{aligned} \tag{3}$$

and by substituting variables from (3) into (2), we obtain the dual Lagrangian:

$$L_D = \sum_{i=1}^{|S|} \alpha_i h^{(i)} - \frac{1}{2} \sum_{i=1}^{|S|} \sum_{j=1}^{|S|} \alpha_i \alpha_j \mathbf{g}^{(i)} \cdot \mathbf{g}^{(j)} \tag{4}$$

where  $h^{(i)} = \frac{1}{n} \sum_{k=1}^n c_{ki}$  and  $\mathbf{g}^{(i)} = -\frac{1}{n} \sum_{k=1}^n c_{ki} y_k \mathbf{x}_k$  are respectively the bias and the subgradient that define a cutting plane model ( $h^{(i)}, \mathbf{g}^{(i)}$ ). Now we can state the dual variant of the optimization problem (1):

$$\begin{aligned} \underset{\alpha \geq 0}{\text{maximize}} \quad & \mathbf{h}^T \alpha - \frac{1}{2} \alpha^T H \alpha \\ \text{subject to} \quad & \alpha^T \mathbf{1} \leq C, \end{aligned} \tag{5}$$

where  $H_{ij} = \mathbf{g}^{(i)} \cdot \mathbf{g}^{(j)}$ . Using the first equation from (3), we get the connection between the primal and dual variables:

$$\mathbf{w} = \sum_{j=1}^{|S|} \alpha_j \left( \frac{1}{n} \sum_{k=1}^n c_{kj} y_k \mathbf{x}_k \right) = - \sum_{j=1}^{|S|} \alpha_j \mathbf{g}^{(j)}, \tag{6}$$

Now we are ready to present the CPA method (Algorithm 1). It starts with an empty set of constraints  $S$  and computes the optimal solution (line 5) to the unconstrained problem (5). Next, the algorithm finds the most violated constraint (lines 9-11) by forming a binary vector  $\mathbf{c}$  that defines the maximum sum of violated constraints. This requires the computation of the following quantity for each training example:

$$\mathbf{w} \cdot \phi(\mathbf{x}_i) = - \sum_{j=1}^{|S|} \alpha_j \mathbf{g}^{(j)} \cdot \phi(\mathbf{x}_i) = \sum_{k=1}^n \left( \sum_{j=1}^{|S|} \frac{1}{n} \alpha_j c_{kj} y_k \right) K(\mathbf{x}_i, \mathbf{x}_k), \tag{7}$$

where  $\phi(\mathbf{x})$  is a mapping from the input to the feature space and  $K(\mathbf{x}_i, \mathbf{x}_k) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_k)$  is a kernel. We then compute the bias  $h^{(t)}$  and the subgradient  $\mathbf{g}^{(t)}$  (lines 12 and 13 respectively) to build a new cutting plane which defines a half-space  $h^{(t)} + \mathbf{w} \cdot \mathbf{g}^{(t)} \leq \xi$  corresponding to the constraint that is most violated by the current solution. Then it is included in the set of active constraints  $S$  (line 14). This way, a series of successively tightening approximations to the original

---

**Algorithm 1.** Cutting Plane Algorithm (dual) with uniform sampling

---

```

1: Input:  $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n), C, \epsilon$ 
2:  $S \leftarrow \emptyset; t = 0;$ 
3: repeat
4: Update matrix  $H$  with a new constraint
5:  $\alpha \leftarrow$  optimize QP problem (5)
6:  $\xi = \frac{1}{C}(\mathbf{h}^T \alpha - \frac{1}{2} \alpha^T H \alpha)$ 
7:  $\mathbf{w} = -\sum_{j=1}^{|S|} \alpha_j \mathbf{g}^{(j)}$ 
8: Sample  $r$  examples from the training set
   /* find the most violated constraint (cutting plane) */
9: for  $i = 1$  to  $r$  do
10:  $c_i \leftarrow \begin{cases} 1 & y_i(\mathbf{w} \cdot \phi(\mathbf{x}_i)) \leq 1 \\ 0 & \text{otherwise} \end{cases}$ 
11: end for
12:  $\mathbf{h}^{(t)} = \frac{1}{r} \sum_{i=1}^r c_i$ 
13:  $\mathbf{g}^{(t)} = -\frac{1}{r} \sum_{i=1}^r c_i y_i \phi(\mathbf{x}_i)$ 
   /* add a constraint to the active set */
14:  $S \leftarrow S \cup \{(\mathbf{h}^{(t)}, \mathbf{g}^{(t)})\}$ 
15:  $t = t + 1$ 
16: until  $h^{(t)} + \mathbf{w} \cdot \mathbf{g}^{(t)} \leq \xi + \epsilon$ 
17: return  $w, \xi$ 

```

---

problem is constructed. The algorithm stops when no constraints are violated by more than  $\epsilon$ , which is formalized by the criteria in line 16.

The analysis of the inner product given by (7) reveals that, since it needs to be computed for each training example, it requires the time  $O(n^2 + Tn)$  after total of  $T$  iterations. Similarly, as we add a cutting plane to  $S$  at each iteration  $t$ , a new column is added to the matrix  $H$  (Algorithm 1, line 4) requiring the computation of

$$H_{it} = \mathbf{g}^{(i)} \cdot \mathbf{g}^{(t)} = \frac{1}{n^2} \sum_{k=1}^n \sum_{l=1}^n c_{ki} c_{lt} y_k y_l K(\mathbf{x}_k, \mathbf{x}_l) \tag{8}$$

which takes  $O(Tn^2)$ . Thus, the obtained  $O(n^2)$  scaling behavior makes cutting plane training no better than conventional decomposition methods.

To address this limitation, we employ the approach of Yu and Joachims [28] to construct approximate cuts by sampling  $r$  examples from the training set. They suggest two strategies to sample examples, namely uniform and importance sampling (the pseudocode of the algorithm using uniform sampling is presented in Algorithm 1). These two strategies derive constant-time and linear-time algorithms. The former uniformly samples  $r$  examples from the training set to approximate the cut. Thus, we approximate a subgradient  $\mathbf{g}^{(t)}$  with only  $r$  examples, which replaces the number of expensive kernel evaluations in (7) over  $n$  by a more tractable:  $\sum_{i,j=1}^r K(\mathbf{x}_i, \mathbf{x}_j)$  (lines 9-13). The importance sampling acts in a more targeted way as it looks through the whole dataset to compute two cutting planes, one to be used in the optimization problem (line 5), and the

other for termination criterion (line 16). The training complexity reduces from  $O(n^2)$  to  $O(T^2r^2)$ , when the uniform sampling algorithm is used, to  $O(Tnr)$  for the importance sampling.

## 4 Tree Kernels

The main idea underlying tree kernels is to compute the number of common substructures between two trees  $T_1$  and  $T_2$  without explicitly considering the whole fragment space. Let  $\mathcal{F} = \{f_1, f_2, \dots, f_{|\mathcal{F}|}\}$  be the set of tree fragments and  $\chi_i(n)$  an indicator function equal to 1 if the target  $f_i$  is rooted at node  $n$  and equal to 0 otherwise. A tree kernel function over  $T_1$  and  $T_2$  is defined as

$$TK(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2),$$

where  $N_{T_1}$  and  $N_{T_2}$  are the sets of nodes in  $T_1$  and  $T_2$ , respectively, and

$$\Delta(n_1, n_2) = \sum_{i=1}^{|\mathcal{F}|} \chi_i(n_1) \chi_i(n_2).$$

The  $\Delta$  function is equal to the number of common fragments rooted in nodes  $n_1$  and  $n_2$  and thus depends on the fragment type.

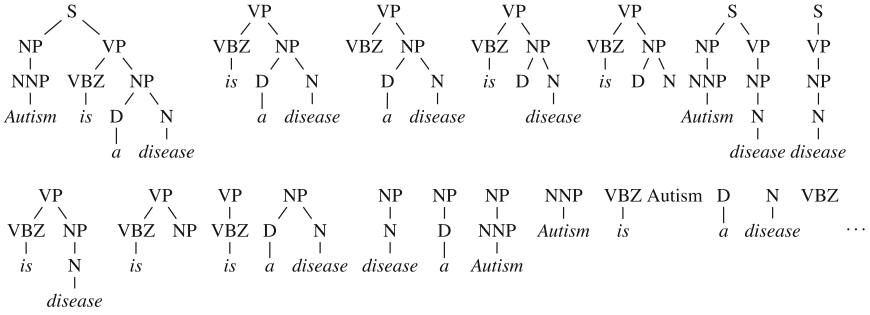
### 4.1 Fragment Types

In [17], we pointed out that there are three main categories of fragments: the subtree (ST), the subset tree (SST) and the partial tree (PT) fragments corresponding to three different kernels. STs are fragments rooted in any node of a tree along with all its descendants. The SSTs are more general structures since, given the root node of an SST, not all its descendants (with respect to the referring tree) have to be included, i.e. the SST leaves can be non-terminal symbols. PT fragments are still more general since their nodes can contain a subset of the children of the original trees, i.e. partial sequences.

For example, Figure 1 illustrates the syntactic parse tree of the sentence *Autism is a disease* on the left along with some of the possible fragments on the right of the arrow. ST kernel generates complete structures like [D a] or [NP [D a] [N disease]]. SST kernel can generate more structures, e.g. [NP [D] [N disease]] whereas PT kernel can also separate children in the fragments, e.g. [NP [N disease]], and generate the individual tree nodes as features, e.g. *Autism* or *VBZ*.

[30] provided a version of SST kernel, which also generates leaves, i.e. words, as features, hereafter, SST-bow. However, such lexical features, when the data is very sparse, tend to cause overfitting. Thus, we give the definition of a variant of PTK, namely, the unlexicalized partial tree kernel (uPTK), which does not include lexicals and individual nodes in the feature space. This will promote the importance of structural information.





**Fig. 1.** A tree for the sentence “Autism is a disease” (top left) with some of its partial tree fragments (PTFs)

### 4.2 Unlexicalized Partial Tree Kernel (uPTK)

The algorithm for the uPTK computation straightforwardly follows from the definition of the  $\Delta$  function of PTK provided in [17]. Given two nodes  $n_1$  and  $n_2$  in the corresponding two trees  $T_1$  and  $T_2$ ,  $\Delta$  is evaluated as follows:

1. if the node labels of  $n_1$  and  $n_2$  are different then  $\Delta(n_1, n_2) = 0$ ;
2. else  $\Delta(n_1, n_2) = \mu \left( \lambda^2 + \sum_{\mathbf{I}_1, \mathbf{I}_2, l(\mathbf{I}_1)=l(\mathbf{I}_2)} \lambda^{d(\mathbf{I}_1)+d(\mathbf{I}_2)} \prod_{j=1}^{l(\mathbf{I}_1)} \Delta(c_{n_1}(\mathbf{I}_{1j}), c_{n_2}(\mathbf{I}_{2j})) \right)$ ,

where: (a)  $\mathbf{I}_1 = \langle h_1, h_2, h_3, \dots \rangle$  and  $\mathbf{I}_2 = \langle k_1, k_2, k_3, \dots \rangle$  are index sequences associated with the ordered child sequences  $c_{n_1}$  of  $n_1$  and  $c_{n_2}$  of  $n_2$ , respectively; (b)  $\mathbf{I}_{1j}$  and  $\mathbf{I}_{2j}$  point to the  $j$ -th child in the corresponding sequence; (c)  $l(\cdot)$  returns the sequence length, i.e. the number of children; (d)  $d(\mathbf{I}_1) = \mathbf{I}_{1l(\mathbf{I}_1)} - \mathbf{I}_{11} + 1$  and  $d(\mathbf{I}_2) + 1 = \mathbf{I}_{2l(\mathbf{I}_2)} - \mathbf{I}_{21} + 1$ ; and (e)  $\mu$  and  $\lambda$  are two decay factors for the size of the tree and for the length of the child subsequences with respect to the original sequence, i.e. we account for gaps.

The uPTK, can be obtained by removing  $\lambda^2$  from the equation in the step 2. An efficient algorithm for the computation of PTK is given in [17]. This evaluates  $\Delta$  by summing the contribution of tree structures coming from different types of sequences, e.g. those composed by  $p$  children such as:

$$\Delta(n_1, n_2) = \mu \left( \lambda^2 + \sum_{p=1}^{lm} \Delta_p(c_{n_1}, c_{n_2}) \right), \tag{9}$$

where  $\Delta_p$  evaluates the number of common subtrees rooted in subsequences of exactly  $p$  children (of  $n_1$  and  $n_2$ ) and  $lm = \min\{l(c_{n_1}), l(c_{n_2})\}$ . It is easy to verify that we can use the recursive computation of  $\Delta_p$  proposed in [17] by simply removing  $\lambda^2$  from Eq. 9.

## 5 Experiments

In these experiments, we study the impact of the cutting plane algorithms (CPAs), reviewed in Section 3, on learning complex text classification tasks in

structural feature spaces. For this purpose, we compare the accuracy and the learning time of CPAs, according to different sample size against the conventional SVMs.

In the second set of experiments, we investigate the possibility of using fast parameter and kernel selection with CPA for conventional SVM. For this purpose, we carried out experiments with different classifiers on two different domains.

## 5.1 Experimental Setup

We integrated two approximate cutting plane algorithms using sampling [28] with SVM-light-TK [17]. For brevity, in this section we will refer to the algorithm that uses uniform sampling as uSVM, importance sampling as iSVM, and SVM-light-TK as SVM. While the implementation of sampling algorithms uses MOSEK to optimize quadratic problem, SVM is based on SVM-light 5.0 solver. As the stopping criteria of the algorithms, we fix the precision parameter  $\epsilon$  at 0.001.

We experimented with five different kernels: the ST, SST, SST-bow, PT, uPT kernels described in Section 4, which are also normalized in the related kernel space. All the experiments that do not involve parameter tuning use the default trade-off parameter (i.e. 1 for normalized kernels) and the default  $\lambda$  fixed at 0.4.

As a measure of classification accuracy we use the harmonic average of the Precision and Recall, i.e.  $F_1$ -score. All the experiments were run on machines equipped with Intel<sup>®</sup> Xeon<sup>®</sup> 2.33GHz CPUs carrying 6Gb of RAM under Linux 2.6.18 kernel.

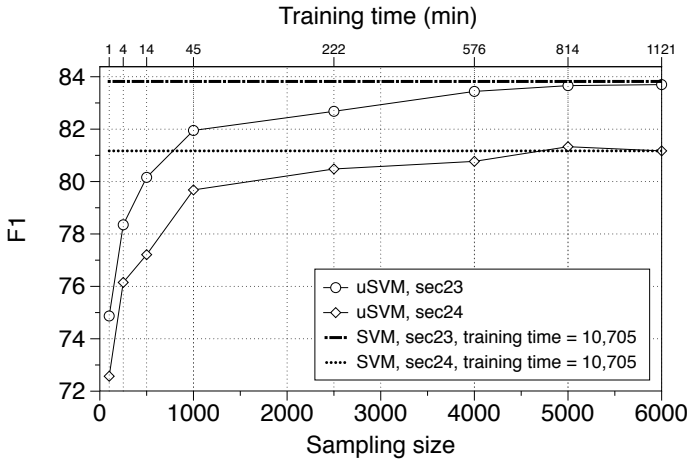
## 5.2 Data

We used two different natural language datasets corresponding to two different tasks: Semantic Role Labeling (SRL) and Question Answering.

The first consists of the Penn Treebank texts [14], PropBank annotation [20] and automatic Charniak parse trees [2] as provided by the CoNLL 2005 evaluation campaign [1]. In particular, we tackle the task of identification of the argument boundaries (i.e. the exact sequence of words compounding an argument). This corresponds to the classification of parse tree nodes in correct or not correct boundaries<sup>2</sup>. For this purpose, we train a binary *Boundary Classifier* (BC) using the AST subtree defined in [15], i.e. the minimal subtree, extracted from the sentence parse tree, including the predicate and the target argument nodes. To test the learned model, we extract two sections, namely *sec23* and *sec24*, that contain 234,416 and 149,140 examples respectively. The models are trained on two subsets of 100,000 and 1,000,000 examples. The proportion of positive examples in the whole corpus is roughly 5%. The dataset along with the exact structural representation is available at <http://danielepighin.net/cms/research/MixedFeaturesForSRL>.

---

<sup>2</sup> In the automatic trees some boundary may not correspond to any node. In this case, we choose the lower node dominating all the argument words.



**Fig. 2.**  $F_1$ -score as a function of the sampling size on SRL dataset (1 million examples). Horizontal axis at top is the training time of the uSVM algorithm.

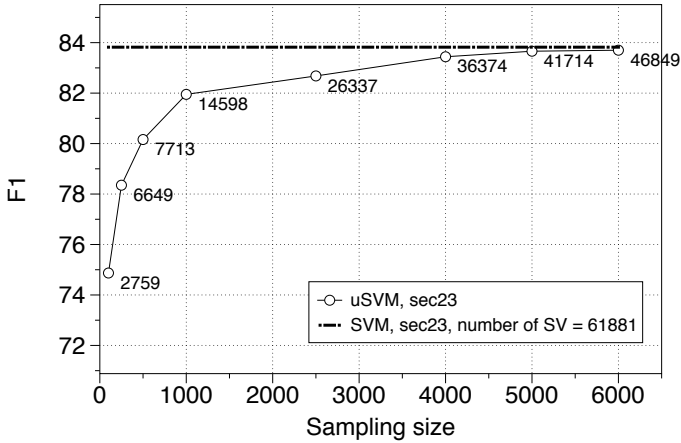
The second corpus, is a Question Classification (QC) dataset, whose testset comes from the TREC 10 - Question Answering evaluation campaign whereas the training set<sup>3</sup> was developed in [13]. The task consists in selecting the most appropriate type of the answer from a set of given possibilities. The *coarse grained* question taxonomy [30,13] consists of six non overlapping classes: Abbreviations (ABBR), Descriptions (DESC, e.g. definitions or explanations), Entity (ENTY, e.g. animal, body or color), Human (HUM, e.g. group or individual), Location (LOC, e.g. cities or countries) and Numeric (NUM, e.g. amounts or dates). For each question, we used the full parse tree as its representation (similarly to [30,17,19]). This is automatically extracted by means of the Stanford parser<sup>4</sup> [10]. We actually have only 5,483 questions in our training set, due to parsing issues with a few of them. The testset is constituted by 500 questions and the size of the categories varies from one thousands to few hundreds.

### 5.3 Accuracy and Efficiency vs. Sampling Size

In these experiments, we test the trade-off between speed and accuracy of CPAs. We use uSVM, since it is faster than iSVM, and compare it against SVM on the SRL task by training on 1 million examples and testing on the two usual testing sections, i.e. *sec23* and *sec24*. We used the SST kernel since it has been indicated as the most accurate in similar tasks, e.g. [17]. Figure 2 plots the  $F_1$ -score for different values of the sampling size. The dashed horizontal lines denote the accuracy achieved by the exact SVM. The training time for the uSVM algorithm is plotted along the top horizontal axis.

<sup>3</sup> <http://l2r.cs.uiuc.edu/cogcomp/Data/QA/QC/>

<sup>4</sup> <http://nlp.stanford.edu/software/lex-parser.shtml>



**Fig. 3.** Number of support vectors (displayed beside each data point) as a function of the sampling size for the 1-slack algorithm

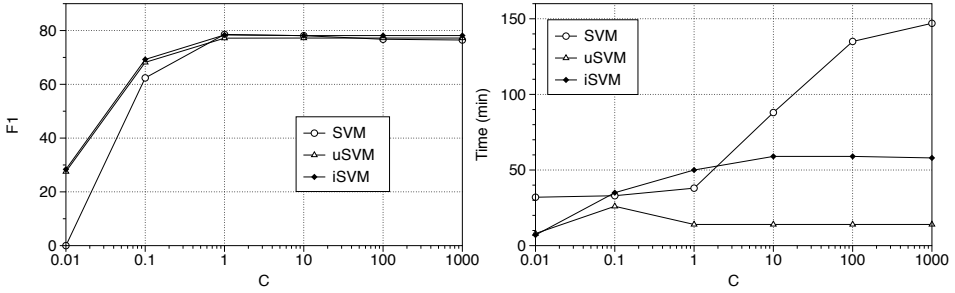
We note that:

- changing the sample size allows us to tune the trade-off between accuracy and speed. Indeed, as we increase the sampling size, the  $F_1$ -score grows until it reaches the accuracy of the exact SVM (at 5,000). In this case, the uSVM produces the same accuracy of SVM while being 10 times faster.
- with a sample size of 2,500 the accuracy of the uSVM is only 1.0 percentage point apart from the exact model whereas the training time savings are of a factor over 50. This corresponds to a training time smaller than 4 hours for uSVM vs. 7.5 days for SVM.
- finally, we note that our reported  $F_1$ -score for boundary classification is state-of-the-art only if tree kernels are used, e.g. [21].

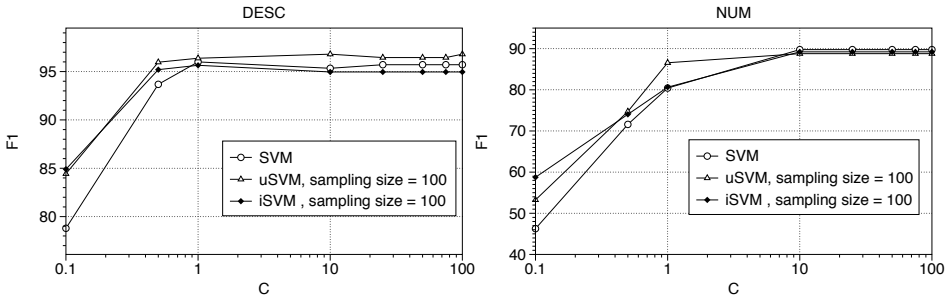
#### 5.4 Producing a Sparse Model

Another interesting dimension to compare sampling algorithms with the exact SVM solver would be to evaluate the sparseness of the produced solutions. As we have already mentioned in Section 3.1, uSVM and iSVM employ the 1-slack formulation that produces very sparse models [7]. This becomes especially important when very large datasets are used, as the improved sparsity can greatly reduce the classification time. Figure 3 is different from Figure 2, as it displays the classification results for sec23 subset of SRL dataset along with the number of support vectors (plotted beside a data point) learned by each model.

Indeed, the number of support vectors grows, as we increase the sampling size in an attempt to match the accuracy of the exact SVM. The model learned by the exact SVM contains 61,881 support vectors, while uSVM model produces 41,714 support vectors. We would like to have more experimental data to make



**Fig. 4.** F<sub>1</sub>-score (left) and training time (right) as a function of margin parameter C on the 100,000 subset of SRL dataset (C is on a logarithmic scale)



**Fig. 5.** F<sub>1</sub>-score as a function of margin parameter C on the DESC (left) and NUM (right) categories of Question Classification dataset (C is on a logarithmic scale)

the comparison more sound but the slow training of the exact SVM makes this endeavor almost infeasible (just the training of the exact SVM on 1 million of examples takes over 7.5 days!)

### 5.5 Fast Parameterization

On very large datasets, finding the best set of parameters for an exact solver, e.g. SVM-light, using structural kernels becomes prohibitively expensive, thus greatly limiting the possibility to find the best performing model. Hence, we test the idea of fast parameter selection for the exact SVM using CPAs and small samples.

We chose the trade-off parameter,  $C$ , as our target parameter and we select a subset of 100,000 examples from SRL data to train uSVM, iSVM, and SVM with  $C \in \{0.01, 0.1, 1, 10, 100, 1000\}$ . We could not use 1 million dataset since SVM prevents to carry out experiments within a tractable time frame. The left plot of Figure 4 shows that F1 of the three models has the same behavior according to  $C$ . Thus, we can select the optimum value according to the fast method (e.g. with a sample size 1000) to estimate the best value of  $C$  of SVM.

Moreover, it is interesting to observe the plot on the right of Figure 4. This shows the training time on the previous subset with respect to  $C$  values. It reveals

**Table 1.**  $F_1$  measured on two testsets, *sec23* and *sec24*, for five different kernels: ST, SST, SST-bow, PT, and uPT kernels, trained on 1 million instances. The best results are shown in bold. The training time is given in minutes. The bottom row is the performance of SVM trained on only **100k** examples.

Sample size	ST			SST			SST-bow			PT			uPT		
	$F_1$		time	$F_1$		time	$F_1$		time	$F_1$		time	$F_1$		time
	<i>sec23</i>	<i>sec24</i>		<i>sec23</i>	<i>sec24</i>		<i>sec23</i>	<i>sec24</i>		<i>sec23</i>	<i>sec24</i>		<i>sec23</i>	<i>sec24</i>	
100	6.8	6.5	3.9	<b>74.9</b>	<b>72.6</b>	<b>1.0</b>	74.3	73.0	2.2	71.9	70.7	3.7	73.3	71.4	3.7
250	14.2	13.0	14.4	<b>78.4</b>	<b>76.2</b>	<b>4.1</b>	78.5	76.3	6.1	74.6	73.2	13.7	76.5	74.6	14.2
500	20.3	18.7	46.6	<b>80.2</b>	<b>77.2</b>	<b>14.0</b>	79.5	77.3	16.9	76.3	74.4	45.0	78.3	76.3	47.6
1000	23.5	21.2	143.7	<b>82.0</b>	<b>79.7</b>	<b>45.3</b>	81.3	79.0	55.9	78.2	76.2	158	79.6	76.9	158.8
SVM	12.6	10.94	213.8	<b>80.78</b>	<b>78.56</b>	<b>37.5</b>	80.38	78.13	42.2	74.39	73.47	89.1	77.54	75.87	100.4

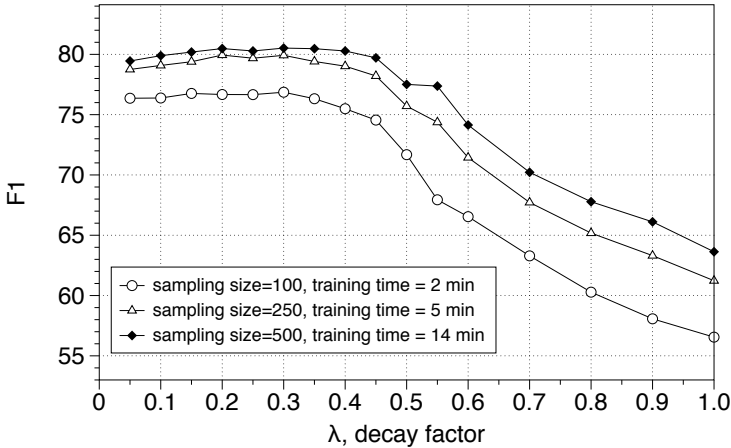
that the use of sampling algorithms, particularly uSVM, provides substantial speed-up in the training time, especially when large values for  $C$  are used. Not surprisingly, uSVM is a preferable choice, since it has a constant-time scaling behavior. These results show that the proposed algorithms can provide fast and reliable parameter search for the best model.

To generalize the findings above we carried out the same experiment on the second dataset. We ran tests for a range of values on all six categories of the QC dataset. Since each category has only 5500 examples, here, however, the main concern is not the training time, but the ability of uSVM and iSVM to match the behavior of the exact SVM with respect to the parameter  $C$ . For brevity, we provide the results only for DESC and NUM categories. Figures 5 shows that both sampling algorithms match well the behavior of the exact SVM for DESC and NUM categories of QC dataset.

## 5.6 Kernel Testing and Selection

The previous section has shown that sampling algorithms can be used for fast selection of the parameter  $C$ . Here we investigate if the same approach can be applied for the efficient selection of the best kernel. The aim is to check if a very fast uSVM algorithm, i.e. using a small sample size, can identify the best performing kernel. Thus, we ran uSVM algorithm on 1 million subset of SRL dataset by varying the sample size and using five different structural kernels: ST, SST, SST-bow, PT and uPT kernels. The results reported in Table 1 show that:

- uSVM has a consistent behavior across different sample sizes for all kernel functions. This suggests that a small sample, e.g. 100 training examples, can be used to select the most suitable kernel for a given task in a matter of a couple of minutes.
- the bottom row of the Table 1 reports the results of SVM trained on a smaller subset of 100k examples (only one experiment with SVM on a subset of 1 million examples takes over 7 days), which demonstrates that the results obtained with uSVM are consistent with the exact solver.



**Fig. 6.**  $F_1$ -score as a function of  $\lambda$  decay factor (length of the child sequences) for SST kernel. Training carried out by uSVM on SRL (1million examples) for samples sizes 100, 250 and 500.

- as already happened in similar tasks, SST kernel appears to provide the best performance with respect to the training time and accuracy whereas the ST kernel, as expected, cannot generate enough features to characterize correct or incorrect boundary. Indeed, its  $F_1$  is very low even when large amount of data is used.
- surprisingly the SST-bow kernel which simply adds bow to SST is less accurate. This suggests that adding words can reduce the generalization ability of the tree kernels, while the syntactic information is very important like in SRL. This aspect is confirmed by the  $F_1$  of uPT, which is higher than PT. Indeed, the former does not generate *pure* lexical features, i.e. words, whereas the latter does.
- finally, since we can quickly pick the most appropriate kernel, we can also perform a fast tuning of kernel hyper-parameters. For this task, we experiment with  $\lambda$ , which is one of the most important factors defining tree kernel performance. Figure 6 displays the behavior of  $F_1$  with respect to the range of  $\lambda$  values. The plot shows that  $F_1$  varies considerably so several values should be tested. We carried out these experiment in a few minutes but using SVM we would have required several weeks.

## 6 Conclusions

In this paper, we have studied the cutting plane technique for training SVMs and we have shown that it is also effective with structural kernels. The experiments on Semantic Role Labeling and Question Classification show very promising results with respect to accuracy and efficiency. Our major achievement is a speed-up factor of over 10 compared to the exact SVM, while obtaining the same precise

solution. In addition, the proposed method gives the flexibility to train very fast models by trading off accuracy for the training time.

The main idea promoted in the paper is that CPA with sampling, while being as accurate as exact SVM, provides the possibility to quickly select optimal kernels or model parameters. Unlike other approximate techniques, which use small subsets of original data to train a model, CPA uses smaller samples only to approximate the subgradient while working on the entire dataset. Our experiments show that parameters/kernels selected by CPA are also optimal for the exact SVM. To the best of our knowledge, this is the first attempt to demonstrate this fact.

## References

1. Carreras, X., Màrquez, L.: Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. In: Proceedings of the 9th Conference on Natural Language Learning, CoNLL-2005, Ann Arbor, MI, USA (2005)
2. Charniak, E.: A maximum-entropy-inspired parser. In: ANLP, pp. 132–139 (2000)
3. Collins, M., Duffy, N.: New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In: ACL, pp. 263–270 (2002)
4. Fine, S., Scheinberg, K.: Efficient svm training using low-rank kernel representations. *Journal of Machine Learning Research* 2, 243–264 (2001)
5. Franc, V., Sonnenburg, S.: Optimized cutting plane algorithm for support vector machines. In: Cohen, W.W., McCallum, A., Roweis, S.T. (eds.) *ICML. ACM International Conference Proceeding Series*, vol. 307, pp. 320–327. ACM, New York (2008)
6. Joachims, T.: Making large-scale SVM learning practical. In: Schölkopf, B., Burges, C., Smola, A. (eds.) *Advances in Kernel Methods - Support Vector Learning*, ch. 11, pp. 169–184. MIT Press, Cambridge (1999)
7. Joachims, T.: Training linear SVMs in linear time. In: *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 217–226 (2006)
8. Joachims, T., Yu, C.N.J.: Sparse kernel svms via cutting-plane training. *Machine Learning* 76(2-3), 179–193 (2009); *European Conference on Machine Learning (ECML) Special Issue*
9. Keerthi, S.S., Chapelle, O., Decoste, D., Bennett, P., Parrado-herndez, E.: Building support vector machines with reduced classifier complexity. *Journal of Machine Learning Research* 8, 2006 (2001)
10. Klein, D., Manning, C.D.: Fast exact inference with a factored model for natural language parsing. In: Becker, S., Thrun, S., Obermayer, K. (eds.) *NIPS*, pp. 3–10. MIT Press, Cambridge (2002)
11. Kudo, T., Matsumoto, Y.: Fast methods for kernel-based text analysis. In: *Proceedings of ACL'03* (2003)
12. Leslie, C., Eskin, E., Cohen, A., Weston, J., Noble, W.S.: Mismatch string kernels for discriminative protein classification. *Bioinformatics* 20(4), 467–476 (2004)
13. Li, X., Roth, D.: Learning question classifiers: the role of semantic information. *Natural Language Engineering* 12(3), 229–249 (2006)
14. Marcus, M., Santorini, B., Marcinkiewicz, M.: Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics* 19(2), 313–330 (1993)
15. Moschitti, A., Pighin, D., Basili, R.: Tree kernels for semantic role labeling. *Computational Linguistics* 34(2), 193–224 (2008)



16. Moschitti, A., Zanzotto, F.: Fast and effective kernels for relational learning from texts. In: Ghahramani, Z. (ed.) *Proceedings of the 24th Annual International Conference on Machine Learning, ICML 2007* (2007)
17. Moschitti, A.: Making tree kernels practical for natural language learning. In: *EACL. The Association for Computer Linguistics* (2006)
18. Moschitti, A.: Kernel methods, syntax and semantics for relational text categorization. In: *Proceeding of CIKM '08, NY, USA* (2008)
19. Moschitti, A., Quarteroni, S., Basili, R., Manandhar, S.: Exploiting syntactic and shallow semantic kernels for question/answer classification. In: *Proceedings of ACL'07* (2007)
20. Palmer, M., Kingsbury, P., Gildea, D.: The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics* 31(1), 71–106 (2005)
21. Pighin, D., Moschitti, A.: Efficient linearization of tree kernel functions. In: *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pp. 30–38. Association for Computational Linguistics, Boulder (June 2009), <http://www.aclweb.org/anthology/W09-1106>
22. Pradhan, S., Hacioglu, K., Krugler, V., Ward, W., Martin, J.H., Jurafsky, D.: Support vector learning for semantic argument classification. *Mach. Learn.* 60(1-3), 11–39 (2005)
23. Rieck, K., Krueger, T., Brefeld, U., Mueller, K.R.: Approximate tree kernels. *Journal of Machine Learning Research* 11, 555–580 (2010)
24. Shalev-Shwartz, S., Singer, Y., Srebro, N.: Pegasos: Primal estimated sub-gradient solver for SVM. In: Ghahramani, Z. (ed.) *ICML. International Conference Proceeding Series*, vol. 227, pp. 807–814. ACM, New York (2007)
25. Shen, L., Joshi, A.K.: An SVM-based voting algorithm with application to parse reranking. In: Daelemans, W., Osborne, M. (eds.) *Proceedings of CoNLL HLT-NAACL 2003*, pp. 9–16 (2003), <http://www.aclweb.org/anthology/W03-0402.pdf>
26. Surdeanu, M., Ciaramita, M., Zaragoza, H.: Learning to rank answers on large online QA collections. In: *Proceedings of ACL-08, HLT, Columbus, Ohio* (2008), <http://www.aclweb.org/anthology/P/P08/P08-1082>
27. Williams, C., Seeger, M.: Using the nystrm method to speed up kernel machines. In: *Advances in Neural Information Processing Systems*, vol. 13, pp. 682–688. MIT Press, Cambridge (2001)
28. Yu, C.N.J., Joachims, T.: Training structural svms with kernels using sampled cuts. In: *ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 794–802 (2008)
29. Zaki, M.J.: Efficiently mining frequent trees in a forest. In: *KDD '02: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 71–80. ACM Press, New York (2002)
30. Zhang, D., Lee, W.S.: Question classification using support vector machines. In: *SIGIR*, pp. 26–32. ACM, New York (2003)
31. Zhang, M., Zhang, J., Su, J.: Exploring Syntactic Features for Relation Extraction using a Convolution tree kernel. In: *Proceedings of NAACL, New York City, USA*, pp. 288–295 (2006), <http://www.aclweb.org/anthology/N/N06/N06-1037>

# Synchronization Based Outlier Detection

Junming Shao<sup>1</sup>, Christian Böhm<sup>1</sup>, Qinli Yang<sup>2</sup>, and Claudia Plant<sup>3</sup>

<sup>1</sup> Institute of Computer Science, University of Munich, Germany

<sup>2</sup> School of Engineering, University of Edinburgh, UK

<sup>3</sup> Department of Scientific Computing, Florida State University, USA

**Abstract.** The study of extraordinary observations is of great interest in a large variety of applications, such as criminal activities detection, athlete performance analysis, and rare events or exceptions identification. The question is: how can we naturally flag these outliers in a real complex data set? In this paper, we study outlier detection based on a novel powerful concept: synchronization. The basic idea is to regard each data object as a phase oscillator and simulate its dynamical behavior over time according to an extensive Kuramoto model. During the process towards synchronization, regular objects and outliers exhibit different interaction patterns. Outlier objects are naturally detected by local synchronization factor (LSF). An extensive experimental evaluation on synthetic and real world data demonstrates the benefits of our method.

**Keywords:** Outlier Detection, Synchronization, Kuramoto model.

## 1 Introduction

*“An outlying observation, or outlier, is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism.”* [1]

Such irregular observations often contain useful information on abnormal behavior of the system described by the data. The detection of these irregular data is thus equally or even more interesting and useful than finding regular patterns applicable to a considerable portion of objects in a data set. For example, the identification of criminal activities, such as credit card fraud, is crucial in electronic commerce applications. The detection of potential outstanding players is critical for athlete performance analysis and management. The wide range of applications also include clinical trials, voting irregularity analysis, data cleansing, network intrusion, gene expression analysis, severe weather prediction, geographic information systems, and may more.

Currently, outlier detection has attracted increasing attention and many algorithms have been proposed (e.g. [2] [3] [4] [5]). However, they suffer from one or more of the following drawbacks: They explicitly or implicitly assume the data to follow a given distribution model, such as Gaussian, uniform or Exponential Power Distribution. The results of many methods strongly depend on suitable parametrization and/or their results are difficult to interpret. In addition, most

approaches are restricted to a flat data structure and do not support complex hierarchical data. A more detailed discussion on these studies will be given in Section 2.

In this paper, we consider outlier detection from a novel different point of view: *synchronization*. Synchronization is the phenomenon that a group of events spontaneously comes into co-occurrence with a common rhythm, despite of the differences between individual rhythms of the events. It is a powerful concept in nature regulating a large variety of complex processes ranging from the metabolism in the cell to social behavior in groups [6]. For example, the effect of synchrony has been described in experiments of people conversation, song or rhythm, or of groups of children interacting to an unconscious beat. In all cases the purpose of the common wave length or rhythm is to strengthen the group bond. The members lacking of synchrony are called “out of synchronization”.

To illustrate synchronization, consider for example opinion formation. In the beginning, each person usually has their own view about the problem. After mutual influence by conversation or discussion, people with similar educational background, age span, hobby, career or experience will easily talk together and finally form a common opinion (synchronization). Over time groups with different opinions emerge. For some people (outliers) with significantly different educational background, experience or other characteristics, it is not easy to join any group for discussion. Therefore, they tend to isolate from other people and keep their own opinions over time (out of synchronization). Inspired by such natural synchronization phenomena, we propose a novel technique for outlier detection. Our approach robustly identifies outliers based on their completely different behaviors in comparison to regular objects during the process towards synchronization.

The remainder of this paper is organized as follows: in the following section, we briefly survey the related work. Section 3 presents our algorithm in detail. Section 4 contains an extensive experimental evaluation and Section 5 concludes the paper.

## 2 Related Work

Currently, most existing approaches to outlier detection can be mainly classified into three categories: distribution-, distance-, and density-based outlier detection. In addition, a brief survey of the application of Kuramoto Model and synchronization is given.

**Distribution-based Outlier Detection.** Methods in this category are mainly developed in the field of statistics. Generally, they assume a known distribution model (Gaussian, Poisson, Exponential Power Distribution, etc.) for the observations based on statistical analysis of a given data set. Outliers are defined as those objects that deviate considerably from the model assumptions [7, 1, 8]. In [7], numerous discordancy tests are discussed for different scenarios. In [9, 10], authors propose SmartSifter (SS), which is an on-line real-time outlier detection

algorithm. The basic principle of SS is to use a probabilistic model (a finite mixture model) to represent the underlying distribution of a given data set. Each time a datum is input and SS employs an on-line learning algorithm to adjust the probability model. An anomaly score is calculated for each datum based on the learned model. However, the method relies on histograms and requires preparing as many Gaussian mixture models as cells in the histogram density. Moreover, in real-world applications, it is not trivial to find an appropriate model to fit an arbitrary data distribution without prior knowledge. Recently, CoCo, an information-theoretic outlier detection approach has been proposed by Böhm, et al. [5]. Based on the MDL principle, outliers are flagged as those objects which need more coding cost than regular objects. For coding each object, the optimal neighborhood size is heuristically determined. Independent Component Analysis and Exponential power distribution (EPD) are used to estimate the probability and the corresponding coding cost. Like most distribution-based methods, CoCo tends to fail if the estimated distribution does not fit the data model well. It is also time consuming to find the optimal neighborhood to estimate the coding cost for each object by screening for suitable neighborhood sizes.

**Distance-based Outlier Detection.** The concept of distance-based outlier detection is proposed by E.M. Knorr and R.T. Ng [3] [4]. These techniques identify potential outliers from ordinary points based on the number of points in the specified neighborhood. It defines a point in a data set  $T$  to be an outlier if at least  $p$  fraction of points in  $T$  have greater distance than  $d$  from it. The basic notion is extended in [11] by computing the distances to the  $k$  nearest neighbors and then ranking the objects based on their proximity to their  $k$ -th nearest neighbors. Consequently top  $n$  outliers are obtained using a partition-based algorithm. However, it is difficult to accurately determine the parameters  $p$  and  $d$  for a arbitrary data set.

**Density-based Outlier Detection.** M. Breunig, et al. [2], introduce a notion of local outlier from a density-based perspective. An object is regarded as an outlier if its local density does not fit well into the density of its neighboring objects. The local outlier factor (LOF) is then proposed to capture the degree to which the object is an outlier. It is defined as the average of the ratio of the local reachability density of the object and those of the objects in its neighborhood. A LOF value of approximately 1 indicates the object is located inside a cluster, while the objects with higher LOF values are more rather considered as outliers. In [12], a connectivity-based outlier factor (COF) scheme is proposed to improve the effectiveness of LOF scheme when a pattern itself has a similar neighborhood density as an outlier. Although these approaches to outlier detection are useful, their performances are sensitive to the parameter  $Minpts$  which can be very difficult to determine. The Local Outlier Integral (LOCI) [13] flags outliers, based on probabilistic reasoning and motivated from the concept of a multi-granularity deviation factor (MDEF). Similar to LOF, the LOCI outlier model takes the local object density into account, but differently, the MDEF of LOCI uses  $\epsilon$ -neighborhoods rather than  $MinPts$  nearest neighbors. The local

neighborhood in LOCI model is defined by two parameters: the counting and the sampling neighborhood. The counting neighborhood specifies some volume of the feature space which is used to estimate the local object density. The sampling neighborhood is larger than the counting neighborhood and contains all points which are used to compute the average object density in the neighborhood. Objects which deviate in their local object density more than three times of the standard deviation are regarded as outliers. The flagging scheme of LOCI thus assumes the object densities follow a Gaussian distribution.

**Kuramoto Model and Synchronization.** Currently, the study of synchronization phenomena have widely used in physical, biological, chemical, and social systems. The Kuramoto model [14] [15] is one of the most famous models to explore collective synchronization. Arenas et al. [16] apply the Kuramoto model for network analysis, and study the relationship between topological scales and dynamic time scales in complex networks. This analysis provides a useful connection between synchronization dynamics, network topology and spectral graph analysis. Recently, the Kuramoto model has attracted some attention in clustering [17] [18]. Aeyels et. al [19] introduce a mathematical model for the dynamics of chaos system. They characterize the data structure by a set of inequalities in the parameters of the model and apply it to a system of interconnected water basins. In summary, previous approaches mainly focus on the synchronization phenomena of a dynamic system from a global perspective. Inspired by ideas from the synchronization phenomena and existing dynamical system analysis, we propose a novel outlier detection technique based on synchronization.

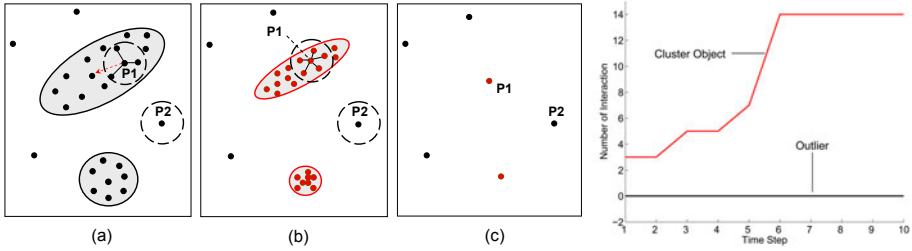
### 3 Synchronization-Based Outlier Detection

In this section, we introduce SOD, to detect outliers based on synchronization principle. We first illustrate the basic idea and then propose an extensive Kuramoto Model for outlier detection. In Section 3.3 we discuss the algorithm SOD and its properties in detail.

#### 3.1 Basic Idea

The concept of synchronization provides a natural way to outlier detection. The basic idea is to flag outliers by distinguishing the object dynamical behaviors during the process towards synchronization. In our work, each data object is regarded as a phase oscillator and interacts dynamically with other objects according to an Extensive Kuramoto model (EKM), which we will introduce in Section 3.2.

To give an intuition of the synchronization-based outlier detection, let's consider a simple data set as illustrated in Figure 1. For an object within a cluster, such as  $P_1$ , many similar objects are located around it and  $P_1$  starts interacting with these similar objects. Through non-linear dynamical interaction, the object changes its initial phase and moves towards the main direction of its interaction



**Fig. 1.** Illustration of synchronization-based outlier detection. (a)-(c): The dynamics of objects towards synchronization. (d): Interaction Plot.

partners (Figure 1(a)). The situation is like the mutual influence of people in discussion. As time evolves, regular objects move gradually closer together through mutual interaction and thus more and more objects can interact with them. Figure 1(b) displays the new positions of the objects for comparison. The objects with similar attributes gradually synchronize together. The initial points (black color) are replaced by the red points after one time stamp. Then, in a sequential process, all these similar objects synchronize together, which finally have the same phase (Figure 1(c)). Outliers, such as  $P2$ , due to the significantly different attributes in comparison with regular objects, have difficulties to interact with other objects and tend to keep their own phases. Therefore, the dynamics of the objects show two different patterns during the process towards synchronization. For each regular object, as time evolves, it interacts with more and more objects and finally synchronized together with other objects. For outliers, there is none or only very minor interaction. Strong outliers keep their unique phase over the whole time during the process towards synchronization. The two different patterns can be easily visualized: Figure 1(d) shows the *Interaction Plot*, which displays for each object the number of interactions on the time scale.

### 3.2 Extensive Kuramoto Model

One of the most successful attempts to understand collective synchronization phenomena is due to Kuramoto [14] [15], who analyzes a model of phase oscillators which are coupled through the sine of their phase differences. The *Kuramoto model* (KM) consists of a population of  $N$  coupled phase oscillators where the phase of the  $i$ -th unit, denoted by  $\theta_i$ , evolves in time according to the following dynamics:

$$\frac{d\theta_i}{dt} = \omega_i + \frac{K}{N} \sum_{j=1}^N \sin(\theta_j - \theta_i), (i = 1, \dots, N), \tag{1}$$

where  $\omega_i$  stands for its natural frequency and  $K$  describes the coupling strength between units.  $\frac{d\theta_i}{dt}$  denotes the instantaneous frequency.  $\theta_i$  describes the phase of  $i$ -th unit.

The Kuramoto model well describes the global synchronization behavior of all coupled phase oscillators. In real-life, this situation rarely occurs. Partial

synchronization is observed more frequently, which is the case when a local ensemble of oscillators are synchronized together. It is also observed that the sets of oscillators with high similarity synchronize more easily than those with large variance. Therefore, partial synchronization provides rich information about the object behaviors. In order to explore the different dynamic behaviors between regular objects and outliers, we extensively reformulate Eq.(II). Formally, we first need to define the notion of  $\epsilon$ -neighborhood.

**Definition 1:** ( $\epsilon$ -neighborhood of an object  $x$ ) The  $\epsilon$ - neighborhood of object  $x$ , which denoted by  $Nb_\epsilon(x)$ , is defined as:

$$Nb_\epsilon(x) = \{y \in \mathcal{D} | dist(y, x) \leq \epsilon\}, \tag{2}$$

where  $dist(y, x)$  is metric distance function.

**Definition 2:** (Extensive Kuramoto model) Let  $x \in \mathbb{R}^d$  be an object in the data set  $\mathcal{D}$  and  $x_i$  be the  $i$ -th dimension of the data object  $x$  respectively. We regard each object  $x$  as a phase oscillator, according to Eq.(II), with a  $\epsilon$ -neighborhood interaction. The dynamics of each dimension  $x_i$  of the object  $x$  is governed by:

$$\frac{dx_i}{dt} = \omega_i + \frac{K}{|Nb_\epsilon(x)|} \sum_{y \in Nb_\epsilon(x)} \sin(y_i - x_i). \tag{3}$$

Let  $dt = \Delta t$ , then:

$$x_i(t + 1) = x_i(t) + \Delta t \cdot \omega_i + \frac{\Delta t \cdot K}{|Nb_\epsilon(x(t))|} \cdot \sum_{y \in Nb_\epsilon(x(t))} \sin(y_i(t) - x_i(t)). \tag{4}$$

For unsupervised outlier detection we assume that all objects having the same frequency  $w$ , since we have no external knowledge on the data. Thus the term  $\Delta t \cdot \omega_i$  is the same for each object and can be ignored. Similarly,  $\Delta t \cdot K$  is a constant which we set to 1. Finally the dynamics of each dimension  $x_i$  of the object  $x$  over time is provided by:

$$x_i(t + 1) = x_i(t) + \frac{1}{|Nb_\epsilon(x(t))|} \cdot \sum_{y \in Nb_\epsilon(x(t))} \sin(y_i(t) - x_i(t)). \tag{5}$$

The object  $x$  at time step  $t = 0$ :  $x(0)$  ( $x_1(0), \dots, x_d(0)$ ) represents the initial phase of the object (the original location of object  $x$ ). The  $x_i(t + 1)$  describes the renewal phase value of  $i$ -th dimension of object  $x$  at the  $t = (0, \dots, T)$  time evolution.

To characterize the level of synchronization between oscillators during the process, an order parameter needs be defined. Instead of considering a global observable, we define a local order parameter  $r$ , measuring the coherence of local oscillator population.

**Definition 3:** (Local Order Parameter) The local order parameter  $r$  characterizing the degree of local synchronization is provided by:

$$r = \frac{1}{N} \sum_{i=1}^N \left( \sum_{y \in Nb_{\epsilon}(x)} e^{-\|y-x\|} \Big|_{x \in \mathcal{D}} \right). \quad (6)$$

The value of  $r$  increases as more neighbors synchronize together with time evolution. The process toward synchronization will terminate when  $r$  converges, which indicates local similar objects achieve phase coherence. At this moment, all local similar objects have the same phase (location).

### 3.3 The SOD Algorithm

In this section, we elaborate the *SOD* algorithm based on our extensive Kuramoto model.

First, without any interaction, all objects in a data set have their own phases. As time evolves, each object starts to interact with its  $\epsilon$ -neighborhood, cf. Definition 1. The traces of all objects are in line with the main direction of their neighborhoods. Gradually, regular objects with similar attributes synchronize together following the intrinsic structure of a data set. In contrast, outliers are difficult to interact with other objects due to the large variance. Finally, the local regular objects with similar attributes synchronize together with same phase while outliers tend to keep their original phases. The objects synchronization process is terminated when the local order parameter converges.

To simply illustrate the objects dynamical movement, the Figure 2 (a)-(d) shows the detailed dynamics of 2-dimensional points at time steps:  $t = 0, 1, 3, 5$ .  $t = 0$  indicates the original data set at the initial time. From that moment on, all objects with similar attributes start to synchronize together through the dynamical interaction according to Eq.(5) and finally, all objects in the data set synchronize at two different phases after 5 time steps. A more intuitive visualization of the objects movement is illustrated in Figure 2(e). Figure 2(f) demonstrates the local order parameter of the data set with time evolution.

**Definition 4:** (Local Synchronization Factor of an object  $x$ ) The local synchronization factor  $LSF(x)$  of object  $x$  is defined as:

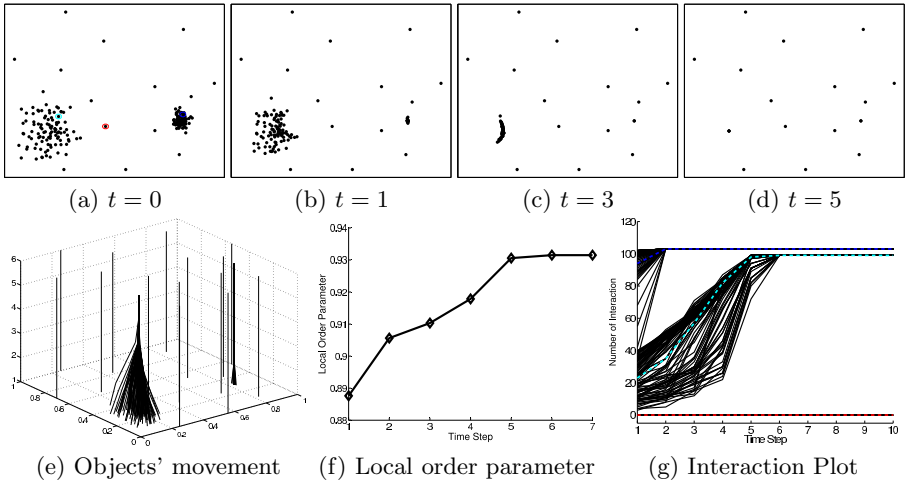
$$LSF(x) = \frac{1}{T} \sum_{t=0}^T \left( \frac{1}{|Nb_{\epsilon}(x(t))|} \sum_{y(t) \in Nb_{\epsilon}(x(t))} \cos(\|y(t) - x(t)\|) \right). \quad (7)$$

where  $T$  is the whole time steps for the process of synchronization. The synchronization factor captures the degree to which the object of being an outlier. The smaller the LSF value, the higher the probability of being an outlier.

According to the definition, LSF shows major desirable properties:

1. *Intuitive.* Since the LSF value indicates each object synchronization factor, it provides an intuitive way to summarize its dynamical interaction behavior





**Fig. 2.** The dynamics of objects toward to synchronization. (a)-(d): The detailed objects' movement during the time resolution. (e): Objects movement. (f): Local order parameter, (g): Interaction Plot for three objects circled in (a).

with other objects during the process towards synchronization. The easier an object synchronizes with other objects, the higher is its LSF value. Outliers are objects which are out of synchronization.

2. *Tightness.* The range of LSF is restricted to  $[0, 1)$ . The lower bound of LSF value is 0, which means that the object does not interact with any other object during the synchronization process. For cluster points which easily synchronize the LSF value is close to 1. The LSF value can thus be easily interpreted as the probability of each object of being an outlier, e.g.  $\text{Probability}(p) = 1 - \text{LSF}(p)$ .
3. *Distinguishable.* Due to the different dynamics between regular objects and outliers during the synchronization process, the values of LSF are fairly distinguishable. For outliers, the LSF value is around 0 while the regular objects nearly to 1. It can easily discern them.

In addition, in order to explore each object dynamics during the process towards synchronization, the *Interaction Plot* is defined to characterize the interaction behavior pattern between objects over time.

**Definition 5:** (*Interaction Plot*) For any object  $x$ , the plot of the number of objects involved in mutual interaction with  $x$  versus the time step, is called *Interaction Plot*.

As a valuable addition to LSF, the *Interaction Plot* provides a detailed visualization of the dynamic behavior of each object according to the Extensive Kuramoto model. With the same data set above, the interaction plot is illustrated in Figure 2(g). From this plot, two distinct interaction patterns become evident. For regular objects, during the synchronization process, more and more

objects interact together along with the time steps. Outliers often fail to interact with other objects (maybe a few at the beginning). As time evolves, the number of objects for interaction tend to keep the same. For example, two regular objects and one outlier are visualized with dash color lines to illustrate the different interaction patterns in Figure 2 (a),(g).

**Outliers Flagging.** After LSF is obtained for each object, all outliers exhibit usually low values in comparison to the regular objects. The denser of the local region of an object, the higher its value of LSF. Therefore, selecting a suitable threshold for flagging outliers could be easily selected since the LSF value is distinct for outliers and regular objects. However, for automatically flagging, in this work, the K-Means algorithm are applied on the LSF values to split the data into two clusters: outliers and regular objects. Finally, the Pseudocode of the *SOD* is illustrated in Algorithm 1.

---

**Algorithm 1.** *SOD*( $D, \epsilon$ )

---

```

LSF := {}; // Synchronization Factors
while loopFlag=true do
  for each object  $p \in D$  do
    Compute  $Nb_\epsilon(p)$ ;
    Obtain new value of object  $p$  using Eq.(5); // Update value
  end for
  Compute local order  $r$  using Eq.(6);
  if  $r$  converges then
    loopFlag=false;
    for each object  $p \in D$  do
      Compute local synchronization factor  $LSF(p)$  using Eq.(7);
    end for
  end if
end while

```

---

Flagging outliers by K-Means based on LSF values.

---

**Runtime Complexity.** For *SOD*, to detect outliers based on synchronization, the runtime complexity with respect to the number of data objects is  $O(T \cdot N^2)$ , where  $N$  is the number of objects and  $T$  is the time evolution. In most cases,  $T$  is small with  $5 \leq T \leq 20$ . If there exists an efficient index, the complexity reduces to  $O(T \cdot N \log N)$ .

**Parameter Setting.** In order to flag outliers based on synchronization principle, an interaction range ( $\epsilon$ ) needs to be specified for EKM. The question is: how to determine the  $\epsilon$  value and how does the LSF value change when the  $\epsilon$  value is adjusted?

Given a data set, theoretically, the  $\epsilon$  value can be 0, which means there is no interaction at all among the objects. In order to generate object interaction, there should be a lower bound of  $\epsilon$ . To generate a stable interaction for most

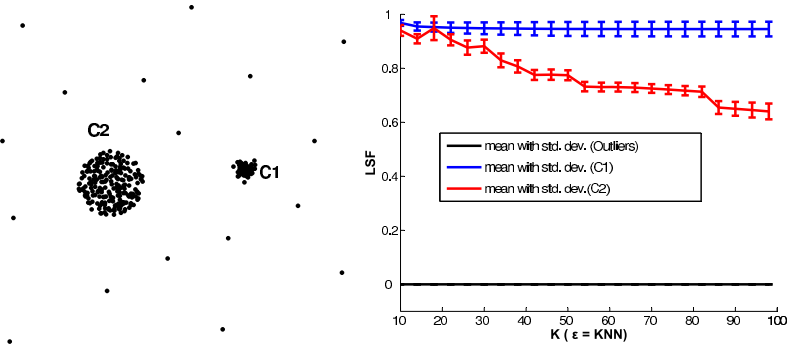


Fig. 3. Influence of  $\epsilon$  on LSF

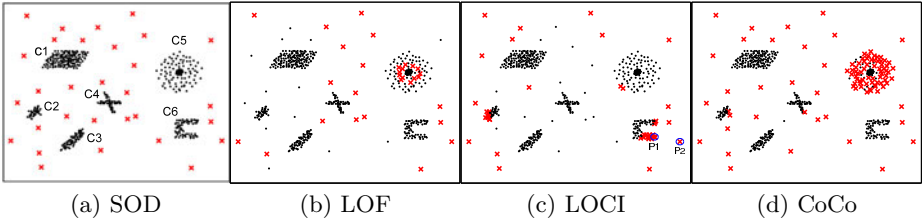
objects, a heuristic way is to use the average value of the  $k$ -nearest neighbor distance determined by a sample from the data set for a small  $k$ . The  $\epsilon$  should not be very large that enclose all data objects for interaction. In that way, we can not detect the local object dynamical behaviors. However, the range of suitable values for  $\epsilon$  is very large. In all experiments on different data sets, we set  $\epsilon$  the the average value of the  $k$ -nearest neighbor distances for  $k$  ranging from 10 to 50.

To asses the impact of  $\epsilon$  value on LSF value and outlier detection, Figure 3 shows a simple data set which consists of 2 clusters with different size (C1:50, C2:200) and density. 17 outliers are added to the data. For each  $\epsilon$  value, the mean and standard deviation of LSF values are calculated for each cluster as well as for the outliers. Figure 3 displays the LSF value with respect to  $\epsilon$ . For all settings of  $\epsilon$ , the mean LSF value for the points in cluster C1 and C2 are clearly much larger than 0, in most cases close to 1 while the mean and standard deviation of LSF value for outliers remain stable at 0. With the increase of  $\epsilon$ , the LSF value of cluster objects begin to decrease. The reason behind it is that more and more objects are enclosed to interact with each other at each time step and thus more difficult to synchronize together. The situation is like two persons are much easier to agree with one thing than a larger group of people. Moreover, since objects in denser cluster are easier to synchronize, the LSF values are much more closer to 1 (e.g. the mean LSF value of objects in C1 are larger than that in C2). For different parameters, outliers and regular objects show distinct LSF values and can be discerned easily. For further evaluation on the robustness of SOD w.r.t. parameter settings please refer to the experimental section.

## 4 Experimental Evaluation

In the following we evaluate our outlier detection *SOD* in comparison to LOF [2], LOCI [13], CoCo [5] on synthetic data set as well as NBA data. We implemented SOD and LOF in Java and obtained the implementation of LOCI and CoCo from the authors.

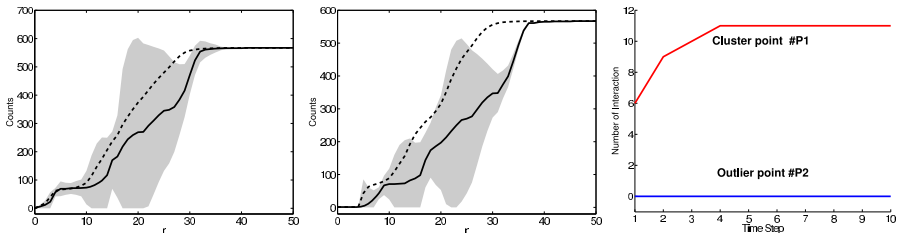
**Synthetic Data.** We start the evaluation with two-dimensional synthetic data sets to facilitate presentation. The data set displayed in Figure 4 consists of six clusters (C1-C6): one Gaussian cluster (C2:39), two correlation clusters (C1:167 and C3:73), two arbitrarily shaped clusters (C4:60 and C6:67) and one a spherical hierarchical cluster (C5:131), including a nested cluster. 30 outliers are added to the data set. Figure 4 provides the results of outlier detection by using SOD, LOF, LOCI and CoCo for the same synthetic data set.



**Fig. 4.** Outlier detection results with different methods. (a): SOD ( $k = 10 - 40$ ), (b): LOF ( $MinPts = 20$ , selecting only the top 30 outliers), (c): LOCI ( $\alpha = 1$ ,  $rmin = 10$  and  $rmax = 50$ ), (d): CoCo. Detected outliers are highlighted with red crosses.

Without any prior knowledge, SOD successfully detects all 30 outlier points (Figure 4 (a)) from the complex data. The outliers are highlighted with red crosses and cluster objects are shown in black. Moreover, SOD obtains the same result with the parameter  $\epsilon$  set to the average  $k$ -nearest neighbor distance for  $k$  ranging from 10 to 40.

For LOF, we try a wide range of different settings for the parameters  $MinPts$  from 10 to 50, which is suggested by authors. The top 30 outliers are obtained according to the LOF value. For different parameters, there are 17, 17, 9, 8 and 10 out of 30 correctly assigned with  $MinPts = \{10, 20, 30, 40, 50\}$ , respectively. Most cluster objects, especially for the objects of hierarchical cluster are wrongly flagged as outliers. The best result is presented in Figure 4 (b) obtained with parameter  $MinPts = 20$ . Obviously, the result of LOF is very much influenced by the parameter  $MinPts$ . In addition, we often have no a priori information about the number of desired outliers.



**Fig. 5.** LOCI Plot of cluster point P1 (left) and outlier P2(middle); Interaction Plot of P1 and P2 (right)

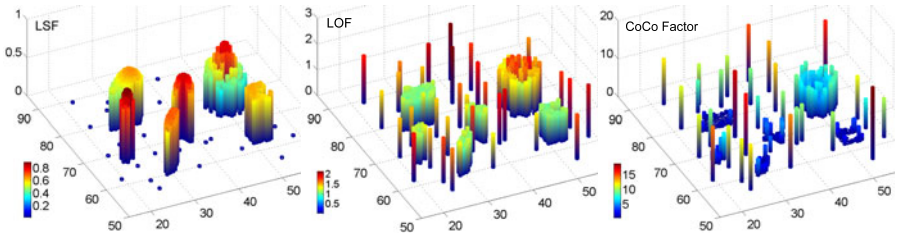


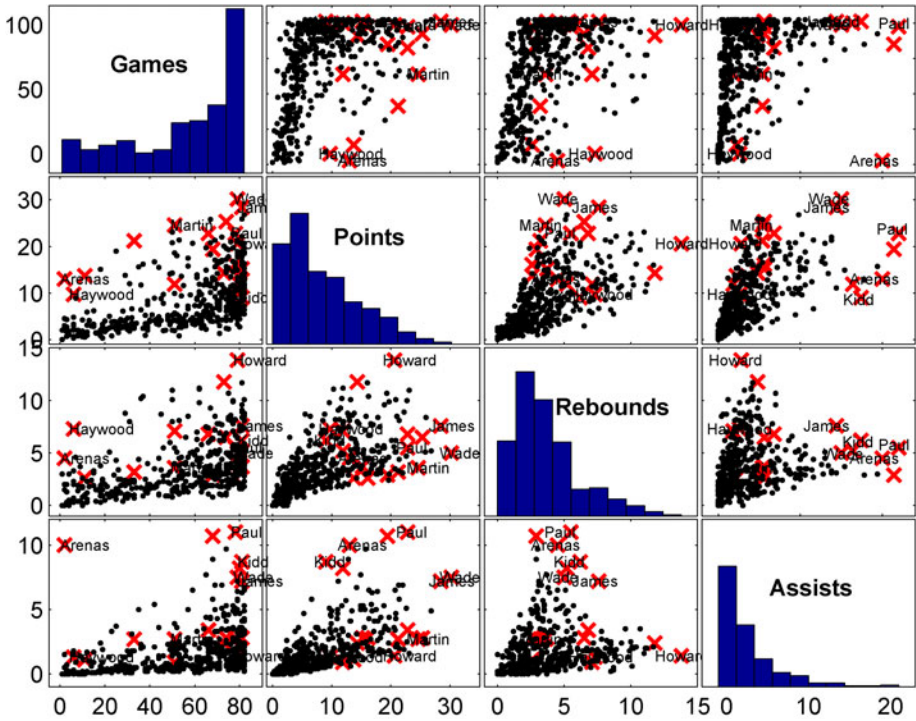
Fig. 6. Visualization of the LSF, LOF and CoCo for the synthetic data set

LOCI is applied to our synthetic data set with  $\alpha = 1$ ,  $rmin = 10$  and  $rmax = 50$  (Figure 4 (c)). 46 outlier points are detected based on the suggested outlier flagging criteria and 16 true outliers are correctly detected. Many cluster points of C2, C4 and C6 are wrongly flagged. With the decrease of  $rmin$  value, more true outlier points are found, but at the same time more cluster points are mislabeled as outliers. As the LOCI plot provides the information for each object, we thus have a closer look at the cluster point (Fig. 4 (c):  $P1$ ) and outlier point ( $P2$ ). It is noticeably that the two LOCI plots look very similar (Figure 5(a)-(b)). For comparison, *Interaction Plot* is displayed for the same two points (Figure 5(c)). It is very clear that the cluster point  $P1$  and outlier point  $P2$  have totally different characteristics.

CoCo identifies all 95 outliers for the synthetic data and only one outlier is missing. However, 66 cluster points are wrongly flagged as outliers. CoCo relies on the assumption that the data follows an Exponential Power Distribution, which is not the case for our example and therefore CoCo yields many false-positives.

To visualize the degree of “outlierness” for each object, the Local Synchronization Factor (LSF) is further compared to the outlier factor of LOF and CoCo in Figure 6. For LSF, the local synchronization factor of all outlier points are nearly 0 and all cluster points are close to 1. Due to the desirable properties of LSF, such as tightness and distinguishable, cluster points and outliers are easily to differentiate. As to LOF and CoCo, the range of values is very wide and the gap between outliers and cluster points is not clear. For example, the range of LOF is from 0.85 to 2.15, which makes it difficult to determine a suitable threshold for outlier flagging.

**NBA Performance Statistics.** After extensive evaluation of SOD on synthetic data sets, we apply our novel outlier detection method to the real data. We use the NBA data available at the NBA website <http://www.nba.com>. In the Season 2008/09, the performance of 444 players are described with four attributes: the number of games played (GP), the number of points (PPG), the rebounds (RPG), and assists (APG) per game. SOD is applied to this NBA data detecting 18 outliers with  $k = 30$ . Figure 7 displays scatter plots of the data with different attributes and the histogram of each attribute in the diagonal respectively. Obviously, the data distribution is non-Gaussian. All 18 outliers are



**Fig. 7.** Outlier detection with SOD for the NBA data set. All 18 outliers are marked in red. The strongest outliers with  $LSF = 0$  are also marked with the name.

highlighted in red. For the most outstanding players with  $LSF = 0$  also the names are provided. For comparison with other techniques, Table 1 lists the top 10 outliers for various settings of  $k$ . For different parametrization ( $k=30,40,50$ ) the same players are among the top 10 outliers of SOD. Eight of 10 players are strongest outliers with a  $LSF$  of 0 for all parameterizations. For comparison, Table 2 lists the top 10 outliers identified by LOF with  $MinPts = 50$ . Only seven players are reproducibly detected as top 10 for  $MinPts = 40$ . LOCI ( $\alpha = 1$ ,  $rmin = 10$  and  $rmax = 50$ ) and CoCo detect the top 10 outliers listed in Table 3 and Table 4, respectively. For the comparison methods, the intersection with SOD is marked in bold. Gilbert Arenas with  $LSF = 0$  is a strong outlier which is among the top 10 for all methods. Having played only 2 games, he has shown outstanding performance in terms of rebounds, points and especially in terms of assists. Most other players with an  $LSF$  of zero are also among the top 10 of at least one of the comparison methods, e. g. the well-known and truly outstanding players Brendan Haywood, Dwyane Wade and LeBron James. Interestingly, Chris Paul is not among the top 10 of any comparison methods although he is especially outstanding in the number of points and assists per game. In the 2006-07 season he has been ranked fourth in the overall NBA in assists ([http://www.nba.com/playerfile/chris\\_paul/bio.html](http://www.nba.com/playerfile/chris_paul/bio.html)). Another strong outlier missed by the comparison methods is Dwight

**Table 1.** Top 10 outliers identified by SOD on NBA data

LSF( $k=30$ )	( $k=40$ )	( $k=50$ )	Name	GP	PPG	RPG	APG
0	0	0	Chris Paul (NOH)	78	22.8	5.5	11
0	0	0	Jason Kidd (DAL)	81	9	6.2	8.7
0	0	0	Dwyane Wade (MIA)	79	30.2	5	7.5
0	0	0	LeBron James (CLE)	81	28.4	7.6	7.2
0	0	0	Kevin Martin (SAC)	51	24.6	3.6	2.7
0	0	0	Dwight Howard (ORL)	79	20.6	13.8	1.4
0	0	0	Gilbert Arenas (WAS)	2	13	4.5	10
0	0	0	Brendan Haywood (WAS)	6	9.7	7.3	1.3
0	0.036	0	Cuttino Mobley (LAC)	11	13.7	2.6	1.1
0	0.260	0.035	Deron Williams (UTA)	68	19.4	2.9	10.7

**Table 2.** Top 10 outliers identified by LOF on NBA data

LOF	Name	GP	PPG	RPG	APG
1.5058	<b>Gilbert Arenas (WAS)*</b>	2	13	4.5	10
1.4938	<b>Brendan Haywood (WAS)*</b>	6	9.7	7.3	1.3
1.4463	DJ White (OKC)*	7	8.9	4.6	0.9
1.4069	Michael Redd (MIL)*	33	21.2	3.2	2.7
1.4011	<b>Cuttino Mobley (LAC)</b>	11	13.7	2.6	1.1
1.3623	Carlos Boozer (UTA)*	37	16.2	10.4	2.1
1.3532	Monta Ellis (GSW)*	25	19	4.3	3.7
1.3492	Elton Brand (PHI)*	29	13.8	8.8	1.3
1.3365	Chris Kaman (LAC)	31	12	8	1.5
1.3294	Tracy McGrady (HOU)	35	15.6	4.4	5

**Table 3.** Top 10 outliers identified by LOCI on NBA data

Name	GP	PPG	RPG	APG
<b>Dwyane Wade (MIA)</b>	79	30.2	5	7.5
<b>LeBron James (CLE)</b>	81	28.4	7.6	7.2
Ben Wallace (CLE)	56	2.9	6.5	0.8
Monta Ellis (GSW)	25	19	4.3	3.7
Pops Mensah-Bonsu (TOR-SAS)	22	5	5.1	0.3
Corey Brewer (MIN)	15	6.2	3.3	1.7
<b>Gilbert Arenas (WAS)</b>	2	13	4.5	10
Kobe Bryant (LAL)	82	26.8	5.2	4.9
<b>Jason Kidd (DAL)</b>	81	9	6.2	8.7
Michael Redd (MIL)	33	21.2	3.2	2.7

Howard who has been named the NBA Defensive Player of the Year in 2008-2009 season ([http://www.nba.com/playerfile/dwight\\_howard/bio.html](http://www.nba.com/playerfile/dwight_howard/bio.html)). Dwight Howard is especially characterized by an outstanding number of 13.8 rebounds per game.

**Table 4.** Top 10 outliers identified by CoCo on NBA data

CoCo o.f.	Name	GP	PPG	RPG	APG
20.76	Michael Redd (MIL)	33	21.2	3.2	2.7
17.39	Andrew Bogut (MIL)	36	11.7	10.2	2
17.31	<b>Kevin Martin (SAC)</b>	51	24.6	3.6	2.7
16.55	Monta Ellis (GSW)	25	19	4.3	3.7
14.42	Elton Brand (PHI)	29	13.8	8.8	1.3
13.84	<b>Gilbert Arenas (WAS)</b>	2	13	4.5	10
13.80	Tracy McGrady (HOU)	35	15.6	4.4	5
12.12	Kobe Bryant (LAL)	57	17.5	3	5
11.85	Cuttino Mobley (LAC)	11	13.7	2.6	1.1
11.45	Tyson Chandler (NOH)	45	8.8	8.7	0.5

## 5 Conclusions

In this paper, we propose SOD, a novel outlier detection algorithm inspired by synchronization phenomenon. The major benefits of SOD can be summarized as follows:

1. *Natural outlier detection.* Based on the synchronization principle, outliers are naturally flagged from a data set due to their unique dynamical interaction pattern during the process towards synchronization.
2. *Intuitive to interpret.* Outliers are represented as those objects which hardly interact with other objects since they have been generated by a different mechanism. Outlier objects are the members of the system which are out of synchronization. The probability for each object of being an outlier can be characterized by a numerical value: The Local Synchronization Factor (LSF). The *Interaction Plot* provides a detailed view of the dynamical behavior pattern of each object.
3. *Without any data distribution assumption.* Without any data distribution assumption or any prior knowledge, outliers are easily flagged by investigating the different interact patterns, which are driven by the intrinsic data structure.
4. *Complex data handling.* The SOD allows to detect outliers from a complex data set including clusters with arbitrary number, shape, size and densities as well as hierarchical data structures.
5. *Robustness to parametrization.* The outlier detection result is insensitive to parameter settings.

In ongoing and future work, we will focus on fully automatic outlier detection based on the powerful concept of synchronization and study online algorithms for outlier detection in data streams, which is essential in many applications.



## References

1. Hawkins, D.: Identification of Outliers. Chapman and Hall, London (1980)
2. Breunig, M.M., Kriegel, H.-P., Ng, R.T., Sander, J.: Lof: identifying density-based local outliers. In: Proceedings of the ACM SIGMOD Conference (2000)
3. Knorr, E.M., Ng, R.T.: Algorithms for mining distance-based outliers in large datasets. In: VLDB, pp. 392–403 (1998)
4. Knorr, E.M., Ng, R.T.: Finding intensional knowledge of distance-based outliers. In: VLDB, pp. 211–222 (1999)
5. Boehm, C., Haegler, K., Mueller, N.S., Plant, C.: CoCo: Coding Cost For Parameter-Free Outlier Detection. In: Proc. ACM SIGKDD 2009, pp. 149–158 (2009)
6. Arenas, J.K.Y.M.A., Guilera, A.D., Zhou, C.S.: Synchronization in complex networks. Phys. Rep. 469, 93–1535 (2008)
7. Barnett, V., Lewis, T.: Outliers in Statistical Data. John Wiley, Chichester (1994)
8. Rousseeuw, P.J., Leroy, A.M.: Robust Regression and Outlier Detection. John Wiley and Sons, Chichester (1987)
9. Yamanishi, K., Takeuchi, J., Williams, G., Milne, P.: On-line unsupervised outlier detection using finite mixtures with discounting learning algorithm. In: Proceedings of KDD 2000, pp. 320–324 (2000)
10. Yamanishi, K., Takeuchi, J.: Discovering Outlier Filtering Rules from Unlabeled Data. In: Proc. ACM SIGKDD 2001, pp. 389–394 (2001)
11. Ramaswamy, S., Rastogi, R., Kyuseok, S.: Efficient Algorithms for Mining Outliers from Large Data Sets. In: Proc. ACM SIGMOD Int. Conf. on Management of Data (2000)
12. Tang, J., Chen, Z., Fu, A.W.-C., Cheung, D.W.: Enhancing effectiveness of outlier detections for low density patterns. In: Chen, M.-S., Yu, P.S., Liu, B. (eds.) PAKDD 2002. LNCS (LNAI), vol. 2336, p. 535. Springer, Heidelberg (2002)
13. Papadimitriou, S., Kitagawa, H., Gibbons, P.B., Faloutsos, C.: LOCI: Fast Outlier Detection Using the Local Correlation Integral. In: Proceedings of IEEE International Conference on Data engineering, Bangalore, India (2003)
14. Kuramoto, Y.: In: Araki, H. (ed.) Proceedings of the International Symposium on Mathematical Problems in Theoretical Physics. Lecture Notes in Physics, pp. 420–422. Springer, New York (1975)
15. Kuramoto, Y.: Chemical oscillations, waves, and turbulence. Springer, New York (1984)
16. Arenas, A., Diaz-Guilera, A., Perez-Vicente, C.J.: Plasticity and learning in a network of coupled phase oscillators. Phys. Rev. Lett. 96 (2006)
17. Kim, C.S., Bae, C.S., Tcha, H.J.: A phase synchronization clustering algorithm for identifying interesting groups of genes from cell cycle expression data. BMC Bioinformatics 9(56) (2008)
18. Böhm, C., Plant, C., Shao, J., Yang, Q.: Clustering by Synchronization. In: Proc. of the 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA (2010)
19. Aeyels, D., Smet, F.D.: A mathematical model for the dynamics of clustering. Physica D: Nonlinear Phenomena 273(19), 2517–2530 (2008)

# Laplacian Spectrum Learning

Pannagadatta K. Shivaswamy and Tony Jebara

Department of Computer Science  
Columbia University  
New York, NY 10027  
{pks2103, jebara}@cs.columbia.edu

**Abstract.** The eigenspectrum of a graph Laplacian encodes smoothness information over the graph. A natural approach to learning involves transforming the spectrum of a graph Laplacian to obtain a kernel. While manual exploration of the spectrum is conceivable, non-parametric learning methods that adjust the Laplacian's spectrum promise better performance. For instance, adjusting the graph Laplacian using kernel target alignment (KTA) yields better performance when an SVM is trained on the resulting kernel. KTA relies on a simple surrogate criterion to choose the kernel; the obtained kernel is then fed to a large margin classification algorithm. In this paper, we propose novel formulations that jointly optimize relative margin and the spectrum of a kernel defined via Laplacian eigenmaps. The large relative margin case is in fact a strict generalization of the large margin case. The proposed methods show significant empirical advantage over numerous other competing methods.

**Keywords:** relative margin machine, graph Laplacian, kernel learning, transduction.

## 1 Introduction

This paper considers the transductive learning problem where a set of labeled examples is accompanied with unlabeled examples whose labels are to be predicted by an algorithm. Due to the availability of additional information in the unlabeled data, both the labeled and unlabeled examples will be utilized to estimate a kernel matrix which can then be fed into a learning algorithm such as the support vector machine (SVM). One particularly successful approach for estimating such a kernel matrix is by transforming the spectrum of the graph Laplacian [8]. A kernel can be constructed from the eigenvectors corresponding to the smallest eigenvalues of a Laplacian to maintain smoothness on the graph. In fact, the diffusion kernel [5] and the Gaussian field kernel [12] are based on such an approach and explore smooth variations of the Laplacian via specific parametric forms. In addition, a number of other transformations are described in [8] for exploring smooth functions on the graph. Through the controlled variation of the spectrum of the Laplacian, a family of allowable kernels can be explored in an attempt to improve classification accuracy. Further, Zhang & Ando [10] provide generalization analysis for spectral kernel design.

Kernel target alignment (KTA for short) [3] is a criterion for evaluating a kernel based on the labels. It was initially proposed as a method to choose a kernel from a family of candidates such that the Frobenius norm of the difference between a label matrix and the kernel matrix is minimized. The technique estimates a kernel independently of the final learning algorithm that will be utilized for classification. Recently, such a method was proposed to transform the spectrum of a graph Laplacian [11] to select from a general family of candidate kernels. Instead of relying on parametric methods for exploring a family of kernels (such as the scalar parameter in a diffusion or Gaussian field kernel), Zhu *et al.* [11] suggest a more general approach which yields a kernel matrix non-parametrically that aligns well with an ideal kernel (obtained from the labeled examples).

In this paper, we propose novel quadratically constrained quadratic programs to jointly learn the spectrum of a Laplacian with a large margin classifier. The motivation for large margin spectrum transformation is straightforward. In kernel target alignment, a simpler surrogate criterion is first optimized to obtain a kernel by transforming the graph Laplacian. Then, the kernel obtained is fed to a classifier such as an SVM. This is a two-step process with a different objective function in each step. It is more natural to transform the Laplacian spectrum jointly with the classification criterion in the first place rather than using a surrogate criterion to learn the kernel.

Recently, another discriminative criterion that generalizes large absolute margin has been proposed. The large *relative* margin [7] criterion measures the margin relative to the spread of the data rather than treating it as an absolute quantity. The key distinction is that large relative margin jointly maximizes the margin while controlling or minimizing the spread of the data. Relative margin machines (RMM) implement such a discriminative criterion through additional linear constraints that control the spread of the projections. In this paper, we consider this aggressive classification criterion which can potentially improve over the KTA approach. Since large absolute margin and large relative margin criteria are more directly tied to classification accuracy and have generalization guarantees, they potentially could identify better choices of kernels from the family of admissible kernels. In particular, the family of kernels spanned by spectral manipulations of the Laplacian will be considered. Since the RMM is more general compared to SVM, by proposing a large relative margin spectrum learning, we encompass large margin spectrum learning as a special case.

## 1.1 Setup and Notation

In this paper we assume that a set of labeled examples  $(\mathbf{x}_i, y_i)_{i=1}^l$  and an unlabeled set  $(\mathbf{x}_i)_{i=l+1}^n$  are given such that  $\mathbf{x}_i \in \mathbb{R}^m$  and  $y_i \in \{\pm 1\}$ . We denote by  $\mathbf{y} \in \mathbb{R}^l$  the vector whose  $i^{\text{th}}$  entry is  $y_i$  and by  $\mathbf{Y} \in \mathbb{R}^{l \times l}$  a diagonal matrix such that  $\mathbf{Y}_{ii} = y_i$ . The primary aim is to obtain predictions on the unlabeled examples; we are thus in a so-called transductive setup. However, the unlabeled examples can be also be utilized in the learning process.

Assume we are given a graph with adjacency matrix  $\mathbf{W} \in \mathbb{R}^{n \times n}$  where the weight  $\mathbf{W}_{ij}$  denotes the edge weight between nodes  $i$  and  $j$  (corresponding to the examples  $\mathbf{x}_i$  and  $\mathbf{x}_j$ ). Define the graph Laplacian as  $\mathbf{L} = \mathbf{D} - \mathbf{W}$  where  $\mathbf{D}$  denotes a diagonal matrix whose  $i^{\text{th}}$  entry is given by the sum of the  $i^{\text{th}}$  row of  $\mathbf{W}$ . We assume that  $\mathbf{L} = \sum_{i=1}^n \theta_i \phi_i \phi_i^\top$  is the eigendecomposition of  $\mathbf{L}$ . It is assumed that the eigenvalues are already arranged such that  $\theta_i \leq \theta_{i+1}$  for all  $i$ . Further, we let  $\mathbf{V} \in \mathbb{R}^{n \times q}$  be the matrix whose  $i^{\text{th}}$  column is the  $(i+1)^{\text{th}}$  eigenvector (corresponding to the  $(i+1)^{\text{th}}$  smallest eigenvalue) of  $\mathbf{L}$ . Note that the first eigenvector (corresponding to the smallest eigenvalue) has been deliberately left out from this definition. Further,  $\mathbf{U} \in \mathbb{R}^{n \times q}$  is defined to be the matrix whose  $i^{\text{th}}$  column is the  $i^{\text{th}}$  eigenvector.  $\mathbf{v}_i$  ( $\mathbf{u}_i$ ) denotes the  $i^{\text{th}}$  column of  $\mathbf{V}^\top$  ( $\mathbf{U}^\top$ ). For any eigenvector (such as  $\phi$ ,  $\mathbf{u}$  or  $\mathbf{v}$ ), we use the horizontal overbar (such as  $\bar{\phi}$ ,  $\bar{\mathbf{u}}$  or  $\bar{\mathbf{v}}$ ) to denote the subvector containing only the first  $l$  elements of the eigenvector, in other words, only the entries that correspond to the labeled examples. We overload this notation for matrices as well; thus  $\bar{\mathbf{V}} \in \mathbb{R}^{l \times q}$  ( $\bar{\mathbf{U}} \in \mathbb{R}^{l \times q}$ ) denotes the first  $l$  rows of  $\mathbf{V}$  ( $\mathbf{U}$ ).  $\Delta$  is assumed to be a  $q \times q$  diagonal matrix whose diagonal elements denote scalar values  $\delta_i$  (i.e.,  $\Delta_{ii} = \delta_i$ ). Finally  $\mathbf{0}$  and  $\mathbf{1}$  denote vectors of all zeros and all ones; their dimensionality can be inferred from the context.

## 2 Learning from the Graph Laplacian

The graph Laplacian has been particularly popular in transductive learning. While we can hardly do justice to all the literature, this section summarizes some of the most relevant previous approaches.

*Spectral Graph Transducer.* The spectral graph transducer [4] is a transductive learning method based on a relaxation of the combinatorial graph-cut problem. It obtains predictions on labeled and unlabeled examples by solving for  $\mathbf{h} \in \mathbb{R}^n$  via the following problem:

$$\min_{\mathbf{h} \in \mathbb{R}^n} \frac{1}{2} \mathbf{h}^\top \mathbf{V} \mathbf{Q} \mathbf{V}^\top \mathbf{h} + C (\mathbf{h} - \boldsymbol{\tau})^\top \mathbf{P} (\mathbf{h} - \boldsymbol{\tau}) \quad \text{s.t. } \mathbf{h}^\top \mathbf{1} = 0, \quad \mathbf{h}^\top \mathbf{h} = n \quad (1)$$

where  $\mathbf{P}$  is a diagonal matrix<sup>2</sup> with  $\mathbf{P}_{ii} = \frac{1}{l_+}$  ( $\frac{1}{l_-}$ ) if the  $i^{\text{th}}$  example is positive (negative);  $\mathbf{P}_{ii} = 0$  for unlabeled examples (i.e., for  $l+1 \leq i \leq n$ ). Further,  $\mathbf{Q}$  is also a diagonal matrix. Typically, the diagonal element  $\mathbf{Q}_{ii}$  is set to  $i^2$  [4].  $\boldsymbol{\tau}$  is a vector in which the values corresponding to the positive (negative) examples are set to  $\sqrt{\frac{l_-}{l_+}}$  ( $\sqrt{\frac{l_+}{l_-}}$ ).

*Non-parametric transformations via kernel target alignment (KTA).* In [11], a successful approach to learning a kernel was proposed which involved transforming

<sup>1</sup> We clarify that  $\bar{\mathbf{V}}^\top$  ( $\bar{\mathbf{U}}^\top$ ) denotes the transpose of  $\bar{\mathbf{V}}$  ( $\bar{\mathbf{U}}$ ).

<sup>2</sup>  $l_+(l_-)$  is the number of positive (negative) labeled examples.

the spectrum of a Laplacian in a non-parametric way. The empirical alignment between two kernel matrices  $\mathbf{K}_1$  and  $\mathbf{K}_2$  is defined as [3]:

$$\hat{A}(\mathbf{K}_1, \mathbf{K}_2) := \frac{\langle \mathbf{K}_1, \mathbf{K}_2 \rangle_F}{\sqrt{\langle \mathbf{K}_1, \mathbf{K}_1 \rangle_F \langle \mathbf{K}_2, \mathbf{K}_2 \rangle_F}}.$$

When the target  $\mathbf{y}$  (the vector formed by concatenating  $y_i$ 's) is known, the ideal kernel matrix is  $\mathbf{y}\mathbf{y}^\top$  and a kernel matrix  $\mathbf{K}$  can be learned by maximizing the alignment  $\hat{A}(\mathbf{K}, \mathbf{y}\mathbf{y}^\top)$ . The kernel target alignment approach [11] learns a kernel via the following formulation:<sup>3</sup>

$$\begin{aligned} \max_{\Delta} \quad & \hat{A}(\bar{\mathbf{U}}\Delta\bar{\mathbf{U}}^\top, \mathbf{y}\mathbf{y}^\top) \\ \text{s.t.} \quad & \text{trace}(\mathbf{U}\Delta\mathbf{U}^\top) = 1 \quad \delta_i \geq \delta_{i+1} \quad \forall 2 \leq i \leq q-1, \quad \delta_q \geq 0, \quad \delta_1 \geq 0. \end{aligned} \quad (2)$$

The above optimization problem transforms the spectrum of the given graph Laplacian  $\mathbf{L}$  while maximizing the alignment score of the labeled part of the kernel matrix ( $\bar{\mathbf{U}}\Delta\bar{\mathbf{U}}^\top$ ) with the observed labels. The trace constraint on the overall kernel matrix ( $\mathbf{U}\Delta\mathbf{U}^\top$ ) is used merely to control the arbitrary scaling. The above formulation can be posed as a quadratically constrained quadratic program (QCQP) that can be solved efficiently [11]. The ordering on the  $\delta$ 's is in reverse order as that of the eigenvalues of  $\mathbf{L}$  which amounts to monotonically inverting the spectrum of the graph Laplacian  $\mathbf{L}$ . Only the first  $q$  eigenvectors are considered in the formulation above due to computational considerations.

The eigenvector  $\phi_1$  is made up of a constant element. Thus, it merely amounts to adding a constant to all the elements of the kernel matrix. Therefore, the weight on this vector (i.e.  $\delta_1$ ) is allowed to vary freely. Finally, note that the  $\phi$ 's are the eigenvectors of  $\mathbf{L}$  so the trace constraint on  $\mathbf{U}\Delta\mathbf{U}^\top$  merely corresponds to the constraint  $\sum_{i=1}^q \delta_i = 1$  since  $\mathbf{U}^\top\mathbf{U} = \mathbf{I}$ .

*Parametric transformations.* A number of methods have been proposed to obtain a kernel from the graph Laplacian. These methods essentially compute the Laplacian over labeled and unlabeled data and transform its spectrum with a particular mapping. More precisely, a kernel is built as  $\mathbf{K} = \sum_{i=1}^n r(\theta_i)\phi_i\phi_i^\top$  where  $r(\cdot)$  is a monotonically decreasing function. Thus, an eigenvector with a small eigenvalue will have a large weight in the kernel matrix. Several methods fall into this category. For example, the diffusion kernel [5] is obtained by the transformation  $r(\theta) = \exp(-\theta/\sigma^2)$  and the Gaussian field kernel [12] uses the transformation  $r(\theta) = \frac{1}{\sigma^2 + \theta}$ . In fact, kernel PCA [6] also performs a similar operation. In kPCA, we retain the top  $k$  eigenvectors of a kernel matrix. From an equivalence that exists between the kernel matrix and the graph Laplacian (shown in the next section), we can in fact conclude that kernel PCA features also fall under the same family of monotonic transformations. While these are very interesting transformations, [11] showed that KTA and learning based approaches are empirically superior to parametric transformations so we will not

<sup>3</sup> In fact, [11] proposes two formulations, we are considering the one that was shown to have superior performance (the so-called improved order method).

elaborate further on these approaches but rather focus on learning the spectrum of a graph Laplacian.

### 3 Why Learn the Laplacian Spectrum?

We start with an optimization problem which is closely related to the spectral graph transducer (II). The main difference is in the choice of the loss function. Consider the following optimization problem:

$$\min_{\mathbf{h} \in \mathbb{R}^n} \frac{1}{2} \mathbf{h}^\top \mathbf{V} \mathbf{Q} \mathbf{V}^\top \mathbf{h} + C \sum_{i=1}^l \max(0, 1 - y_i \mathbf{h}_i), \quad (3)$$

where  $\mathbf{Q}$  is assumed to be an invertible diagonal matrix to avoid degeneracies<sup>4</sup>. The values on the diagonal of  $\mathbf{Q}$  depend on the particular choice of the kernel. The above optimization problem is essentially learning the predictions on all the examples by minimizing the so-called hinge loss and the regularization defined by the eigenspace of the graph Laplacian. The choice of the above formulation is due to its relation to the large margin learning framework given by the following theorem.

**Theorem 1.** *The optimization problem (3) is equivalent to*

$$\min_{\mathbf{w}, b} \frac{1}{2} \mathbf{w}^\top \mathbf{w} + C \sum_{i=1}^l \max(0, 1 - y_i (\mathbf{w}^\top \mathbf{Q}^{-\frac{1}{2}} \mathbf{v}_i + b)). \quad (4)$$

*Proof.* The predictions on all the examples (without the bias term) for the optimization problem (4) are given by  $\mathbf{f} = \mathbf{V} \mathbf{Q}^{-\frac{1}{2}} \mathbf{w}$ . Therefore  $\mathbf{Q}^{\frac{1}{2}} \mathbf{V}^\top \mathbf{f} = \mathbf{Q}^{\frac{1}{2}} \mathbf{V}^\top \mathbf{V} \mathbf{Q}^{-\frac{1}{2}} \mathbf{w} = \mathbf{w}$  since  $\mathbf{V}^\top \mathbf{V} = \mathbf{I}$ . Substituting this expression for  $\mathbf{w}$  in (4), the optimization problem becomes,

$$\min_{\mathbf{f}, b} \frac{1}{2} \mathbf{f}^\top \mathbf{V} \mathbf{Q} \mathbf{V}^\top \mathbf{f} + C \sum_{i=1}^l \max(0, 1 - y_i (\mathbf{f}_i + b)).$$

Let  $\mathbf{h} = \mathbf{f} + b\mathbf{1}$  and consider the first term in the objective above,

$$\begin{aligned} & (\mathbf{h} - b\mathbf{1})^\top \mathbf{V} \mathbf{Q} \mathbf{V}^\top (\mathbf{h} - b\mathbf{1}) \\ &= \mathbf{h}^\top \mathbf{V} \mathbf{Q} \mathbf{V}^\top \mathbf{h} + 2\mathbf{h}^\top \mathbf{V} \mathbf{Q} \mathbf{V}^\top \mathbf{1} + \mathbf{1}^\top \mathbf{V} \mathbf{Q} \mathbf{V}^\top \mathbf{1} = \mathbf{h}^\top \mathbf{V} \mathbf{Q} \mathbf{V}^\top \mathbf{h}, \end{aligned}$$

where we have used the fact that  $\mathbf{V}^\top \mathbf{1} = \mathbf{0}$  since the eigenvectors in  $\mathbf{V}$  are orthogonal to  $\mathbf{1}$ . This is because  $\mathbf{1}$  is always an eigenvector of  $\mathbf{L}$  and other eigenvectors are orthogonal to it. Thus, the optimization problem (3) follows.  $\square$

<sup>4</sup> In practice,  $\mathbf{Q}$  can be non-invertible, but we consider an invertible  $\mathbf{Q}$  to elucidate the main point.

The above theorem<sup>5</sup> thus implies that learning predictions with Laplacian regularization in (3) is equivalent to learning in a large margin setting (4). It is easy to see that the implicit kernel for the learning algorithm (4) (over both labeled and unlabeled examples) is given by  $\mathbf{V}\mathbf{Q}^{-1}\mathbf{V}^\top$ . Thus, computing predictions on all examples with  $\mathbf{V}\mathbf{Q}\mathbf{V}^\top$  as the regularizer in (3) is equivalent to large margin learning with the kernel obtained by inverting the spectrum  $\mathbf{Q}$ . However, it is not clear why inverting the spectrum of a Laplacian is the right choice for a kernel. The parametric methods presented in the previous section construct this kernel by exploring specific parametric forms. On the other hand, the kernel target alignment approach constructs this kernel by maximizing alignment with labels while maintaining an ordering on the spectrum. The spectral graph transducer in Section 2 uses<sup>6</sup> the transformation  $i^2$  on the Laplacian for regularization. In this paper, we explore a family of transformations and allow the algorithm to choose the one that best conforms to a large (relative) margin criterion. Instead of relying on parametric forms or using a surrogate criteria, this paper presents approaches that jointly obtain a transformation and a large margin classifier.

### 4 Relative Margin Machines

Relative margin machines (RMM) [7] measure the margin relative to the data spread; this approach has yielded significant improvement over SVMs and has enjoyed theoretical guarantees as well. In its primal form, the RMM solves the following optimization problem<sup>7</sup>

$$\begin{aligned} \min_{\mathbf{w}, b, \xi} \quad & \frac{1}{2} \mathbf{w}^\top \mathbf{w} + C \sum_{i=1}^l \xi_i & (5) \\ \text{s.t.} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad |\mathbf{w}^\top \mathbf{x}_i + b| \leq B \quad \forall 1 \leq i \leq l. \end{aligned}$$

Note that when  $B = \infty$ , the above formulation gives back the support vector machine formulation. For values of  $B$  below a threshold, the RMM gives solutions that differ from SVM solutions. The dual of the above optimization problem can be shown to be:

$$\begin{aligned} \max_{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\eta}} \quad & -\frac{1}{2} \boldsymbol{\gamma}^\top \mathbf{X}^\top \mathbf{X} \boldsymbol{\gamma} + \boldsymbol{\alpha}^\top \mathbf{1} - B(\boldsymbol{\beta}^\top \mathbf{1} + \boldsymbol{\eta}^\top \mathbf{1}) & (6) \\ \text{s.t.} \quad & \boldsymbol{\alpha}^\top \mathbf{y} - \boldsymbol{\beta}^\top \mathbf{1} + \boldsymbol{\eta}^\top \mathbf{1} = 0, \quad \mathbf{0} \leq \boldsymbol{\alpha} \leq C\mathbf{1}, \quad \boldsymbol{\beta} \geq \mathbf{0}, \quad \boldsymbol{\eta} \geq \mathbf{0}. \end{aligned}$$

In the dual, we have defined  $\boldsymbol{\gamma} := \mathbf{Y}\boldsymbol{\alpha} - \boldsymbol{\beta} + \boldsymbol{\eta}$  for brevity. Note that  $\boldsymbol{\alpha} \in \mathbb{R}^l$ ,  $\boldsymbol{\beta} \in \mathbb{R}^l$  and  $\boldsymbol{\eta} \in \mathbb{R}^l$  are the Lagrange multipliers corresponding to the constraints in (5).

<sup>5</sup> Although we excluded  $\phi_1$  in the definition of  $\mathbf{V}$  in these derivation, typically we would include it in practice and allow the weight on it to vary freely as in the kernel target alignment approach. However, experiments show that the algorithms typically choose a negligible weight on this eigenvector.

<sup>6</sup> Strictly speaking, the spectral graph transducer has additional constraints and a different motivation.

<sup>7</sup> The constraint  $|\mathbf{w}^\top \mathbf{x}_i + b| \leq B$  is typically implemented as two linear constraints.

#### 4.1 RMM on Laplacian Eigenmaps

Based on the motivation from earlier sections, we consider the problem of jointly learning a classifier and weights on various eigenvectors in the RMM setup. We restrict the family of weights to be the same as that in (2) in the following problem:

$$\begin{aligned}
 \min_{\mathbf{w}, b, \xi, \Delta} \quad & \frac{1}{2} \mathbf{w}^\top \mathbf{w} + C \sum_{i=1}^l \xi_i & (7) \\
 \text{s.t.} \quad & y_i (\mathbf{w}^\top \Delta^{\frac{1}{2}} \mathbf{u}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 & \forall 1 \leq i \leq l \\
 & |\mathbf{w}^\top \Delta^{\frac{1}{2}} \mathbf{u}_i + b| \leq B & \forall 1 \leq i \leq l \\
 & \delta_i \geq \delta_{i+1} \quad \forall 2 \leq i \leq q-1, \quad \delta_1 \geq 0, \quad \delta_q \geq 0, \\
 & \text{trace}(\mathbf{U} \Delta \mathbf{U}^\top) = 1.
 \end{aligned}$$

By writing the dual of the above problem over  $\mathbf{w}$ ,  $b$  and  $\xi$ , we get:

$$\begin{aligned}
 \min_{\Delta} \quad & \max_{\alpha, \beta, \eta} -\frac{1}{2} \boldsymbol{\gamma}^\top \bar{\mathbf{U}} \Delta \bar{\mathbf{U}}^\top \boldsymbol{\gamma} + \boldsymbol{\alpha}^\top \mathbf{1} - B (\boldsymbol{\beta}^\top \mathbf{1} + \boldsymbol{\eta}^\top \mathbf{1}) & (8) \\
 \text{s.t.} \quad & \boldsymbol{\alpha}^\top \mathbf{y} - \boldsymbol{\beta}^\top \mathbf{1} + \boldsymbol{\eta}^\top \mathbf{1} = 0, \quad \mathbf{0} \leq \boldsymbol{\alpha} \leq C \mathbf{1}, \quad \boldsymbol{\beta} \geq \mathbf{0}, \quad \boldsymbol{\eta} \geq \mathbf{0}, \\
 & \delta_i \geq \delta_{i+1} \quad \forall 2 \leq i \leq q-1, \quad \delta_1 \geq 0, \quad \delta_q \geq 0, \quad \sum_{i=1}^q \delta_i = 1.
 \end{aligned}$$

where we exploited the fact that  $\text{trace}(\mathbf{U} \Delta \mathbf{U}^\top) = \sum_{i=1}^q \delta_i$ . Clearly, the above optimization problem, without the ordering constraints (*i.e.*,  $\delta_i \geq \delta_{i+1}$ ) is simply the multiple kernel learning<sup>8</sup> problem (using the RMM criterion instead of the standard SVM). A straightforward derivation—following the approach of [1]—results in the corresponding multiple kernel learning optimization. Even though the optimization problem (8) without the ordering on  $\delta$ 's is a more general problem, it may not produce smooth predictions over the entire graph. This is because, with a small number of labeled examples (*i.e.*, small  $l$ ), it is unlikely that multiple kernel learning will maintain the spectrum ordering unless it is explicitly enforced. In fact, this phenomenon can frequently be observed in our experiments where multiple kernel learning fails to maintain a meaningful ordering on the spectrum.

## 5 STORM and STOAM

This section poses the optimization problem (8) in a more canonical form to obtain practical large-margin (denoted by STOAM) and large-relative-margin (denoted by STORM) implementations. These implementations achieve globally optimal joint estimates of the kernel and the classifier of interest. First, the min

<sup>8</sup> In this paper, we restrict our attention to convex combination multiple kernel learning algorithms.



and the max in (8) can be interchanged since the objective is concave in  $\Delta$  and convex in  $\alpha, \beta$  and  $\eta$  and both are strictly feasible [2]<sup>9</sup>. Thus, we can write:

$$\begin{aligned} \max_{\alpha, \beta, \eta} \min_{\Delta} & -\frac{1}{2} \gamma^\top \sum_{i=1}^q \delta_i \bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^\top \gamma + \alpha^\top \mathbf{1} - B(\beta^\top \mathbf{1} + \eta^\top \mathbf{1}) \quad (9) \\ \text{s.t. } & \alpha^\top \mathbf{y} - \beta^\top \mathbf{1} + \eta^\top \mathbf{1} = 0, \\ & \mathbf{0} \leq \alpha \leq C\mathbf{1}, \beta \geq \mathbf{0}, \eta \geq \mathbf{0}, \\ & \delta_i \geq \delta_{i+1} \quad \forall 2 \leq i \leq q-1, \delta_1 \geq 0, \delta_q \geq 0, \sum_{i=1}^q \delta_i = 1. \end{aligned}$$

### 5.1 An Unsuccessful Attempt

We first discuss a naive attempt to simplify the optimization that is not fruitful. Consider the inner optimization over  $\Delta$  in the above optimization problem (9):

$$\begin{aligned} \min_{\Delta} & -\frac{1}{2} \sum_{i=1}^q \delta_i \gamma^\top \bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^\top \gamma \quad (10) \\ \text{s.t. } & \delta_i \geq \delta_{i+1} \quad \forall 2 \leq i \leq q-1, \delta_1 \geq 0, \delta_q \geq 0, \sum_{i=1}^q \delta_i = 1. \end{aligned}$$

**Lemma 1.** *The dual of the above formulation is:*

$$\max_{\tau, \lambda} -\tau \quad \text{s.t.} \quad \frac{1}{2} \gamma^\top \bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^\top \gamma = \lambda_{i-1} - \lambda_i + \tau, \lambda_i \geq 0 \quad \forall 1 \leq i \leq q.$$

where  $\lambda_0 = 0$  is a dummy variable.

*Proof.* Start by writing the Lagrangian of the optimization problem:

$$\mathcal{L} = -\frac{1}{2} \sum_{i=1}^q \delta_i \gamma^\top \bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^\top \gamma - \sum_{i=2}^{q-1} \lambda_i (\delta_i - \delta_{i+1}) - \lambda_q \delta_q - \lambda_1 \delta_1 + \tau (\sum_{i=1}^q \delta_i - 1),$$

where  $\lambda_i \geq 0$  and  $\tau$  are Lagrange multipliers. The dual follows after differentiating  $\mathcal{L}$  with respect to  $\delta_i$  and equating the resulting expression to zero.  $\square$

*Caveat.* While the above dual is independent of  $\delta$ 's, the constraints  $\frac{1}{2} \gamma^\top \bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^\top \gamma = \lambda_{i-1} - \lambda_i + \tau$  involve a quadratic term in an equality. It is not possible to simply leave out  $\lambda_i$  to make this constraint an inequality since the same  $\lambda_i$  occurs in two equations. This is non-convex in  $\gamma$  and is problematic since, after all, we eventually want an optimization problem that is jointly convex in  $\gamma$  and the other variables. Thus, a reformulation is necessary to pose relative margin kernel learning as a jointly convex optimization problem.

---

<sup>9</sup> It is trivial to construct such  $\alpha, \beta, \eta$  and  $\Delta$  when not all the labels are the same.

## 5.2 A Refined Approach

We proceed by instead considering the following optimization problem:

$$\begin{aligned} \min_{\Delta} & -\frac{1}{2} \sum_{i=1}^q \delta_i \gamma^\top \bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^\top \gamma \\ \text{s.t.} & \delta_i - \delta_{i+1} \geq \epsilon \quad \forall 2 \leq i \leq q-1, \quad \delta_1 \geq \epsilon, \quad \delta_q \geq \epsilon, \quad \sum_{i=1}^q \delta_i = 1 \end{aligned} \quad (11)$$

where we still maintain the ordering of the eigenvalues but require that they are separated by at least  $\epsilon$ . Note that  $\epsilon > 0$  is not like other typical machine learning algorithm parameters (such as the parameter  $C$  in SVMs), since it can be *arbitrarily* small. The only requirement here is that  $\epsilon$  remains positive. Thus, we are not really adding an extra parameter to the algorithm in posing it as a QCQP. The following theorem shows that a change of variables can be done in the above optimization problem so that its dual is in a particularly convenient form; *note, however, that directly deriving the dual of (11) fails to give the desired property and form.*

**Theorem 2.** *The dual of the optimization problem (11) is:*

$$\begin{aligned} \max_{\lambda \geq 0, \tau} & -\tau + \epsilon \sum_{i=1}^q \lambda_i \\ \text{s.t.} & \frac{1}{2} \gamma^\top \sum_{j=2}^i \bar{\mathbf{u}}_j \bar{\mathbf{u}}_j^\top \gamma = \tau(i-1) - \lambda_i \quad \forall 2 \leq i \leq q \\ & \frac{1}{2} \gamma^\top \bar{\mathbf{u}}_1 \bar{\mathbf{u}}_1^\top \gamma = \tau - \lambda_1. \end{aligned} \quad (12)$$

*Proof.* Start with the following change of variables:

$$\kappa_i := \begin{cases} \delta_1 & \text{for } i = 1, \\ \delta_i - \delta_{i+1} & \text{for } 2 \leq i \leq q-1, \\ \delta_q & \text{for } i = q. \end{cases}$$

This gives:

$$\delta_i = \begin{cases} \kappa_1 & \text{for } i = 1, \\ \sum_{j=i}^q \kappa_j & \text{for } 2 \leq i \leq q. \end{cases} \quad (13)$$

Thus, (11) can be stated as,

$$\begin{aligned} \min_{\kappa} & -\frac{1}{2} \sum_{i=2}^q \sum_{j=i}^q \kappa_j \gamma^\top \bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^\top \gamma + \kappa_1 \gamma^\top \bar{\mathbf{u}}_1 \bar{\mathbf{u}}_1^\top \gamma \\ \text{s.t.} & \kappa_i \geq \epsilon \quad \forall 1 \leq i \leq q, \quad \text{and} \quad \sum_{i=2}^q \sum_{j=i}^q \kappa_j + \kappa_1 = 1. \end{aligned} \quad (14)$$

Consider simplifying the following term within the above formulation:

$$\sum_{i=2}^q \sum_{j=i}^q \kappa_j \gamma^\top \bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^\top \gamma = \sum_{i=2}^q \kappa_i \sum_{j=2}^i \gamma^\top \bar{\mathbf{u}}_j \bar{\mathbf{u}}_j^\top \gamma \quad \text{and} \quad \sum_{i=2}^q \sum_{j=i}^q \kappa_j = \sum_{i=2}^q (i-1) \kappa_i.$$

It is now straightforward to write the Lagrangian to obtain the dual. □

Even though the above optimization appears to have non-convexity problems mentioned after Lemma 1, these can be avoided. This is facilitated by the following helpful property.

**Lemma 2.** *For  $\epsilon > 0$ , all the inequality constraints are active at the optimum of the following optimization problem:*

$$\begin{aligned} \max_{\lambda \geq \mathbf{0}, \tau} \quad & -\tau + \epsilon \sum_{i=1}^q \lambda_i & (15) \\ \text{s.t.} \quad & \frac{1}{2} \gamma^\top \sum_{j=2}^i \bar{\mathbf{u}}_j \bar{\mathbf{u}}_j^\top \gamma \leq \tau(i-1) - \lambda_i & \forall 2 \leq i \leq q \\ & \frac{1}{2} \gamma^\top \bar{\mathbf{u}}_1 \bar{\mathbf{u}}_1^\top \gamma \leq \tau - \lambda_1. \end{aligned}$$

*Proof.* Assume that  $\lambda^*$  is the optimum for the above problem and constraint  $i$  (corresponding to  $\lambda_i$ ) is not active. Then, clearly, the objective can be further maximized by increasing  $\lambda_i^*$ . This contradicts the fact that  $\lambda^*$  is the optimum. □

In fact, it is not hard to show that the Lagrange multipliers of the constraints in problem (15) are equal to the  $\kappa_i$ 's. Thus, replacing the inner optimization over  $\delta$ 's in (9), by (15), we get the following optimization problem, which we call STORM (Spectrum Transformations that Optimize the Relative Margin):

$$\begin{aligned} \max_{\alpha, \beta, \eta, \lambda, \tau} \quad & \alpha^\top \mathbf{1} - \tau + \epsilon \sum_{i=1}^q \lambda_i - B(\beta^\top \mathbf{1} + \eta^\top \mathbf{1}) & (16) \\ \text{s.t.} \quad & \frac{1}{2} (\mathbf{Y}\alpha - \beta + \eta)^\top \sum_{j=2}^i \bar{\mathbf{u}}_j \bar{\mathbf{u}}_j^\top (\mathbf{Y}\alpha - \beta + \eta) \leq (i-1)\tau - \lambda_i & \forall 2 \leq i \leq q \\ & \frac{1}{2} (\mathbf{Y}\alpha - \beta + \eta)^\top \bar{\mathbf{u}}_1 \bar{\mathbf{u}}_1^\top (\mathbf{Y}\alpha - \beta + \eta) \leq \tau - \lambda_1 \\ & \alpha^\top \mathbf{y} - \beta^\top \mathbf{1} + \eta^\top \mathbf{1} = 0, \quad \mathbf{0} \leq \alpha \leq C\mathbf{1}, \quad \beta \geq \mathbf{0}, \quad \eta \geq \mathbf{0}, \quad \lambda \geq \mathbf{0}. \end{aligned}$$

The above optimization problem has a linear objective with quadratic constraints. This equation now falls into the well-known family of quadratically constrained quadratic optimization (QCQP) problems whose solution is straightforward in practice. Thus, we have proposed a novel QCQP for large relative margin spectrum learning. Since the relative margin machine is strictly more general than the support vector machine, we obtain STOAM (Spectrum Transformations that Optimize the Absolute Margin) by simply setting  $B = \infty$ .

*Obtaining  $\delta$  values.* Interior point methods obtain both primal and dual solutions of an optimization problem simultaneously. We can use equation (13) to obtain the weight on each eigenvector to construct the kernel.

*Computational complexity.* STORM is a standard QCQP with  $q$  quadratic constraints of dimensionality  $l$ . This can be solved in time  $\mathcal{O}(ql^3)$  with an interior point solver. We point out that, typically, the number of labeled examples  $l$  is much smaller than the total number of examples (which is  $n$ ). Moreover,  $q$  is typically a fixed constant. Thus the runtime of the proposed QCQP compares favorably with the  $\mathcal{O}(n^3)$  time for the initial eigendecomposition of  $\mathbf{L}$  which is required for all the spectral methods described in this paper.

## 6 Experiments

To study the empirical performance of STORM and STOAM with respect to previous work, we performed experiments on both text and digit classification problems. Five binary classification problems were chosen from the 20-newsgroups text dataset (separating categories like baseball-hockey (b-h), pc-mac (p-m), religion-atheism (r-a), windows-xwindows (w-x), and politics.mideast-politics.misc (m-m)). Similarly, five different problems were considered from the MNIST dataset (separating digits 0-9, 1-2, 3-8, 4-7, and 5-6). One thousand randomly sampled examples were used for each task.

A mutual nearest neighbor graph was first constructed using five nearest neighbors and then the graph Laplacian was computed. The elements of the weight matrix  $\mathbf{W}$  were all binary. In the case of MNIST digits, raw pixel values (note that each feature was normalized to zero-mean and unit variance) were used as features. For digits, nearest neighbors were determined by Euclidean distance, whereas, for text, the cosine similarity and tf-idf was used. In the experiments, the number of eigenvalues  $q$  was set to 200. This was a uniform choice for all methods which would not yield any unfair advantages for one approach over any other. In the case of STORM and STOAM,  $\epsilon$  was set to a negligible value of  $10^{-6}$ .

The entire dataset was randomly divided into labeled and unlabeled examples. The number of labeled examples was varied in steps of 20; the rest of the examples served as the test examples (as well as the unlabeled examples in graph construction). We then ran KTA to obtain a kernel; the estimated kernel was then fed into an SVM (this was referred to as KTA-S in the Tables) as well as to an RMM (referred to as KTA-R). To get an idea of the extent to which the ordering constraints matter, we also ran multiple kernel learning optimization which are similar to STOAM and STORM but without any ordering constraints. We refer to the multiple kernel learning with the SVM objective as MKL-S and with the RMM objective as MKL-R. We also included the spectral graph transducer (SGT) and the approach of [9] (described in the Appendix) in the experiments. Predictions on all the unlabeled examples were obtained for all the methods. Error rates were evaluated on the unlabeled examples. Twenty such runs were done for various values of hyper-parameters (such as  $C, B$ ) for all the methods.

The values of the hyper-parameters that resulted in minimum average error rate over unlabeled examples were selected for all the approaches. Once the hyper-parameter values were fixed, the entire dataset was again divided into labeled and unlabeled examples. Training was then done but with fixed values of various hyper-parameters. Error rates on unlabeled examples were then obtained for all the methods over hundred runs of random splits of the dataset.

**Table 1.** Mean and std. deviation of percentage error rates on text datasets. In each row, the method with minimum error rate is shown in dark gray. All the other algorithms whose performance is not significantly different from the best (at 5% significance level by a paired t-test) are shown in light gray.

DATA	$l$	<a href="#">[9]</a>	MKL-S	MKL-R	SGT	KTA-S	KTA-R	STOAM	STORM
r-a	30	44.89 $\pm$ 5.2	37.14 $\pm$ 5.6	37.14 $\pm$ 5.6	19.46 $\pm$ 1.4	22.98 $\pm$ 4.8	22.99 $\pm$ 4.8	25.81 $\pm$ 6.1	25.81 $\pm$ 6.1
	50	42.18 $\pm$ 3.8	29.93 $\pm$ 5.1	30.01 $\pm$ 5.2	18.92 $\pm$ 1.1	19.87 $\pm$ 3.1	19.87 $\pm$ 3.1	21.49 $\pm$ 4.0	21.49 $\pm$ 4.0
	70	40.15 $\pm$ 2.5	25.18 $\pm$ 4.4	25.43 $\pm$ 4.3	18.44 $\pm$ 1.0	18.30 $\pm$ 2.4	18.30 $\pm$ 2.4	18.48 $\pm$ 3.1	18.48 $\pm$ 3.1
	90	38.86 $\pm$ 2.5	22.33 $\pm$ 3.3	22.67 $\pm$ 3.3	18.22 $\pm$ 0.9	17.32 $\pm$ 1.5	17.32 $\pm$ 1.5	17.21 $\pm$ 1.8	17.23 $\pm$ 1.9
	110	37.74 $\pm$ 2.3	20.43 $\pm$ 2.4	20.41 $\pm$ 2.4	18.10 $\pm$ 1.0	16.46 $\pm$ 1.3	16.46 $\pm$ 1.3	16.40 $\pm$ 1.2	16.41 $\pm$ 1.2
w-m	30	46.98 $\pm$ 2.4	22.74 $\pm$ 8.7	22.74 $\pm$ 8.7	41.88 $\pm$ 8.5	16.03 $\pm$ 8.8	16.08 $\pm$ 8.8	14.26 $\pm$ 5.9	14.26 $\pm$ 5.9
	50	45.47 $\pm$ 3.5	15.08 $\pm$ 3.8	15.08 $\pm$ 3.8	35.63 $\pm$ 9.3	13.54 $\pm$ 3.4	13.56 $\pm$ 3.4	11.49 $\pm$ 3.4	11.52 $\pm$ 3.4
	70	43.62 $\pm$ 4.0	13.03 $\pm$ 1.6	13.04 $\pm$ 1.6	29.03 $\pm$ 7.8	12.75 $\pm$ 4.8	12.89 $\pm$ 5.0	10.72 $\pm$ 0.9	10.76 $\pm$ 1.0
	90	42.85 $\pm$ 3.6	12.20 $\pm$ 1.6	12.20 $\pm$ 1.6	22.55 $\pm$ 6.3	11.30 $\pm$ 1.5	11.41 $\pm$ 1.7	10.43 $\pm$ 0.6	10.43 $\pm$ 0.6
	110	41.91 $\pm$ 3.8	11.84 $\pm$ 1.0	11.85 $\pm$ 1.0	18.16 $\pm$ 5.0	10.87 $\pm$ 1.4	10.99 $\pm$ 1.7	10.31 $\pm$ 0.6	10.28 $\pm$ 0.6
p-m	30	46.48 $\pm$ 2.7	41.21 $\pm$ 4.9	40.99 $\pm$ 5.0	39.58 $\pm$ 3.8	28.00 $\pm$ 5.8	28.05 $\pm$ 5.8	30.58 $\pm$ 6.6	30.58 $\pm$ 6.6
	50	44.08 $\pm$ 3.5	35.98 $\pm$ 5.3	35.94 $\pm$ 4.9	37.46 $\pm$ 3.8	24.34 $\pm$ 4.8	24.34 $\pm$ 4.8	25.72 $\pm$ 4.6	25.72 $\pm$ 4.6
	70	42.05 $\pm$ 3.5	31.48 $\pm$ 4.6	31.18 $\pm$ 4.3	35.52 $\pm$ 3.4	22.14 $\pm$ 3.6	22.14 $\pm$ 3.6	22.33 $\pm$ 4.9	22.33 $\pm$ 4.9
	90	39.54 $\pm$ 3.2	28.15 $\pm$ 3.8	28.30 $\pm$ 3.8	33.57 $\pm$ 3.4	20.58 $\pm$ 2.8	20.59 $\pm$ 2.7	20.44 $\pm$ 3.0	20.77 $\pm$ 3.2
	110	38.10 $\pm$ 3.2	25.88 $\pm$ 3.1	26.16 $\pm$ 2.9	32.16 $\pm$ 3.2	19.53 $\pm$ 2.2	19.56 $\pm$ 2.2	19.74 $\pm$ 2.4	19.70 $\pm$ 2.4
b-h	30	47.04 $\pm$ 2.1	4.35 $\pm$ 0.8	4.35 $\pm$ 0.8	3.95 $\pm$ 0.2	3.91 $\pm$ 0.4	3.80 $\pm$ 0.3	3.90 $\pm$ 0.3	3.87 $\pm$ 0.3
	50	46.11 $\pm$ 2.2	3.90 $\pm$ 0.1	3.91 $\pm$ 0.1	3.93 $\pm$ 0.2	3.81 $\pm$ 0.3	3.80 $\pm$ 0.4	3.87 $\pm$ 0.3	3.73 $\pm$ 0.3
	70	45.92 $\pm$ 2.4	3.91 $\pm$ 0.2	3.90 $\pm$ 0.2	3.90 $\pm$ 0.2	3.76 $\pm$ 0.3	3.76 $\pm$ 0.3	3.78 $\pm$ 0.3	3.68 $\pm$ 0.3
	90	45.30 $\pm$ 2.5	3.88 $\pm$ 0.2	3.89 $\pm$ 0.2	3.85 $\pm$ 0.3	3.69 $\pm$ 0.3	3.67 $\pm$ 0.3	3.75 $\pm$ 0.3	3.61 $\pm$ 0.3
	110	44.99 $\pm$ 2.6	3.88 $\pm$ 0.2	3.88 $\pm$ 0.2	3.83 $\pm$ 0.3	3.71 $\pm$ 0.4	3.66 $\pm$ 0.3	3.67 $\pm$ 0.3	3.56 $\pm$ 0.3
m-m	30	48.11 $\pm$ 4.7	12.35 $\pm$ 5.2	12.35 $\pm$ 5.2	41.30 $\pm$ 3.5	7.35 $\pm$ 3.6	7.36 $\pm$ 3.8	7.60 $\pm$ 3.9	6.88 $\pm$ 2.9
	50	46.36 $\pm$ 3.3	7.47 $\pm$ 3.1	7.25 $\pm$ 2.9	31.18 $\pm$ 7.5	6.25 $\pm$ 2.8	6.19 $\pm$ 2.9	5.45 $\pm$ 1.0	5.39 $\pm$ 1.2
	70	45.31 $\pm$ 5.7	6.05 $\pm$ 1.3	5.98 $\pm$ 1.4	22.30 $\pm$ 7.5	5.43 $\pm$ 1.0	5.35 $\pm$ 1.1	5.20 $\pm$ 0.7	4.90 $\pm$ 0.6
	90	42.52 $\pm$ 5.0	5.71 $\pm$ 1.0	5.68 $\pm$ 1.0	15.39 $\pm$ 5.9	5.13 $\pm$ 0.9	5.14 $\pm$ 1.1	5.09 $\pm$ 0.6	4.76 $\pm$ 0.6
	110	41.94 $\pm$ 5.2	5.44 $\pm$ 0.7	5.16 $\pm$ 0.6	10.96 $\pm$ 3.9	4.97 $\pm$ 0.8	4.92 $\pm$ 0.9	4.95 $\pm$ 0.5	4.65 $\pm$ 0.5

The results are presented in Table [1](#) and Table [2](#). It can be seen that STORM and STOAM perform much better than all the methods. Results in the two tables are further summarized in Table [3](#). It can be seen that both STORM and STOAM have significant advantages over all the other methods. Moreover, the formulation of [\[9\]](#) gives very poor results since the learned spectrum is independent of  $\alpha$ .

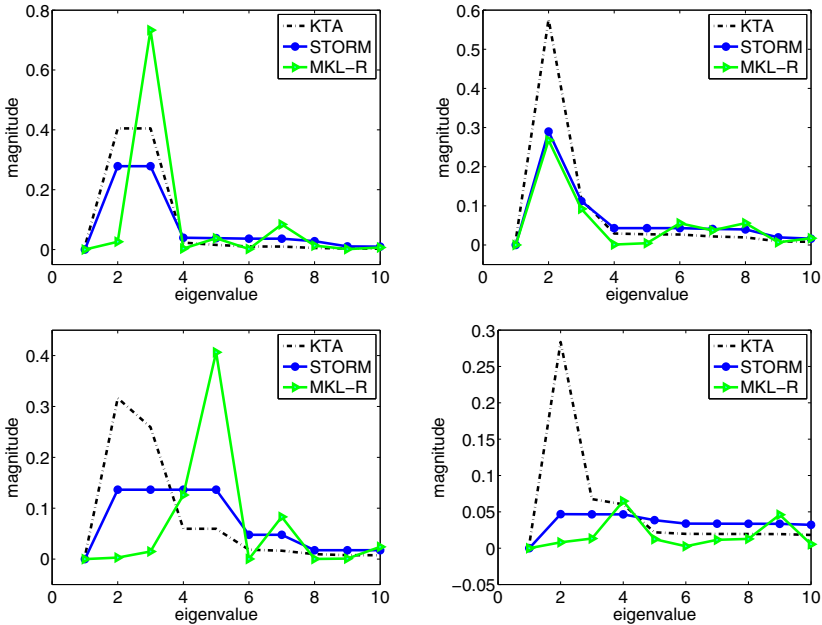
**Table 2.** Mean and std. deviation of percentage error rates on digits datasets. In each row, the method with minimum error rate is shown in dark gray. All the other algorithms whose performance is not significantly different from the best (at 5% significance level by a paired t-test) are shown in light gray.

DATA	$l$	<b>9</b>	MKL-S	MKL-R	SGT	KTA-S	KTA-R	STOAM	STORM
0-9	30	46.45 $\pm$ 1.5	0.89 $\pm$ 0.1	0.89 $\pm$ 0.1	0.83 $\pm$ 0.1	0.90 $\pm$ 0.1	0.90 $\pm$ 0.1	0.88 $\pm$ 0.1	0.88 $\pm$ 0.1
	50	45.83 $\pm$ 1.9	0.89 $\pm$ 0.1	0.90 $\pm$ 0.1	0.85 $\pm$ 0.1	0.91 $\pm$ 0.1	0.91 $\pm$ 0.1	0.89 $\pm$ 0.1	0.89 $\pm$ 0.1
	70	45.55 $\pm$ 2.0	0.88 $\pm$ 0.1	0.87 $\pm$ 0.1	0.87 $\pm$ 0.1	0.89 $\pm$ 0.1	0.93 $\pm$ 0.2	0.88 $\pm$ 0.1	0.88 $\pm$ 0.1
	90	45.68 $\pm$ 1.6	0.90 $\pm$ 0.1	0.85 $\pm$ 0.2	0.86 $\pm$ 0.1	0.91 $\pm$ 0.2	0.91 $\pm$ 0.2	0.87 $\pm$ 0.1	0.86 $\pm$ 0.1
	110	45.40 $\pm$ 2.0	0.85 $\pm$ 0.2	0.90 $\pm$ 0.2	0.87 $\pm$ 0.1	0.89 $\pm$ 0.1	0.89 $\pm$ 0.1	0.92 $\pm$ 0.3	0.86 $\pm$ 0.1
1-2	30	47.22 $\pm$ 2.0	3.39 $\pm$ 3.3	4.06 $\pm$ 5.9	11.81 $\pm$ 6.8	2.92 $\pm$ 0.6	2.92 $\pm$ 0.6	2.88 $\pm$ 0.5	2.85 $\pm$ 0.4
	50	46.02 $\pm$ 2.0	2.85 $\pm$ 0.5	2.58 $\pm$ 0.4	3.57 $\pm$ 2.7	2.78 $\pm$ 0.4	2.84 $\pm$ 0.5	2.80 $\pm$ 0.7	2.80 $\pm$ 0.7
	70	45.56 $\pm$ 2.4	2.64 $\pm$ 0.3	2.34 $\pm$ 0.3	2.72 $\pm$ 0.5	2.74 $\pm$ 0.3	2.76 $\pm$ 0.4	2.61 $\pm$ 0.3	2.70 $\pm$ 0.3
	90	45.00 $\pm$ 2.7	2.71 $\pm$ 0.3	2.35 $\pm$ 0.3	2.60 $\pm$ 0.2	2.76 $\pm$ 0.3	2.73 $\pm$ 0.4	2.70 $\pm$ 0.4	2.70 $\pm$ 0.3
	110	44.97 $\pm$ 2.3	2.77 $\pm$ 0.3	2.36 $\pm$ 0.3	2.61 $\pm$ 0.2	2.61 $\pm$ 0.6	2.61 $\pm$ 0.6	2.51 $\pm$ 0.3	2.51 $\pm$ 0.3
3-8	30	45.42 $\pm$ 3.0	13.02 $\pm$ 3.7	12.63 $\pm$ 3.6	9.86 $\pm$ 0.9	8.54 $\pm$ 2.7	7.58 $\pm$ 2.2	7.93 $\pm$ 2.2	7.68 $\pm$ 1.8
	50	43.72 $\pm$ 3.0	9.54 $\pm$ 2.3	9.04 $\pm$ 2.2	8.76 $\pm$ 0.9	6.93 $\pm$ 1.8	6.61 $\pm$ 1.6	6.42 $\pm$ 1.5	6.37 $\pm$ 1.4
	70	42.77 $\pm$ 3.1	7.98 $\pm$ 2.1	7.39 $\pm$ 1.7	8.00 $\pm$ 0.8	6.31 $\pm$ 1.6	6.07 $\pm$ 1.4	5.85 $\pm$ 1.3	5.85 $\pm$ 1.1
	90	41.28 $\pm$ 3.4	7.02 $\pm$ 1.6	6.60 $\pm$ 1.3	7.33 $\pm$ 0.8	5.69 $\pm$ 1.1	5.69 $\pm$ 1.1	5.45 $\pm$ 1.0	5.40 $\pm$ 0.9
	110	41.09 $\pm$ 3.5	6.56 $\pm$ 1.2	6.15 $\pm$ 1.0	6.91 $\pm$ 0.9	5.35 $\pm$ 0.9	5.43 $\pm$ 0.9	5.25 $\pm$ 0.8	5.24 $\pm$ 0.9
4-7	30	44.85 $\pm$ 3.5	5.74 $\pm$ 3.4	5.54 $\pm$ 3.3	5.60 $\pm$ 1.2	4.27 $\pm$ 1.9	4.09 $\pm$ 1.9	3.64 $\pm$ 1.4	3.57 $\pm$ 1.1
	50	43.65 $\pm$ 3.3	4.31 $\pm$ 1.2	3.97 $\pm$ 0.9	4.50 $\pm$ 0.5	3.50 $\pm$ 0.9	3.40 $\pm$ 0.8	3.24 $\pm$ 0.7	3.17 $\pm$ 0.6
	70	44.05 $\pm$ 3.3	3.66 $\pm$ 0.8	3.31 $\pm$ 0.6	4.04 $\pm$ 0.4	3.38 $\pm$ 0.8	3.23 $\pm$ 0.7	3.11 $\pm$ 0.6	3.04 $\pm$ 0.5
	90	42.04 $\pm$ 3.3	3.46 $\pm$ 0.8	3.13 $\pm$ 0.6	3.77 $\pm$ 0.4	3.12 $\pm$ 0.6	3.00 $\pm$ 0.6	2.92 $\pm$ 0.5	2.89 $\pm$ 0.5
	110	41.85 $\pm$ 3.1	3.28 $\pm$ 0.7	3.00 $\pm$ 0.5	3.60 $\pm$ 0.4	2.99 $\pm$ 0.6	2.98 $\pm$ 0.6	2.92 $\pm$ 0.5	2.91 $\pm$ 0.5
5-6	30	46.75 $\pm$ 2.6	5.18 $\pm$ 2.7	4.91 $\pm$ 3.2	2.49 $\pm$ 0.2	3.48 $\pm$ 1.3	3.32 $\pm$ 1.1	3.19 $\pm$ 1.4	2.96 $\pm$ 0.9
	50	45.98 $\pm$ 3.1	3.30 $\pm$ 1.3	2.93 $\pm$ 0.8	2.46 $\pm$ 0.2	2.94 $\pm$ 0.7	2.86 $\pm$ 0.5	2.73 $\pm$ 0.4	2.67 $\pm$ 0.4
	70	45.75 $\pm$ 3.5	2.80 $\pm$ 0.5	2.62 $\pm$ 0.3	2.49 $\pm$ 0.2	2.70 $\pm$ 0.4	2.65 $\pm$ 0.4	2.63 $\pm$ 0.3	2.83 $\pm$ 0.6
	90	45.19 $\pm$ 3.8	2.68 $\pm$ 0.3	2.60 $\pm$ 0.3	2.49 $\pm$ 0.2	2.62 $\pm$ 0.4	2.60 $\pm$ 0.4	2.60 $\pm$ 0.3	2.52 $\pm$ 0.4
	110	43.59 $\pm$ 2.8	2.62 $\pm$ 0.3	2.52 $\pm$ 0.3	2.51 $\pm$ 0.2	2.57 $\pm$ 0.4	2.53 $\pm$ 0.4	2.55 $\pm$ 0.4	2.49 $\pm$ 0.4

To gain further intuition, we visualized the learned spectrum in each problem to see if the algorithms yield significant differences in spectra. We present four typical plots in Figure 1. We show the spectra obtained by KTA, STORM and MKL-R (the difference between the spectra obtained by STOAM (MKL-S) was much closer to that obtained by STORM (MKL-R) compared to other methods). Typically KTA puts significantly more weight on the top few eigenvectors. By not maintaining the order among the eigenvectors, MKL seems to put haphazard weights on the eigenvectors. However, STORM is less aggressive and its eigenspectrum decays at a slower rate. This shows that STORM obtains a markedly different spectrum compared to KTA and MKL and is recovering a qualitatively different kernel. It is important to point out that MKL-R (MKL-S) solves a more general problem than STORM (STOAM). Thus, it can always achieve a better objective value compared to STORM (STOAM). However, this causes over-fitting and the experiments show that the error rate on the unlabeled

**Table 3.** Summary of results in Tables 1 & 2. For each method, we enumerate the number of times it performed best (dark gray), the number of times it was not significantly worse than the best performing method (light gray) and the total number of times it was either best or not significantly worse from best.

	9	MKL-S	MKL-R	SGT	KTA-S	KTA-R	STOAM	STORM
#dark gray	0	1	5	9	5	2	8	22
#light gray	0	1	2	4	8	12	16	13
#total	0	2	7	13	13	14	24	35



**Fig. 1.** Magnitudes of the top 15 eigenvalues recovered by the different algorithms. Top: problems 1-2 and 3-8. Bottom: m-m and p-m. The plots show average eigenspectra over all runs for each problem.

examples actually increases when the order of the spectrum is not preserved. In fact, MKL obtained competitive results in only one case (digits:1-2) which could be attributed to chance.

## 7 Conclusions

We proposed a large relative margin formulation for transforming the eigenspectrum of a graph Laplacian. A family of kernels was explored which maintains smoothness properties on the graph by enforcing an ordering on the eigenvalues of the kernel matrix. Unlike the previous methods which used two distinct criteria at each phase of the learning process, we demonstrated how jointly optimizing

the spectrum of a Laplacian while learning a classifier can result in improved performance. The resulting kernels, learned as part of the optimization, showed improvements on a variety of experiments. The formulation (3) shows that we can learn predictions as well as the spectrum of a Laplacian jointly by convex programming. This opens up an interesting direction for further investigation. By learning weights on an appropriate number of matrices, it is possible to explore all graph Laplacians. Thus, it seems possible to learn both a graph structure and a large (relative) margin solution jointly.

*Acknowledgments.* The authors acknowledge support from DHS Contract N66001-09-C-0080—“Privacy Preserving Sharing of Network Trace Data (PPSNTD) Program” and “NetTrailMix” Google Research Award.

## References

1. Bach, F.R., Lanckriet, G.R.G., Jordan, M.I.: Multiple kernel learning, conic duality, and the smo algorithm. In: International Conference on Machine Learning (2004)
2. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press, Cambridge (2003)
3. Cristianini, N., Shawe-Taylor, J., Elisseeff, A., Kandola, J.S.: On kernel-target alignment. In: NIPS, pp. 367–373 (2001)
4. Joachims, T.: Transductive learning via spectral graph partitioning. In: ICML, pp. 290–297 (2003)
5. Kondor, R.I., Lafferty, J.D.: Diffusion kernels on graphs and other discrete input spaces. In: ICML, pp. 315–322 (2002)
6. Schölkopf, B., Smola, A.J., Müller, K.-R.: Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation* 10, 1299–1319 (1998)
7. Shivaswamy, P., Jebara, T.: Maximum relative margin and data-dependent regularization. *Journal of Machine Learning Research* 11, 747–788 (2010)
8. Smola, A.J., Kondor, R.I.: Kernels and regularization on graphs. In: Schölkopf, B., Warmuth, M.K. (eds.) COLT/Kernel 2003. LNCS (LNAI), vol. 2777, pp. 144–158. Springer, Heidelberg (2003)
9. Xu, Z., Zhu, J., Lyu, M.R., King, I.: Maximum margin based semi-supervised spectral kernel learning. In: IJCNN, pp. 418–423 (2007)
10. Zhang, T., Ando, R.: Analysis of spectral kernel design based semi-supervised learning. In: NIPS, pp. 1601–1608 (2006)
11. Zhu, X., Kandola, J.S., Ghahramani, Z., Lafferty, J.D.: Nonparametric transforms of graph kernels for semi-supervised learning. In: NIPS (2004)
12. Zhu, X., Lafferty, J., Ghahramani, Z.: Semi-supervised learning: From gaussian fields to gaussian processes. Technical report, Carnegie Mellon University (2003)

## A Approach of Xu *et al.* [9]

It is important to note that, in a previously published article [9], other authors attempted to solve a problem related to STOAM. While this section is not the main focus of our paper, it is helpful to point out that the method in [9] is completely different from our formulation and contains serious flaws. The



previous approach attempted to learn a kernel of the form  $\mathbf{K} = \sum_{i=1}^q \delta_i \mathbf{u}_i \mathbf{u}_i^\top$  while maximizing the margin in the SVM dual. They start with the problem (Equation (13) in [9] but using our notation):

$$\begin{aligned} & \max_{\mathbf{0} \leq \boldsymbol{\alpha} \leq C\mathbf{1}, \boldsymbol{\alpha}^\top \mathbf{y} = 0} \boldsymbol{\alpha}^\top \mathbf{1} - \frac{1}{2} \boldsymbol{\alpha}^\top \mathbf{Y} \mathbf{K}^{\text{tr}} \mathbf{Y} \boldsymbol{\alpha} & (17) \\ \text{s.t. } & \delta_i \geq w \delta_{i+1} \quad \forall 1 \leq i \leq q-1, \quad \delta_i \geq 0, \quad \mathbf{K} = \sum_{i=1}^q \delta_i \mathbf{u}_i \mathbf{u}_i^\top, \quad \text{trace}(\mathbf{K}) = \mu \end{aligned}$$

which is the SVM dual with a particular choice of kernel. Here  $\mathbf{K}^{\text{tr}} = \sum_{i=1}^q \delta_i \bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^\top$ . It is assumed that  $\mu$ ,  $w$  and  $C$  are fixed parameters. The authors discuss optimizing the above problem while exploring  $\mathbf{K}$  by adjusting the  $\delta_i$  values. The authors then claim, without proof, that the following QCCQP (Equation (14) of [9]) can jointly optimize  $\delta$ 's while learning a classifier:

$$\begin{aligned} & \max_{\boldsymbol{\alpha}, \delta, \rho} 2\boldsymbol{\alpha}^\top \mathbf{1} - \mu\rho & (18) \\ \text{s.t. } & \mu = \sum_{i=1}^q \delta_i t_i, \quad \mathbf{0} \leq \boldsymbol{\alpha} \leq C\mathbf{1}, \quad \boldsymbol{\alpha}^\top \mathbf{y} = 0, \quad \delta_i \geq 0 \quad \forall 1 \leq i \leq q \\ & \frac{1}{t_i} \boldsymbol{\alpha}^\top \mathbf{Y} \bar{\mathbf{u}}_i \bar{\mathbf{u}}_i^\top \mathbf{Y} \boldsymbol{\alpha} \leq \rho \quad \forall 1 \leq i \leq q, \quad \delta_i \geq w \delta_{i+1} \quad \forall 1 \leq i \leq q-1 \end{aligned}$$

where  $t_i$  are *fixed* scalar values (whose values are irrelevant in this discussion). The only constraints on  $\delta$ 's are: non-negativity,  $\delta_i \geq w \delta_{i+1}$ , and  $\sum_{i=1}^q \delta_i t_i = \mu$  where  $w$  and  $\mu$  are fixed parameters. Clearly, in this problem,  $\delta$ 's can be set independently of  $\boldsymbol{\alpha}$ ! Further, since  $\mu$  is also a fixed constant,  $\delta$  no longer has any effect on the objective. Thus,  $\delta$ 's can be set without affecting either the objective or the other variables ( $\boldsymbol{\alpha}$  and  $\rho$ ). Therefore, the formulation ([18]) certainly does not maximize the margin while learning the spectrum. This conclusion is further supported by empirical evidence in our experiments. Throughout all the experiments, the optimization problem proposed by [9] produced extremely weak results.

# *k*-Version-Space Multi-class Classification Based on *k*-Consistency Tests

Evgueni Smirnov<sup>1</sup>, Georgi Nalbantov<sup>1</sup>, and Nikolay Nikolaev<sup>2</sup>

<sup>1</sup> Department of Knowledge Engineering, Maastricht University,  
P.O. BOX 616, 6200 MD Maastricht, The Netherlands

{smirnov, g.nalbantov}@maastrichtuniversity.nl

<sup>2</sup> Department of Computing, Goldsmiths College, University of London,  
London SE14 6NW, United Kingdom  
n.nikolaev@gold.ac.uk

**Abstract.** *k*-Version spaces were introduced in [6] to handle noisy data. They were defined as sets of *k*-consistent hypotheses; i.e., hypotheses consistent with all but *k* instances. Although *k*-version spaces were applied, their implementation was intractable due to the boundary-set representation.

This paper argues that to classify with *k*-version spaces we do not need an explicit representation. Instead we need to solve a general *k*-consistency problem and a general *k*0-consistency problem. The general *k*-consistency problem is to test the hypothesis space for classifier that is *k*-consistent with the data. The general *k*0-consistency problem is to test the hypothesis space for classifier that is *k*-consistent with the data and 0-consistent with a labeled test instance. Hence, our main result is that the *k*-version-space classification can be (tractably) implemented if we have (tractable) *k*-consistency-test algorithms and (tractable) *k*0-consistency-test algorithms. We show how to design these algorithms for any learning algorithm in multi-class classification setting.

**Keywords:** Classification, *k*-Version Spaces, Consistency Problems.

## 1 Introduction

Version spaces form a well-known approach to two-class classification [6–8]. Given a hypothesis space, version spaces are sets of hypotheses consistent with training data  $D$ . The consistency criterion implies that version spaces provide correct instance classifications if  $D$  is noise-free. Otherwise, version spaces can misclassify instances.

To tackle the problem with noisy data Mitchell introduced *k*-version spaces in [6]. *k*-Version spaces were defined as sets of all the hypotheses that are *k*-consistent with the training data  $D$ ; i.e., hypotheses that are consistent with all but *k* instances in  $D$ . The key idea is that if we have  $m$  class-mislabeled instances in  $D$  and  $m \leq k$ , we can still have a hypothesis in the *k*-version space that is inconsistent with these  $m$ -instances and consistent with most of the remaining data in  $D$ . Thus, the parameter *k* is capable of filtering the noise in the training data  $D$ .

Representing *k*-version spaces is a difficult problem. Any *k*-version space consists of  $\binom{|D|}{|D|-k}$  number of 0-version spaces that are not overlapped in the worst case. Thus, any

version-space representation employed (such as boundary sets [6–8], one-sided boundary sets [4], instance-based boundary sets [9, 10, 12] etc.) results in intractable representation of  $k$ -version spaces. This implies that the classification algorithm of  $k$ -version spaces is intractable as well in this case.

Following [5] this paper focuses on the classification side of the problem instead of the representational. We prove that the problem of  $k$ -version-space classification is a recursive problem that consists of the problem of  $k - 1$ -version-space classification, the general  $k$ -consistency problem, and the general  $k0$ -consistency problem. We define the general  $k$ -consistency problem as a problem of determining whether there exists classifier in the hypothesis space that is  $k$ -consistent with the data. We define the general  $k0$ -consistency problem as a problem of determining whether there exists classifier in the hypothesis space that is  $k$ -consistent with the data and 0-consistent with a labeled test instance. Thus, our main result is that the  $k$ -version-space classification can be (tractably) implemented as soon as we have a (tractable)  $k$ -consistency-test algorithm and a (tractable)  $k0$ -consistency-test algorithm for the hypothesis space. The consistency-test algorithms can be applied to any class. Thus, we allow  $k$ -version spaces to be applied for multi-class classification tasks. This contrasts with the original formulation of  $k$ -version spaces proven for two-class classification tasks only due to the Boolean nature of version-space representations [6–8].

The practical contribution of our paper is that we show how to design  $k$ -consistency-test algorithms and  $k0$ -consistency-test algorithms for any learning algorithm and its hypothesis space. We demonstrate how consistency-test algorithms can be used in logical and probabilistic settings. Hence, our work converts  $k$ -version spaces to a meta framework applicable for any learning algorithm.

The paper is organized as follows. Section 2 formalizes the classification task. The  $k$ -version spaces are introduced in Section 3. Subsections 3.1, 3.2, and 3.3 provide our generalization of the  $k$ -consistency criterion,  $k$ -version spaces, and their classification rule for multi-class classification. The classification function of  $k$ -version spaces together with the general  $k$ -consistency problem and the general  $k0$ -consistency problem are introduced in Subsection 3.4. Sections 4 and 5 show how to implement  $k$ -consistency-test and  $k0$ -consistency-test algorithms in logical and probabilistic settings. The experiments are given in Section 6. Finally, Section 7 concludes the paper.

## 2 Classification Task

Let  $X$  be a non-empty instance space and  $Y$  be a non-empty class set s.t.  $|Y| > 1$ . A labeled instance is defined as a tuple  $(x, y)$  where  $x \in X$  and  $y \in Y$ . Training data  $D$  is a multi-set of labeled instances.<sup>1</sup> Given data  $D$  and instance  $x \in X$  to be classified, the classification task is to assign a class  $y \in Y$  to  $x$ .

To assign a class  $y \in Y$  to an instance  $x \in X$  we need a scoring classifier  $h : X \rightarrow \mathcal{P}(\mathbb{R})$  from a hypothesis space  $H$ . The classifier  $h$  outputs for  $x$  a posterior distribution of scores  $\{s(y)\}_{y \in Y}$  over the classes in  $Y$ . The final class  $y \in Y$  for  $x$  is determined by a class function  $c : \mathcal{P}(\mathbb{R}) \rightarrow Y$ . The function  $c$  receives as argument the score distribution  $\{s(y)\}_{y \in Y}$  and then outputs a class  $y \in Y$  according to some rule (usually

<sup>1</sup> The multi-set notation employed in this paper is that from [14].

the class with highest score  $s(y)$ ). Hence, the function composition  $c(h(x))$  forms a discrete classifier that assigns the final class  $y \in Y$  to the instance  $x$  to be classified.

To identify the scoring classifier  $h$  we need a learning algorithm  $l : \mathcal{P}(X \times Y) \rightarrow \mathcal{P}(\mathbb{R})^X$ . Given training data  $D$ , the algorithm  $l$  searches the hypothesis space  $H$  and then outputs the scoring classifier  $h \in H$ . The goal is to find the classifier  $h$  s.t. the discrete classifier  $c(h)$  classifies correctly future unseen instances iid drawn from the same probability distribution from which  $D$  was drawn.

### 3 Multi-class $k$ -Version Spaces

Mitchell proposed  $k$ -version spaces in [6] for two-class classification in the presence of noisy training data. This section extends the  $k$ -version spaces for multi-class classification. We first introduce the  $k$ -consistency criterion. Then we define  $k$ -version spaces and show how they can be used for classification if we can implement two consistency tests.

#### 3.1 $k$ -Consistency Criterion

The multi-class  $k$ -consistency criterion generalizes the two-class  $k$ -consistency criterion [6]. Its definition and properties are given below.

**Definition 1.** Given hypothesis space  $H$ , class function  $c$ , data  $D$ , and integer  $k \leq |D|$ , scoring classifier  $h \in H$  is said to be  $k$ -consistent with  $D$ , denoted by  $cons_k(h, D)$ , iff:

$$(\exists D_k \in \mathcal{P}_k(D))(\forall (x, y) \in D_k)c(h(x)) = y,$$

where  $\mathcal{P}_k(D) = \{D_k \subseteq D \mid |D_k| = |D| - k\}$ .

The integer  $k$  determines the extent of consistency of classifier  $h$  with respect to  $D$ . The boundary cases for  $k$  are given in Corollary 1.

**Corollary 1.** Consider data  $D$  and integer  $k \leq |D|$ . Then:

- if  $k < 0$ ,  $(\forall h \in H) \neg cons_k(h, D)$ ;
- if  $k = 0$ ,  $(\forall h \in H)(cons_k(h, D) \leftrightarrow (\forall (x, y) \in D)c(h(x)) = y)$ ;
- if  $k = |D|$ ,  $(\forall h \in H)cons_k(h, D)$ .

The  $k$ -consistency has an important implication property formulated in Theorem 1.

**Theorem 1.** Consider scoring classifier  $h \in H$ , data  $D_1$  and  $D_2$  s.t.  $D_2 \subseteq D_1$ , and integers  $k_1$  and  $k_2$  s.t.  $k_1 \leq k_2 \leq |D_2|$ . Then:  $cons_{k_1}(h, D_1) \rightarrow cons_{k_2}(h, D_2)$ .

**Proof.** Consider arbitrary scoring classifier  $h \in H$  s.t.  $cons_{k_1}(h, D_1)$ . By Definition 1:

$$(\exists D_{k_1} \in \mathcal{P}_{k_1}(D_1))(\forall (x, y) \in D_{k_1})c(h(x)) = y. \tag{1}$$

$k_1 \leq k_2$  and  $D_2 \subseteq D_1$  imply  $(\forall D_{k_1} \in \mathcal{P}_{k_1}(D_1))(\exists D_{k_2} \in \mathcal{P}_{k_2}(D_2))D_{k_1} \supseteq D_{k_2}$ . Thus, formula (1) implies  $(\exists D_{k_2} \in \mathcal{P}_{k_2}(D_2))(\forall (x, y) \in D_{k_2})c(h(x)) = y$ . The latter by Definition 1 is equivalent to  $cons_{k_2}(h, D_2)$ .  $\square$

### 3.2 $k$ -Version Spaces: Definition and Properties

The multi-class  $k$ -version space for training data  $D$  is the set of all the classifiers in a hypothesis space  $H$  that are  $k$ -consistent with  $D$ .

**Definition 2.** Given data  $D$  and integer  $k \leq |D|$ , the  $k$ -version space  $VS_k(D)$  equals:

$$\{h \in H | \text{cons}_k(h, D)\}.$$

The key idea of  $k$ -version spaces is that if we have  $m$  class-mislabeled instances in the training data  $D$  and  $m \leq k$ , then we can still have a scoring classifier  $h$  in the  $k$ -version space that is inconsistent with these  $m$ -instances and consistent with most of the remaining data in  $D$ . Thus, the integer  $k$  is capable of filtering the noise in  $D$ . We consider three boundary cases for  $k$  formulated in Corollary 2

**Corollary 2.** Consider data multi-set  $D$  and integer  $k \leq |D|$ . Then:

- (1) if  $k < 0$ , then  $VS_k(D) = \emptyset$ ,
- (2) if  $k = 0$ , then  $VS_k(D) = \{h \in H | (\forall (x, y) \in D) c(h(x)) = y\}$ ,
- (3) if  $k = |D|$ , then  $VS_k(D) = H$ .

The implication property of the  $k$ -consistency from Theorem 1 entails a sub-set property of the  $k$ -version spaces formulated in Theorem 2

**Theorem 2.** Consider data  $D_1$  and  $D_2$  s.t.  $D_2 \subseteq D_1$ , and integers  $k_1$  and  $k_2$  s.t.  $k_1 \leq k_2$  and  $k_2 \leq |D_2|$ . Then:  $VS_{k_1}(D_1) \subseteq VS_{k_2}(D_2)$ .

**Proof.** The proof follows from Theorem 1 □

### 3.3 $k$ -Version-Space Classification

Given  $k$ -version space  $VS_k(D)$ , the  $k$ -version-space classification rule assigns a class set  $VS_k(D)(x) \subseteq Y$  to any instance  $x \in X$ . The class set  $VS_k(D)(x)$  includes any class  $y \in Y$  s.t. there exists a scoring classifier  $h \in VS_k(D)$  that is 0-consistent with  $\lambda(x, y)$ ; i.e., the discrete classifier  $c(h(x))$  assigns class  $y$  to  $x$ .

**Definition 3.** Given data  $D$ , integer  $k < |D|$ , and  $k$ -version space  $VS_k(D)$ , instance  $x \in X$  receives a  $k$ -class set  $VS_k(D)(x)$  equal to:

$$\{y \in Y | (\exists h \in VS_k(D)) \text{cons}_0(h, \lambda(x, y))\}.$$

Our  $k$ -version-space classification rule is more general than the Mitchell's one [6]. It provides a  $k$ -class set  $VS_k(D)(x)$  for any instance  $x \in X$  instead of just one class or no class. To test whether  $VS_k(D)(x)$  is empty or not we introduce Theorem 3. The Theorem states that  $VS_k(D)(x)$  is empty iff the  $k$ -version space  $VS_k(D)$  is empty.

**Theorem 3.** For any data  $D$  and instance  $x \in X$ :

$$VS_k(D)(x) = \emptyset \leftrightarrow VS_k(D) = \emptyset.$$

**Proof.** For any data  $D$  and instance  $x \in X$ :

$$VS_k(D)(x) = \emptyset \text{ iff [by Definition 3]}$$

$$(\forall y \in Y) \neg (\exists h \in VS_k(D)) \text{cons}_0(h, \{(x, y)\}) \text{ iff [by Definition 1]}$$

$$(\forall y \in Y) \neg (\exists h \in VS_k(D)) c(h(x)) = y \text{ iff}$$

$$(\forall y \in Y) (\forall h \in VS_k(D)) c(h(x)) \neq y \text{ iff [} c \text{ is a function from } \mathcal{P}(\mathbb{R}) \text{ to } Y]$$

$$VS_k(D) = \emptyset. \quad \square$$

The problem to classify an instance  $x \in X$  by a  $k$ -version space  $VS_k(D)$  according to Definition 3 is called *the  $k$ -version-space classification problem*. In the next Subsection 3.4 we propose one solution to this problem based on two consistency tests.

### 3.4 $k$ -Consistency Tests

Consider a  $k$ -version space  $VS_k(D)$  and an instance  $x \in X$  to be classified. By Theorem 3 if  $VS_k(D)$  is empty, then the  $k$ -class set  $VS_k(D)(x)$  is empty. If  $VS_k(D)$  is non-empty, then to classify  $x$  we need to compute the non-empty  $k$ -class set  $VS_k(D)(x)$ . Therefore, we divide the  $k$ -version-space classification problem into two sub-problems:

- (1)  $k$ -collapse problem: to decide whether the  $k$ -version space  $VS_k(D)$  is empty;
- (2)  $k$ -class-set problem: to compute the  $k$ -class set  $VS_k(D)(x)$  for the instance  $x$  if  $VS_k(D)$  is non-empty.

For the  $k$ -collapse problem we formulate Theorem 4. The Theorem introduces a test to decide whether the  $k$ -version space  $VS_k(D)$  is empty.

**Theorem 4.** Consider data  $D$  and integer  $k \leq |D|$ . Then:

$$VS_k(D) \neq \emptyset \leftrightarrow (VS_{k-1}(D) \neq \emptyset \vee (\exists h \in H)(\text{cons}_k(h, D) \wedge \neg \text{cons}_{k-1}(h, D))).$$

**Proof.** For any data  $D$  and integer  $k \leq |D|$ :

$$VS_k(D) \neq \emptyset \text{ iff [by Definition 2]}$$

$$(\exists h \in H) \text{cons}_k(h, D) \text{ iff}$$

$$(\exists h \in H)((\text{cons}_k(h, D) \wedge \text{cons}_{k-1}(h, D)) \vee (\text{cons}_k(h, D) \wedge \neg \text{cons}_{k-1}(h, D))) \text{ iff}$$

$$\text{[by Theorem 1 } \text{cons}_k(h, D) \leftarrow \text{cons}_{k-1}(h, D)]$$

$$(\exists h \in H)(\text{cons}_{k-1}(h, D) \vee (\text{cons}_k(h, D) \wedge \neg \text{cons}_{k-1}(h, D))) \text{ iff}$$

$$(\exists h \in H) \text{cons}_{k-1}(h, D) \vee (\exists h \in H)(\text{cons}_k(h, D) \wedge \neg \text{cons}_{k-1}(h, D)) \text{ iff}$$

$$\text{[by Definition 2]}$$

$$VS_{k-1}(D) \neq \emptyset \vee (\exists h \in H)(\text{cons}_k(h, D) \wedge \neg \text{cons}_{k-1}(h, D)). \quad \square$$

By Theorem 4 a  $k$ -version space  $VS_k(D)$  is non-empty iff the  $k - 1$ -version space  $VS_{k-1}(D)$  is non-empty or there exists a scoring classifier  $h \in H$  that is  $k$ -consistent and  $k - 1$ -inconsistent with data  $D$ . We note that the problem to decide whether the  $k - 1$ -version space  $VS_{k-1}(D)$  is empty is the  $k - 1$ -collapse problem. The problem to decide whether there exists a scoring classifier  $h \in H$  that is  $k$ -consistent and  $k - 1$ -inconsistent with  $D$  is a new problem that we call *exact  $k$ -consistency problem*. Thus,

the  $k$ -collapse problem is a recursive problem. By Corollary 1 the recursion is restricted below for  $k = -1$ , since  $\text{cons}_{-1}(h, D)$  is false.

Theorem 4 does not specify whether to solve first the  $k - 1$ -collapse problem or the exact  $k$ -consistency problem. However, by Lemma 1 if the test for the  $k - 1$ -collapse problem is negative (i.e.,  $VS_{k-1}(D) = \emptyset$ ), then the exact  $k$ -consistency problem is simplified to a problem to decide whether there exists a scoring classifier  $h \in H$  that is only  $k$ -consistent with  $D$ .

**Lemma 1.** Consider data  $D$  and integer  $k \leq |D|$ . If  $VS_{k-1}(D) = \emptyset$ , then:

$$(\exists h \in H)(\text{cons}_k(h, D) \wedge \neg \text{cons}_{k-1}(h, D)) \leftrightarrow (\exists h \in H)\text{cons}_k(h, D).$$

**Proof.** The  $(\rightarrow)$  part of the proof is obvious. Thus, we provide the  $(\leftarrow)$  part only. Consider data  $D$  and integer  $k \leq |D|$ . If  $VS_{k-1}(D) = \emptyset$ , then  $\neg(\exists h \in H)\text{cons}_{k-1}(h, D)$ ; i.e.,  $(\forall h \in H)\neg \text{cons}_{k-1}(h, D)$ . The latter and  $(\exists h \in H)\text{cons}_k(h, D)$  imply:

$$(\exists h \in H)(\text{cons}_k(h, D) \wedge \neg \text{cons}_{k-1}(h, D)). \quad \square$$

The problem to decide whether there exists a scoring classifier  $h \in H$  that is  $k$ -consistent with data  $D$  is a new problem that we call *general  $k$ -consistency problem*.

**Definition 4. (General  $k$ -Consistency Problem).** Given hypothesis space  $H$  and data  $D$ , the general  $k$ -consistency problem is to determine:  $(\exists h \in H)\text{cons}_k(h, D)$ .

By combining the results of Theorem 4 and Lemma 1 we determine the order of computation for the  $k$ -collapse problem. First we solve the  $k - 1$ -collapse problem. If  $VS_{k-1}(D) \neq \emptyset$ , then by Theorem 4  $VS_k(D) \neq \emptyset$ . If  $VS_{k-1}(D) = \emptyset$ , then we solve the general  $k$ -consistency problem since this problem by Lemma 1 is equivalent to the exact  $k$ -consistency problem. Thus, we conclude that the  $k$ -collapse problem is a recursive problem that consists of the  $k - 1$ -collapse problem and the general  $k$ -consistency problem in the proposed order of computations.

For the  $k$ -class-set problem we formulate Theorem 5. The Theorem introduces a test for any class  $y \in Y$  to determine whether  $y$  belongs to the  $k$ -class set  $VS_k(D)(x)$  assigned to instance  $x$  to be classified.

**Theorem 5.** For any data  $D$ , integer  $k \leq |D|$ , instance  $x \in X$ , and class  $y \in Y$ :

$$y \in VS_k(D)(x) \leftrightarrow (y \in VS_{k-1}(D)(x) \vee (\exists h \in H)(\text{cons}_k(h, D) \wedge \text{cons}_0(h, \lambda(x, y)) \wedge \neg \text{cons}_{k-1}(h, D))).$$

**Proof.** For any data  $D$ , integer  $k \leq |D|$ , instance  $x \in X$ , and class  $y \in Y$ :

$$\begin{aligned} y \in VS_k(D)(x) & \text{ iff [by Definition 3]} \\ (\exists h \in VS_k(D))\text{cons}_0(h, \lambda(x, y)) & \text{ iff [by Theorem 2 } VS_{k-1}(D) \subseteq VS_k(D)] \\ (\exists h \in VS_{k-1}(D))\text{cons}_0(h, \lambda(x, y)) & \vee \\ (\exists h \in VS_k(D) \setminus VS_{k-1}(D))\text{cons}_0(h, \lambda(x, y)) & \text{ iff [by Definitions 2 and 3]} \\ y \in VS_{k-1}(D)(x) \vee & \\ (\exists h \in H)(\text{cons}_k(h, D) \wedge \text{cons}_0(h, \lambda(x, y)) \wedge \neg \text{cons}_{k-1}(h, D)). & \quad \square \end{aligned}$$

By Theorem 5 a class  $y \in Y$  belongs to the  $k$ -class set  $VS_k(D)(x)$  iff  $y$  belongs to the  $k - 1$ -class set  $VS_{k-1}(D)(x)$ , or there exists a scoring classifier  $h \in H$  that is  $k$ -consistent with  $D$ , 0-consistent with  $\lambda(x, y)$ , and  $k - 1$ -inconsistent with  $D$ . We note that the problem to determine whether the class  $y$  belongs to the class set  $VS_{k-1}(D)(x)$  is the  $k - 1$ -class-set problem for the class  $y$ . The problem to decide whether there exists a scoring classifier  $h \in H$  that is  $k$ -consistent with  $D$ , 0-consistent with  $\lambda(x, y)$ , and  $k - 1$ -inconsistent with  $D$  is a new problem that we call *exact  $k0$ -consistency problem*. Thus, the  $k$ -class-set problem is a recursive problem. By Corollary 1 the recursion is restricted below for  $k = -1$ , since  $cons_{-1}(h, D)$  is false.

Theorem 5 does not specify whether we have first to solve the  $k - 1$ -class-set problem or the exact  $k0$ -consistency problem. However, by Lemma 2 if for some class  $y \in Y$  the result of the  $k - 1$ -class-set problem is negative (i.e.,  $y \notin VS_{k-1}(D)(x)$ ), the exact  $k0$ -consistency problem is simplified to a problem to decide whether there exists a classifier  $h \in H$  that is *only*  $k$ -consistent with  $D$  and 0-consistent with  $\lambda(x, y)$ .

**Lemma 2.** Consider data  $D$ , integer  $k \leq |D|$ , instance  $x \in X$ , and class  $y \in Y$ . If  $y \notin VS_{k-1}(D)(x)$ , then:

$$(\exists h \in H)(cons_k(h, D) \wedge cons_0(h, \lambda(x, y))) \wedge \neg cons_{k-1}(h, D) \leftrightarrow (\exists h \in H)(cons_k(h, D) \wedge cons_0(h, \lambda(x, y)))$$

**Proof.** The  $(\rightarrow)$  part of the proof is obvious. Hence, we provide the  $(\leftarrow)$  part only. Consider data  $D$ , integer  $k \leq |D|$ , instance  $x \in X$ , and class  $y \in Y$ . If  $y \notin VS_{k-1}(D)(x)$ , by Definition 3  $\neg(\exists h \in VS_{k-1}(D))cons_0(h, \lambda(x, y))$ . By Definition 2 the latter implies  $\neg(\exists h \in H)(cons_{k-1}(h, D) \wedge cons_0(h, \lambda(x, y)))$  which is equivalent to:

$$(\forall h \in H)(\neg cons_{k-1}(h, D) \vee \neg cons_0(h, \lambda(x, y)))$$

Thus,

$$\begin{aligned} & (\exists h \in H)(cons_k(h, D) \wedge cons_0(h, \lambda(x, y))) \text{ iff} \\ & (\exists h \in H)(cons_k(h, D) \wedge cons_0(h, \lambda(x, y)) \wedge \\ & \quad (\neg cons_{k-1}(h, D) \vee \neg cons_0(h, \lambda(x, y)))) \text{ iff} \\ & (\exists h \in H)(cons_k(h, D) \wedge cons_0(h, \lambda(x, y)) \wedge \neg cons_{k-1}(h, D)). \quad \square \end{aligned}$$

The problem to decide whether there exists a scoring classifier  $h \in H$  that is  $k$ -consistent with data  $D$  and 0-consistent with  $\lambda(x, y)$  is a new problem that we call *general  $k0$ -consistency problem*. Below we provide a definition of this problem.

**Definition 5. (General  $k0$ -Consistency Problem)** Given hypothesis space  $H$ , data  $D$ , integer  $k \leq |D|$ , instance  $x \in X$ , and class  $y \in Y$ , the general  $k0$ -consistency problem is to determine:  $(\exists h \in H)(cons_k(h, D) \wedge cons_0(h, \lambda(x, y)))$ .

By combining the results of Theorem 5 and Lemma 2 we determine the order of computation for the  $k$ -class-set problem for any  $k$ -version space  $VS_k(D)$ , instance  $x \in X$ , and class  $y \in Y$ . First we solve the  $k - 1$ -class-set problem. If  $y \in VS_{k-1}(D)(x)$ , then by Theorem 5  $y \in VS_k(D)(x)$ . If  $y \notin VS_{k-1}(D)(x)$ , then we solve the general  $k0$ -consistency problem since this problem by Lemma 2 is equivalent to the exact  $k0$ -consistency problem. Thus, we conclude that the  $k$ -class-set problem is a recursive



---

```

Function Classify:
Input: integer  $k$ , data  $D$ , instance  $x$ .
Output: class set  $VS_k(D)(x)$  assigned to  $x$ .
if  $k < 0$  then
    return  $\emptyset$ ;
 $Y_{k-1} := \text{Classify}(k - 1, D, x)$ ;
if  $Y_{k-1} = \emptyset$  then
    if  $\neg(\exists h \in H) \text{cons}_k(h, D)$  then
        return  $\emptyset$ ;
     $Y_k := Y_{k-1}$ ;
for each class  $y \in Y \setminus Y_{k-1}$  do
    if  $(\exists h \in H)(\text{cons}_k(h, D) \wedge \text{cons}_0(h, \lambda(x, y)))$  then
         $Y_k := Y_k \cup \{y\}$ ;
return  $Y_k$ .
    
```

---

**Fig. 1.** Classification function of  $k$ -version spaces based on the  $k$ -consistency tests

problem that consists of the  $k - 1$ -class-set problem and the general  $k0$ -consistency problem in the proposed order of computations.

So far we showed that the  $k$ -version-space classification problem consists of the  $k$ -collapse problem and the  $k$ -class-set problem. The  $k$ -collapse problem consists of the  $k - 1$ -collapse problem and the general  $k$ -consistency problem. The  $k$ -class-set problem consists of the  $k - 1$ -class-set problem and the general  $k0$ -consistency problem. Thus, since the  $k - 1$ -collapse problem and the  $k - 1$ -class-set problem form the  $k - 1$ -version-space classification problem, it follows that *the  $k$ -version-space classification problem is a recursive problem that consists of the  $k - 1$ -version-space classification problem, the general  $k$ -consistency problem, and the general  $k0$ -consistency problem.* This result implies that the  $k$ -version-space classification can be (tractably) implemented as soon as we can (tractably) test for  $k$ -consistency and  $k0$ -consistency in the given hypothesis space. The consistency tests can be applied to any class. Thus, we allow  $k$ -version spaces to be applied for multi-class classification tasks.

The classification function of  $k$ -version spaces based on the consistency tests is given in Figure 1. The function input includes data  $D$ , integer  $k$ , and instance  $x \in X$ . The output is the  $k$ -class set  $VS_k(D)(x)$  for  $x$  provided according to Definition 3.

The function is recursive. It first checks whether  $k < 0$ . If  $k < 0$ , then by Corollary 2 the  $k$ -version space  $VS_k(D)$  is empty. This implies by Theorem 3 that the  $k$ -class set  $VS_k(D)(x)$  is empty. Thus, the function returns empty set.

If  $k \geq 0$ , we note that the  $k$ -version-space classification problem includes the  $k - 1$ -version-space classification problem. Hence, the function calls itself recursively for  $k - 1$ . The result of the call is the set  $Y_{k-1}$  of classes assigned by the  $k - 1$ -version space  $VS_{k-1}(D)$  to the instance  $x$ . If the class set  $Y_{k-1}$  is empty, then by Theorem 3 the  $k - 1$ -version space  $VS_{k-1}(D)$  is empty. Thus, by Theorem 4 and Lemma 1 in order to decide whether the  $k$ -version space  $VS_k(D)$  is non-empty we solve the general  $k$ -consistency problem; i.e., we test whether there exists a scoring classifier in  $H$  that is

*k*-consistent with *D*. If the test is negative, by Definition 2 the *k*-version space  $VS_k(D)$  is empty and by Definition 3 the function returns  $\emptyset$ . If the test is positive, by Definition 2  $VS_k(D)$  is non-empty. Therefore, the function continues the classification process by initializing the class set  $Y_k$  (assigned to the instance  $x$  by  $VS_k(D)$ ). By Theorem 5  $Y_{k-1} \subseteq Y_k$ . Thus,  $Y_k$  is initialized equal to  $Y_{k-1}$ . Then the function tests whether the classes from  $Y \setminus Y_{k-1}$  can be added to  $Y_k$ . We note that for each of these classes Lemma 2 holds. Thus, by Theorem 5 for each class  $y \in Y \setminus Y_{k-1}$  we solve the general *k*0-consistency problem for the data *D* and  $\{(x, y)\}$ . This is done by testing whether there exists a scoring classifier  $h \in H$  that is *k*-consistent with *D* and 0-consistent with  $\{(x, y)\}$ . If so, then by Theorem 5 the class  $y$  is added to the set  $Y_k$ . Once all the classes in  $Y \setminus Y_{k-1}$  have been visited the class set  $Y_k$  is outputted.

Let  $T_k$  be the time complexity of the general *k*-consistency test and  $T_{k0}$  be the time complexity  $T_{k0}$  of the general *k*0-consistency test. Assuming that  $T_k < T_{k0}$  the worst-case time complexity of the classification function of *k*-version spaces equals:

$$O(T_k + k|Y|T_{k0}). \tag{2}$$

## 4 Consistency Algorithms

To implement the classification function of *k*-version spaces based on the consistency tests we need *k*-consistency-test algorithms and *k*0-consistency-test algorithms. Below we propose two approaches to implement these algorithms. The first one is for the case when there exists a 0-consistent learning algorithm  $l$  for the hypothesis space  $H$ . It allows designing consistency-test algorithms valid for the whole hypothesis space  $H$ . Hence, it is called hypothesis-unrestrictive approach. The second approach is for the case when there exists no 0-consistent learning algorithm  $l$  for the hypothesis space  $H$ . It allows designing consistency-test algorithms valid for a sub-space of the hypothesis space  $H$ . Hence, it is called hypothesis-restrictive approach.

### 4.1 Hypothesis-Unrestrictive Approach

The hypothesis-unrestrictive approach assumes that there exists a 0-consistent learning algorithm  $l$  for the hypothesis space  $H$ . Thus,  $H$  contains hypothesis that is 0-consistent with data *D* iff  $l$  succeeds; i.e.,  $l$  outputs for *D* some hypothesis (that by definition is consistent with *D*). This implies that the 0-consistency-test algorithm in this case is the 0-consistent learning algorithm  $l$  plus a success test. In the past (cf. [5]) 0-consistency-test algorithms were proposed for different hypothesis spaces such as 1-decision lists, monotone depth two formulas, halfspaces etc. They guarantee tractable 0-version-space classification, if they are tractable.

By Definition 1 if we can test for 0-consistency, we can test for *k*-consistency. Thus, given data *D* and integer *k*, we design a *k*-consistency-test algorithm as follows. We start with  $m = 0$  and then for each  $D_m \subseteq D$  with size  $|D| - m$  we apply a 0-consistency-test algorithm. If the 0-consistency-test algorithm identifies 0-consistency for at least one  $D_m$ , by Theorem 1 there is 0-consistency for some  $D_k \subseteq D_m$  and we return value “true”. Otherwise, we continue with the next  $D_m$  or increment  $m$  in the boundary of *k*. If this is not possible, we return value “false”. Thus, the worst-case time complexity

of the  $k$ -consistency-test algorithm is  $O(\binom{|D|}{|D|-k} T_l)$  where  $T_l$  is the time complexity of the learning algorithm  $l$  used in the 0-consistency-test algorithm.

The  $k0$ -consistency-test algorithms and their worst-case time complexity are analogous. Thus, we conclude that the  $k$ -consistency-test and  $k0$ -consistency-test algorithms based on 0-consistency learning algorithms are intractable in the worst case. Thus, according to formula (2) the  $k$ -version space classification is intractable in this case.

## 4.2 Hypothesis-Restrictive Approach

The hypothesis-restrictive approach assumes that the learning algorithm  $l$  provided is not a 0-consistent learning algorithm for the hypothesis space  $H$ . The approach restricts  $H$  s.t. we can implement the tests for the  $k$ -consistency and  $k0$ -consistency in the constrained space  $H(k, D) \subseteq H$  using the algorithm  $l$ . Below we define  $H(k, D)$  and condition s.t. the consistency tests can be implemented using the learning algorithm  $l$ .

The restricted hypothesis space  $H(k, D)$  is defined for the learning algorithm  $l$ , data  $D$ , and integer  $k$ . It is non-empty if the scoring classifier  $l(D) \in H$  is  $k$ -consistent with  $D$ . In this case  $H(k, D)$  consists of  $l(D)$  plus any scoring classifier  $l(D \uplus \lambda(x, y)) \in H$  for some instance  $(x, y) \in X \times Y$  that is  $k$ -consistent with the data  $D \uplus \lambda(x, y)$ .

**Definition 6.** Consider hypothesis space  $H$ , integer  $k \leq |D|$ , and data  $D$ . If scoring classifier  $l(D)$  is  $k$ -consistent with  $D$ , the hypothesis sub-space  $H(k, D) \subseteq H$  equals:

$$\{l(D)\} \cup \{l(D \uplus \lambda(x, y)) \in H \mid (x, y) \in X \times Y \wedge \text{cons}_k(l(D \uplus \lambda(x, y)), D \uplus \lambda(x, y))\}.$$

Otherwise,  $H(k, D) = \emptyset$ .

Any learning algorithm  $l$  can be used for consistency testing in the hypothesis sub-space  $H(k, D)$  if the instance property holds. This property often holds for stable classifiers like Naive Bayes [3].

**Definition 7. (Instance Property)** Learning algorithm  $l$  has the instance property iff for any data  $D$  and instance  $(x, y) \in X \times Y$  if there exists instance  $(x', y') \in X \times Y$  s.t. the classifier  $l(D \uplus \lambda(x', y'))$  is  $k$ -consistent with  $D \uplus \lambda(x, y)$  and 0-consistent with  $\lambda(x, y)$ , then the classifier  $l(D \uplus \lambda(x, y))$  is  $k$ -consistent with  $D \uplus \lambda(x, y)$  and 0-consistent with  $\lambda(x, y)$ .

Below we describe algorithm  $C_k$  for the  $k$ -consistency test and algorithm  $C_{k0}$  for  $k0$ -consistency test in the restricted hypothesis space  $H(k, D)$ . The algorithm  $C_{k0}$  employs a learning algorithm  $l$  under the assumption that the instance property holds.

**Algorithm  $C_k$  for  $k$ -consistency test:** The algorithm  $C_k$  tests whether there exists a scoring classifier  $h$  in  $H(k, D)$  that is  $k$ -consistent with data  $D$ . For that purpose it first builds the scoring classifier  $l(D)$ . Then, the algorithm tests if  $l(D)$  is  $k$ -consistent with  $D$ . If so, by Definition 6  $H(k, D)$  is non-empty and includes the desired classifier. Otherwise,  $H(k, D)$  is empty; i.e., it does not include the desired classifier.

**Algorithm  $C_{k0}$  for  $k0$ -consistency test:** given a labeled instance  $(x, y) \in X \times Y$ , the algorithm  $C_{k0}$  tests whether there exists a classifier  $h$  in  $H(k, D)$  that is  $k$ -consistent

with data  $D$  and 0-consistent with data  $\lambda(x, y)$ . For that purpose it first builds the scoring classifier  $l(D)$  and then tests whether  $l(D)$  is  $k$ -consistent with  $D$ . If  $l(D)$  is not  $k$ -consistent with  $D$  by Definition 6  $H(k, D)$  is empty and thus it does not include the desired classifier. Otherwise,  $H(k, D)$  is non-empty and the algorithm tests whether the scoring classifier  $l(D)$  is 0-consistent with  $\lambda(x, y)$ . If  $l(D)$  is 0-consistent with  $\lambda(x, y)$ , then  $H(k, D)$  includes the desired classifier. Otherwise, the algorithm makes a second attempt. It builds the scoring classifier  $l(D \uplus \lambda(x, y))$  and then tests whether  $l(D \uplus \lambda(x, y))$  is  $k$ -consistent with  $D \uplus \lambda(x, y)$  and 0-consistent with  $\lambda(x, y)$ . If both tests are positive, then by Definition 6  $l(D \uplus \lambda(x, y)) \in H(k, D)$ , and  $l(D \uplus \lambda(x, y))$  is  $k$ -consistent with  $D$  and 0-consistent with  $\lambda(x, y)$ ; i.e.,  $H(k, D)$  includes the desired classifier. If at least one of the tests is negative, then by Definition 7 it follows that there exists no instance  $(x', y') \in X \times Y$  s.t.  $l(D \uplus \lambda(x', y'))$  is consistent with  $D \uplus \lambda(x, y)$  and 0-consistent with  $\lambda(x, y)$ ; i.e., there exists no instance  $(x', y') \in X \times Y$  s.t.  $l(D \uplus \lambda(x', y'))$  is consistent with  $D$  and 0-consistent with  $\lambda(x, y)$ . Thus, by Definition 6  $H(k, D)$  does not include the desired classifier.

The correctness of the algorithm  $C_{k0}$  is proven in Theorem 6.

**Theorem 6.** *If the instance property holds, then for any data  $D \subseteq X \times Y$ , integer  $k \leq |D|$ , instance  $x \in X$ , and class  $y \in Y$  we have:*

$$\begin{aligned} (\exists h \in H(k, D))(cons_k(h, D) \wedge cons_0(h, \lambda(x, y))) \leftrightarrow \\ (cons_0(l(D), \lambda(x, y)) \vee \\ (cons_k(l(D \uplus \lambda(x, y)), D \uplus \lambda(x, y)) \wedge cons_0(l(D \uplus \lambda(x, y)), \lambda(x, y)))). \end{aligned}$$

**Proof.** ( $\rightarrow$ ) Consider arbitrary data  $D \subseteq X \times Y$ , integer  $k \leq |D|$ , instance  $x \in X$ , and class  $y \in Y$  so that:

$$(\exists h \in H(k, D))(cons_k(h, D) \wedge cons_0(h, \lambda(x, y))).$$

Thus, by Definition 6:

$$\begin{aligned} cons_0(l(D), \lambda(x, y)) \vee \\ (\exists(x', y') \in X \times Y)(cons_k(l(D \uplus \lambda(x', y')), D \uplus \lambda(x', y')) \wedge \\ cons_k(l(D \uplus \{(x', y')\}), D) \wedge \\ cons_0(l(D \uplus \lambda(x', y')), \lambda(x, y))). \end{aligned}$$

which implies:

$$\begin{aligned} cons_0(l(D), \lambda(x, y)) \vee \\ (\exists(x', y') \in X \times Y)(cons_k(l(D \uplus \{(x', y')\}), D \uplus \lambda(x, y)) \wedge \\ cons_0(l(D \uplus \lambda(x', y')), \lambda(x, y))). \end{aligned}$$

By Definition 7 the latter implies:

$$\begin{aligned} cons_0(l(D), \lambda(x, y)) \vee \\ (cons_k(l(D \uplus \lambda(x, y)), D \uplus \lambda(x, y)) \wedge cons_0(l(D \uplus \lambda(x, y)), \lambda(x, y))). \end{aligned}$$

( $\leftarrow$ ) This part of the Theorem follows from Definitions 6 and 7. □

The time complexity of the consistency-test algorithms  $C_k$  and  $C_{k0}$  is  $O(T_l + |D|T_c)$  where  $T_l$  is the time complexity of the learning algorithm  $l$  and  $T_c$  is the time complexity to classify with the scoring classifier  $l(D)$  (derived by  $l$ ). Thus, according to formula (2) the hypothesis-restrictive approach guarantees tractable  $k$ -version-space classification iff the learning algorithm  $l$  and corresponding scoring classifier  $l(D)$  are tractable.

We conclude this Section with a remark. The hypothesis-unrestrictive approach and hypothesis-restrictive approach together allow implementing  $k$  and  $k0$ -consistency-test algorithms independent on the type of learning algorithms. Thus, they together with the  $k$ -version-space classification function (from Figure 1) form a meta  $k$ -version-space framework that is applicable for any type of learning algorithms.

## 5 Implementing $k$ -Consistency

The key to success of our hypothesis restrictive approach is the problem of implementing  $k$ -consistency: how to decide whether a classifier is  $k$ -consistent with data. In this Section we consider two possible implementations: logical and probabilistic.

### 5.1 Logical $k$ -Consistency

The logical implementation of  $k$ -consistency follows Definition 1. To decide whether a classifier  $h$  is  $k$ -consistent with data  $D$  we first test  $h$  on  $D$  and determine a multi-set  $D_c \subseteq D$  of correctly classified instances from  $D$ . If  $|D_c| \geq |D| - k$ , then we output value “true”; otherwise, we output value “false”.

### 5.2 Probabilistic $k$ -Consistency

The probabilistic implementation of  $k$ -consistency is based on Definition 1 and the generalized binomial distribution [3]. Consider a scoring classifier  $h \in H$ . It outputs for any instance a distribution of scores  $\{s(y)\}_{y \in Y}$ . If the scores are normalized, we receive a distribution of estimated probabilities  $\{p(y)\}_{y \in Y}$ . If  $y_r \in Y$  is the known class of an instance  $x \in X$ , the experiment to assign class to  $x$  according to the distribution  $\{p(y)\}_{y \in Y}$  (provided by  $h(x)$ ) is a binary trial with probability of success  $p(y_r)$ . Thus, to classify all the instances from data  $D$  we receive a sequence of  $|D|$  independent binary trials, each with different probability of success  $p(y_r)$ . The probabilities  $p(|D| - k)$  that we have  $|D| - k$  successes in the sequence of  $|D|$  trials for  $k \in 0..|D|$  form generalized binomial distribution. The probability  $F(|D| - k, |D|)$  that we have  $|D| - k$  or more successes equals  $\sum_{i=0}^k p(|D| - i)$ . This probability is actually the probability that the scoring classifier  $h$  is  $k$ -consistent with  $D$ .

**Definition 8.** Given the generalized binomial distribution  $\{p(i)\}_{i \in 0..|D|}$  of scoring classifier  $h \in H$  for data  $D$  and a probability threshold  $p_t \in [0, 1]$ ,  $h$  is probabilistically  $k$ -consistent with  $D$ , denoted by  $p\text{-cons}_k(h, D)$ , iff  $F(|D| - k, |D|) > p_t$ .

---

<sup>2</sup> The generalized binomial distribution is a discrete probability distribution of the number of successes in a sequence of  $n$  independent binary experiments with *different* success probability.

Implementing the probabilistic  $k$ -consistency is as follows. First, we test the scoring classifier  $h$  on data  $D$  to compute the probabilities of success  $p(y_r)$ . Then, we derive the generalized binomial distribution  $\{p(i)\}_{i \in 0..|D|}$  and compute the probability  $F(|D| - k, |D|)$ . Finally, by Definition 8 we output the truth value of the probabilistic consistency predicate  $p-cons_k(h, D)$ .

If the probabilistic consistency predicate is employed in Definition 2 we receive the definition of  $k$ -version spaces based on the probabilistic  $k$ -consistency. We note two advantages of the probabilistic  $k$ -consistency over the logical one. First, it employs the information from the probabilities  $p(y_r)$  of the true classes  $y_r$  for all the instances in the data  $D$ . Second, it provides solution in the whole range of  $k$  from 0 to the size of  $D$ .

## 6 Experiments

This Section presents experiments with our meta  $k$ -version-space framework. We employed the  $k$ -version-space classification function from Figure 1. The base classifier used in the framework was the Naive-Bayes classifier (NB) [8]. The consistency algorithms and the hypothesis space were designed using the hypothesis-restrictive approach, since NB is not 0-consistent classifier. The  $k$ -consistency was implemented using probabilistic  $k$ -consistency predicate (see Definition 8), since NB is a probabilistic classifier. The resulting combination we call  $k$ -Naive-Bayes Version Spaces ( $k$ -NBVSs). Note that  $k$ -NBVSs have two parameters, parameter  $k$  of  $k$ -version spaces and parameter  $p_t$ , the probability threshold of the probabilistic  $k$ -consistency predicate.

We tested  $k$ -NBVSs in the context of the reliable-classification task [1, 11, 15]. The task is to derive a classifier that outputs only reliable instance classifications. This implies that instances with unreliable classifications remain unclassified. Hence, reliable classifiers are evaluated using two measures: coverage rate and accuracy rate. The coverage rate is the proportion of the instances that receive classifications while the accuracy rate is the accuracy on the classified instances.

The reliable-classification task is chosen since it is typical for version spaces [13]. We assume that  $k$ -NBVSs provide reliable classification for an instance if only one class is outputted for that instance, i.e., we have an unanimous-voting rule.

We compared  $k$ -NBVSs with the NB classifier, since they employ this classifier. NB for reliable classification uses a rejection technique: the class with the highest posterior probability is outputted for an instance if this probability is greater than a user-defined probability threshold [16]. NB with the rejection technique is denoted by  $NB_r$ .

The reliable classification experiments were performed on 14 UCI datasets [2] (see Table 1). The evaluation method was 10-fold cross validation. Table 2 reports the best results for  $k$ -NBVSs and NB obtained by a grid search for tuning the parameters of these classifiers<sup>3</sup>. More precisely, it shows the maximal coverage rates of each of the classifiers for which the accuracy rate of 1.0 is achieved. The maximal coverage rates show that  $k$ -NBVSs outperform the  $NB_r$  classifier on 13 out of 14 datasets, 6 times significantly (level 0.05).

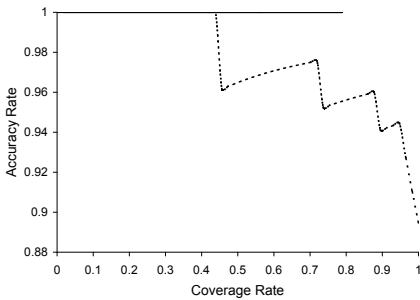
<sup>3</sup> Similar experiments with internal parameter tuning were performed. The results are compatible with those from Table 2.

**Table 1.** The UCI data sets employed in the experiments.  $A$  is the number of attributes,  $I$  is the number of instances, and  $C$  is the number of classes.

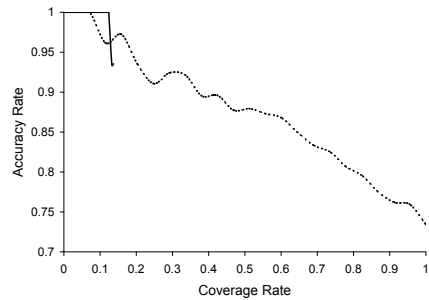
Data Set	$A$	$I$	$C$
audiology	70	226	24
breast-cancer	10	286	2
colic	23	386	2
diabetes	9	768	2
heart-statlog	14	270	2
hepatitis	20	155	2
ionosphere	35	351	2
iris	5	150	3
labor	17	57	2
lymphography	19	148	4
sonar	61	208	2
wisc. breast-cancer	10	699	2
wine	14	178	3
zoo	18	101	7

**Table 2.** Coverage rate for accuracy rate of 1.0:  $k$ -NBVS and  $NB_r$ . The numbers in bold present statistically better results on significance level 0.05.

Data Set	$k$ -NBVS	$NB_r$
audiology	0.1238	0.0708
breast-cancer	0.0734	0.0525
colic	<b>0.1250</b>	0.0055
diabetes	<b>0.0730</b>	-
heart-statlog	0.1222	0.0334
hepatitis	<b>0.2903</b>	0.0259
ionosphere	<b>0.2706</b>	0.1453
iris	0.8933	0.8333
labor	<b>0.7884</b>	0.4386
lymphography	0.1959	0.1825
sonar	0.0528	0.0048
wisc. breast-cancer	<b>0.3147</b>	0.0473
wine	0.9269	0.8821
zoo	0.9208	0.9208



**Fig. 2.** Coverage/accuracy graphs of  $k$ -NBVS (solid line) and  $NB_r$  (dashed line) for the labor data when the parameter  $k$  equals 0



**Fig. 3.** Coverage/accuracy graphs of  $k$ -NBVS (solid line) and  $NB_r$  (dashed line) for the audiology data when the parameter  $k$  equals 6

In addition we derived coverage/accuracy graphs for all the 14 datasets. One point in the graphs represents a  $k$ -NBVSs for some values of the parameters  $k$  and  $p_t$ . Subsequent points represent  $k$ -NBVSs for the same value of  $k$  and increased values of  $p_t$  in the range  $(p_t, 1.0]$ . Hence, the graphs show the potential of  $k$ -NBVSs for reliable classification. Due to page limit Figures 2 and 3 present the coverage/accuracy graphs for the labor data and audiology data only. The graphs of  $NB_r$  are derived analogously and are present in the Figures. The coverage/accuracy graphs show two features of  $k$ -NBVSs. First, when the unanimous-voting rule is used the coverage rate of  $k$ -NBVSs is never 1.0. This is due the fact that  $k$ -NBVSs is a set of scoring classifiers that disagree on some part of the instance space  $X$ . Second,  $k$ -NBVSs are sensitive for parameter  $k$ : the bigger  $k$  the bigger is the size of  $k$ -NBVSs. Thus, when the unanimous-voting rule is

used bigger  $k$  implies less classified instances; i.e., lower coverage rate. For example the parameter  $k$  for the labor data equals 0 and for the audiology data equals 6. Therefore the coverage rates of  $k$ -NBVSs for the labor data is bigger.

## 7 Conclusions

This paper has theoretical and practical contributions. The theoretical contributions are two. The first one is that we proved that the problem of  $k$ -version-space classification is a recursive problem that consists of the problem of  $k - 1$ -version-space classification, the general  $k$ -consistency problem, and the general  $k0$ -consistency problem. Thus, the  $k$ -version-space classification can be (tractably) implemented as soon as we can (tractably) test for  $k$ -consistency and  $k0$ -consistency. In this respect our work is a continuation of [5] showing that the 0-version-space classification problem is equivalent to the 0-consistency problem. The second theoretical contribution is that we extended  $k$ -version spaces to multi-class classification. This is due to the consistency tests that can be applied to any class. This contrasts with the original formulation of  $k$ -version spaces proven and used for two-class classification only due to the Boolean nature of version-space representations [6–8].

The practical contributions are two. The first one consists of two approaches to designing consistency-test algorithms for any type of learning algorithms. The most important is the hypothesis-restrictive approach applicable for nonzero-consistent learning algorithms. The second practical contribution is that we introduced two implementations of  $k$ -consistency. They allow logical and probabilistic implementations of consistency-test algorithms, and thus of the  $k$ -version-space classification.

The theoretical and practical contributions convert  $k$ -version spaces to a meta framework applicable for any type of learning algorithms. The framework is practical for nonzero-consistent learning algorithms. More precisely it guarantees tractable  $k$ -version-space classification iff these algorithms are tractable.

**Acknowledgements.** The authors would like to thank the anonymous referees for the useful and detailed reviews.

## References

1. Bosnic, Z., Kononenko, I.: An overview of advances in reliability estimation of individual predictions in machine learning. *Intell. Data Anal.* 13(2), 385–401 (2009)
2. Frank, A., Asuncion, A.: UCI machine learning repository (2010), <http://archive.ics.uci.edu/ml>
3. Hastie, T., Tibshirani, R., Friedman, J.: *Statistical Learning: Data Mining, Inference, and Prediction*. Springer, Heidelberg (2009)
4. Hirsh, H.: Polynomial-time learning with version spaces. In: *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-1992)*, pp. 117–122. AAAI Press, Menlo Park (1992)
5. Hirsh, H., Mishra, N., Pitt, L.: Version spaces and the consistency problem. *Artificial Intelligence* 156(2), 115–138 (2004)
6. Mitchell, T.: *Version Spaces: An Approach to Concept Learning*. Ph.D. thesis, Electrical Engineering Dept., Stanford University, Stanford, CA (1978)
7. Mitchell, T.: Generalization as search. *Artificial Intelligence* 18(2), 203–226 (1982)



8. Mitchell, T.: *Machine Learning*. McGraw-Hill, New York (1997)
9. Sebag, M.: Delaying the choice of bias: a disjunctive version space approach. In: *Proceedings of the Thirteenth International Conference on Machine Learning (ICML-1996)*, pp. 444–452. Morgan Kaufmann, San Francisco (1996)
10. Sebag, M., Rouveirol, C.: Any-time relational reasoning: Resource-bounded induction and deduction through stochastic matching. *Machine Learning* 38(1-2), 41–62 (2000)
11. Shafer, G., Vovk, V.: A tutorial on conformal prediction. *Journal of Machine Learning Research* 9(3), 371–421 (2008)
12. Smirnov, E., van den Herik, H., Sprinkhuizen-Kuyper, I.: A unifying version-space representation. *Ann. Math. Artif. Intell.* 41(1), 47–76 (2004)
13. Smirnov, E., Sprinkhuizen-Kuyper, I., Nalbantov, G., Vanderlooy, S.: S.Vanderlooy: Version space support vector machines. In: *Proceedings of the Seventeenth European Conference on Artificial Intelligence (ECAI-2006)*, pp. 809–810. IOS Press, Amsterdam (2006)
14. Syropoulos, A.: Mathematics of multisets. In: *Proceedings of the First International Workshop on Multiset Processing (MP-2000)*, pp. 347–358 (2000)
15. Tzikas, D., Kukar, M., Likas, A.: Transductive reliability estimation for kernel based classifiers. In: R. Berthold, M., Shawe-Taylor, J., Lavrač, N. (eds.) *IDA 2007*. LNCS, vol. 4723, pp. 37–47. Springer, Heidelberg (2007)
16. Vanderlooy, S., Sprinkhuizen-Kuyper, I., Smirnov, E., van den Herik, H.: The ROC isometrics approach to construct reliable classifiers. *Intell. Data Anal.* 13(1), 3–37 (2009)

# Complexity Bounds for Batch Active Learning in Classification

Philippe Rolet and Olivier Teytaud

TAO team (Inria, LRI, UMR 8623) (CNRS - Université Paris-Sud)  
bat. 490 Université Paris-Sud 91405 Orsay Cedex France  
`name@lri.fr`

**Abstract.** Active learning [1] is a branch of Machine Learning in which the learning algorithm, instead of being directly provided with pairs of problem instances and their solutions (their labels), is allowed to choose, from a set of unlabeled data, which instances to query. It is suited to settings where labeling instances is costly. This paper analyzes the speed-up of batch (parallel) active learning compared to sequential active learning (where instances are chosen 1 by 1): how faster can an algorithm become if it can query  $\lambda$  instances at once?

There are two main contributions: proving lower and upper bounds on the possible gain, and illustrating them by experimenting on usual active learning algorithms. Roughly speaking, the speed-up is asymptotically logarithmic in the batch size  $\lambda$  (i.e. when  $\lambda \rightarrow \infty$ ). However, for some classes of functions with finite VC-dimension  $V$ , a linear speed-up can be achieved until a batch size of  $V$ . Practically speaking, this means that parallelizing computations on an expensive-to-label problem which is suited to active learning is very beneficial until  $V$  simultaneous queries, and less interesting (yet still bringing improvement) afterwards.

## 1 Introduction

Active learning [1] (AL) is a statistical Machine Learning setting in which data comes unlabeled, the learning algorithm chooses which data points (i.e. instances) are important, and queries an *oracle* to get their labels. Active learning can be particularly useful if the oracle labelling the examples is expensive.

Batch active learning [12,22,13] is the particular case of active learning in which the algorithm can choose  $\lambda$  examples at a time, meaning the oracle provides  $\lambda$  answers at once— $\lambda$  is the *batch size*, that is the number of simultaneous requests to the oracle. This setting is for instance suited to cases where the oracle is a computational code (such as in numerical engineering applications): if  $\lambda$  computing units are available, the code can be run simultaneously on each machine.

This paper provides rigorous bounds on the number of iterations before a given precision is reached for batch active learning in binary classification, in particular as a function of  $\lambda$ . This model of complexity, based on the number of iterations only, is relevant for cases in which almost all the cost is in the calls to

the oracle function (*expensive* oracle), and when at least  $\lambda$  computation units are available. The internal cost of the learning algorithm is not taken into account.

The quantity of interest when analyzing batch active learning w.r.t. *sequential* active learning (i.e. when  $\lambda = 1$ ), is the *speed-up* at  $\lambda$ : it is the ratio of the number of iterations (number of calls to the oracle) of a sequential algorithm and the number of iterations of an algorithm using batches of examples of size  $\lambda$  at each iteration. Obviously, under this assumption, *passive* learning—when instances to be queried are selected i.i.d. from the natural input distribution—has a linear speed-up: an algorithm querying  $\lambda$  instances at time in an i.i.d. fashion is  $\lambda$  times faster than the sequential algorithm querying 1 instance at a time. We here investigate to which extent such a good speed-up can be recovered for active learning.

This work was motivated notably by the task of approximating a big numerical code, simulating some physical process or some engineering process, that require hours or days to compute an outcome given an input. The authors have been in contact with nuclear physicists that had such a code simulating nuclear fusion ignition through lasers, with a low input dimension (5 to 10).

The simulation code is not parallelized itself. However, it can of course be run simultaneously on multiple inputs if multiple processors are available, for instance on a single cluster (e.g. a hundred cores), or on a grid (e.g. five thousand cores). In the first case, outputs are given by batches of 100, and in the second by batches of 5000. However, the second requires much more deployment effort. This study shows that in such a case, if a good active learning strategy is available, it is not worth wasting time and computational power on using the grid, since the speed-up as compared to using the cluster would be quite low. This is naturally specific to active learning, and would not be true for passive learning scenarios.

The paper is organized as follows:

- Section 2 presents the framework and notations, so that complexity bounds can be properly formalized;
- Section 3 shows bounds for batch active learning using covering and packing numbers. Results include lower and upper bounds on the speed-up of batch active learning, seen as a parallel algorithm;
- Section 4 presents some experiments; these experiments are aimed at comparing predicted speed-ups (for optimal algorithms) to speed-ups of simple or usual algorithms;
- Section 5 concludes.

## State of the Art

The query learning models introduced by [1] can be viewed as the first attempts of the learning algorithm to directly interact with the oracle. Another early work [15] establishes a lower bound for the instance size in any active learning (AL) classification setting, logarithmic in the  $\epsilon$ -packing number of the hypothesis space  $\mathcal{F}$ —the number of disjoint  $\epsilon$ -radius balls needed that can be put in  $\mathcal{F}$  (see section 2 below).

**Classification.** [4] devised a heuristic for error-free learning (i.e., in the realizable setting) of a binary classifier (i.e. a  $\{0, 1\}$ -valued target function to learn), considering a large pool of unlabeled examples and selecting the best example to be labeled in each time step (pool-based adaptive sampling). Considering the set of hypotheses compatible with the available examples—the *version space* (VS) defined in [17]—the selected examples were meant to prune the version space.

[10] analyzed another algorithm based on a Bayesian prior on the hypothesis space called *Query-by-committee* (QBC) from [21]; it directly reduces the VS volume. A related research direction focuses on *error reduction*, meant as the expected generalization error improvement brought by an instance. Many criteria reflecting various measures of the expected error reduction have been proposed [5,14,18,16,8], with sometimes encouraging results in, for instance, pharmaceutical industry [25]. Specific algorithms and methods have been developed for active learning in linear and kernel spaces, either heuristically [20] or with theoretical foundations [3,8,2].

On the theoretical side, [10] related the efficiency of QBC to a statistical criterion called *Information Gain*, measuring how efficiently the VS can be divided. [6] has shown that with a Bayesian prior, greedily choosing examples that most evenly divide the VS is an almost optimal AL strategy.

Dasgupta also studied the non-Bayesian setting [7], deriving upper and lower complexity bounds based on a criterion called *splitting index*.

**Batch active learning for classification.** Batch active learning has received less attention. [12] assesses the information brought by batches of examples via a criterion based on Fisher information matrix reduction. [11] seeks sets of examples with low uncertainty; they phrase this as an optimization problem (NP-hard), and devise a method to find an acceptable approximation of the solution. Both works provide empirical evidence of the soundness of their strategies. However, they do not provide any formal proof guaranteeing their behavior. Further, we are not aware of any theoretical study of the speed-up of batch Active Learning over sequential Active Learning, in terms of sample complexity bounds (speed-up is in terms of gain with respect to the number of iterations, see section 2 below). Clearly, batch active learning can not reduce the overall number of evaluations when compared to sequential active learning; the advantage is only in the case of a parallel use of the oracle querying  $\lambda$  instances at a time.

## 2 Framework

In all the paper,  $\log$  refers to the binary logarithm. If  $a, b \in \mathbb{N}$ ,  $[[a, b]] = \mathbb{N} \cap [a, b]$ . If  $x, y \in [0, \infty[^d$ , then we note  $[x, y] = \{a \in \mathbb{R}^d; \forall i, x_i \leq a_i \leq y_i\}$ . The *speedup* of a parallel algorithm  $\mathcal{A}_\lambda$  over its sequential counterpart  $\mathcal{A}$  (or equivalently  $\mathcal{A}_1$ ) is the ratio of  $\mathcal{A}$ 's complexity in terms of number of iterations (i.e. of batch calls to the oracle) on  $\mathcal{A}_\lambda$ 's complexity.

Only deterministic algorithms are considered here; the lower bounds can be extended, nonetheless, to stochastic cases within logarithmic dependencies on

the risk  $\delta$  (i.e. case with confidence  $1 - \delta$ )<sup>1</sup> and upper bounds (Theorem 2, referred to as a *simulation* result) can also be extended to the stochastic case. The framework of batch active learning is presented in Algorithm 1. A batch active learning algorithm  $\mathcal{A}_\lambda$  is defined by the triplet  $(learn_\lambda, generate_\lambda, update_\lambda)$ . Let  $D$  be a domain,  $P_D$  a probability measure on this domain, and  $f^* : D \rightarrow \{0, 1\}$  be the unknown oracle, supposed to be deterministic and to belong to some set  $\mathcal{F} \subset \{0, 1\}^D$ . The *generalization error*  $\mathbf{d}(f, f^*)$  of an approximation  $f \in \mathcal{F}$  of  $f^*$  is defined as  $P_D(\{x | f(x) \neq f^*(x)\})$ . Note that  $\mathbf{d}$  is a distance for the space  $\mathcal{F}$ .<sup>2</sup>

We assume that the considered concept class  $\mathcal{F}$  has a finite VC-dimension  $V$ . VC-dimension is a classical measure of complexity for classes of functions, for which the reader is referred to [23,9]. It is common to consider finite VC-dimension in active learning settings since the improvement over passive learning is potentially much bigger in this case [15]. Indeed, the number of examples required to learn  $f^*$  with precision  $\epsilon$ , i.e. to find  $f$  such that  $\mathbf{d}(f^*, f) \leq \epsilon$ , is  $N = \Theta(V \log(1/\epsilon))$  in good cases, see for instance [7,10].

---

**Algorithm 1.** Batch active learning algorithm  $\mathcal{A}_\lambda = (learn_\lambda, generate_\lambda, update_\lambda)$ .  $\lambda$  is the number of visited points per iteration.

---

```

 $I_0 =$  initial state // Global state of the algorithm
 $n \leftarrow 0$ 
while true do
     $f_n \leftarrow learn_\lambda(I_n)$ 
     $(x_{n\lambda+1}, \dots, x_{(n+1)\lambda}) = generate_\lambda(I_n)$ 
    for  $i \in [1, \lambda]$  do
         $y_{n\lambda+i} = f(x_{n\lambda+i})$  // label  $\lambda$  instances at once
    end for
     $I_{n+1} \leftarrow update_\lambda(I_n, x_{n\lambda+1}, \dots, x_{(n+1)\lambda}, y_{n\lambda}, \dots, y_{(n+1)\lambda})$ 
     $n \leftarrow n + 1$ 
end while

```

---

For a given  $\lambda$  and a given algorithm  $\mathcal{A}$ , the number of iterations for reaching precision  $\epsilon$  is noted

$$N_\lambda^{\mathcal{A}}(\epsilon) = \sup_{f \in \mathcal{F}} \min\{n; \|f_n[\mathcal{A}_\lambda] - f\|_1 \leq \epsilon\}$$

with  $\|\cdot\|_1$  the  $L_1$  norm and  $f_n[\mathcal{A}_\lambda]$  the approximation learned after  $n$  iterations via the framework presented in Algorithm 1. The smallest achievable number of iterations for any algorithm will be noted

$$L_\lambda(\epsilon) = \inf_{\mathcal{A}} N_\lambda^{\mathcal{A}}(\epsilon).$$

$L_\lambda$  depends on the considered function class  $\mathcal{F}$  (so does  $N_\lambda^{\mathcal{A}}$ ), and will be noted  $L_\lambda^{\mathcal{F}}$  when it is needed to make explicit which  $\mathcal{F}$  is under study. Let  $pack_{\mathcal{F}}(\epsilon)$  be

<sup>1</sup> Precisely, the sample complexity is increased by a factor  $-O(\log(\delta))$  if we request that the algorithm finds the solution with probability  $1 - \delta$ .

<sup>2</sup> More precisely, a pseudo-metric.

the maximum number of points in  $\mathcal{F}$  with pairwise distance  $\mathbf{d}$  at least  $2\epsilon$  for the  $L_1$  norm. In the sequel, for some of our results, the following equation will be assumed:

$$\forall \epsilon, \text{pack}_{\mathcal{F}}(\epsilon) \geq (M/\epsilon)^{C \times V} \tag{1}$$

for constants  $C$  and  $M$ . It states that the log-packing numbers of target function class  $\mathcal{F}$  are at least  $-CV \log(\epsilon/M)$ . The product  $C \times V$  in Eq. 1 stems from many results emphasizing some constant  $C$ , and the VC-dimension  $V$  [7], such as, for example, the well-known case of homogeneous linear separators [3] of the sphere with homogeneous distribution [10,8,2], in which case  $V$  is equal to the dimension of the domain.

### 3 Covering Numbers and Batch Active Learning

Eq. 1 has the following consequence (see [15,24]):

$$L_1(\epsilon) \geq \lceil CV \log(M/\epsilon) \rceil. \tag{2}$$

Eq. 2 states a lower bound on the sample complexity of AL, and has the following consequence (which is a lower bound on the sample complexity of batch AL):

$$L_\lambda(\epsilon) \geq \lceil CV \log(M/\epsilon)/\lambda \rceil \tag{3}$$

Eq. 3 is the ultimate limit for batch active learning: it is the case of a linear speed-up. The following explores the extent to which it can be reached, w.r.t.  $\lambda$ . In a parallel setting, in which  $\lambda$  calls to the oracle are performed in parallel, Eq. 3 refers to a linear speed-up for the parallel (i.e. batch) form of active learning.

The first contribution of this work is the following extension of the classical bound 2:

**Theorem 1 (Lower bound for batch AL).** *If  $\mathcal{F}$  has packing number*

$$\text{pack}_{\mathcal{F}}(\epsilon) \geq (M/\epsilon)^{C \cdot V}, \tag{4}$$

*then the following holds for  $\lambda > 1$ :*

$$L_\lambda(\epsilon) \geq CV \log(M/\epsilon) / \log(K) \tag{5}$$

*where  $K = \lambda^V$  if  $V \geq 3$ ,  $\lambda^V + 1$  if  $V \leq 2$ , i.e.*

$$L_\lambda(\epsilon) \geq C \log(M/\epsilon) / (\log(\lambda)). \tag{6}$$

**Remark.**  $K$  is an upper bound on the number of possible classifications of  $\lambda$  points, given a class of function with VC-dimension  $V$ .  $K = \lambda^V$  stems from Sauer’s lemma (see [19]).  $K$  is exponential in  $V$ , but finite, thanks to the finite

---

<sup>3</sup> That is, linear separators whose value in the null vector is 0.

number of possible classifications of  $\lambda$  points when the VC-dimension is  $V$ . Having this upper bound on the number of reachable states for one batch, the number of possible branches in a run of the algorithm can be bounded accordingly.

**Proof:** Consider an algorithm realizing  $L_\lambda(\epsilon)$ .

As the algorithm is deterministic, there is one and only one possible value for the  $\lambda$ -uple  $(x_1, \dots, x_\lambda)$ , independently of  $f$ :

$$(x_1, \dots, x_\lambda) = \text{generate}(I_0).$$

Thanks to the finite VC-dimension and to Sauer’s lemma, there are at most  $K$  possible values for  $y_1, \dots, y_\lambda$ ; therefore there are at most  $K$  possible values for  $I_1$  (since the algorithm is assumed to be deterministic).

Similarly, for each possible value of  $I_1$ , there are at most  $K$  possible values for  $I_2$ ; therefore the total number of possible values for  $I_2$  is at most  $K^2$ .

By induction, there are at most  $K^i$  possible values for  $I_i$ . After  $L_\lambda(\epsilon)$  iterations, each possible state  $I_{L_\lambda(\epsilon)}$  corresponds to a function learned with  $\lambda L_\lambda(\epsilon)$  examples. Since the algorithm realizes the bound  $L_\lambda(\epsilon)$ , for any 2 oracle functions distant of  $\epsilon$  or more, the algorithm must have 2 different states. Thus, the final number of states is at least as big as the packing number:  $K^{L_\lambda(\epsilon)} \geq \text{pack}_{\mathcal{F}}(\epsilon)$ . As a consequence,

$$L_\lambda(\epsilon) \geq \log(\text{pack}_{\mathcal{F}}(\epsilon)) / \log(K). \tag{7}$$

Eqs. 7 and 4 yield the expected result. □

The next result shows that this bound is tight, at least asymptotically ( $\lambda \rightarrow \infty$ ).

**Theorem 2 (Upper bound for batch AL).** *In AL framework of Algo. 1, assume  $\mathcal{F}$  has VC-dimension  $V$ . Define  $K = \lambda^V + 1$  and*

$$\lambda' = \lambda \frac{K^D - 1}{K - 1}. \tag{8}$$

*Then the following holds for all  $D \geq 1$ :*

$$L_{\lambda'}(\epsilon) \leq \lceil L_\lambda(\epsilon) / D \rceil \tag{9}$$

**Remark.** Eq. 9 leads to

$$L_{\lambda'}(\epsilon) = O(\lceil L_\lambda(\epsilon) / \log(\lambda') \rceil) \tag{10}$$

for fixed  $V$  and  $\lambda$ . This is a logarithmic speed-up: see for instance that if  $\lambda = 1$ , then  $\forall D, L_{2^D}(\epsilon) = O(\frac{L_1(\epsilon)}{D})$

**Proof:** The proof exhibits an algorithm realizing Eq. 9. Consider an algorithm  $\mathcal{A}_\lambda = (\text{learn}_\lambda, \text{generate}_\lambda, \text{update}_\lambda)$  realizing  $L_\lambda(\epsilon)$  and consider some  $D \geq 1$ . Define  $\lambda' = \lambda \frac{K^D - 1}{K - 1}$ . Consider, then, another algorithm  $\mathcal{A}_{\lambda'} = (\text{learn}_{\lambda'}, \text{generate}_{\lambda'}, \text{update}_{\lambda'})$  which generates  $\lambda'$  points by simulating  $\mathcal{A}_\lambda$  on  $D$  steps; if  $\mathcal{A}_\lambda$  has internal state  $I_n$ , then  $\mathcal{A}_{\lambda'}$  has  $n^{\text{th}}$  internal state  $I'_n = I_{Dn}$ . At each iteration:

–  $generate_{\lambda'}$  simulates the  $K^D$  possible internal paths

$$(I_{Dn}, I_{Dn+1}, \dots, I_{Dn+D})_k \text{ with } k \in [[0, K^D]] \tag{11}$$

and generates for each iteration all the  $\lambda'$  points visited in any of those paths. At state  $I_{Dn}$   $\lambda$  points are to be labeled. Then, there are  $K$  possible states  $I_{Dn+1}$  resulting in  $K\lambda$  other points. Repeating the process for  $I_{Dn+2}, \dots$ , one can see that  $\lambda'$  as in Eq. 8 is enough;

- the target  $f$  is computed at these  $\lambda'$  points. It is then possible to figure out which path among the  $K^D$  possible paths is the one that would actually have happened during  $D$  steps of  $\mathcal{A}_\lambda$
- $update_{\lambda'}$  is the result of  $update_\lambda$  for the path selected in  $generate_{\lambda'}$  4;
- the output of  $learn_{\lambda'}$  is the output of  $learn_\lambda$  on all points visited in the selected path 5. □

Eqs. 6 and 9 show that  $\log(\lambda)$  is the optimal speed-up when no assumption on  $\lambda$  are made: theorem 2 shows that in all cases, a logarithmic speed-up is achievable and Theorem 1 shows that we cannot do much better for  $\lambda$  large. The rate of batch AL is therefore at least logarithmic as a function of  $\lambda$ , at most linear, and for  $\lambda$  large enough at most logarithmic.

The remaining question is what happens for moderate values of  $\lambda$ , and in particular how many simultaneous queries we need for removing the dependency in  $V$ . We now show that  $\lambda = V$  leads to a nearly linear speed-up, for some families  $\mathcal{F}$ . This removes the dependency in  $V$  in runtimes—this means that the curse of dimensionality can be broken with a batch size of  $V$ , whereas  $\lambda > V$  will only provide a logarithmic speed-up.

Let us remind that for binary classification, function classes of a domain  $X$  can equivalently be described as sets of subsets of  $X$ . In our case, the convention is that a set describes all the instances on which the function values are 1 (the complement of the set is thus where the function values are 0).

**Theorem 3 (Linear speed-up until  $\lambda = V$ ).** *Consider  $\mathcal{F}_V = \{[0, x], x \in [0, 1]^V\}$ . Then, for some  $M > 0, M' > 0$ ,*

$$\exists C > 0, \forall V, \exists \epsilon_0, \forall \epsilon < \epsilon_0, L_1^{\mathcal{F}_V}(\epsilon) \geq CV \log(M/\epsilon) \tag{12}$$

and

$$\exists C'; \forall V, \exists \epsilon_0, \forall \epsilon < \epsilon_0, L_V^{\mathcal{F}_V}(\epsilon) \leq C' \log(M'/\epsilon). \tag{13}$$

It is a classical result that  $VC - \dim(\mathcal{F}_V) = V$ . Eqs. 12 and 13 state the linear speed-up for batch active learning with  $\lambda = V$  for this family of functions (within the constants  $C$  and  $C'$ ).

<sup>4</sup> Therefore, only  $D\lambda$  points are actually used among the  $\lambda'$  labeled points.

<sup>5</sup> A big part of the points for which the target value has been computed is discarded. This is necessary for the formal proof of tightness of complexity bounds. For real-world applications, we guess that applying  $learn_{\lambda'}$  on all points might be much better, within constant factors however.



**Proof:** We first show that the following holds:

$$\exists C > 0, \forall V, \exists \epsilon_0, \forall \epsilon < \epsilon_0, \text{pack}_{\mathcal{F}_V}(\epsilon) \geq M/\epsilon^{(C \times V)}. \tag{14}$$

Eq. 14 is a version of Eq. 1 modified for considering only  $\epsilon$  small; it is weaker than Eq. 1 and sufficient for our purpose.

Eq. 14 is proved as follows:

- For  $x$  and  $y$  in  $[\frac{1}{2}, 1]^V$ , the  $L^1$  distance between  $[0, x]$  and  $[0, y]$  is lower bounded by  $\Theta(\|x - y\|_1)$ .
- Therefore, a regular grid of edge  $\Theta(\epsilon)$  can be constructed in  $[0, 1]^V$ , yielding  $\Theta(1/\epsilon^V)$  points for the sole  $[1/2, 1]^V$  part. Consequently, the packing number of  $\{[0, x]; x \in [0, 1]^V\}$  is  $\Theta(1/\epsilon^V)$  and is therefore  $\omega(1/\epsilon^{V/2})$ .
- This shows Eq. 14 for  $C = \frac{1}{2}$ .

Then, Eq. 14 classically leads to Eq. 12 (this is analogous to the proof of Eq. 2 from Eq. 1, see section 3). This is the first part of the theorem (Eq. 12). Let us now show Eq. 13, by considering the following algorithm described at iteration  $n$ , with  $\lambda = V$ :

- *generate* $_\lambda$  prepares the batch  $((x_{n\lambda+1}, \dots, x_{(n+1)\lambda})$  as follows: for each  $x_{n\lambda+i}$ , all coordinates  $j \neq i$  are set to 0. The  $i^{th}$  coordinate of  $x_{n\lambda+i}$  is chosen by looking at the  $n - 1$  previous points in position  $i$  of each of the  $n - 1$  previous batches. It is defined as the middle of the segment defined by the lowest previously-observed  $i^{th}$  coordinate whose label is 0, and the highest previously observed  $i^{th}$  coordinate whose label is 1. More formally: 6

$$(x_{n\lambda+i})_i = \frac{1}{2} \left( \min_{n' \leq n} \{(x'_{n'\lambda+i})_i | y'_{n'\lambda+i} = 0\} + \max_{n' \leq n} \{(x'_{n'\lambda+i})_i | y'_{n'\lambda+i} = 1\} \right). \tag{15}$$

- *learn* $_\lambda$  selects any function  $f_n \in \mathcal{F}_V$  which is consistent with  $x_1, \dots, x_{n\lambda}$ .

At a given iteration  $n$  each point  $x_{n,i}$  of the batch of size  $\lambda$  makes sure that the domain will be halved along the  $i^{th}$  coordinate. Thus, after  $N$  iterations, it is known that the target oracle/classifier is in a square of edge size  $2^{-N}$ . As a consequence, precision  $\epsilon$  is reached in at most  $\Theta(\log(1/\epsilon))$  iterations, which shows Eq. 13. □

This theorem shows that, at least for  $\mathcal{F}$  as above, we can have a linear speed-up until  $\lambda = V$ ; this is the tightness of Eq. 3 for  $\lambda \leq V$ —similarly to the tightness of Eq. 6 (*i.e.* logarithmic speed-up) shown by Eq. 10 for  $\lambda$  large.

---

<sup>6</sup> In Eq. 15, if no point  $x_{n'\lambda+i}$  has been labeled as 0, the minimum is set to 1; equivalently, if no point has been labeled as 1, the maximum is set to 0.

## 4 Experiments

We have formally proved both lower and upper bounds on batch AL. The following shows that both a simple algorithm and a more sophisticated (yet usual) AL algorithm behave as predicted by the theorems of previous sections when adapted to the batch setting.

### 4.1 Experiments with Naive AL

We here experiment a simple batch AL algorithm for  $\mathcal{F} = \{[0, x]; x \in [0, 1]^d\}$  (VC-dimension  $V = d$ ). The new sample(s)  $x_{n\lambda+1}, \dots, x_{(n+1)\lambda}$  are  $\lambda$  points randomly drawn in  $v$  where

$$v = \{x \in [0, 1]^d; \forall j \in [[1, n\lambda]] y_j = 0 \Rightarrow \neg(x_j \leq x)\} \\ \cap \{x \in [[0, 1]^d; \forall j \in [[1, n\lambda]] y_j = 1 \Rightarrow x_j \leq x\}.$$

This means that we randomly sample the version space. Note that this parallel algorithm is straightforward to derive from its sequential counterpart that queries one random sample from the version space at each iteration. This is not true of all active learning algorithms: in many cases, it is not clear how to efficiently turn a sequential active learning into a parallel one.

The plot shows the inverse of the number of iterations for reaching precision  $0.001d^2$ , depending on  $\lambda$ ; this means that what lies on the Y-axis are *rates* of convergence. Results are presented in Fig. 1. Each point is averaged over 33 runs.

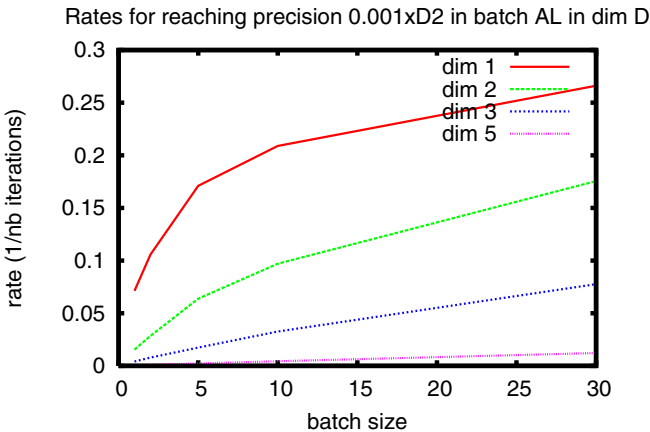
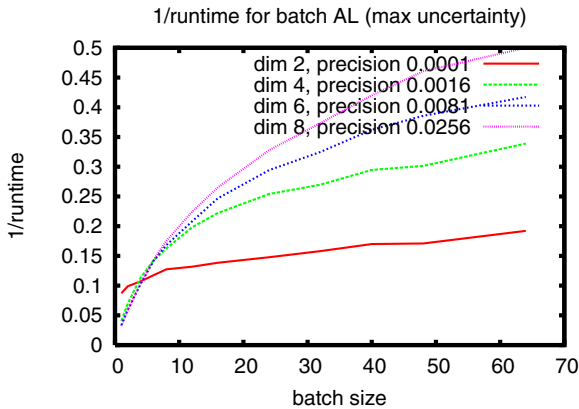


Fig. 1. Speed-up of batch AL for a simple AL algorithm (see text)

## 4.2 Experiments with Max-uncertainty

This part of the experiments is concerned with a straightforward adaptation of a good, classical active learning heuristic that we call *Maximum Uncertainty* to the batch setting. The idea is to choose the most *uncertain* examples, meaning the ones for which many approximations of the target function that are still good candidates disagree on the label.

The possible approximations lie in the *version space*, the space of all possible functions of  $\mathcal{F}$  consistent with the examples observed so far. Each example  $x$  splits the version space in two: the functions labeling  $x$  by 1, and those labeling  $x$  by 0. Thus, the goal is to find examples separating the version space the most evenly. This criterion has been studied empirically and theoretically; multiple algorithms enforcing this criterion or related criteria have been proposed [21,6].



**Fig. 2.** Speed-up of batch maximum uncertainty active learning

Experiments learn homogeneous linear separators of  $\mathbb{R}^d$ , where examples lie on the hypersphere  $\mathbb{S}^{d-1}$  for dimensions  $d = 2, 4, 6, 8$ . This setting has been widely studied for sequential active learning [2,8,10] and is thus fitted to a speed-up analysis for the batch setting.

In such a setting, for  $d > 2$ , an infinite number of points of the hypersphere maximize uncertainty given previously witnessed instances—whereas if  $d = 2$ , the maximum is unique. Consequently, a possible batch strategy may consist in selecting  $\lambda$  of those points maximizing uncertainty, at each iteration. Note that contrary to the algorithm of the preceding subsection, it is less obvious, at first sight, that this parallelization will be an efficient one (although a posteriori the results emphasize good speed-ups).

Batch sizes are  $\lambda = 1, 2, 4, 6, 8, 12, 16, 20, 24, 32, 40, 48, 64$ . Precision is set to  $0.0001(d/2)^4$ . For each  $(d, \lambda)$ , the rate are averaged over 160 runs.

## Interpretation

In both cases, the results resemble the expected behavior: a steady (linear) speed-up for small batch sizes, when  $\lambda < d$  (where  $d$  is the dimension), and a slow speed-up when  $\lambda$  becomes much bigger than  $d$ , that somewhat resembles a logarithmic speed-up. Thus, as expected, the gain of parallelization in high dimension is bigger.

## Remarks

- We did not study the behavior of the speed-up for values of  $\lambda$  in-between  $d$  and  $\lambda$  large; it might be that the speed-up can remain good even for a while when  $\lambda > d$ , but asymptotically it will end in a logarithmic improvement;
- The rates for high dimensions seem low (although linear) on the first figure, while they seem higher in the second. This is due to the fact that the precision to be reached is bigger in the second figure than in the first, in an attempt to be more “fair” to high dimensions—since for a given number of examples, a concept is harder to learn if the dimension of the domain is higher.

## 5 Discussion

This paper shows that batch active learning exhibits:

- a linear speed-up until  $\lambda = V$  for some families of target functions;
- a speed-up at least logarithmic in all cases;
- and a logarithmic speed-up at most for  $\lambda$  large.

Please note that the logarithmic speed-up is a simulation result. The point is not to analyze the convergence rate of active learning in general, but to emphasize that *any* active learning algorithm can be transformed into a batch active learning algorithm (with  $\lambda$  computation units) which simulates it with speedup  $D$ , where  $D$  is logarithmic as a function of  $\lambda$ .

All proofs have been made for deterministic algorithms. Extending them to randomized algorithms, however, is straightforward.

Experiments have been performed only in moderate dimension and for easy families of functions; the extension of the experiments to bigger dimensions and to other families of functions, is a possible further work.

**Acknowledgements.** This work was supported by the French National Research Agency (ANR) grant No. ANR-08-COSI-007-12, and through COSINUS program (project EXPLO-RA n ANR-08-COSI-004).

## References

1. Angluin, D.: Queries and concept learning. *Mach. Learn.* 2(4), 319–342 (1988)
2. Balcan, M.-F., Broder, A., Zhang, T.: Margin based active learning. In: *Proc. of the 20 th Conference on Learning Theory* (2007)

3. Cesa-bianchi, N., Conconi, A., Gentile, C.: Learning probabilistic linear-threshold classifiers via selective sampling. In: Schölkopf, B., Warmuth, M.K. (eds.) COLT/Kernel 2003. LNCS (LNAI), vol. 2777, pp. 373–386. Springer, Heidelberg (2003)
4. Cohn, D., Atlas, L., Ladner, R.: Improving generalization with active learning. *Mach. Learn.* 15(2), 201–221 (1994)
5. Cohn, D., Ghahramani, Z., Jordan, M.: Active Learning with Statistical Models. *Journal of Artificial Intelligence Research* 4, 129–145 (1996)
6. Dasgupta, S.: Analysis of a greedy active learning strategy. In: Saul, L.K., Weiss, Y., Bottou, L. (eds.) *Advances in Neural Information Processing Systems*, vol. 17, pp. 337–344. MIT Press, Cambridge (2005)
7. Dasgupta, S.: Coarse sample complexity bounds for active learning. In: Weiss, Y., Schölkopf, B., Platt, J. (eds.) *Advances in Neural Information Processing Systems*, vol. 18, pp. 235–242. MIT Press, Cambridge (2006)
8. Dasgupta, S., Kalai, A.T., Monteleoni, C.: Analysis of perceptron-based active learning. In: Auer, P., Meir, R. (eds.) COLT 2005. LNCS (LNAI), vol. 3559, pp. 249–263. Springer, Heidelberg (2005)
9. Devroye, L., Györfi, L., Lugosi, G.: *A Probabilistic Theory of Pattern Recognition*. Springer, Heidelberg (1997)
10. Freund, Y., Seung, H.S., Shamir, E., Tishby, N.: Selective sampling using the query by committee algorithm. *Mach. Learn.* 28(2-3), 133–168 (1997)
11. Guo, Y., Schuurmans, D.: Discriminative batch mode active learning. In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 593–600. MIT Press, Cambridge (2008)
12. Hoi, S.C.H., Jin, R., Zhu, J., Lyu, M.R.: Batch mode active learning and its application to medical image classification. In: *ICML '06: Proceedings of the 23rd International Conference on Machine Learning*, pp. 417–424. ACM, New York (2006)
13. Hoi, S.C.H., Jin, R., Zhu, J., Lyu, M.R.: Semisupervised svm batch mode active learning with applications to image retrieval. *ACM Trans. Inf. Syst.* 27(3), 1–29 (2009)
14. Iyengar, V.S., Apte, C., Zhang, T.: Active learning using adaptive resampling. In: *Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 91–98 (2000)
15. Kulkarni, S.R., Mitter, S.K., Tsitsiklis, J.N.: Active learning using arbitrary binary valued queries. *Mach. Learn.* 11(1), 23–35 (1993)
16. Lindenbaum, M., Markovitch, S., Rusakov, D.: Selective sampling for nearest neighbor classifiers. *Machine Learning* 54, 125–152 (2004)
17. Mitchell, T.M.: Generalization as search. *Artificial Intelligence* 18, 203–226 (1982)
18. Roy, N., McCallum, A.: Toward optimal active learning through sampling estimation of error reduction. In: *Proc. 18th International Conf. on Machine Learning*, pp. 441–448. Morgan Kaufmann, San Francisco (2001)
19. Sauer, N.: On the density of families of sets. *J. Comb. Theory, Ser. A* 13(1), 145–147 (1972)
20. Schohn, G., Cohn, D.: Less is more: Active learning with support vector machines. In: *Proceedings of the Seventeenth International Conference on Machine Learning*, vol. 282, pp. 285–286 (2000)
21. Seung, H.S., Opper, M., Sompolinsky, H.: Query by committee. In: COLT '92: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pp. 287–294. ACM, New York (1992)

22. Sugiyama, M., Rubens, N.: A batch ensemble approach to active learning with model selection. *Neural Netw.* 21(9), 1278–1286 (2008)
23. Vapnik, V.: *The Nature of Statistical Learning Theory*. Springer, N.Y. (1995)
24. Vidyasagar, M.: *A Theory of Learning and Generalization*. Springer, New York (1997)
25. Warmuth, M.K., Liao, J., Rätsch, G., Mathieson, M., Putta, S., Lemmen, C.: Support vector machines for active learning in the drug discovery process. *Journal of Chemical Information Sciences* 43, 667–673 (2003)

# Semi-supervised Projection Clustering with Transferred Centroid Regularization

Bin Tong<sup>1</sup>, Hao Shao<sup>1</sup>, Bin-Hui Chou<sup>2</sup>, and Einoshin Suzuki<sup>1,2</sup>

<sup>1</sup> Graduate School of Systems Life Sciences, Kyushu University, Fukuoka, Japan

<sup>2</sup> Department of Informatics, ISEE, Kyushu University, Fukuoka, Japan

**Abstract.** We propose a novel method, called Semi-supervised Projection Clustering in Transfer Learning (SPCTL), where multiple source domains and one target domain are assumed. Traditional semi-supervised projection clustering methods hold the assumption that the data and pairwise constraints are all drawn from the same domain. However, many related data sets with different distributions are available in real applications. The traditional methods thus can not be directly extended to such a scenario. One major challenging issue is how to exploit constraint knowledge from multiple source domains and transfer it to the target domain where all the data are unlabeled. To handle this difficulty, we are motivated to construct a common subspace where the difference in distributions among domains can be reduced. We also invent a *transferred centroid regularization*, which acts as a bridge to transfer the constraint knowledge to the target domain, to formulate this geometric structure formed by the centroids from different domains. Extensive experiments on both synthetic and practical data sets show the effectiveness of our method.

## 1 Introduction

Over the past decades, clustering for high dimensional data, referred as projection clustering, has witnessed an increasing interest in the clustering literature. It is designed to derive a subspace where the clustering performance can be enhanced, and the curse of high dimensionality [18] can be mitigated to some extent as well. The projection clustering can be divided into two categories, which are unsupervised ones [22,8] and semi-supervised ones [20]. An important branch of semi-supervised projection clustering is to exploit additional background information posed on data points, i.e., pairwise constraints, to aid unsupervised projection clustering. Each pairwise constraint indicates whether a pair of data points must reside in the same cluster [21]. However, one limitation of semi-supervised projection clustering techniques is that all the data points are drawn from the same distribution. Unfortunately, many scenarios in real applications do not follow this requirement. Typically, many related data sets with different distributions are available, which may be handled by transfer learning [17,13,6].

In this paper, we investigate a new transfer clustering task that would be met in real applications. One example [10] is clustering Web pages from four

universities, i.e., Cornell, Texas, Wisconsin and Washington. The Web pages of each university are grouped into 7 categories, i.e., student, faculty, staff, department, course, project and others. The Web pages satisfy the typical setting of transfer learning. The contents from four universities are related to each other. However, their distributions are different, since different universities exhibit different features. In the first three universities, referred as three source domains, we are given pairwise constraints inside each university. We are going to utilize the pairwise constraint knowledge to cluster unlabeled Web pages of Washington university, referred as the target domain. Since the distributions of the source domains are different from that of the target domain, the pairwise constraints from multiple source domains cannot be directly employed in the target domain. Therefore, one challenging issue in this task is how to exploit constraint knowledge from multiple source domains to aid the clustering in the target domain. To the best of our knowledge, this issue has not been addressed in the transfer learning literature.

To handle this challenging issue, we are motivated to invent a novel clustering method, called Semi-supervised Projection Clustering in Transfer Learning (SPCTL), to transfer the constraint knowledge from multiple source domains to the target domain. In the first step, we seek a common subspace where the difference in distributions among the domains can be reduced. We do not expect that the data from different domains have the same cluster structure in the original space, but assume the cluster centroids from all the domains follow a certain manifold structure in the common subspace. That is, similar clusters have similar centroids in the common subspace. For example, as to the faculty clusters from different universities, since the common subspace is spanned by the most relevant features among faculties, the centroids of faculty clusters from different universities are assumed to be located nearby in the common subspace. It has inspired us to invent a *transferred centroid regularization* to formulate the manifold structure formed by the centroids in the common subspace and to implicitly transfer the constraint knowledge to the target domain.

## 2 Problem Setting and Motivation

The problem setting of our method is described as follows. Suppose that we are given source domains  $\mathcal{D}^s$ ,  $s = 1, 2, \dots, P$  ( $P \geq 2$ ), where  $P$  is the number of the source domains, and a target domain  $\mathcal{D}_T$ . We define the data set of the source domains as  $D_s = \{\mathbf{x}_i^s\}_{i=1}^{n_s}$ ,  $s = 1, 2, \dots, P$ , and the data set of the target domain as  $D_T = \{\mathbf{x}_i'\}_{i=1}^{n_T}$ , where  $n_s$  denotes the number of data points in the  $s$ -th source domain,  $n_T$  represents the number of data points in the target domain, and  $\mathbf{x}_i^s, \mathbf{x}_i' \in \mathbb{R}^d$ . We assume that there is no irrelevant source domain to the target domain. Let  $\mathcal{P}(D_s)$  and  $\mathcal{P}(D_T)$  be the marginal distribution of  $D_s$  and  $D_T$ , respectively. In general, they are somehow related to each other yet different. For the  $s$ -th source domain  $\mathcal{D}_s$ ,  $X_s = \{D_s, D_T\}$  represents the data points from both the  $s$ -th source domain and the target domain, and  $N_s = n_s + n_T$  represents the number of data points in  $X_s$ . We also assume a set of pairwise constraints for

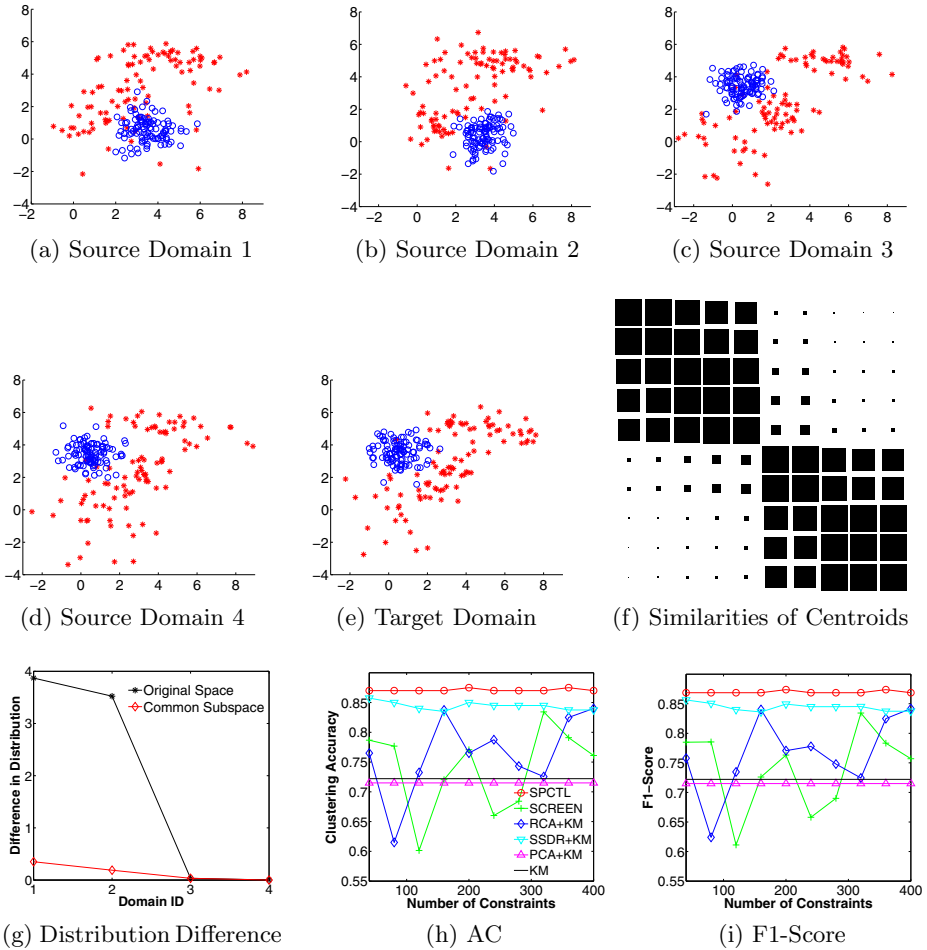


each source domain, which are must-link constraints and cannot-link constraints. There is no assumption posed on the pairwise constraints. In general, the pairwise constraints are randomly selected by the users. In the  $s$ -th source domain, the set of must-link constraints is represented as  $\mathbf{M}^s = \{m_1^s, m_2^s, \dots, m_{r_s}^s\}$ , and the set of cannot-link constraints is denoted by  $\mathbf{C}^s = \{c_1^s, c_2^s, \dots, c_{l_s}^s\}$ , where  $r_s$  and  $l_s$  denote the number of must-link constraints and cannot-link constraints, respectively. More precisely,  $m_i^s$  consists of a pair of points belonging to the same class while  $c_i^s$  consists of a pair of points belonging to different classes. For the target domain, we assume that all the data points are unlabeled. Moreover, we assume that each domain has  $K$  clusters, and the  $K$  clusters in a domain conceptually correspond to the  $K$  clusters in another domain. The output of our method is a cluster indicator matrix  $Q_T$  for the target domain and a  $d \times l$  transformation matrix  $W$  with an orthogonal constraint  $W^T W = I$ .  $W$  consists of  $l$  projective vectors  $\{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_l\}$ , where  $l$  is the dimensionality of a common subspace shared by all the domains.

The goal of our method is to obtain a good clustering performance for the target domain in the common subspace. To achieve it, we are going to appropriately transfer the constraint knowledge from multiple source domains  $\mathcal{D}_s$ ,  $s = 1, 2, \dots, P$ , to the target domain  $\mathcal{D}_T$ .

## 2.1 Motivation for Transferred Centroid Regularization

In this subsection, we show a motivating example that empirically explains why we build a *transferred centroid regularization* in a common subspace where the difference in distributions among domains is reduced. We follow the way of [14] to generate five domains, where the first four domains are regarded as the source domains while the last domain is the target domain, as shown from Fig. 1a to Fig. 1e. Each domain consists of examples belonging to one of two classes. The examples of class one (denoted with stars) are drawn from a Gaussian Mixture Model (GMM) and the examples of class two (denoted with circles) are drawn from a single Gaussian. For the domains 1 and 2, the GMM parameters of class one are prescribed by a three-component mixture defined as follows. The mixture weights are (0.3, 0.3, 0.4); their respective means are (1, 1), (3, 3) and (5, 5); their respective covariance matrices are  $\Sigma_1 = \begin{pmatrix} 0.3 & 0.7 \\ 0.7 & 3.0 \end{pmatrix}$ ,  $\Sigma_2 = \begin{pmatrix} 3.0 & 0.0 \\ 0.0 & 0.3 \end{pmatrix}$  and  $\Sigma_3 = \begin{pmatrix} 3.0 & -0.5 \\ -0.5 & 0.3 \end{pmatrix}$ . The examples of class two are drawn from a single Gaussian with mean (3.5, 0.5) and diagonal covariance with symmetric variance 0.5. For the domains 3, 4 and the target domain, the three component GMM parameters of class one are prescribed as follows. The mixture weights are (0.3, 0.3, 0.4); their respective means are (1, 1), (3, 3) and (5, 5); their respective covariance matrices are  $\Sigma_2$ ,  $\Sigma_3$  and  $\Sigma_1$ . The examples of class two are drawn from a single Gaussian with mean (0.5, 3.5) and diagonal covariance with symmetric variance 0.5. Note that each source domain has 50 must-link constraints and 50 cannot-link constraints, which are generated randomly. To keep the



**Fig. 1.** Motivating example

Figures readable, the pairwise constraints are not shown in the Figures. This motivating example is designed to obtain a one-dimensional common subspace where the clustering for the target domain is performed.

From Fig. 1a to Fig. 1e, we can see that the data between domains 1 and 2 are similar and the data among domains 3, 4 and 5 are similar. However, the distributions between two groups of domains are obviously different. In Fig. 1f, we show that the similarities among ten centroids in the common subspace from all the source domains and the target domain in Hinton diagram 1f, in which the square size of the  $(i, j)$ -th element denotes the degree of similarity between the  $i$ -th and  $j$ -th centroids. The similarity is computed by  $\exp(-\|f_i - f_j\|^2 / 2)$ , where  $f_i$  represents the  $i$ -th centroid. Note that a larger square in the Hinton diagram indicates a larger value of the similarity between two centroids. It is obvious from

Fig. 1f that the centroids from all the domains form two clusters in the common subspace, each of which is composed of centroids from the corresponding clusters in different domains. This specific geometric distribution of centroids can be considered as a manifold structure. In addition, we employ *Maximum Mean Discrepancy* (MMD) [16,15] to measure the distribution difference between each source domain and the target domain. As shown in Fig. 1g, we can see that the distribution difference between each source domain and the target domain in the common subspace is much smaller than that in the original space, especially for the domains 1 and 2. In Fig. 1h and Fig. 1i, we compare our method, SPCTL, with other methods. Except for  $k$ -means (KM) and PCA+ $k$ means (PCA+KM), the competitive methods, RCA [1]+ $k$ means (RCA+KM), SSDR [24]+ $k$ means (SSDR+KM), and SCREEN [20], simply combine the pairwise constraints from different source domains together. We can see that SPCTL outperforms others in terms of two clustering evaluation measures, clustering accuracy (AC) [23] and F1-Score [6].

From the motivating example, we observe that the centroids from all the domains form a certain manifold structure in the common subspace where the difference in distributions is reduced. This fact has motivated us to embed the constraint knowledge from the source domains into the centroids, and exploit the *transferred centroid regularization* to transfer the constraint knowledge from multiple source domains to the target domain.

### 3 Proposed Framework

Our framework consists of two steps. In the first step, we invent a new semi-supervised projection clustering method which considers the domain adaptation [15,16] and the constraint knowledge. We obtain the centroids, where constraint knowledge is embedded, from all the source domains in the common subspace where the difference in distributions is reduced. In the second step, we propose the *transferred centroid regularization* to formulate the manifold structure of centroids in the common subspace such that the constraint knowledge can be transferred to the target domain.

#### 3.1 Projection Clustering

Before introducing any further, we firstly review a projection clustering method, named AML [22], used in our method. For the  $s$ -th source domain, each vector  $\mathbf{x}_i^s$  in the  $d$ -dimensional space is transformed by  $W$  to a vector  $\mathbf{y}_i^s$  in the  $l$ -dimensional space, where  $\mathbf{y}_i^s = W^T \mathbf{x}_i^s$ . The distance between two points in the common subspace is measured by the *Mahalanobis* distance [1] defined as follows:

$$d_M(\mathbf{x}_i^s, \mathbf{x}_j^s) = \sqrt{(\mathbf{y}_i^s - \mathbf{y}_j^s)^T \Sigma_s^{-1} (\mathbf{y}_i^s - \mathbf{y}_j^s)} \quad (1)$$

where  $\Sigma_s$  is the covariance matrix defined as follows:

$$\Sigma_s = \frac{1}{n_s} \sum_{i=1}^{n_s} (\mathbf{y}_i^s - \mathbf{u}^s)(\mathbf{y}_i^s - \mathbf{u}^s)^T \quad (2)$$

where  $\mathbf{u}^s = \frac{1}{n_s} \sum_{i=1}^{n_s} \mathbf{y}_i^s$ . In general, maximization of the *Sum of Squared Intra-cluster Error* (SSIE) can be equivalently regarded as the cost function of the standard  $k$ -means algorithm. SSIE is defined as follows:

$$\text{SSIE} = \sum_{j=1}^K n_s^j d_M(\mathbf{u}_j^s, \mathbf{u}^s)^2 \quad (3)$$

where  $n_s^j$  is the sample size of the  $j$ -th cluster  $C_j^s$  in the  $s$ -th source domain, and  $\mathbf{u}_j^s$  is the mean of  $C_j^s$ . Let  $Q_s$  be a  $n_s \times K$  cluster indicator matrix defined as follows.  $Q_s = \{r_{ij}^s\}_{n_s \times K}$ , where  $r_{ij}^s = 1$ , if  $\mathbf{x}_i^s \in C_j^s$ . We define the *weighted cluster indicator* matrix as  $L_s = [L_s^1, L_s^2, \dots, L_s^K]$ , so that  $L_s = Q_s(Q_s^T Q_s)^{-\frac{1}{2}}$  [8]. It follows that the  $j$ -th column of  $L_s^j$  is given by

$$L_s^j = (0, \dots, 0, \overbrace{1, \dots, 1}^{n_s^j}, 0, \dots, 0)^T / \sqrt{n_s^j} \quad (4)$$

Therefore, SSIE can be succinctly rewritten as the following form [22], which needs to be maximized with respect to  $W$  and  $L_s$ :

$$\xi_s = \frac{1}{n_s} \text{tr}(L_s^T D_s^T W (W^T \Sigma_s W)^{-1} W^T D_s L_s) \quad (5)$$

where  $\text{tr}$  denotes the trace operator.

### 3.2 Semi-supervised Projecting Clustering via Domain Adaptation

The key idea of the new semi-supervised projection clustering is to integrate two factors, which are constraint knowledge and domain adaptation, into projection clustering. Traditional domain adaptation methods only apply to the case of one source domain and one target domain. In our problem setting, we extend it to the case of multiple source domains and one target domain.

As to the  $s$ -th source domain, following the idea of [12], we change the penalties of violations in the constraints in  $\mathbf{M}^s$  into the *awards*. In addition, we place a constraint for all the source domains that the clustering result in the common subspace is the same as that in the original space. The *awards* in the common subspace can thus be written as follows:

$$\varphi_s = \sum_{\substack{\{\mathbf{x}_i^s, \mathbf{x}_j^s\} \in \mathbf{C}^s \\ \text{s.t. } l_i^s = l_j^s}} \vartheta_{ij} - \sum_{\substack{\{\mathbf{x}_i^s, \mathbf{x}_j^s\} \in \mathbf{M}^s \\ \text{s.t. } l_i^s = l_j^s}} \theta_{ij} \quad (6)$$

where  $l_i^s$  denotes the class label of  $\mathbf{x}_i^s$ . Eq. (6), which needs to be minimized with respect to  $L_s$ , can be rewritten as follows:

$$\varphi_s = \text{tr}(L_s^T \Theta^s L_s) \quad (7)$$

where  $\Theta^s$  is a  $n_s \times n_s$  matrix with its  $(i, j)$ -th entry defined as follows:

$$\Theta_{ij}^s = \begin{cases} \vartheta_{ij}, & \text{when } \{\mathbf{x}_i^s, \mathbf{x}_j^s\} \in \mathbf{C}^s \\ -\theta_{ij}, & \text{when } \{\mathbf{x}_i^s, \mathbf{x}_j^s\} \in \mathbf{M}^s \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

where  $\vartheta_{ij}$  and  $\theta_{ij}$  are both positive reals. To consider the domain adaptation, in the common subspace, we employ *Maximum Mean Discrepancy* (MMD) [16,15] to measure the mismatch of distributions between the  $s$ -th source domain and the target domain. The criterion [6], which needs to be minimized with respect to  $W$ , is defined as follows:

$$\varpi_s = \text{MMD}[D^s, D_T] = \text{tr}(W^T X_s \mathbf{K}_s X_s^T W) \quad (9)$$

where

$$\mathbf{K}_{sij} = \begin{cases} \frac{1}{n_s^2}, & \text{when } \mathbf{x}_i^s, \mathbf{x}_j^s \in D_s \\ \frac{1}{n_s n_T}, & \text{when } \mathbf{x}_i^s, \mathbf{x}_j^T \in D_s \times D_T \\ \frac{-1}{n_s n_T}, & \text{otherwise} \end{cases} \quad (10)$$

So far, we only consider the  $s$ -th source domain and the target domain. In order to exploit the constraint knowledge from all the source domains and to derive the common subspace shared by all the domains, we then extend the objective function considering all the domain pairs that are composed of each source domain and the target domain. The final objective function, which needs to be minimized with respect to  $W$  and  $L_s$ , can be written as follows.

$$\begin{aligned} J &= \sum_{s=1}^P (\varphi_s + \alpha \varpi_s - \beta \xi_s) \\ \text{s.t } &W^T W = I, L_s \in \{0, 1\}^{n_s \times K} \end{aligned} \quad (11)$$

Note that by its definition, the elements of  $L_s$  can only take binary values, which makes the minimization of Eq. (11) very difficult. Therefore, we relax  $L_s$  into a nonnegative continuous domain. Then the objective function in Eq. (11) turns out to be

$$\begin{aligned} J &= \sum_{s=1}^P (\varphi_s + \alpha \varpi_s - \beta \xi_s) \\ \text{s.t } &W^T W = I, L_s \geq 0 \end{aligned} \quad (12)$$

where both  $\alpha$  and  $\beta$  are regularization parameters.

### 3.3 Target Clustering via Transferred Centroid Regularization

In the setting of transfer learning, we are more concerned about the performance of the target domain than the source domains. As shown in the previous sections, we assume that the cluster centroids from all the source domains form a certain manifold structure that the centroid of a cluster in a source domain is located nearby with the centroids of the corresponding clusters in other source domains. It is natural assumed that the cluster centroids from the target domain follow the same manifold structure. To exploit the constraint knowledge that is implicitly embedded into the centroids in the common subspace, we thus employ the manifold regularization [2] to formulate this specific structure.

Since the *weighted cluster indicator* matrix  $L_s$ ,  $s = 1, 2, \dots, P$ , can be derived from Eq. (12), we can compute the centroid matrix  $F_s$  by using  $L_s$ . We define  $F_T$  by the centroid matrix for the target domain. By combining  $F_T$  with  $F_s$ ,  $s = 1, 2, \dots, P$ , we can obtain the centroid matrix for all the domains, denoted by  $F = [F_T, F_1, \dots, F_P] \in \mathbb{R}^{K \times (P+1)}$ . We use  $f_i$  to denote the  $i$ -th column of  $F$ . We then invent the *transferred centroid regularization* to depict the graph formed by the centroids.

$$\delta = \frac{1}{2} \sum_{i,j} G_{ij} (f_i - f_j)^2 \tag{13}$$

where  $G$  is a similarity matrix defined as follows:

$$G_{ij} = \begin{cases} \exp(\|f_i - f_j\|^2 / \gamma), & f_i \in N(f_j) \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

where  $N(f_i)$  is defined as the  $P$  nearest neighbor set of  $f_i$ . Eq. (13) can be rewritten as follows:

$$\delta = \text{tr}(F^T M F) \tag{15}$$

where  $M = U - G$  is defined as a graph Laplacian matrix, where  $U$  is a diagonal matrix whose entries are column sums of  $G$ ,  $U_{ii} = \sum_j G_{ij}$ . In the common subspace, we then develop a new cost function for the  $k$ -means algorithm by integrating the *transferred centroid regularization*, which is written as the following form:

$$H = \|W^T D_T - F_T Q_T^T\| + \lambda \delta \tag{16}$$

where  $\lambda$  is a regularization parameter. Note that Eq. (16) needs to be minimized. From Eq. (16), our method can be seen as a *implicit transfer* method, as the constraint knowledge is transferred by formulating the centroid structure.

### 3.4 Optimization

We notice that, for minimizing Eq. (12) with respect to  $W$  and  $L_s$ ,  $s = 1, \dots, P$ , we cannot give a closed-form solution. However, Eq. (12) can be iteratively optimized with respect to  $L_s$  by fixing  $W$ , and vice versa.

Specially, when  $W$  is fixed, optimizing Eq. (12) with respect to  $L_s$  is equivalent to optimizing

$$\begin{aligned} J_{L_s} &= \text{tr}(L_s^T \Theta^s L_s) - \frac{\beta}{n_s} \text{tr}(L_s^T (D_s^T W (W^T \Sigma_s W)^{-1} W^T D_s) L_s) \\ \text{s.t. } L_s &\geq 0 \end{aligned} \tag{17}$$

We introduce the Lagrangian multiplier  $\psi \in \mathbb{R}^{n_s \times K}$ , and the Lagrangian function is

$$\begin{aligned} J_{L_s}^* &= \text{tr}(L_s^T \Theta^s L_s) - \frac{\beta}{n_s} \text{tr}(L_s^T (D_s^T W (W^T \Sigma_s W)^{-1} W^T D_s) L_s) \\ &\quad - \text{tr}(\psi L_s^T) \end{aligned} \tag{18}$$

By setting  $\frac{\partial J_{L_s}^*}{\partial L_s} = 0$ , we obtain

$$\psi = 2\mathbf{A}L_s \tag{19}$$

where  $\mathbf{A} = \Theta^s - \frac{\beta}{n_s} [D_s^T W (W^T \Sigma_s W)^{-1} W^T D_s]$ . Using the Karush-Kuhn-Tucker condition [5]  $\psi_{ij} L_{s_{ij}} = 0$ , we get

$$[\mathbf{A}L_s]_{ij} L_{s_{ij}} = 0 \tag{20}$$

By introducing  $\mathbf{A} = \mathbf{A}^+ - \mathbf{A}^-$  where  $\mathbf{A}^+ = (|\mathbf{A}_{ij}| + \mathbf{A}_{ij})/2$  and  $\mathbf{A}^- = (|\mathbf{A}_{ij}| - \mathbf{A}_{ij})/2$  [9], we obtain

$$[\mathbf{A}^+ L_s - \mathbf{A}^- L_s]_{ij} L_{s_{ij}} = 0 \tag{21}$$

Eq. (21) leads to the following updating formula.

$$L_{s_{ij}} \leftarrow L_{s_{ij}} \sqrt{\frac{[\mathbf{A}^- L_s]_{ij}}{[\mathbf{A}^+ L_s]_{ij}}} \tag{22}$$

Eq. (17) can be proven to be non-increasing by using the auxiliary function approach [9]. Due to space limitation, we leave it to the later version of our paper.

When  $L_s$  is fixed, optimizing Eq. (12) with respect to  $W$  is equivalent to optimizing

$$J_W = \sum_{s=1}^P \left[ \alpha \text{tr}(W^T \mathbf{G}_s W) - \frac{\beta}{n_s} \text{tr}((W^T \mathbf{C}_s W)^{-1} W^T \mathbf{E}_s W) \right] \\ \text{s.t. } W^T W = I \tag{23}$$

where  $\mathbf{G}_s = X_s \mathbf{K}_s X_s^T$ ,  $\mathbf{C}_s = \Sigma_s$ , and  $\mathbf{E}_s = D_s L_s L_s^T D_s^T$ . We use the *gradient descent* to obtain the optimal  $W$  for Eq.(23). Starting with an initial  $W^0$ , the gradient descent method successively updates  $W$  by

$$W^t = W^{t-1} - v \left. \frac{\partial J_W}{\partial W} \right|_{W=W^{t-1}} \tag{24}$$

where

$$\frac{\partial J_W}{\partial W} = 2\alpha \sum_{s=1}^P (\mathbf{G}_s W) + 2 \sum_{s=1}^P \left[ \frac{\beta}{n_s} \mathbf{C}_s W (W^T \mathbf{C}_s W)^{-1} W^T \mathbf{E}_s W (W^T \mathbf{C}_s W)^{-1} \right] \\ + 2 \sum_{s=1}^P \left[ \frac{\beta}{n_s} \mathbf{E}_s W (W^T \mathbf{C}_s W)^{-1} \right] \tag{25}$$

where  $v$  is the step size and  $t$  is the iteration number. We also notice that the updating rule of Eq. (24) is given without the orthogonality constraint  $W^T W = I$ . After each updating step of  $W^t$ , let  $W^t = W^t R$  be the *QR* decomposition [5] of  $W^t$ , where  $W^t$  has orthogonal columns and  $R$  is an upper triangle.  $W^t$  is then replaced by  $W^t$  for the next iteration. Eq. (23) is iteratively solved until a convergence condition, which is explained in detail in the next subsection, is satisfied.

### 3.5 Clustering Framework

This clustering framework mainly includes two objective functions, which are Eq.(12) and Eq.(16). Eq.(12) is designed to obtain the common subspace and the *weighted cluster indicator* matrix  $L_s$ ,  $s = 1, 2, \dots, P$ . The optimization of Eq.(12) is achieved by both Eq. (17) and Eq. (23). In Eq.(16), besides the cost function of the standard  $k$ -means algorithm, the *transferred centroid regularization* is taken into consideration. The cluster indicator matrix  $Q_T$  of the target domain can be derived through the standard  $k$ -means optimization by using Eq.(16). After obtaining  $Q_T$ , the LDA criterion is employed to select the most discriminative subspace. The optimizations for Eq.(12), Eq. (23) and Eq.(16) are adaptively performed until the convergence is reached. Here, we define a function, named *IsConvergent*( $\rho$ ,  $l$ ), to determine whether the convergence condition  $|\rho_l - \rho_{l-1}|/|\rho_{l-1}| < \epsilon$  holds, where  $l$  is the iteration number,  $\rho_l$  is the value of the objective function in the  $l$ -th iteration, and  $\epsilon$  is set to be 0.05. The main steps of clustering framework are presented in Algorithm 1. Although the whole framework is optimized by iterative steps, the empirical result shows it converges fast and iteration times for  $t_1$ ,  $t_2$  and  $t$  are less than five in most cases.

## 4 Evaluation by Experiments

### 4.1 Experiments Setup

We performed experiments on the 20 Newsgroups corpus<sup>1</sup>, which consists of approximately 20000 newsgroup documents collected evenly from 20 different newsgroups. The documents from some different newsgroups are related. For example, the newsgroups *rec.sport.baseball* and *rec.sport.hockey* are relevant to recreation. According to the typical setting of transfer learning, we employed the strategy from [6] to construct the data sets in the following way. First, we applied the typical pre-processing steps [19]: (1) removed stop words; (2) ignored file headers; (3) selected the top words by mutual information. In our experiments, we selected the top 100 words by ranking the mutual information value of each word. For each domain, we then randomly selected 200 documents for a given upper level category, i.e., comp, rec, sci, and talk. Specific details of the data sets are described in Table 1. We denote the data set NG1- $i$  ( $i = 1, 2, 3$ ) by the data set where the  $i$ -th domain from NG1 is regarded as the target domain and the remaining domains are the source domains. This specification also applies to the data set NG2- $i$  ( $i = 1, 2, 3$ ).

In our experiments, we compare SPCTL with the following typical methods:  $k$ -means (KM), PCA+KM, SDR [24]+KM, RCA [1]+KM, SCREEN [20]. We also use two metrics, the clustering accuracy (AC) [23] and F1-Score [6], to measure the clustering performance. Since SDR+KM, RCA+KM and SCREEN belong to the semi-supervised methods exploiting the pairwise constraints, for these three methods, we simply combine the data from all the domains together

<sup>1</sup> <http://people.csail.mit.edu/jrennie/20Newsgroups/>



---

**Algorithm 1**

---

**Input:**  $D_s, D_T, \mathbf{M}^s, \mathbf{C}^s, c$  ( $s = 1, 2, \dots, P$ )**Output:**  $W, L_s, Q_T$ 

```

1: Initialize the transformation matrix  $W$  by using PCA for the data of the target
   domain
2: Initialize the indicator matrix  $L_s$  by  $k$ -means on the data for each source domain
3: Set  $t_1 = t_2 = t = 1$ 
4: while  $IsConvergent(H, t)$  and  $t \leq T$  do
5:   while  $IsConvergent(J, t_1)$  and  $t_1 \leq T_1$  do
6:     for  $s = 1$  to  $P$  do
7:       Update  $L_s$  by using Eq. (22)
8:     end for
9:     while  $IsConvergent(J_W, t_2)$  and  $t_2 \leq T_2$  do
10:      Update  $W$  by using Eq. (24)
11:      Take QR decomposition for  $W$ 
12:       $t_2 = t_2 + 1$ 
13:    end while
14:     $t_1 = t_1 + 1$ 
15:  end while
16:  Use the  $k$ -means criterion to optimize Eq. (16) and obtain  $Q_T$ 
17:  Exploit LDA criterion to select  $W$  by using  $Q_T$ 
18:   $t = t + 1$ 
19: end while

```

---

and the pairwise constraints from all the source domains. However, we only consider the data from target domain when computing ACC and F1. For KM and PCA+KM, we directly applied them to the target domain. For each data set, we repeated the experiments for 10 trails, and report the averages.

The parameter setting of SPCTL is listed as follows. The Gaussian kernel parameter  $\gamma$ , which is used to construct the *transferred centroid regularization*, is set to be 10. Since our method is iterative, we prescribe the maximum number of iterations as  $T = T_1 = T_2 = 10$ . The parameters  $\vartheta$  and  $\theta$ , which are used in Eq. (8) to code the information of pairwise constraints, are both set to be 0.5. The step size  $\nu$  for updating  $W$  is set to be 0.01. The balancing parameters  $\alpha$ ,  $\beta$ , and  $\lambda$  are set by searching the grid  $\{0, 10^0, 10^1, 10^2, 10^3\}$ . In order to compare methods fairly, for the parameter settings of other competitive methods, we follow the parameters recommended by them, which are considered to be optimal. Without specific explanation, each source domain has the same number of pairwise constraints. The number of must-link constraints is always set to be equal to that of cannot-link constraints. Must-link constraints and cannot-link constraints are randomly selected according to the ground-truth of class labels.

## 4.2 Analysis of Experiments

In this subsection, we show the effectiveness of SPCTL. First, our performance is exhibited by varying the number of constraints when the reduced dimension is fixed. The reduced dimension for all the data sets is set to be 50. As illustrated

**Table 1.** Statistics for our data sets from Newsgroup corpus

Dataset	Domain	Class label				#doc
		comp	rec	sci	talk	
NG1	1	graphics	autos	crypt	N/A	600
	2	os.ms-windows.misc	motorcycles	electronics	N/A	600
	3	sys.ibm.pc.hardware	sport.baseball	med	N/A	600
	4	sys.mac.hardware	sport.hockey	space	N/A	600
NG2	1	ibm.pc.hardware	motorcycles	electronics	politics.guns	800
	2	sys.mac.hardware	sport.baseball	med	politics.mideast	800
	3	os.ms-windows.misc	sport.hockey	crypt	politics.misc	800

in Fig. 2, in most cases, SPCTL keeps the best performance when the number of available constraints from each source domain increases from 10 to 100. The performances of the three semi-supervised methods are inferior to that of SPCTL. A possible reason is the direct use of data and pairwise constraints from source domains without considering the distribution difference would not guarantee a good performance. As shown in Fig. 2c and Fig. 2d, the performances of SCREEN are fluctuated. The reason is probably that the dimension reduction step of SCREEN depends only on the new generated cannot-link constraints. Because the pairwise constraints are randomly selected, the new generated cannot-link constraints are more likely to be affected by the random selection, so that the subspace is largely influenced by the randomness of the new generated cannot-link constraints. It can be seen from Fig. 2g and Fig. 2h that SPCTL is unable to outperform others. The most probable reason is that SPCTL fails to obtain the optimal common subspace, so that the *transferred centroid regularization* behaves inappropriately.

Second, we show the performance of SPCTL when the reduced dimension varies, i.e., 80, 40, 20, and 10, and the average performances over the four dimensions are also presented. In this experiment, we set the number of pairwise constraints to be 100 for each source domain. As shown in Fig. 3, it can be seen that, in most cases, the average performances of SPCTL over different reduced dimensions are kept the highest, indicating that SPCTL is robust against the change of the reduced dimensions. In addition, although PCA+KM is slightly better than the standard  $k$ -means, PCA+KM is in general not competitive, probably because the constraint knowledge is not used. We also observe that, in most cases, the average performances of RCA+KM and SDR+KM are better than those of PCA+KM. The possible reason is, although the pairwise constraints come from different domains, they are still informative in the subspace selection. The reason that RCA+KM and SDR+KM are inferior to SCREEN is probably that they were not proposed for a clustering task. It is not surprising that SPCTL outperforms SCREEN in most cases. It consists in the fact that they behave in a different way of exploiting constraint knowledge from multiple source domains. This validates our conjecture that transferring constraint knowledge by using the *transferred centroid regularization* is more effective than the way of simply combining the pairwise constraints.

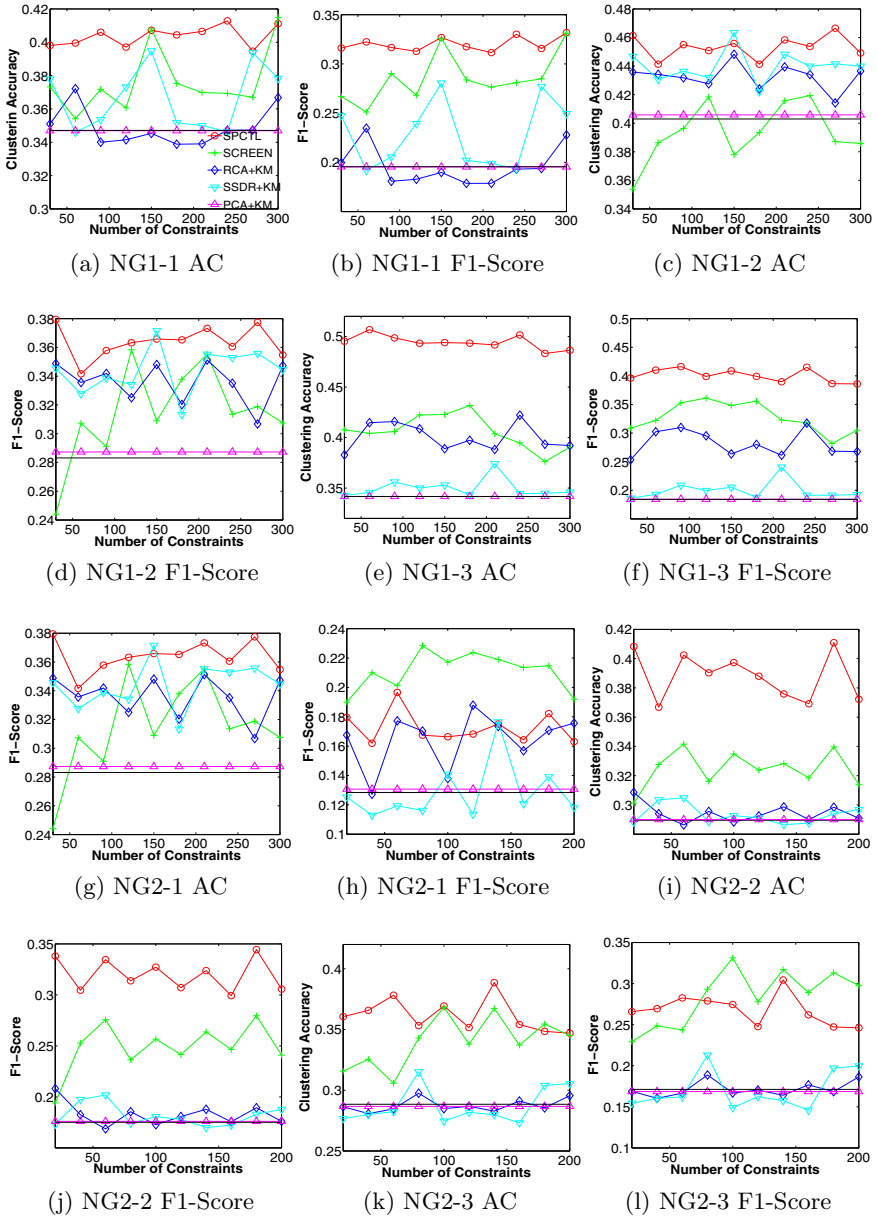


Fig. 2. The performance with different numbers of constraints

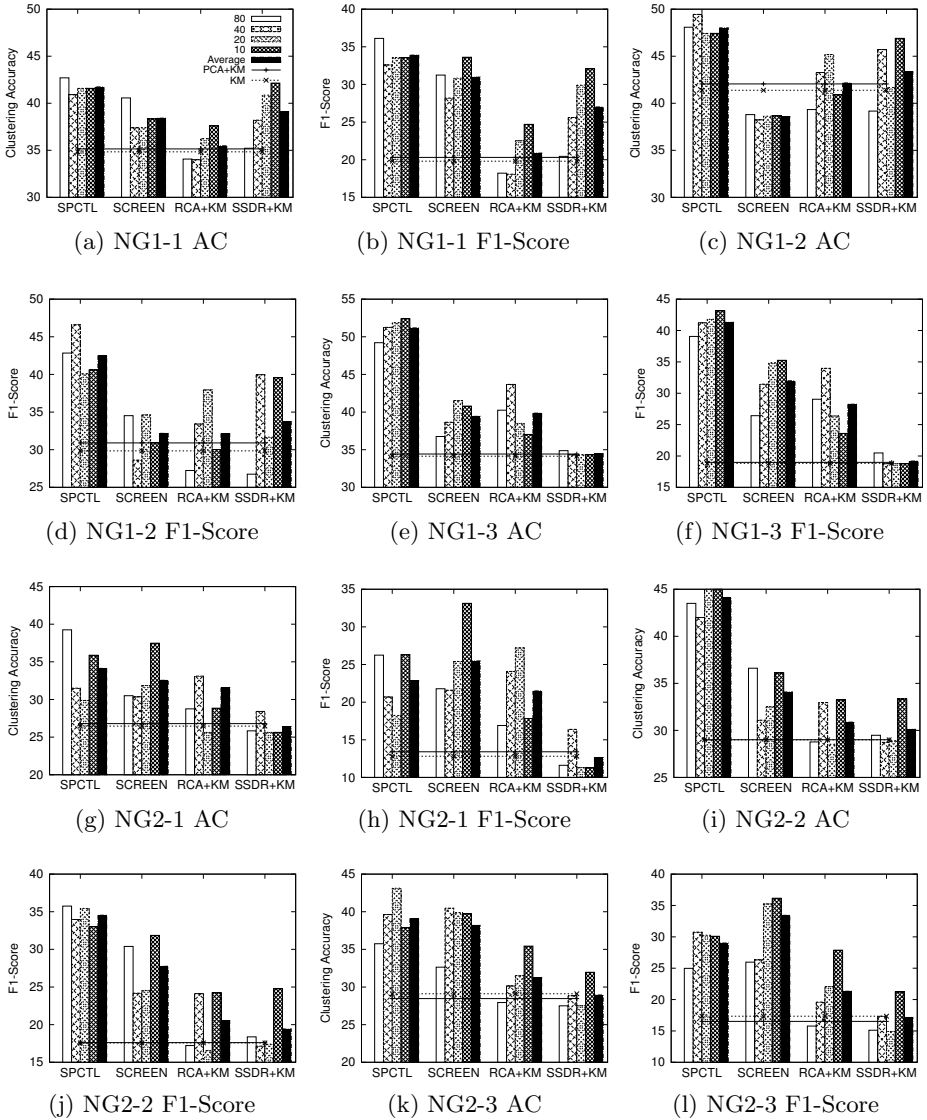


Fig. 3. The performance with different dimensions

## 5 Related Works

In this section, we review past research works related to our work, including semi-supervised clustering, domain adaptation and transfer clustering.

Semi-supervised clustering can be traced back to [21], where pairwise constraints were employed to aid the unsupervised clustering. Semi-supervised clustering for high dimensional data can be implemented by metric-based methods

[20,11,24]. However, semi-supervised clustering assumes the data are drawn from the same domain.

Domain adaptation is aimed to obtain a good feature representation that is able to reduce the difference in distributions between domains. Domain adaptation is one of the most significant issues in transfer learning, because different domains would probably cause the *negative transfer* [17]. Blitzer et al. [4] proposed a Structural Correspondence Learning (SCL) method to select domain independent pivot features that appear frequently in different domains, so that the same feature structure can be shared. Recently, Pan et al. [15] invented a new dimension reduction method, named MMDE, for domain adaptation by exploiting the Maximum Mean Discrepancy (MMD) criterion. MMDE is extended in [16] to the out-of-sample case, and reduced the computational time. However, in these two methods, the final classification tasks were independent from the domain adaptation. Chen et al. [6] employed the domain adaptation method to extend a classification framework for transfer learning. It is different from our task setting as our task is for clustering.

Our work is closely related to the transfer clustering of which goal is to improve the clustering performance of the target domain. Dai et al. [7] proposed a Self-taught Clustering (STC) method that aims at clustering a small collection of target unlabeled data with the help of a large amount of auxiliary unlabeled data from source domains. Bhattacharya et al. [3] invented a cross-guided clustering method, in which the clustering partitions of the source domains are manually prescribed. The cross-domain distance measure is then designed to aid the alignment for centroids across domains. Unlike these methods, the input of our transfer clustering task includes pairwise constraints.

## 6 Conclusions and Future Works

In this paper, we investigated a new task of handling the semi-supervised projection clustering in the transfer learning setting. In this task, we carefully designed an iterative framework to obtain a common subspace where the difference in distributions among domains is reduced. Inspired by the specific structure formed by the centroids in the common subspace, we transfer the constraint knowledge to the target domain by inventing the *transferred centroid regularization*, such that the clustering performance of the target domain in the common subspace can be enhanced. The experimental results show that our method is effective.

Although we have obtained the common subspace where the distribution differences between each source domain and the target domain are reduced, the analysis to the similarities between source domains in the common subspace is not sufficiently discussed. Therefore, in the future work, we are interested in incorporating the similarities between source domains into our framework.

**Acknowledgments.** This work is partially supported by the grant-in-aid for scientific research on fundamental research (B) 21300053 from the Japanese Ministry of Education, Culture, Sports, Science and Technology. Bin Tong and Hao Shao are sponsored by the China Scholarship Council (CSC).

## References

1. Bar-Hillel, A., Hertz, T., Shental, N., Weinshall, D.: Learning a Mahalanobis Metric from Equivalence Constraints. *J. Mach. Learn. Res.* 6, 937–965 (2005)
2. Belkin, M., Niyogi, P., Sindhvani, V.: Manifold Regularization: A Geometric Framework for Learning from Labeled and Unlabeled Examples. *J. Mach. Learn. Res.* 7, 2399–2434 (2006)
3. Bhattacharya, I., Godbole, S., Joshi, S., Verma, A.: Cross-Guided Clustering: Transfer of Relevant Supervision across Domains for Improved Clustering. In: *ICDM*, pp. 41–50 (2009)
4. Blitzer, J., McDonald, R., Pereira, F.: Domain Adaptation with Structural Correspondence Learning. In: *EMNLP*, pp. 120–128 (2006)
5. Boyd, S., Vandenberghe, L.: *Convex Optimization*. Cambridge University Press, Cambridge (March 2004)
6. Chen, B., Lam, W., Tsang, I., Wong, T.L.: Extracting Discriminative Concepts for Domain Adaptation in Text Mining. In: *KDD*, pp. 179–188 (2009)
7. Dai, W., Yang, Q., Xue, G.R., Yu, Y.: Self-taught Clustering. In: *ICML*, pp. 200–207 (2008)
8. Ding, C., He, X., Simon, H.D.: On the Equivalence of Nonnegative Matrix Factorization and Spectral Clustering. In: *SDM* (2005)
9. Ding, C., Li, T., Jordan, M.I.: Convex and Semi-Nonnegative Matrix Factorizations. *IEEE Trans. PAMI* 32, 45–55 (2010)
10. Gu, Q., Zhou, J.: Learning the Shared Subspace for Multi-task Clustering and Transductive Transfer Classification. In: *ICDM*, pp. 159–168 (2009)
11. Hinton, G.E., Sejnowski, T.J.: Learning and Relearning in Boltzmann Machines, pp. 282–317 (1986)
12. Kulis, B., Basu, S., Dhillon, I., Mooney, R.: Semi-supervised Graph Clustering: A Kernel Approach. In: *ICML*, pp. 457–464 (2005)
13. Ling, X., Dai, W., Xue, G.R., Yang, Q., Yu, Y.: Spectral Domain-transfer Learning. In: *KDD*, pp. 488–496 (2008)
14. Liu, Q., Liao, X., Carin, H.L., Stack, J.R., Carin, L.: Semisupervised Multitask Learning. *IEEE Trans. PAMI* 31(6), 1074–1086 (2009)
15. Pan, S.J., Kwok, J.T., Yang, Q.: Transfer Learning via Dimensionality Reduction. In: *AAAI*, pp. 677–682 (2008)
16. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain Adaptation via Transfer Component Analysis. In: *IJCAI*, pp. 1187–1192 (2009)
17. Pan, S.J., Yang, Q.: A Survey on Transfer Learning. *IEEE Trans. TKDE* 99 (2009)
18. Parsons, L., Haque, E., Liu, H.: Subspace Clustering for High Dimensional Data: A Review. *SIGKDD Explor. Newsl.* 6(1), 90–105 (2004)
19. Slonim, N., Tishby, N.: Document Clustering Using Word Clusters via the Information Bottleneck Method. In: *SIGIR*, pp. 208–215 (2000)
20. Tang, W., Xiong, H., Zhong, S., Wu, J.: Enhancing Semi-supervised Clustering: A Feature Projection Perspective. In: *KDD*, pp. 707–716 (2007)
21. Wagstaff, K., Cardie, C.: Clustering with Instance-level Constraints. In: *ICML*, pp. 1103–1110 (2000)
22. Ye, J., Zhao, Z., Liu, H.: Adaptive Distance Metric Learning for Clustering. In: *CVPR* (2007)
23. Ye, J., Zhao, Z., Wu, M.: Discriminative K-means for Clustering. In: *NIPS* (2007)
24. Zhang, D., Zhou, Z., Chen, S.: Semi-supervised Dimensionality Reduction. In: *SDM* (2007)

# Permutation Testing Improves Bayesian Network Learning

Ioannis Tsamardinos and Giorgos Borboudakis

Computer Science Department, University of Crete and  
Institute of Computer Science, Foundation for Research and Technology, Hellas

**Abstract.** We are taking a peek “under the hood” of constraint-based learning of graphical models such as Bayesian Networks. This mainstream approach to learning is founded on performing statistical tests of conditional independence. In all prior work however, the tests employed for categorical data are only asymptotically-correct, i.e., they converge to the exact  $p$ -value in the sample limit. In this paper we present, evaluate, and compare exact tests, based on standard, adjustable, and semi-parametric Monte-Carlo permutation testing procedures appropriate for small sample sizes. It is demonstrated that (a) permutation testing is calibrated, i.e. the actual Type I error matches the significance level  $\alpha$  set by the user; this is not the case with asymptotic tests, (b) permutation testing leads to more robust structural learning, and (c) permutation testing allows learning networks from multiple datasets sharing a common underlying structure but different distribution functions (e.g. continuous vs. discrete); we name this problem the *Bayesian Network Meta-Analysis* problem. In contrast, asymptotic tests may lead to erratic learning behavior in this task (error increasing with total sample-size). The semi-parametric permutation procedure we propose is a reasonable approximation of the basic procedure using 5000 permutations, while being only 10-20 times slower than the asymptotic tests for small sample sizes. Thus, this test should be practical in most graphical learning problems and could substitute asymptotic tests. The conclusions of our studies have ramifications for learning not only Bayesian Networks but other graphical models too and for related causal-based variable selection algorithms, such as HITON. The code is available at mensxmachina.org.

## 1 Introduction

Graphical models such as Bayesian Networks (BNs) are often at the heart of decision support systems and employed for prediction and diagnosis [4]. Graphical-model theory has also led to successful variable selection algorithms [3]. In addition, most causal discovery methods induce some type of a graphical model from data representing various types of causal relations among variables. This family of models includes, among others, (static, dynamic, and causal) Bayesian Networks (BNs) [13], Partially Directed Acyclic Graphs (PDAGs), Maximal Ancestral Graphs (MAGs), Partially Oriented Ancestral Graphs (PAGs) [12], and

Pairwise Causal Graphs (PCG) [16]. Often, these models represent a set of conditional dependencies and independencies that hold in the data distribution.

A major approach to learning such models from data is called the *constraint-based approach*: a number of tests of conditional independence are performed on the data whose results (dependence or independence) constrain the possible models fitting the data. A suitable test-strategy can then converge to a single model or all models equally fitting the data and thus are statistically indistinguishable (also called Markov Equivalent networks). Examples include the PC [13], the Fast Causal Inference (FCI) [13], and the recently introduced cSAT+ algorithm [16] that learn a PDAG, a PAG, and a PCG respectively. The constraint-based approach has also been employed for variable selection with excellent results [3].

In this paper, we take a closer look into the main operation that makes all of the above algorithm “tick”: the statistical test of conditional independence. We denote as  $T(X; Y|\mathbf{Z})$  the test of independence of variables  $X$  with  $Y$  given variables  $\mathbf{Z}$ . We denote by  $Ind(X; Y|\mathbf{Z})$  and  $Dep(X; Y|\mathbf{Z})$  the actual independence and dependence.  $T(X; Y|\mathbf{Z})$  returns a  $p$ -value denoted by  $p_{XY|\mathbf{Z}}$  corresponding to the probability of obtaining a test statistic equal to or more extreme than the observed test statistic on the data, given the hypothesis  $Ind(X; Y|\mathbf{Z})$  holds. Typically, given a significance level  $a$ , the algorithm rejects  $Ind(X; Y|\mathbf{Z})$  (accepts  $Dep(X; Y|\mathbf{Z})$ ) if  $p_{XY|\mathbf{Z}} \leq a$  and accepts  $Ind(X; Y|\mathbf{Z})$  otherwise.

While foundational, testing conditional independence in the context of graphical model learning has not been studied in depth. In particular, for nominal categorical data two tests have been employed, namely the Pearson’s  $\chi^2$  test and the likelihood ratio (LR) test [14,13,17]. Both of them are *asymptotic* tests, i.e., the returned  $p$ -value is approximate and converges to the true value in the sample limit. Statisticians have long warned that the approximation is often poor in many circumstances, particularly when the sample size is low or the probabilities of the distribution are extreme (close to 0 or 1). However, prior research has not been able to fully characterize these cases (see 9.8.4. [1]) to allow automatic detection of a poor approximation.

Ideally, one would prefer to use exact tests of independence. Unfortunately, in the general case such tests have a high computational overhead that prohibits their use in the context of learning large graphical models. In addition, they require highly specialized software that is often proprietary [10]. To address the problem, we advocate and study easy-to-implement, Monte-Carlo permutation tests. These tests are exact in the sense that  $E(\hat{p}) = p$ , where  $p$  is the true  $p$ -value of the test and  $\hat{p}$  the value returned by the test. We develop and compare three procedures (a) a standard Monte-Carlo simulation, (b) an adjustable permutation method, and (c) a semi-parametric permutation method. Similar techniques have been successfully developed and employed for attribute selection and pruning in Decision Trees [5], relation learning settings [11,9], rule learning and model selection [8]. We demonstrate empirically that in terms of efficiency:

- The adjustable and the semi-parametric procedures require on average about 450 and 100 permutations respectively to achieve the same learning performance in BNs as the basic procedure using 5000 permutations.



- The semi-parametric procedure is only 10 to 20 times slower than an asymptotic test for small samples sizes ( $< 500$ ).

The efficiency results show that simple permutation procedures could replace the asymptotic tests without a prohibitive efficiency cost in many practical situations. In terms of the learning-performance benefits, we show that:

- Permutation procedures are more effective in distinguishing between dependence and independence than the asymptotic tests for small sample sizes.
- Permutation procedures are calibrated, i.e, their actual Type I error matches the significance level  $\alpha$ ; this is not the case with the asymptotic tests.
- Permutation procedures lead to more robust BN structural learning.
- Perhaps most importantly, exact tests allow the development of algorithms that are able to learn from multiple datasets non-identically distributed. In contrast, asymptotic tests with the heuristics lead to erratic behavior where the error rate increases as the available data increase.

We name the problem of learning a BN from multiple datasets assumed to be sharing the same structure, defined over the same variables, and obtained under similar experimental and sampling conditions *Bayesian Network Meta-Analysis* learning (BNMA). This is because it generalizes statistical meta-analysis: the latter aims to induce a single dependency relation from multiple similar studies, while the former aims to learning a complete BNs (a set of dependencies and independencies). This situation may occur when different studies measure the same variables but on different scales or using different methods or equipments. For example, in one study *Smoking* maybe taking a dichotomous No/Yes value, while in another the values No/Light/Regular/Heavy smoker. In different psychology studies the level of depression may be measured using different questionnaires and methods. In different gene-expression micro-array experiments, gene expression values from different experiments cannot be easily translated to a common scale for various technical reasons [7]. In all these cases, one could not pool all the data together in a single dataset without the risk of losing information or performing an inappropriate data transformation. Techniques for BNMA learning have been recently introduced independently by our group [18] and Tillman et. al. [14]. Permutation tests allow these techniques to be applicable in practical cases where each dataset has low sample size. We suspect exact testing to be important in other BN learning-related procedures that depend on the exact value of the  $p$ -values returned. Such a procedure is a technique for controlling the False Discovery Rate of the identified edges in a BN [19].

## 2 Asymptotic Tests of Independence

We now consider a test  $T(X; Y | \mathbf{Z})$  and denote by  $N_{xyz}$  the number of samples where  $X = x$ ,  $Y = y$ , and  $\mathbf{Z} = \mathbf{z}$  in the data, where  $\mathbf{z}$  is a vector of values of  $\mathbf{Z}$  in case there are more than one variable in the conditioning set. We denote with  $N_{x+\mathbf{z}} = \sum_y N_{xyz}$  and similarly for  $N_{+y\mathbf{z}}$ ,  $N_{xy+}$ ,  $N_{++\mathbf{z}}$ , and  $N_{+++} = N$  the total

sample size. Finally, we denote with  $|X|$  the size of the domain of variable  $X$  and with  $|\mathbf{Z}|$  the number of joint states of the variables in  $\mathbf{Z}$ . Assuming  $Ind(X; Y|\mathbf{Z})$  the expected number of samples where  $X = x, Y = y$ , and  $\mathbf{Z} = \mathbf{z}$  is:

$$E_{xyz} = \frac{N_{x+\mathbf{z}} \cdot N_{+y\mathbf{z}}}{N_{++\mathbf{z}}}$$

when the actual observed number is  $N_{xyz}$ . The overall discrepancy between these two quantities in all cells of the contingency tables is captured by the following two test statistics:

$$\chi^2 = \sum_{x,y,\mathbf{z}} \frac{(N_{xyz} - E_{xyz})^2}{E_{xyz}} \quad \mathcal{G} = 2 \sum_{x,y,\mathbf{z}} N_{xyz} \ln \frac{N_{xyz}}{E_{xyz}}$$

Whenever  $E_{xyz}$  is zero the corresponding term in the summation is defined as zero as well. Both of these statistics are *asymptotically* distributed as  $\chi^2_{df}$  with  $df = (|X|-1)(|Y|-1)|\mathbf{Z}|$  degrees of freedom. Using the  $\chi^2$  as a test statistic leads to the Pearson’s  $\chi^2$  test of independence, while using the  $\mathcal{G}$  leads to a likelihood ratio test of independence, also called a G-test [1]. The  $p_{XY|\mathbf{Z}}$  is calculated as  $1 - F(S_o)$  where  $F$  is the cumulative distribution function of  $\chi^2_{df}$  and  $S_o$  the observed value of the statistic (either the  $\chi^2$  or  $\mathcal{G}$ ) in the data.

Let us denote with  $m = \#\{N_{xyz}\}$  the number of counts to calculate. When all variables are ternary then for  $T(X; Y|Z_1, Z_2)$  we get  $m = 3^4$ . Thus, the average number of samples to estimate each count drops exponentially with the number of variables to condition. This reduces the statistical power and increases the number of cells with zero counts. In other words, with a large enough conditioning set, any  $T(X; Y|\mathbf{Z})$  will return a high  $p_{XY|\mathbf{Z}}$  with high probability leading the algorithm to accept  $Ind(X; Y|\mathbf{Z})$ .

Two heuristic solutions have appeared in the literature. First, some algorithms (PC, MMPC) [13,17] do not perform  $T(X; Y|\mathbf{Z})$  if they determine there are not enough samples to achieve large enough power. The algorithms require that the average sample per count  $N/m$  is at least  $\pi$ , where  $\pi$  is a user defined parameter. Typical values for  $\pi$  are 10 [13], 5 [17], 0 [15] (in chronological order). We call this the *heuristic power rule*.

Second, several of the zero counts  $N_{xyz} = 0$  that appear in the contingency tables are heuristically declared as “structural zeros”, i.e., they are judged to stem from structural constraints of the problem and not as random events. A structural zero for example, would appear if  $X$  is “taking contraceptives”,  $Y$  measures “osteoporosis” and  $Z$  is gender. We expect that  $N_{Yes,y, Male}$  to be zero. Since structural zeros are not free to vary, the degrees of freedom of the test should be adjusted. Spirtes et. al. [13] consider as structural any zero that appears in the contingency tables. They subtract one from  $df$  for every such zero, which may actually lead to negative degrees of freedom. In [17] we present a different heuristic where we consider as structural zero any case  $N_{xyz} = 0$  and also either of the marginals are  $N_{+y\mathbf{z}} = 0$  or  $N_{x+\mathbf{z}} = 0$ . For example, if  $N_{+y\mathbf{z}} = 0$ , then we consider  $y$  as a structurally forbidden value for  $Y$  when  $\mathbf{Z} = \mathbf{z}$  and we reduce the degrees of freedom by  $|X| - 1$  (as if we had one column less in

the contingency table where  $\mathbf{Z} = \mathbf{z}$ ). We call the latter method the *degrees of freedom adjustment* heuristic. There have been several pieces of work examining the appropriateness of the approximate tests [1] and several attempts and rules to characterize cases of poor approximation. However, a full characterization is still lacking. The alternative is to use exact tests of independence that is presented next.

### 3 Permutation Tests of Independence

The first exact test for categorical data to appear has been Fisher's exact test [1] that treats the special case in  $2 \times 2$  tables (i.e.,  $T(X; Y|\emptyset)$  with  $|X| = |Y| = 2$ ). However, generalizing exact testing in the general case of  $i \times j \times k$  (conditional test with unrestricted sizes of domains for all variables) has been proven a difficult task. A mainstream approach to exact testing is called the *exact conditional approach* that considers the row and column marginals in each table  $N_{x+\mathbf{z}}$ ,  $N_{+y\mathbf{z}}$ , and  $N_{++\mathbf{z}}$  as fixed [2,11,8]. The distribution of the test statistic under the null hypothesis is then calculated conditioned on these marginals. Specifically, to calculate the exact  $p$ -value one needs to calculate  $P(S_o \geq S|Ind(X; Y|\mathbf{Z}))$ , where  $S_o$  is the observed test statistic. This in turn implies identifying all contingency tables with the same marginals and whose test statistic is larger or equal to the observed one. The number of possible tables with the same marginals quickly explodes: "a  $4 \times 4$  table ... with a 100 observations can have about  $7 \times 10^9$  such tables" [2]. Various computational methods have appeared that attempt to do the computations implicitly without enumeration of all tables, some of which have led to the development of the StatXact package [10]. These methods however, are still relatively slow, hard to implement, and not freely available.

In this section, we present intuitive, easy-to-implement, and relatively efficient permutation testing procedures. Notice that, each table (where  $\mathbf{Z} = \mathbf{z}$ ) with the same marginals as the observed table can be produced by permuting the values of  $X$  or  $Y$  of the samples (while retaining  $\mathbf{Z} = \mathbf{z}$ ). For example, for binary variables  $X, Y, Z$ , suppose we have the observations  $\langle 0, 1, 0 \rangle$  and  $\langle 1, 0, 0 \rangle$  giving  $N_{0+0} = N_{1+0} = N_{+00} = N_{+10} = 1$ . Permuting the two values of  $Y$  between the only two observations provides the permuted data  $\langle 0, 0, 0 \rangle$  and  $\langle 1, 1, 0 \rangle$  with the same marginals. Under the null hypothesis of independence, this is justified as follows: since  $X$  and  $Y$  are assumed independent given  $\mathbf{Z}$ , any such permutation has the same probability of being observed.

Calculating all such possible permutations is equivalent to enumerating all possible tables with the same marginals. However, one can sample from the space of all possible permutations (tables) randomly to estimate  $\hat{P}(S_o \geq S|Ind(X; Y|\mathbf{Z}))$ . Such methods are called Monte Carlo Permutation methods [6]. We denote with  $D_0 = \{\langle x, y, \mathbf{z} \rangle\}_{j=1}^N$  the unpermuted, observed data. We obtain permuted data  $D_i, i > 0$  as follows: for each possible value  $\mathbf{z}$  of  $\mathbf{Z}$ , randomly permute the values of  $Y$  in  $D_0$  *only* among the cases where  $\mathbf{Z} = \mathbf{z}$  (i.e., ensuring all marginals remain the same). We denote with  $S(D_i)$  the value of the statistic (either  $\chi^2$  or the  $\mathcal{G}$  statistic) on the data  $D_i$ . The basic procedure is shown in Algorithm [1].

**Algorithm 1.** Basic Permutation  $T(X; Y|\mathbf{Z})$ 


---

**Input:** Data  $D_0 = \{(x, y, \mathbf{z})\}_{j=1}^N$ , Test Statistic  $S$ , number of permutations  $B$   
**Output:**  $\hat{p}_{XY|\mathbf{Z}}$

- 1 **for**  $i = 1, \dots, B$  **do**
- 2   | Randomly permute data to create  $D_i$ ; calculate  $S(D_i)$
- 3 **end**
- 4 **return**  $\hat{p}_{XY|\mathbf{Z}} = \#\{S(D_0) \leq S(D_i), i = 1, \dots, B\}/B$

---

The procedure requires the computation of  $S(D_i)$  an additional  $B$  times compared to the asymptotic test, so as given it is at least that many times slower. One obvious optimization to the procedure is to notice that the values  $E_{xy\mathbf{z}}$  depend only on the marginals that remain the same across all datasets (observed and permuted) and so can be computed only once.

A sufficient number of permutations  $B$  seems to range between 1000 to 5000 which makes the procedure quite costly for learning large graphical models (we used 5000 in our experiments). To improve the computational requirements, we design an adjustable procedure that may stop early the computation of more permuted statistics. The procedure infers whether the current approximation  $\hat{p}_{XY|\mathbf{Z}}$  is sufficiently close to the true  $p_{XY|\mathbf{Z}}$  to make a decision at significance level  $a$ , i.e., to determine whether  $p_{XY|\mathbf{Z}} \leq a$  or not.

First, we implement a heuristic rule based on asymptotic tests in an effort to completely avoid permutation testing in easy-to-determine cases. Assuming a typical range for the significance level to be between 0.01 and 0.1, then if a conservative asymptotic test (in our experiments the  $\mathcal{G}$  test) returns a relatively much lower  $p$ -value (lower than 0.001 in our experiments) we immediately accept dependence. Similarly, if a liberal asymptotic test ( $\mathcal{X}^2$  with the  $df$  heuristic adjustment) returns a high  $p$ -value (larger than 0.5), we accept independence.

**Rule 1 :** if  $p_{\mathcal{G}} < 0.001$  return Dep., else if  $p_{\mathcal{X}^2} > 0.5$  return Ind.

If the rule does not apply, we begin permutation testing. To bound the error of approximation at the current iteration  $b$ , we proceed as follows. We define a Bernoulli trial  $X_i = I(S(D_0) \leq S(D_i))$ , i.e., the event of a random permutation obtaining a larger statistic than the observed. The probability of success  $P(X_i = 1)$  is equal to the exact  $p$ -value of the test by definition. Thus,  $\sum_{i=1}^b X_i$  follows a Binomial( $B, p_{XY|\mathbf{Z}}$ ). For relatively large  $b$  and non-extreme  $p$ -values we can approximate this distribution with a normal distribution  $N(\mu', \sigma')$ , where  $\mu' = b \cdot p_{XY|\mathbf{Z}}$  and  $\sigma'^2 = b \cdot p_{XY|\mathbf{Z}} \cdot (1 - p_{XY|\mathbf{Z}})$ . Thus,  $\hat{p}_{XY|\mathbf{Z}} = \sum_{i=1}^b X_i/b$  follows  $N(\mu, \sigma)$ , where  $\mu = p_{XY|\mathbf{Z}}$  and  $\sigma^2 = p_{XY|\mathbf{Z}} \cdot (1 - p_{XY|\mathbf{Z}})/b$ . Based on this approximation, we find the confidence interval  $p_{XY|\mathbf{Z}} = \hat{p}_{XY|\mathbf{Z}} \pm \epsilon$ ,  $\epsilon > 0$ . The maximum magnitude of  $\epsilon$  called  $r(b)$ , where  $b$  is the current iteration, is obtained for  $\sigma^2 = 1/4$  and  $p$ -value=0.5 . Then, with probability  $\delta$  (exercise 3.45 at [II]):

$$r(b) = \frac{\Phi^{-1}\left(\frac{1+\delta}{2}\right)}{2 \cdot \sqrt{b}}$$

---

**Algorithm 2.** Adjustable Permutation  $T(X; Y|\mathbf{Z})$ 

---

**Input:** Data  $D_0 = \{(x, y, \mathbf{z})\}_{j=1}^N$ , Test Statistic  $S$ , maximum number of permutations  $B$ , significance level  $a$ **Output:** Independent/Dependent

```

1 Let  $\delta = 0.99$ 
2 Apply Rule 1
3 for  $b = 1, \dots, B$  do
4   | Randomly permute data to create  $D_b$ ; calculate  $S(D_b)$ 
5   | Apply Rule 2
6   | Apply Rule 3
7 end
8 if  $\hat{p}_{XY|\mathbf{Z}} > a$  then
9   | return Independent
10 else
11   | return Dependent
12 end

```

---



---

**Algorithm 3.** Semi-Parametric (fitted) Permutation  $T(X; Y|\mathbf{Z})$ 

---

**Input:** Data  $D_0 = \{(x, y, \mathbf{z})\}_{j=1}^N$ , Test Statistic  $S$ , number of permutations  $B$ **Output:**  $\hat{p}_{XY|\mathbf{Z}}$ 

```

1 for  $i = 1, \dots, B$  do
2   | Randomly permute data to create  $D_i$ ; calculate  $S(D_i)$ 
3 end
4  $df = \overline{S(D_i)}$ 
5 return  $\hat{p}_{XY|\mathbf{Z}} = 1 - F(S(D_0))$ , where  $F$  is the cumulative distribution of  $\chi_{df}^2$ 

```

---

The adjustable procedure periodically checks whether the following rule applies to the current approximation of the  $p$ -value  $\hat{p}_{XY|\mathbf{Z}}(b)$  at iteration  $b$ :

**Rule 2:** if  $\hat{p}_{XY|\mathbf{Z}}(b) + r(b) \leq a$  return Dep., else if  $a \leq \hat{p}_{XY|\mathbf{Z}}(b) - r(b)$  return Ind.

We additionally implement an idea described in [6] that prematurely aborts further permutations based on worst-case reasoning. Specifically, if assuming all the remaining permutations give  $S(D_0) \leq S(D_i)$  and our estimate  $\hat{p}_{XY|\mathbf{Z}}$  would still be greater than  $a$ , we can immediately determine independence. Similarly, if assuming all remaining permutations turn out  $S(D_0) > S(D_i)$  and our  $\hat{p}_{XY|\mathbf{Z}}$  would still be less than  $a$ , we can immediately determine dependence. Given these observations, a lower bound  $LB$  of  $\hat{p}_{XY|\mathbf{Z}}$  at any iteration  $b$  during the algorithm is:  $LB(b) = \#\{S(D_0) \leq S(D_i), i = 1, \dots, b\} / B$  where  $B$  is the maximum allowed number of permutations. Similarly, we find an upper bound  $UB$  as  $UB(b) = 1 - \#\{S(D_0) > S(D_i), i = 1, \dots, b\} / B$  and so, the final early-stopping rule implemented is:

**Rule 3 :** if  $UB(b) < a$  return Dep., else if  $LB(b) > a$  return Ind.

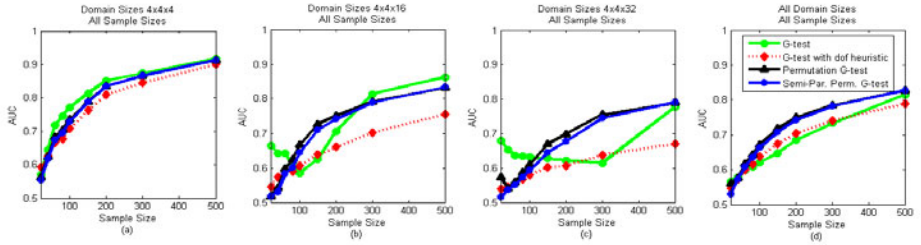
We formalize the procedure in Algorithm 2. Finally, we consider a semi-parametric approach often used in resampling (permutation and bootstrapping) methods to reduce the number of permutations required. When the distribution of the test statistic is suspected to follow a specific parametric form of unknown parameters it is not necessary to perform a larger number of permutations to identify the shape of the distribution. Only a relatively small number of permutations are performed to estimate the parameters. In our case, we assume that the distribution is well-approximated by a  $\chi_{df}^2$  distribution with a single unknown parameter, the degrees of freedom  $df$ . In preliminary experiments we determine that a reasonable number of permutations required to estimate  $df$  is about 100. The maximum likelihood estimate of  $df$  given samples from a  $\chi_{df}^2$  is the sample mean. Once the  $df$  has been approximated, the test calculates the  $p$ -value as in the asymptotic tests. The procedure is presented in Algorithm 3.

## 4 Distinguishing between Dependence and Independence

In this section we empirically evaluate the ability of the tests to distinguish between dependence and independence situations. We consider the following two network structures  $A \leftarrow C \rightarrow B$  and  $A \rightarrow C \leftarrow B$  because they are heavily involved in the basic reasoning operations performed by typical network algorithms such as the PC. In the former structure it holds that  $Dep(A; B|\emptyset)$  and  $Ind(A; B|C)$ . The latter independence would lead network algorithms to remove the edge  $A - B$  in the graph. For the latter structure it holds that  $Ind(A; B|\emptyset)$  and  $Dep(A; B|C)$ . This subgraph is called a v-structure [13] and allows network algorithms to orient edges. For each structure we perform the unconditional test  $T(A; B|\emptyset)$  and the conditional  $T(A; B|C)$ . We consider the following cases for the sizes of the domains of the variables  $A$ ,  $B$ , and  $C$  respectively:  $2 \times 2 \times k$ ,  $k = 2, 4, 8$  and  $4 \times 4 \times k$ ,  $k = 4, 16, 32$ . For each such case, we randomly sample with uniform probability the parameters of the network 20 times. For each parametrization of the networks and for each sample size in the set  $\{20, 40, 60, 80, 100, 150, 200, 300, 500\}$  we randomly create 25 datasets from the distribution of the network. The total number of tests to perform is  $2$  (networks)  $\times 2$  (types of tests)  $\times 6$  (domain definitions)  $\times 9$  (sample sizes)  $\times 20$  (parameterizations)  $\times 25$  samplings = 108K tests per method to evaluate.

The methods under study are: asymptotic tests with and without the heuristic adjustment based on the  $G$  and the  $\mathcal{X}^2$  test statistic, denoted by AG, AX, AGh, and AXh, A denoting an asymptotic test, the second letter referring to the test statistic used, and h referring to the employment of the heuristic; the basic and the fitted permutations based on the two statistics, denoted with PGB, PXB, PGF, PXF, P denoting a permutation based test, the second referring to the test statistic, and the last letter to the basic or fitted procedure. The adjustable procedure is not included in this set of experiments because it returns a binary decision, not a  $p$ -value. Thus, there is a total of 8 methods to evaluate.

The hardest cases (smaller ratio of samples over counts-to-estimate) are summarized in Figure 1(a)-(c) that show the AUCs over sample size of all datasets



**Fig. 1. Distinguishing between  $Ind(X; Y|Z)$  and  $Dep(X; Y|Z)$ .** (a)-(c) AUC with sample size for the tests based on the  $G$  statistic for the  $4 \times 4 \times k$  cases. The asymptotic tests have an erratic behavior for the more difficult cases. (d) AUC over sample size for all experiments. The permutation-based tests show statistically significantly ( $p < 0.0001$ ) increased AUC ranging between 1.75% and 3.39% according to the 95% confidence intervals (Table II).

where the domain counts  $4 \times 4 \times k$ ,  $k = 4, 16, 32$ . The AUCs were calculated as follows: the  $p$ -values returned on all tests performed by a method on a group of tests were ordered and thresholded by the significance level. Taking all possible thresholds gives the ROC curve and the AUC for that method on that group of tests. To avoid clutter in the figures we only present the tests based on the  $G$  statistic; the results are similar for the  $\chi^2$ . In figure (a) all tests have a reasonable behavior with increasing performance as sample size increases. As  $k$  increases and the sample is split to 16 and then 32 contingency tables (figures (b) and (c)) the behavior of the asymptotic tests becomes erratic and large dips in the curves appear. For the  $4 \times 4 \times 32$  case there are 512 counts  $N_{xyz}$  so in the best case on average there is about 1 sample to estimate each count and a large number of expected zero counts. The asymptotic approximations of the tests fail in these cases while the permutation-based procedures are quite robust. Figure II(d) shows the overall AUCs on all datasets; this includes the easier cases of  $2 \times 2 \times k$ ,  $k = 2, 4, 8$  so the average behavior is smoother for all tests. Finally, the AUCs of all methods are presented at Table II. We also note that, the permutation procedures do not depend as much on the choice of the test statistic (figures omitted for brevity).

Note that, the permutation methods are worse than the asymptotic methods for sample sizes 20-60 and  $k = 16, 32$ . This is because the number of possible permutations (that maintain the marginal counts) is too low to estimate the distribution of the test statistic. When sample size increases, the number of admissible permutations grows exponentially and the methods quickly improve over the parametric ones. E.g. the percentage of unique values of the statistic in 10000 permutations for the  $4 \times 4 \times 32$  case and sample size 20, 40, 100, 150 turns out to be on average 0.01%, 0.06%, 3%, and 12% respectively.

To compare whether the differences between the methods are statistically significant, we used (what else?) a permutation testing procedure. We define  $AUC_m$  the AUC returned by a method  $m$  on all results and define the statistic  $\Sigma_m = AUC_{PGF} - AUC_m$ . To generate a single round of permuted data, we

**Table 1.** AUC: The Area Under the ROC Curve of the testing procedures over all networks, domain sizes, sample sizes, and samplings. CI(%): 95% confidence interval of the difference with PGF  $\Sigma_m$  times 100.

	Asymptotic Tests				Permutation Tests			
	AG	AX	AGh	AXh	PGB	PXB	PGF	PXF
AUC	0.6432	0.6197	0.6548	0.5918	0.6766	0.6749	0.6744	0.6718
CI(%)	[2.85 3.39]	[5.18 5.76]	[1.75 2.16]	[7.91 8.61]	[-0.34 -0.10]	[-0.19 0.07]	0	[0.037 0.16]

permute each pair of corresponding  $p$ -values (whether they come from PGF or  $m$ ) with 50% probability. We estimate the empirical distribution of  $\Sigma_m$  using 10000 such rounds. The PGF test is statistically significantly better ( $\Sigma_m > 0$ ) than all asymptotic tests ( $p < 0.0001$ ) and than PXF ( $p < 0.0001$ ). However, the PGF performs worse than the PGB ( $p < 0.0001$ ) and PXB ( $p = 0.19$ ). To estimate the performance difference between PGF and the other tests, we present the 95% confidence intervals for  $\Sigma_m$  in Table 1. The improvements range from at least 1.75% improvement in AUC over the AGh, to at least 7.91% over the AXh. Based on these results, we forgo the use of tests based on the  $\chi^2$  statistic in the rest of the paper.

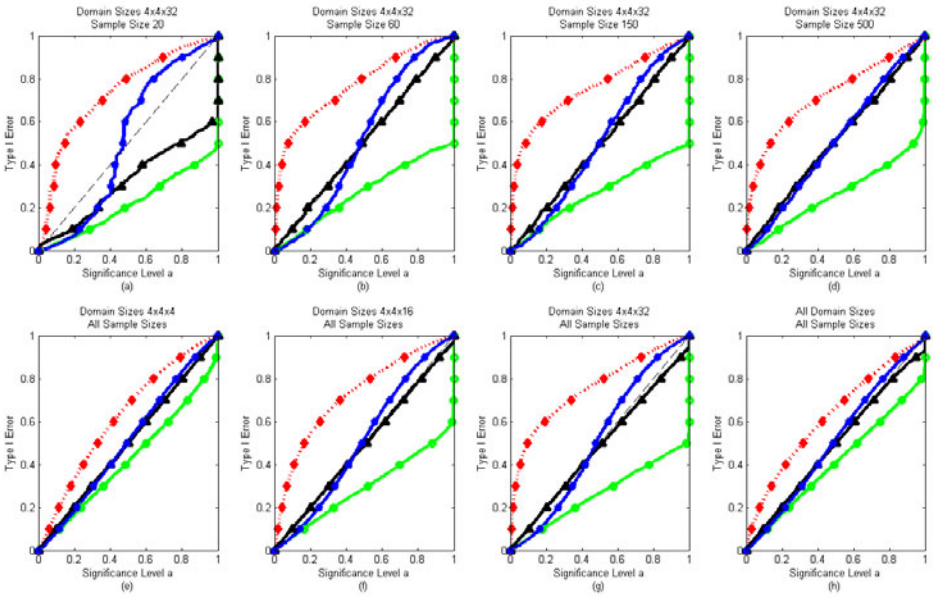
In terms of execution time, the asymptotic tests require about the same time given that the d.f. adjusting heuristic takes time linear to the number of counts computed. The PXB and PGB optimized versions take approximately 500-700 times more than the asymptotic procedures for the sample and domain sizes used in the study. Finally, the PGF takes about only 10-20 times more than the asymptotic tests. Given that the difference in performance between PGB and PGF is less than 0.4% AUC we consider PGF a good trade-off between performance and computational effort spent.

## 5 Evaluating the Calibration of the Tests

We now evaluate the relation between the actual Type I error when rejecting the null hypothesis at confidence level  $a$  ( $P(p \leq a | \text{null})$ ), where  $p$  is the  $p$ -value returned by the test, and the level  $a$ . For a calibrated test these two quantities should be equal, i.e.,  $P(p \leq a | \text{null}) = a$ . Notice that, a test may not be calibrated (e.g., always return half the true  $p$ -value) and still achieve a high AUC if the cases of  $Dep(X; Y | \mathbf{Z})$  have a lower  $p$ -value than the cases of  $Ind(X; Y | \mathbf{Z})$ .

Figure 2 shows  $P(p \leq a | \text{null})$  (Type I error) vs. the confidence level  $a$ .  $P(p \leq a | \text{null})$  is estimated as the proportion of cases where  $p \leq a$  in datasets where  $Ind(X; Y | \mathbf{Z})$  holds. Figure 2(a)-(d) focus on the hardest case of  $4 \times 4 \times 32$  experiments. With sample size 20 all tests return non-calibrated  $p$ -values. At sample size 60 the basic permutation procedure is already well-calibrated. At sample size 150 the semi-parametric procedure catches up. The asymptotic tests fail miserably to return calibrated values even for the largest sample size attempted of 500 cases. In the second row, Figures 2(e)-(g) show the results for the  $4 \times 4 \times k$  domain sizes averaged out on all sample sizes. The last sub-figure shows the average overall behavior including the easier cases of  $2 \times 2 \times k$ . The results for



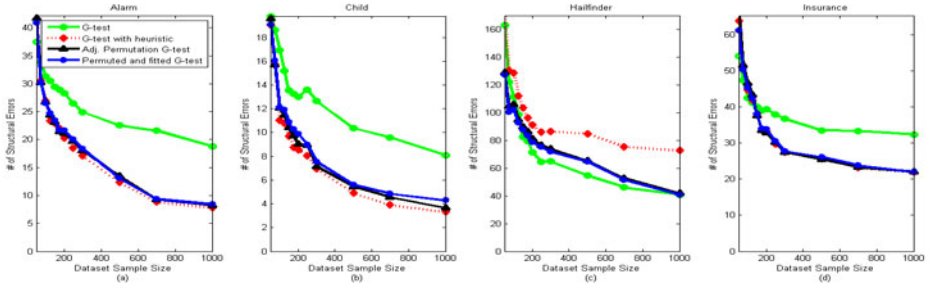


**Fig. 2. Calibration Properties of Tests.** The subfigures plot the Type I error ( $P(p \leq a | \text{null})$ ) vs. the significance level  $a$  for various cases. The legend is the same as in Figure 1. The asymptotic  $G$  test with the heuristic (red line) underestimates the true  $p$ -value; the asymptotic  $G$  test (green line) overestimates it. The permutation procedures are relatively well-calibrated.

the tests based on the  $\chi^2$  statistic lead to similar conclusions, not shown due to lack of space. In conclusion, when independence holds *the asymptotic  $G$  test with the heuristic underestimates the true  $p$ -value, while the asymptotic counterpart without the heuristic adjustment overestimates it. The permutation procedures are well-calibrated for the complete range of the significance levels.*

## 6 Improving Bayesian Network Learning

We now examine whether the improvements in AUC and calibration of the permutation procedures translate to improved learning rates and robustness. We consider four typical Bayesian Networks in the literature used in decision support systems, namely the ALARM, Insurance, Hailfinder, and Child [17]. These have 37, 27, 56, 20 variables and 46, 52, 66, 25 edges respectively. From the distribution of each network and for each sample size in  $\{50, 75, 100, 125, 150, 175, 200, 250, 300, 500, 700, 1000, 2500, 5000\}$  we sample 20 times simulated data. The prototypical PC algorithm [13] is then employed to attempt to reconstruct the network from the data. We used our implementation of the PC algorithm with one main difference from the original version: the algorithm begins from the empty graph and not the complete one, then adds the edges corresponding to detectable pairwise associations (see [17] for a detailed justification).



**Fig. 3. Learning the Skeleton of Bayesian Networks.** The number of structural errors (extra and omitted edges) is presented for each network and procedure. The performance of the asymptotic procedures highly depends on the network. Permutation-based method robustly perform in all networks and sample sizes.

The PC algorithm is executed with different testing procedures to evaluate their efficacy. PC accepts two parameters, the significance level  $\alpha$  for the tests and the threshold  $\pi$  in the heuristic power rule that determines which tests to omit. The significance threshold used in all cases is the standard 0.05. In our experiments  $\pi = 2$  selected as the value of  $\pi \in \{0, \dots, 15\}$  that optimizes the performance of the asymptotic tests.

Based on the results of Section 4, we focus on tests based only on the  $G$  statistic. In addition, we exclude the basic permutation procedure from the evaluation as having a large computational overhead. Instead, we employ the adjustable permutation procedure (PGA) defined in Algorithm 2. In preliminary experiments (not shown) this algorithm was found to be a good approximation of the basic permutation procedure. Thus, overall there are four tests in the evaluation: AG, AGh, PGF, and the PGA.

The structural errors are defined as the number of extra and omitted edges and are shown in Figure 3. The adjustable and semi-parametric permutation procedures are always close to the maximum accuracy achieved by any method in all four networks. The permutation-based methods perform similarly in terms of structural errors. In terms of computations, the number of permutations required for the adjustable procedure is on average about 450 while the semi-parametric procedure performs a fixed number of only 100 permutations. *The semi-parametric permutation test performs surprisingly well; the results show that permutation-based tests can be practical for graphical model learning.*

When it comes to the asymptotic tests, it is easy to observe that their performance highly depends on the network structure. The AGh test outperforms the AG test in three out of the four networks. However, the reverse relation holds in the Hailfinder network. This is explained due to the existence of several variables with a large domain and many adjacencies: AGh in general underestimates the true p-value and so the large-domain variables initially obtain many adjacencies at the first iteration of the algorithm. Once this happens, it becomes impossible to perform tests involving these variables due to the power rule, which eventually leads to a large number of false positives. The AG tests never includes edges

with these variables and so there are many fewer false positives. E.g., for sample size 500 on Hailfinder the average number of structural errors for nodes 3 and 40 (domain size 11) and its neighbors (19 nodes) is 61 and 31 for the AGh and the AG respectively. For all other 37 nodes it is 23 and 23.

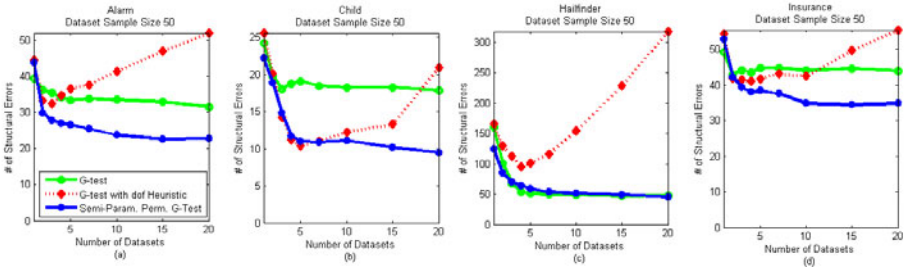
In contrast, the permutation procedures are highly robust over all networks. Over all networks and sample sizes, the PGF procedure statistically significantly outperforms the asymptotic ones (single-sided, 10000 permutations  $p < 0.0001$ ) in terms of the number of structural errors. However, we would like to note that the learning performance of an algorithm depends on the sparseness of the graph (ratio between edges and non-edges) and the cost associated with a false positive (extra edge) and a false negative (omitted edges). The number of structural errors in particular penalizes both types of errors by the same amount. A more complete evaluation should optimize the threshold  $a$  of a test relative to the performance measure. Given this discussion, *the important conclusion from this set of experiments in our opinion is the robustness of the permutation tests to the different characteristics of the networks relative to asymptotic tests.*

## 7 Bayesian Network Meta-analysis

In this set of experiments, we demonstrate the importance of exact testing procedures to the BNMA problem. Recall that in such tasks cases, one can not pool all the data together in a single dataset. Fortunately, we can overcome this problem. Most constraint-based algorithms that learn graphical models only require the ability to perform conditional tests. A straightforward approach that employs all datasets is to perform the test  $T_i(X; Y | \mathbf{Z})$  individually in each available dataset  $D_i$  obtaining the  $p$ -values  $\{p_i\}$ . Fisher's Inverse  $\chi^2$  test can then be used to compute a combined statistic  $S = -2 \sum \log p_i$ .  $S$  follows a  $\chi^2$  distribution with  $2q$  degrees of freedom, where  $q$  is the number of datasets contributing data to the test, from which we can obtain the combined  $p$ -value  $p^*$  for the test  $T(X; Y | \mathbf{Z})$  on all datasets (see also [7, 14] for other methods).

When combining multiple datasets, each dataset may not have enough sample to perform the test, but their combination could have. So, we generalized the heuristic power rule to perform a test when  $\pi \leq \sum N_i / m_i$  over all datasets  $i$  where  $N_i$  is the sample size of  $D_i$  and  $m_i$  the number of parameters estimated by the specific test (e.g.,  $m_i$  may be different from  $m_j$  if a variable takes 3 possible values in one dataset and 4 in the other). When all datasets have the same number of parameters for the test, the rule results in testing whether  $\pi \leq N/m$  as in the single-dataset case. Again, the value of  $\pi$  is set to 2.

To evaluate the performance of the PC on multiple-datasets we employed the same four networks as in Section 6. For each network we decide on the number of datasets to feed the PC in the set  $\{1, 2, 3, 4, 5, 7, 10, 15, 20\}$  and the size of these datasets within the set  $\{50, 100, 300, 500\}$ . For each case (4 networks  $\times$  9 dataset-collection size  $\times$  4 sample sizes) we sample 10 times from the distribution of the network. All reported results are averaged over the 10 samplings (repeats of an experiment). We then execute the PC algorithm equipped with Fisher's Inverse



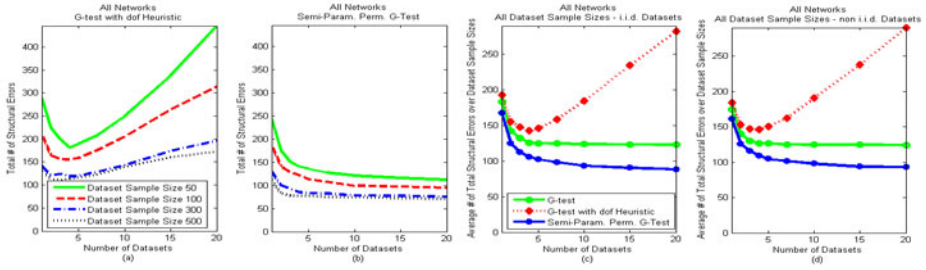
**Fig. 4. Learning the Skeleton of Bayesian Networks from Multiple i.i.d. Datasets: the effect of increasing number of datasets.** The permutation procedure dominates performance. The  $G$ -test with the heuristic adjustment has counter-intuitive behavior showing more errors with total available sample.

$\chi^2$  test to combine  $p$ -values from the different datasets, and the AG, AGh, and PGF to compute the  $p$ -values on each individual dataset, giving rise to three versions of the algorithm. Obviously, in this set of experiments the datasets fed to the PC could be pooled together; the results are to validate the methods in the simplest case of i.i.d. datasets.

The results for sample size 50 are shown in Figures 4(a)-(d) where the number of structural errors is shown over the number of datasets combined. An interesting phenomenon is that the AGh exhibits decreasing performance with available datasets (and total sample size)! This is explained considering the lack of calibration of the test demonstrated at Section 5. The  $p$ -values of the test are underestimated to be closer to zero. When several of them are combined together in the statistic  $S = -2 \sum \log p_i$  the errors accumulate and falsely provide confidence that dependency can be accepted. For example, consider combining four exact  $p$ -values all equal to 0.2. This gives  $S = 12.8755$  and combined  $p^* = 0.1162$ . If the approximate  $p$ -values are computed instead as half the exact values (i.e., as 0.1) then  $S = 18.4207$  and the combined  $p^* = 0.0183$ , i.e, 6 times lower. Thus, as the number of datasets and  $p$ -values that are combined each time increases, the probability of accepting dependence also increases.

Figures 5(a)-(c) show the performance (total number of structural errors in all four networks) of each algorithm respectively, as the sample size per dataset increases. As expected all algorithms benefit from the increased available size per dataset. In Figure 5(a) we observe that the counter-intuitive behavior of the AGh test is ameliorated as the asymptotic approximations of the  $p$ -values become more accurate with increased sample size. Figure 5(c) shows the average over all sample sizes of the total structural errors on all four networks: *the permutation procedure dominates the asymptotic ones in learning quality.*

In a second set of experiments using the same exact settings we simulate non-identically distributed networks. Specifically, for each non-binary variable in a dataset, with probability 10% we collapse pairs of neighboring values to a single one. For example, a variable Smoking taking values No/Light/Regular/Heavy becomes a binary No/Yes variable. The behavior of all algorithms is very similar



**Fig. 5.** (a)-(b) **The effect of increasing sample size per dataset.** The counter-intuitive behavior of the  $G$ -test with the heuristic adjustment (a) is ameliorated with increased sample size per dataset. The permutation based procedure well-behaves for small and large datasets (b). **Learning the Skeleton from Multiple i.i.d. and non i.i.d. Datasets.** The permutation procedure dominates performance in both cases. The performance for all procedures is similar in the i.i.d. (c) and the non i.i.d. (d) data.

to the results employing i.i.d. datasets, so we select to present only the average results over all networks and sample sizes in Figure 5(d). The maximum difference in average performance between figures (c)(i.i.d.) and (d) (non i.i.d.) cases is between 6-9 total structural errors for each algorithm.

## 8 Conclusions

Procedures for testing conditional independence are foundational for learning graphical models using constraint-based methods. The tests used in prior work for discrete data are all based on asymptotic approximations that are not robust to small sample sizes. We counter-suggest the use of exact tests based on permutation procedures. We show that the latter are both practical (10 to 20 times slower than asymptotic tests) and provide several benefits in learning behavior. Specifically, we show that (a) permutation testing is calibrated, i.e, the actual Type I error matches the significance level  $\alpha$  set by the user; this is not the case with asymptotic tests, (b) permutation testing leads to more robust structural learning, and (c) permutation testing allows learning networks from multiple datasets that cannot be pooled together; in contrast, asymptotic tests may lead to erratic learning behavior in this task (error increasing with total sample-size). The semi-parametric permutation procedure we propose is a reasonable approximation of the basic procedure using 5000 permutations, while being only 10-20 times slower than the asymptotic tests for small sample sizes. While our evaluation considers learning BNs, the conclusions of our studies should be applicable in learning other graphical models and related causal-based variable selection algorithms.

**Acknowledgements.** The work was partially funded by ELKE, University of Crete contract 2934 and the REACTION GA 248590 EU project.

## References

1. Agresti, A.: *Categorical Data Analysis*, 2nd edn. Wiley Series in Probability and Statistics. Wiley-Interscience, Hoboken (2002)
2. Agresti, A.: A survey of exact inference for contingency tables. *Statistical Science* 7(1), 131–153 (1992)
3. Aliferis, C.F., et al.: Local causal and markov blanket induction for causal discovery and feature selection for classification part i: Algorithms and empirical evaluation. *JMLR* 11, 171–234 (2010)
4. Beinlich, I., Suermondt, G., Chavez, R., Cooper, G.: The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. *Artificial Intelligence in Medicine*, 247–256 (1989)
5. Frank, E., Witten, I.H.: Using a permutation test for attribute selection in decision trees. In: *ICML*, pp. 152–160. Morgan Kaufmann, San Francisco (1998)
6. Good, P.: *Permutation, Parametric, and Bootstrap Tests of Hypotheses*, 3rd edn. Springer Series in Statistics. Springer, Heidelberg (2004)
7. Hong, F., Breitling, R.: A comparison of meta-analysis methods for detecting differentially expressed genes in microarray experiments. *Bioinformatics* 24, 374–382 (2008)
8. Jensen, D.: *Induction with Randomization Testing: Decision-Oriented Analysis of Large Data Sets*. Ph.D. thesis, Washington University (1992)
9. Jensen, D., Neville, J.: Randomization tests for relational learning. Tech. Rep. 03-05, Department of Computer Science, University of Massachusetts Amherst (2003)
10. Mehta, C.P.: Statxact: A statistical package for exact nonparametric inference. *The American Statistician* 45, 74–75 (1991)
11. Neville, J., Jensen, D., Friedland, L., Hay, M.: Learning relational probability trees. In: *9th ACM SIGKDD* (2003)
12. Richardson, T., Spirtes, P.: Ancestral graph markov models. *Annals of Statistics* 30(4), 962–1030 (2002)
13. Spirtes, P., Glymour, C., Scheines, R.: *Causation, Prediction, and Search*, 2nd edn. MIT Press, Cambridge (2000)
14. Tillman, R.E.: Structure learning with independent non-identically distributed data. In: *26th International Conference on Machine Learning, ICML 2009* (2009)
15. Tillman, R.E., Danks, D., Glymour, C.: Integrating locally learned causal structures with overlapping variables. In: *NIPS* (2008)
16. Triantafyllou, S., Tsamardinos, I., Tollis, I.G.: Learning causal structure from overlapping variable sets. In: *AI and Statistics* (2010)
17. Tsamardinos, I., Brown, L., Aliferis, C.: The Max-Min Hill-Climbing Bayesian Network Structure Learning Algorithm. *Machine Learning* 65(1), 31–78 (2006)
18. Tsamardinos, I., Triantafyllou, S.: The possibility of integrative causal analysis: Learning from different datasets and studies. *Journal of Engineering Intelligent Systems* (to appear, 2010)
19. Tsamardinos, I., Brown, L.E.: Bounding the false discovery rate in local bayesian network learning. In: *AAAI*, pp. 1100–1105 (2008)

# Example-dependent Basis Vector Selection for Kernel-Based Classifiers\*

Antti Ukkonen<sup>1</sup> and Marta Arias<sup>2</sup>

<sup>1</sup> Aalto University and  
Helsinki Institute for Information Technology  
Helsinki, Finland

`antti.ukkonen@hiit.fi`

<sup>2</sup> Universitat Politècnica de Catalunya, Barcelona, Spain  
`marias@lsi.upc.edu`

**Abstract.** We study methods for speeding up classification time of kernel-based classifiers. Existing solutions are based on explicitly seeking sparse classifiers during training, or by using budgeted versions of the classifier where one directly limits the number of basis vectors allowed. Here, we propose a more flexible alternative: instead of using the same basis vectors over the whole feature space, our solution uses different basis vectors in different parts of the feature space. At the core of our solution lies an optimization procedure that, given a set of basis vectors, finds a good partition of the feature space and good subsets of the existing basis vectors. Using this procedure repeatedly, we build trees whose internal nodes specify feature space partitions and whose leaves implement simple kernel classifiers. Experiments suggest that our method reduces classification time significantly while maintaining performance. In addition, we propose several heuristics that also perform well.

## 1 Introduction

Kernel-based classifiers such as support vector machines are among the most popular classification methods. When working with big datasets or costly kernels, however, these classifiers can be slow not only during train, but also when classifying new instances. Setting the kernel computation cost aside, the classification cost of a kernel classifier is mainly governed by the number of basis vectors that it contains. Existing solutions to speed up classification time are based on explicitly seeking classifiers with few basis vectors (*sparse* classifiers) during training. This is the approach taken for example in [17] in the context of support vector machines. Another approach is to limit the number of basis vectors explicitly by means of an a-priori imposed bound, a *budget*, as is the case of the budget perceptron [10] in online learning.

In this paper we propose an alternative solution that is also based on reducing the number of basis vectors. The main difference of our approach with respect to

---

\* This work was carried out while the author was visiting Yahoo! Research Barcelona.

existing solutions is that instead of using the same sparse classifier in the whole feature space, we use different sparse classifiers in different parts of the feature space. That is, *the set of basis vectors that we use to classify a new example now depends on the example*, or more accurately, depends on the region where the example lies. This added flexibility represents a challenge for learning: one not only needs to partition the feature space, but also needs to select basis vectors in each partition.

The idea of using different classifiers in different regions of the feature space is not new, and is in fact used in popular methods for regression and other learning problems, e.g. piecewise regression or functional trees [16]. However, to the best of our knowledge, this is the first time this has been applied in the context of sparse kernel methods. We use the framework of online learning and the kernel perceptron algorithm [14] as the basis of our solution. We note however, that our method can be applied as a post-processing step to any kernel-based classifier.

## 1.1 Related Work

Kernel methods and in particular *Support Vector Machines* (SVMs) [30] have been intensely studied during the past couple of decades. Finding *sparse* SVMs has been one of the main concerns from the start. Early works on this topic include [6,5], in which the authors devise post-processing algorithms for finding a reduced set of basis vectors from a given solution. Most of the approaches that followed differ from [6,5] (and our paper) by formulating a version of the SVM learning problem that attempts to directly find a sparse solution. Early examples of this include [27,13,20,24]. More recently, [32] discuss a method to build sparse SVMs and report results where the accuracy of the full SVM is in some cases achieved using only a very small fraction (5%) of the original vectors. Unlike other related work, [8] propose an ensemble-based solution. The most recent contribution along this line of work is [17], where authors show that they can reduce the number of basis vectors by two orders of magnitude without suffering a decrease in classification accuracy.

Most of the approaches mentioned so far determine automatically the number of basis vectors used in the final sparse SVM (or use some regularization parameter that determines the trade-off between sparsity and accuracy of the solution). In contrast, the approach in [12] admits a parameter specifying the maximum number of vectors allowed in the final solution (the *budget*). The works in [17,32] have this ability also.

A different line of work towards finding sparse kernel-based classifiers has been done in the context of the perceptron algorithm [26]. Perceptrons use the same type of classification functions as SVMs but are much faster to train. They can also be used in conjunction with kernels [14], which has gained them much popularity [14,3,18,9]. Starting with the work of [10], a new line of algorithms known as the *budget perceptron* has emerged, of which several variants exist [10,31,11]. All of these impose an upper bound on the number of basis vectors allowed in the final classifier, but differ in how they behave when this budget is exceeded. For example, on exceeding the budget [31,10,11] use different criteria



---

**Algorithm 1.** The perceptron [26] and kernel perceptron [14] algorithms
 

---

PERCEPTRON

```

1:  $\mathbf{w} = 0$ 
2: for  $i = 1, \dots, n$  do
3:    $\text{pred} = \text{sign}(\mathbf{w}^T \mathbf{x}_i)$ 
4:   if  $\text{pred} \neq y_i$  then
5:      $\mathbf{w} = \mathbf{w} + y_i \mathbf{x}_i$ 
6:   end if
7: end for

```

KERNELPERCEPTRON

```

1:  $S = \{\}$ 
2: for  $i = 1, \dots, n$  do
3:    $\text{pred} = \text{sign}(\sum_{(a,b) \in S} aK(\mathbf{b}, \mathbf{x}_i))$ 
4:   if  $\text{pred} \neq y_i$  then
5:      $S = S \cup \{(y_i, \mathbf{x}_i)\}$ 
6:   end if
7: end for

```

---

to chose which “old” vector is going to be replaced. In contrast, the *projectron* [21] attempts to project the new example back to the space spanned by the existing basis vectors if possible, or adds it otherwise. The *projectron* does not have an apriori upper bound, but it can be shown that classifiers cannot grow unboundedly.

All related work described so far produces a single, global, sparse classifier. In this paper, we depart from this by allowing different solutions in different regions of the feature space. The only works we know of that use this idea are [28] and [2]. Especially [2] discusses a method that seems similar to the one presented here as it combines decision trees with support vector machines. However, the main motivation in [2] is not to reduce the number of kernel evaluations, and the resulting algorithm is a fairly complex combination of gradient descent and tabu search. In contrast to these, the work proposed here is done in the context of online learning with kernel perceptrons. In particular, our tree-based approach combines an online partitioning of the feature space with fast classification with budgeted kernel perceptrons. We call this new model the *perceptron tree*.

## 2 Perceptron Trees

### 2.1 Background on Perceptrons

The *kernel perceptron* [14,19] is an online learning algorithm which has been shown to perform very well in practice. Its simplicity, well-understood behavior, and the fact that it has the ability to incorporate kernels, make it an excellent candidate for being the basis for our solution. Given a sequence of  $n$  examples described by feature vectors  $\mathbf{x}_i$  together with their  $y_i \in \pm 1$  class labels

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)$$

the standard perceptron algorithm [26] keeps a hypothesis  $\mathbf{w}$  in the form of a weight vector of the same dimensionality as the examples representing a linear classifier, which is updated as indicated in Algorithm 1 (left column).

Notice that  $\mathbf{w}$  is a linear combination of examples  $\mathbf{x}_i$  on which the perceptron makes a mistake [14,18]. This suggests representing  $\mathbf{w}$  in its dual form, namely, by maintaining the set  $S$  of coefficients  $y_i$  and vectors  $\mathbf{x}_i$  on which the algorithm

---

**Algorithm 2.** An online algorithm for learning perceptron trees

---

```

BUILDTREE( $\mathbf{x}, y, N$ )
  if  $N$  is a leaf node then
    if  $P_N(\mathbf{x}) \neq y$  then
      add  $\mathbf{x}$  as a basis vector of  $P_N$ .
    end if
    if  $|P_N| > B$  then
      ( $f_N, N_L, N_R$ )  $\leftarrow$  SPLIT( $N$ )
    end if
  else
    BUILDTREE( $\mathbf{x}, y, f_N(\mathbf{x})$ )
  end if

```

---

makes mistakes during training. The advantage of this representation is that it allows the use of a kernel function  $K(\cdot, \cdot)$ . In the pseudocode for kernel perceptron in Algorithm [1](#) (right column),  $S$  is the set of pairs  $(a, \mathbf{b})$  such that  $a$  is the coefficient of the term  $K(\mathbf{b}, \cdot)$  in the decision function, and  $\mathbf{b}$  is a basis vector.

Clearly, the cost of classifying a new example requires the computation of a weighted sum of kernel products between the example to classify and the existing basis vectors. This cost is proportional to both the number of vectors in our hypothesis  $|S|$  and the time it takes to compute the kernel.

## 2.2 Solution Overview

Our solution uses different budget perceptrons in different regions of the feature space. Therefore, we need to devise a method that is able to learn mappings from the feature space to small sets of basis vectors. On arrival of a new example  $\mathbf{x}$  to be classified, we use the mapping to get the basis vectors  $S(\mathbf{x})$  operating in the region that  $\mathbf{x}$  belongs to, and use the rule  $\text{sign}(\sum_{(a, \mathbf{b}) \in S(\mathbf{x})} aK(\mathbf{b}, \mathbf{x}))$  to make the final prediction. Since we are trying to reduce classification time, we should be able to compute the region where an example falls quickly.

We propose a top-down, tree-like approach that starts with a single node training a budget perceptron. When we need to add a new basis vector and the budget is exceeded, we *split* that perceptron node and start training two new perceptrons as new left and right leaves to this node. The new perceptron leaf nodes are initialized with suitable subsets of the existing basis vectors. The split node is assigned a *branching function*  $f$  that sends examples to its left or right child. The leaves continue to split as we add new examples. When the process is finished, we end up with a tree where the branching functions  $f_i$  at the internal nodes induce a partition of the feature space, and the budget perceptrons  $P_j$  in the leaves classify examples.

Pseudocode for the tree building algorithm is shown in Algorithm [2](#). To construct a tree given training data, we call BUILDTREE once for each example  $\mathbf{x}$  with  $N$  being the root of the tree. When the size of the perceptron at a leaf node exceeds a predefined budget  $B$ , we split the node. This is done by the SPLIT function that returns the branching function  $f_N$  for node  $N$ , as well as

the left and right child nodes,  $N_L$  and  $N_R$ , rooted at  $N$ . SPLIT also initializes the perceptrons at  $N_L$  and  $N_R$  using the basis vectors at node  $N$ .

Naturally, all kinds of questions related to growing trees arise, such as when do we stop growing the tree, do we use pruning, etc. Although important in practice, these issues are not the focus of this paper. Our focus is in how to do the split: namely, how to find the branching functions, and how to select good subsets of existing basis vectors for the children of a split node.

In the following sections, we present several ways of implementing the SPLIT function. The main method that we propose is based on solving several quadratic and linear programming problems, although we also present a few heuristics that perform well in our experiments.

### 3 Splitting Nodes

Suppose that we are at a leaf node  $N$  that has made a mistake on a new example, and  $N$  has exhausted its budget. The perceptron at this leaf node needs to be split into two new perceptrons. To carry out the split we need to define two things:

1. *Partition problem*: what examples are assigned to the left and right subtrees, respectively?
2. *Selection problem*: what basis vectors of node  $N$  form the initial perceptron in the left and right subtrees, respectively?

Ideally we would like to solve both of these tasks in a way that minimizes the number of errors on unseen examples. It is not clear to us how to do this properly. Instead, we propose a number of different approaches. Of these the most important one is based on approximating the existing classifier at node  $N$ . This strategy is the main focus in this section, while the remaining ones are covered in Section 4.

Observe that the partition and selection problems are not independent. A good partition may depend on the selection of basis vectors, and selecting the basis vectors is affected by the partition. This suggests that in order to find good solutions the two problems must be solved together in a joint optimization step. Alternatively, we can decouple the problems by first using some criteria to define the partition, and subsequently selecting the basis vectors given this partition. We discuss the joint optimization approach in this section.

#### 3.1 Basic Definitions

For now our main objective when computing the split is to maintain the current classifier as well as possible. Let the set  $S_N = \{(a_1, \mathbf{b}_1), \dots, (a_m, \mathbf{b}_m)\}$  contain all basis vectors  $\mathbf{b}_j$  together with their class labels  $a_j$  that are used by node  $N$ . Moreover, let the set  $X_N = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$  contain all examples that node  $N$  has seen so far. Define the matrix  $\mathbf{G}$ , such that  $\mathbf{G}_{ij} = a_j K(\mathbf{x}_i, \mathbf{b}_j)$ , and let  $\mathbf{r}_i = \sum_j \mathbf{G}_{ij}$ . The perceptron at node  $N$  classifies  $\mathbf{x}_i$  according to  $\text{sign}(\mathbf{r}_i)$ . In

order to maintain the behavior of this classifier, *we seek to approximate  $\mathbf{r}_i$  on both sides of the split by using different subsets of  $S_N$* . To make this more formal, we define the following:

- *Partition:* Denote by  $\mathbf{z}$  an  $n$ -dimensional binary vector that indicates how to split  $X_N$ . We let  $\mathbf{z}_i = 1$  if  $\mathbf{x}_i \in X_N$  is assigned to the left subtree, and  $\mathbf{z}_i = 0$  if  $\mathbf{x}_i$  is assigned to the right subtree. For simplicity, we only consider perfectly balanced partitions, that is, partitions for which  $\sum_i \mathbf{z}_i = n/2$ .
- *Left selection:* Denote by  $\mathbf{p}$  an  $m$ -dimensional binary vector that indicates which basis vectors in  $S_N$  are assigned to the left child of  $N$ . We let  $\mathbf{p}_j = 1$  if basis vector  $\mathbf{b}_j$  is to be placed to the left, and  $\mathbf{p}_j = 0$  indicates otherwise.
- *Right selection:* We define the binary vector  $\mathbf{q}$  in the same way as  $\mathbf{p}$  for selecting basis vectors of the right child of  $N$ .

Using these definitions a split is represented by the triple  $(\mathbf{z}, \mathbf{p}, \mathbf{q})$ . Here  $\mathbf{z}$  defines the assignment of examples to the subtrees, while  $\mathbf{p}$  and  $\mathbf{q}$  select the basis vectors to be used in the left and right child nodes. But notice that while vector  $\mathbf{z}$  tells us which examples in  $X_N$  go to the left and right subtrees, it tells us nothing about new examples. In practice, we must learn the *branching function*  $f_N$  that allows us to classify new, unseen examples. Let us put this problem aside for now, however, and see how to find good splits  $(\mathbf{z}, \mathbf{p}, \mathbf{q})$ . Later, in Section 3.3, we show that a simple modification of this procedure allows us to learn  $\mathbf{p}$  and  $\mathbf{q}$  together with  $f_N$ .

### 3.2 Joint Optimization with Unconstrained Branching

In this section we consider the case where vectors in  $X_N$  can be placed arbitrarily to the left and right subtree. This means that the branching function is essentially unconstrained, and has no predefined form. We define the costs of vectors  $\mathbf{p}$  and  $\mathbf{q}$  as follows:

$$c_{\mathbf{p}}(\mathbf{x}_i) = \left( \sum_j \mathbf{p}_j \mathbf{G}_{ij} - \mathbf{r}_i \right)^2 \text{ and } c_{\mathbf{q}}(\mathbf{x}_i) = \left( \sum_j \mathbf{q}_j \mathbf{G}_{ij} - \mathbf{r}_i \right)^2, \quad (1)$$

where  $\mathbf{G}$  and  $\mathbf{r}$  are defined as above. We are thus seeking to minimize the squared difference of using all the basis vectors vs. using the selection only. (Notice that  $\mathbf{p}$  and  $\mathbf{q}$  need not be integer for Equation 1 to make sense.) For all  $\mathbf{x}_i$  that are placed to the left we pay  $c_{\mathbf{p}}(\mathbf{x}_i)$ , and for the remaining ones we pay  $c_{\mathbf{q}}(\mathbf{x}_i)$ . The total cost of the split is thus defined as

$$c(\mathbf{z}, \mathbf{p}, \mathbf{q}) = \sum_i \mathbf{z}_i c_{\mathbf{p}}(\mathbf{x}_i) + (1 - \mathbf{z}_i) c_{\mathbf{q}}(\mathbf{x}_i), \quad (2)$$

which is equivalent (up to an additive constant, see Appendix A) to

$$\hat{c}(\mathbf{z}, \mathbf{p}, \mathbf{q}) = \mathbf{p}^T \mathbf{Q}_L(\mathbf{z}) \mathbf{p} - 2\mathbf{w}_L^T(\mathbf{z}) \mathbf{p} + \mathbf{q}^T \mathbf{Q}_R(\mathbf{z}) \mathbf{q} - 2\mathbf{w}_R^T(\mathbf{z}) \mathbf{q}. \quad (3)$$

Here  $\mathbf{Q}_L(\mathbf{z})$  and  $\mathbf{Q}_R(\mathbf{z})$  are  $m \times m$  matrices, and  $\mathbf{w}_L(\mathbf{z})$  and  $\mathbf{w}_R(\mathbf{z})$  are  $m$ -dimensional vectors that *depend on  $\mathbf{z}$* . If we fix the vector  $\mathbf{z}$ , then minimizing

Equation 3 is a matter of solving two independent quadratic integer programs, of which the one for  $\mathbf{p}$  is shown below:

$$\begin{aligned} \text{QP}_{\mathbf{p}} : \min & \frac{1}{2} \mathbf{p}^T \mathbf{Q}_L(\mathbf{z}) \mathbf{p} - \mathbf{w}_L^T(\mathbf{z}) \mathbf{p} \\ \text{st.} & \sum_j \mathbf{p}_j \leq C, \\ & \mathbf{p}_j \in \{0, 1\} \text{ for all } j. \end{aligned}$$

The program  $\text{QP}_{\mathbf{q}}$  for finding  $\mathbf{q}$  is defined analogously. Notice that the matrices  $\mathbf{Q}_L(\mathbf{z})$  and  $\mathbf{Q}_R(\mathbf{z})$  are positive semidefinite and therefore the objective function is convex.

The constraint  $\sum_j \mathbf{p}_j \leq C$  is imposed so that the left subtree uses at most  $C$  basis vectors. This constant should be set by the user depending on the needs of the application at hand or by some other empirical method beyond the scope of our present discussion. If given a budget  $B$ , a reasonable value for  $C$  could be set for example to  $B/2$ , so that the new perceptrons at the leaves are initialized with half of their available budget. This is, in fact, the strategy that we adopt in the experiments.

Finding  $\mathbf{p}$  and  $\mathbf{q}$  is therefore easy if we know  $\mathbf{z}$ . We continue by showing that given  $\mathbf{p}$  and  $\mathbf{q}$  it is easy to find  $\mathbf{z}$ . Clearly Equation 2 can be rewritten as

$$c(\mathbf{z}, \mathbf{p}, \mathbf{q}) = \sum_i \mathbf{z}_i \underbrace{\left( c_{\mathbf{p}}(\mathbf{x}_i) - c_{\mathbf{q}}(\mathbf{x}_i) \right)}_{c_i(\mathbf{p}, \mathbf{q})} + \sum_i c_{\mathbf{q}}(\mathbf{x}_i).$$

From this we see that if we know the costs  $c_{\mathbf{p}}(\mathbf{x}_i)$  and  $c_{\mathbf{q}}(\mathbf{x}_i)$  for all  $i$ , i.e., the vectors  $\mathbf{p}$  and  $\mathbf{q}$  are fixed, the optimal  $\mathbf{z}$  is found by solving the linear integer program

$$\begin{aligned} \text{LP}_{\mathbf{z}} : \min & \mathbf{z}^T \mathbf{c}(\mathbf{p}, \mathbf{q}) \\ \text{st.} & \sum_i \mathbf{z}_i = n/2, \\ & \mathbf{z}_i \in \{0, 1\} \text{ for all } i. \end{aligned}$$

Note that solving this is simply a matter of setting those indices of  $\mathbf{z}$  to 1 that correspond to the  $n/2$  smallest values of  $\mathbf{c}(\mathbf{p}, \mathbf{q})$ .

Ideally we would like to solve (3) for  $\mathbf{z}$ ,  $\mathbf{p}$ , and  $\mathbf{q}$  simultaneously. This, however, does not seem easy. But as argued above, we can find  $\mathbf{p}$  and  $\mathbf{q}$  given  $\mathbf{z}$ , and vice versa. Our solution, shown in Algorithm 3, relies on this. It iteratively optimizes each of  $\mathbf{z}$ ,  $\mathbf{p}$ , and  $\mathbf{q}$  while keeping the two other vectors fixed.

The situation is further complicated by the integer constraints present in the quadratic programs given above. Our strategy here is to use fractional solutions (variables bounded in  $[0, 1]$ ) during the iteration and round these to integers before returning a solution. We experimented with a number of rounding schemes, and observed the following to perform well. For each  $i$ , set  $\mathbf{p}_i = 1$  with probability equal to the value of  $\mathbf{p}_i$  in the fractional solution. Repeat this a number of times and keep the best.

---

**Algorithm 3.** Joint optimization with unconstrained branching

---

- 1: Initialize  $\mathbf{p}$  and  $\mathbf{q}$  at random.
  - 2: **while** objective function value decreases **do**
  - 3:   Find  $\mathbf{z}$  given  $\mathbf{q}$  and  $\mathbf{p}$  by solving  $\text{LP}_{\mathbf{z}}$ .
  - 4:   Find  $\mathbf{q}$  given  $\mathbf{z}$  by solving a fractional relaxation of  $\text{QP}_{\mathbf{q}}$ .
  - 5:   Find  $\mathbf{p}$  given  $\mathbf{z}$  by solving a fractional relaxation of  $\text{QP}_{\mathbf{p}}$ .
  - 6: **end while**
  - 7: Round  $\mathbf{p}$  and  $\mathbf{q}$  to integers.
  - 8: **return**  $(\mathbf{z}, \mathbf{p}, \mathbf{q})$
- 

While Algorithm 3 clearly solves the selection part of the split, it is less obvious whether we can use the vector  $\mathbf{z}$  to construct a good branching function  $f_N$ . A simple approach is to find the vector  $\mathbf{x}_i^* \in X_N$  that is closest to an unseen example  $\mathbf{x}$ , and forward  $\mathbf{x}$  to the left subtree if  $z_i = 1$ , and to the right if  $z_i = 0$ . We refer to this approach as ZPQ in the experiments of Section 5.

Notice that finding the nearest neighbor can be slow, and ideally we want the branching functions to be (orders of magnitude) faster than classifying  $\mathbf{x}$ . To this end, we discuss next an alternative formulation of the split that directly includes  $f_N$  as part of the optimization problem.

### 3.3 Joint Optimization with Linear Branching

In this section we propose a split where the vectors in  $X_N$  may no longer be partitioned arbitrarily. In particular, the branching function  $f_N$  is constrained to be a linear separator in the feature space. The basis vector selection remains as before. Denote the linear separator by  $\mathbf{w}$ . We let  $\mathbf{z}$  depend on  $\mathbf{w}$  as follows:

$$z_i = \frac{1}{2} \text{sign}(\mathbf{w}^T \mathbf{x}_i) + \frac{1}{2},$$

where  $\mathbf{x}_i \in X_N$  as above. If we further assume that  $\|\mathbf{x}_i\| = 1$ ,  $\|\mathbf{w}\| = 1$ , and remove the sign from  $\mathbf{w}^T \mathbf{x}_i$ , we can let  $z_i = (\frac{1}{2} \mathbf{w}^T \mathbf{x}_i + \frac{1}{2}) \in [0, 1]$ . Using this the objective function of  $\text{LP}_{\mathbf{z}}$  from the previous section can be re-written to depend on  $\mathbf{w}$  instead of  $\mathbf{z}$ :

$$\mathbf{z}^T \mathbf{c} = \sum_i (\frac{1}{2} \mathbf{w}^T \mathbf{x}_i + \frac{1}{2}) c_i = \frac{1}{2} \mathbf{w}^T \underbrace{\sum_i \mathbf{x}_i c_i}_{\mathbf{d}} + \frac{1}{2} \sum_i c_i = \frac{1}{2} \mathbf{w}^T \mathbf{d} + D.$$

Omitting constants, our new formulation becomes

$$\begin{aligned} \text{LP}_{\mathbf{w}} : \min \quad & \mathbf{w}^T \mathbf{d} \\ \text{st.} \quad & \|\mathbf{w}\| = 1, \end{aligned}$$

where the  $\mathbf{d}$  is a column vector such that  $\mathbf{d}_j = \sum_i \mathbf{x}_{ij} c_i$ ,  $\mathbf{x}_{ij}$  is the value of example  $\mathbf{x}_i$  along dimension  $j$  after normalizing  $\mathbf{x}_i$ , and  $c_i$  is the cost associated with example  $\mathbf{x}_i$  which can be computed for fixed  $\mathbf{p}, \mathbf{q}$ .

It can be shown that  $LP_{\mathbf{w}}$  is minimized for  $\mathbf{w} = -\frac{\mathbf{d}}{\|\mathbf{d}\|}$ . That is, as with  $LP_{\mathbf{z}}$ , we do not need to solve a linear program. Notice that we have ignored the constraint present in  $LP_{\mathbf{z}}$  that requires a balanced solution. In this case, this can be enforced by using a threshold  $\theta = \text{median}\{\mathbf{w}^T \mathbf{x}_i | \mathbf{x}_i \in X_N\}$ . We acknowledge that we could include  $\theta$  in our optimization, but this straightforward approach leads to good results in practice.

Again we use Algorithm 3, but modify it so that line 3 reads: “Find  $\mathbf{w}$  given  $\mathbf{q}$  and  $\mathbf{p}$  by solving  $LP_{\mathbf{w}}$ . Let  $\theta = \text{median}\{\mathbf{w}^T \mathbf{x}_i | \mathbf{x}_i \in X_N\}$ , and compute the associated  $\mathbf{z}$  by setting  $\mathbf{z}_i = \text{sign}(\mathbf{w}^T \mathbf{x}_i + \theta)$ .”. Naturally, the same considerations regarding the rounding procedure to obtain integer solutions apply here.

With this method it is obvious that we have found both a fast to compute branching function,  $\text{sign}(\mathbf{w}^T \mathbf{x}_i + \theta)$ , and a good set of basis vectors for the child nodes. The algorithm presented here finds these simultaneously in the sense that the solution for  $\mathbf{w}$  depends on  $\mathbf{p}$  and  $\mathbf{q}$ , and vice versa. In the next section we discuss some other approaches where the partition and selection tasks are decoupled. We refer to this approach as WPQ in the experiments of Section 5.

## 4 Heuristics and Baselines for Splitting Nodes

The algorithm discussed above may be too complex for some situations as it involves solving a number of quadratic programs. Moreover, it is difficult to say how fast the objective function value converges. In practice we have observed the convergence to be rapid, but there are no theoretical guarantees for this. As a remedy we present some heuristics that are in general faster to compute.

We describe simple heuristics and baseline methods for the partition and the selection problems, respectively. These can be combined in order to develop reasonable splitting methods. As we already observed earlier, solving the selection problem becomes easy if we fix the partition.

### 4.1 Partition Heuristics

With partition heuristics we mean heuristics for constructing the branching function  $f_N$ . We have already seen two alternatives in the previous section. The first one made use of a nearest neighbor algorithm, while the second one was based on a linear separator of the feature space.

A simple baseline, which we call RNDW, consists in obtaining a random partition by drawing a hyperplane uniformly at random.

In addition, we propose a class of branching functions that can be represented as axis-aligned hyperplanes in the kernel space. That is, an example  $\mathbf{x}$  is directed to the left subtree if  $K(\mathbf{x}, \mathbf{b}_j) \leq t$  for some  $\mathbf{b}_j \in S_N$  and  $t$ , otherwise  $\mathbf{x}$  goes to the right. This type of branching was used in our previous work [28]. Furthermore, we only consider *balanced* branching functions, i.e., those mapping half of the examples in  $X_N$  to the left, and the other half to the right.

Algorithm 4 shows how to split a node using such balanced branching functions. Essentially, the algorithm goes through all basis vectors in  $S_N$  and uses

**Algorithm 4.**BALANCEDBRANCHINGFUNCTION( $S_N, X_N, K$ )

---

```

for  $(a_j, \mathbf{b}_j) \in S_N$  do
   $t_j \leftarrow \text{median}\{K(\mathbf{x}, \mathbf{b}_j) : \mathbf{x} \in X_N\}$ 
   $X_L \leftarrow \{\mathbf{x} \in X_N : K(\mathbf{x}, \mathbf{b}_j) \leq t_j\}$ 
   $X_R \leftarrow \{\mathbf{x} \in X_N : K(\mathbf{x}, \mathbf{b}_j) > t_j\}$ 
   $s_j \leftarrow \text{EVALUATE}(X_L, X_R)$ 
end for
 $j^* \leftarrow \text{argmax}_j s_j$ 
return  $(\mathbf{b}_{j^*}, t_{j^*})$ 

```

---

each in turn to construct a branching function. The partition proposed by each is evaluated using the EVALUATE function. The algorithm returns the basis vector together with the associated  $t$  that has the highest score. We can alter the optimization criteria by changing the EVALUATE function, below we describe two alternatives.

**BAL** This heuristic is used in combination with any of the selection methods described below. The split is evaluated in terms of the squared error between the approximation and the true  $\mathbf{r}_i$  values (see Section 3.1) after solving the selection problem in the leaves.

**ENT** This heuristic is inspired by the impurity measure used in decision trees. Let  $H(\mathbf{z}_i = 1)$  and  $H(\mathbf{z}_i = 0)$  be the entropies of the class labels in left and right partitions, respectively. This heuristic chooses the split that minimizes  $\min(H(\mathbf{z}_i = 1), H(\mathbf{z}_i = 0))$ .

## 4.2 Selection Heuristics

Now suppose that we have selected a branching function according to some partition heuristic from above. We have to determine how to select subsets of existing basis vectors for the left and right children. We have experimented with the following baselines and heuristics for this:

**RNDS** Pick two random subsets of size  $C$  from  $S_N$  to be used as the initial basis vectors in the left and right subtrees of  $N$ .

**INDS** Use the branching function  $f_N$  to assign existing basis vectors in  $S_N$  either to the left or right subtree. Notice that the bound  $C$  on the number of basis vectors may be violated, so some post-processing may be needed.

**OPTS** Solve fractional relaxations of QP $_{\mathbf{p}}$  and QP $_{\mathbf{q}}$ , round  $\mathbf{p}$  and  $\mathbf{q}$  to obtain integer solutions.

In our experiments, we use identifiers such as BAL-OPTS or ENT-INDS for our heuristics; these mean the combination of the *balanced* partition heuristic using our *optimization* as evaluation function, and the combination of the *entropy* heuristic for partition and *partition-induced* heuristic for selection, respectively.



**Table 1.** Data sets used in the experimental section

<i>name</i>	<i>dataset size</i>	<i>nr. attributes</i>	<i>source</i>
ADULT	48842	123	[11]
IJCNN	141691	22	[25]
COD-RNA	488565	8	[29]

## 5 Experiments

In this section we test empirically the validity of our method and the heuristics proposed. Recall that the main goal of this work is to devise a classifier where the basis vectors of the decision rule depend on the input example  $\mathbf{x}$ . Our approach is to partition the input space and use a different perceptron in each region. The most straightforward strategy is to divide the feature space into two disjoint regions. The first experiment is designed to show that already in this very simple setting we can gain in accuracy by using two different perceptrons in the regions.

In the second experiment we study how well the tree-based approach compares with other budgeted online learning algorithms. We do not expect them to outperform these in accuracy. Instead our main argument is that we can achieve a similar performance by using a considerably smaller number of basis vectors. While our trees can grow very large, and may hence contain a large number of leaf nodes, the total number of kernel computations for each example is bounded by the user-specified budget  $B$ . We set  $B = 100$  in these experiments. A more thorough study of the effects of  $B$  is left for a longer version of this paper.

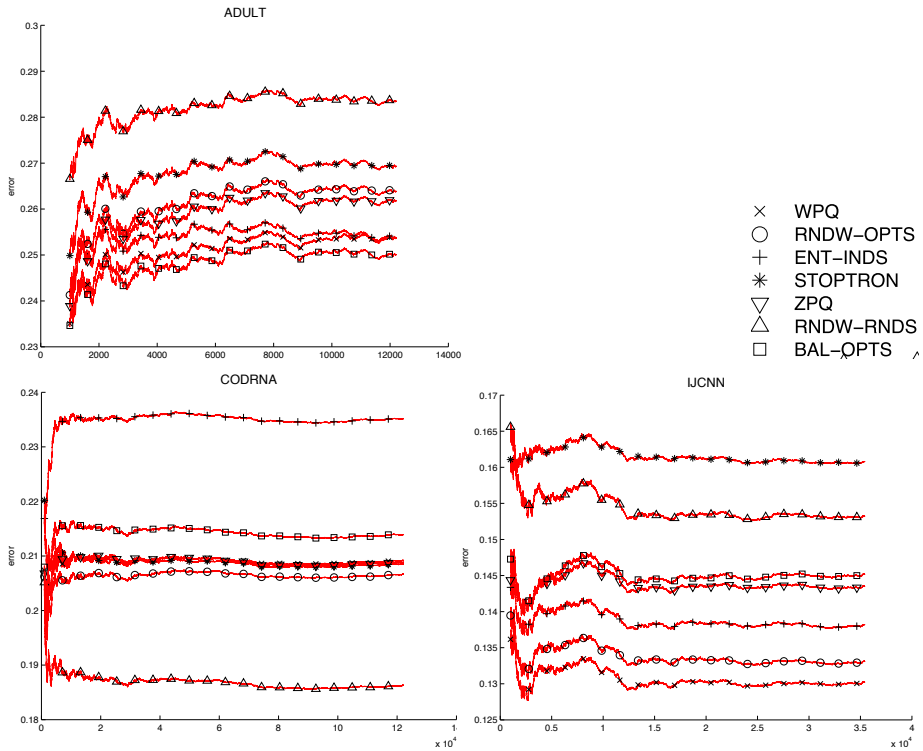
We compare our methods with the FORGETRON [11] and PROJECTRON [23] algorithms<sup>1</sup>, and the STOPTRON, a simple budget perceptron baseline that learns a kernel perceptron but stops updating after budget is reached. The data sets we use are publicly available at [7]. Table 1 shows details of the data sets. Each data set was split into training (50%), testing (25%), and validation (25%) sets. In the experiments that follow we use different portions of the datasets, which will be appropriately described in the text.

In each case we use a Gaussian kernel. The  $\gamma$  parameter of the kernel was optimized for each data by running the basic perceptron algorithm on the training data using different values of  $\gamma$  and choosing the one with the best performance on the validation set.

### 5.1 Comparing Splitting Functions

This experiment compares the performance of using a single STOPTRON against using two STOPTRONS at the leaves of a single split, varying and thus comparing the different splitting criteria. The setting is as follows: we start training a STOPTRON on the training data. Once the number of basis vectors hits  $B$  (set to 100), we split the node using each of the techniques described above. At this point training is stopped, and we switch to testing. For each splitting method,

<sup>1</sup> We use implementations of the DOGMA library [22].



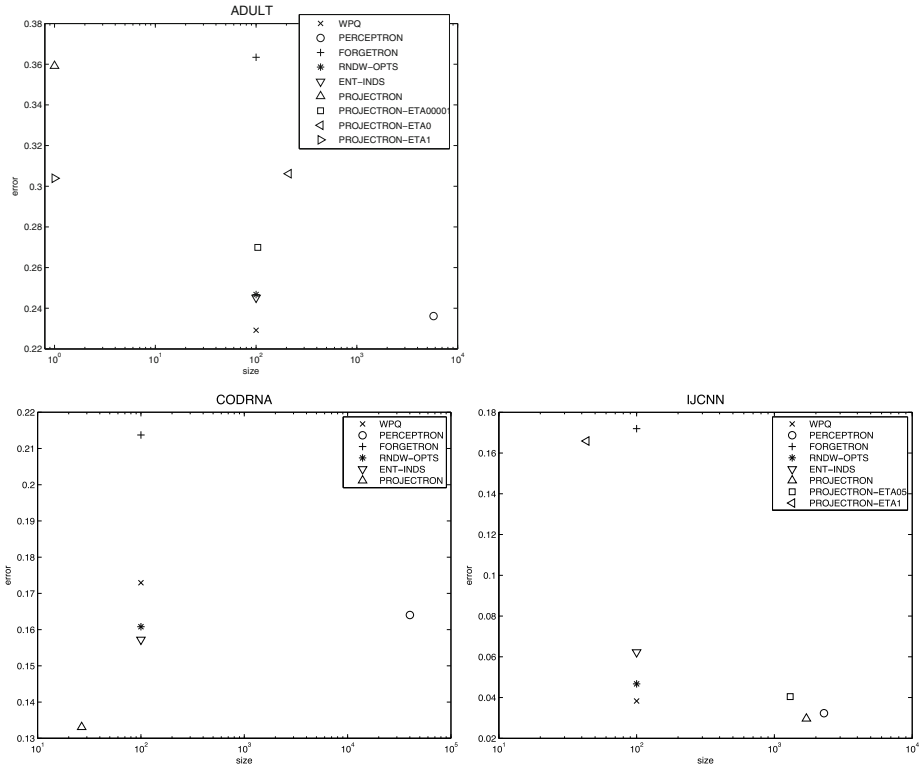
**Fig. 1.** Results of the splitting experiment showing medians over 30 independent runs of the average cumulative error. The x-axis represents the number of test examples processed so far, and the y-axis shows the average cumulative error obtained up until that point. The different runs are obtained by permuting the training set and so the initial *Stoptron* may differ in each separate run.

the resulting model is applied to the test data. We make a pass over the entire test data, and two *STOPTRONS* are grown on either side of the splits.

The results in Figure 1 show that splitting in combination with *STOPTRON* outperforms the global *STOPTRON*, supporting our hypothesis that splitting may be very useful. Moreover, the results also suggest that it is important to do the split in a smart way. This is indicated by the relatively poor performance of the completely random *RNDW-RDNS* heuristic in case of *ADULT* and *IJCNN*. Interestingly, *RNDW-OPTS* performs quite well, suggesting that if the basis vectors are selected carefully, any balanced linear branching function will have a reasonable performance. Finally, our *WPQ* method is the only one that consistently shows good performance across all three datasets.

## 5.2 Algorithm Comparison

This experiment measures the trade-off between classification time and classification accuracy. We use the number of basis vectors as a proxy for classification



**Fig. 2.** Number of basis vectors versus the error. X-axis: the budget, i.e., the maximum number of basis vectors that are used to classify an unseen example. Y-axis: the average cumulative test error at the end of learning.

time, as this essentially determines the run-time complexity of evaluating the decision function. We compare our perceptron tree approach with the regular perceptron and recently proposed budgeted variants [11,23]. Again we set  $B$  to 100. In case of the PROJECTRON algorithm it is not possible to set the budget explicitly. Instead, there is a parameter  $\eta$  that indirectly affects the size of the resulting model. The results include the values which we found to work best after manual tuning.

Figure 2 shows size vs. error for a representative set of methods. The errors are averages over 10 runs of the mean number of mistakes made by the algorithms after doing one pass over the training data. Ideally one seeks a small model with low error. These can be found in the lower left corner of the plots. We want to point out that for the perceptron trees, the size is set to  $B$  because this is the number of basis vectors that contribute to an example’s classification. The actual size of the tree is much larger, but only  $B$  basis vectors are used to classify an example.

In general our methods show good performance. With the ADULT data the WPQ algorithm is not only the quickest to evaluate, but outperforms every other method in terms of the error. Unlike reported in [23] we observe a fairly poor performance of the projectron with the ADULT data. In the case of IJCNN, WPQ has a slightly higher error rate than the PROJECTRON, but is an order of magnitude smaller. With the COD-RNA data the PROJECTRON algorithm clearly outperforms all other methods both in size and error. However, our methods are competitive with the perceptron while using a much smaller number of basis vectors, which is the main objective of this work.

## 6 Conclusions and Future Work

We have proposed a solution to the problem of speeding up classification time with kernel methods that is based on partitioning the feature space into disjoint regions and using budgeted models in each region. This idea follows up preliminary work by one of the authors in [28] and is, to the best of our knowledge, the first time an example-dependent solution is applied to this problem. Clearly, data sets that present different structure in different parts of the feature space should benefit from this approach. Our experiments indicate the potential of this method, and we want to further study the behavior of the algorithms using more datasets and explore the effect of different parameter settings.

Our optimization criterion in this paper is to approximate the behavior of the perceptron while using a smaller number of basis vectors. The motivation behind this is that in some other learning scenarios, such as ranking for example, one actually does care about the values output by the methods, and not just the labels. Naturally, other objective functions could be considered, such as directly minimizing the classification error in the context of a classification problem. In fact, the ENT heuristic from Section 4 is a first step in this direction. We plan to work on more principled approaches in the future.

The idea of example-dependent selection of basis functions carries naturally over to other domains, such as ensemble or boosting methods, e.g. random forests [4] or gradient boosted decision trees [15]. From a more theoretical perspective, it would be desirable to obtain an understanding of the generalization ability of example-dependent classifiers.

## References

1. Asuncion, A., Newman, D.: UCI machine learning repository (2007), <http://archive.ics.uci.edu/ml/>
2. Bennett, K.P., Blue, J.A.: A support vector machine approach to decision trees. In: Proceedings of The 1998 IEEE International Joint Conference on Neural Networks, pp. 2396–2401 (1998)
3. Bordes, A., Ertekin, S., Weston, J., Bottou, L.: Fast kernel classifiers with online and active learning. *The Journal of Machine Learning Research* 6, 1619 (2005)
4. Breiman, L.: Random forests. *Machine Learning* 45(1), 5–32 (2001)

5. Burges, C.J.C.: Simplified support vector decision rules. In: ICML, pp. 71–77 (1996)
6. Burges, C.J.C., Schölkopf, B.: Improving the accuracy and speed of support vector machines. In: NIPS, pp. 375–381 (1996)
7. Chang, C.C., Lin, C.J.: LIBSVM dataset site (April 2010), <http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/>
8. Chen, J.H., Chen, C.S.: Reducing svm classification time using multiple mirror classifiers. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 34(2), 1173–1183 (2004)
9. Collins, M., Duffy, N.: New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 263–270. Association for Computational Linguistics, Morristown (2001)
10. Crammer, K., Kandola, J., Singer, Y.: Online classification on a budget. *Advances in Neural Information Processing Systems* 16 (2004)
11. Dekel, O., Shalev-Shwartz, S., Singer, Y.: The Forgetron: A kernel-based Perceptron on a budget. *SIAM Journal on Computing* 37(5), 1342–1372 (2008)
12. Dekel, O., Singer, Y.: Support vector machines on a budget. In: *Advances in Neural Information Processing Systems, Proceedings of the 2006 Conference*, vol. 19, p. 345. The MIT Press, Cambridge (2007)
13. Downs, T., Gates, K., Masters, A.: Exact simplification of support vector solutions. *The Journal of Machine Learning Research* 2, 293–297 (2002)
14. Freund, Y., Schapire, R.E.: Large margin classification using the perceptron algorithm. *Machine Learning* 37(3), 277–296 (1999)
15. Friedman, J.: Greedy function approximation: a gradient boosting machine. *Annals of Statistics* 29(5), 1189–1232 (2001)
16. Gama, J.: Functional trees. *Machine Learning* 55(3), 219–250 (2004)
17. Joachims, T., Yu, C.N.J.: Sparse kernel svms via cutting-plane training. *Machine Learning* 76(2-3), 179–193 (2009); *European Conference on Machine Learning (ECML) Special Issue*
18. Khardon, R., Wachman, G.: Noise tolerant variants of the perceptron algorithm. *Journal of Machine Learning Research* 8, 227–248 (2007)
19. Kivinen, J., Smola, A., Williamson, R.: Online learning with kernels. *IEEE Transactions on Signal Processing* 52(8), 2165–2176 (2004)
20. Nair, P., Choudhury, A., Keane, A.: Some greedy learning algorithms for sparse regression and classification with mercer kernels. *The Journal of Machine Learning Research* 3, 801 (2003)
21. Orabona, F., Keshet, J., Caputo, B.: Bounded Kernel-Based Online Learning. *Journal of Machine Learning Research* 10, 2643–2666 (2009)
22. Orabona, F.: DOGMA: a MATLAB toolbox for Online Learning (2009), software available at, <http://dogma.sourceforge.net>
23. Orabona, F., Keshet, J., Caputo, B.: The projectron: a bounded kernel-based perceptron. In: *ICML '08: Proceedings of the 25th International Conference on Machine Learning*, pp. 720–727. ACM, New York (2008)
24. Osuna, E., Girosi, F.: Reducing the run-time complexity of support vector machines. In: *Advances in Kernel Methods: Support Vector Learning*, pp. 271–284. MIT Press, Cambridge (1999)
25. Prokhorov, D.: IJCNN 2001, Neural Network Competition (2001)
26. Rosenblatt, F.: The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review* 65(1), 386–407 (1958)
27. Tipping, M.E.: Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research* 1, 211–244 (2001)

28. Ukkonen, A.: The support vector tree. Algorithms and Applications, 244–259 (2010)
29. Uzilov, A., Keegan, J., Mathews, D.: Detection of non-coding RNAs on the basis of predicted secondary structure formation free energy change. BMC Bioinformatics 7(1), 173 (2006)
30. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, Heidelberg (2000)
31. Weston, J., Bordes, A., Bottou, L.: Online (and offline) on an even tighter budget. In: Proceedings of International Workshop on Artificial Intelligence and Statistics, Citeseer (2005)
32. Wu, M., Schölkopf, B., Bakır, G.: A direct method for building sparse kernel learning algorithms. The Journal of Machine Learning Research 7, 624 (2006)

## A Derivation of Equation 3

Denote by  $\mathbf{G}_i$  the  $i$ th row of  $\mathbf{G}$ . We write

$$\begin{aligned}
 \sum_i \mathbf{z}_i c_{\mathbf{p}}(\mathbf{x}_i) &= \sum_i \mathbf{z}_i \left( \left( \sum_j \mathbf{p}_j \mathbf{G}_{ij} \right)^2 - 2\mathbf{r}_i \sum_j \mathbf{p}_j \mathbf{G}_{ij} + \mathbf{r}_i^2 \right) \\
 &= \sum_i \mathbf{z}_i \left( \mathbf{p}^T \mathbf{G}_i^T \mathbf{G}_i \mathbf{p} - 2\mathbf{r}_i \mathbf{G}_i \mathbf{p} + \mathbf{r}_i^2 \right) \\
 &= \mathbf{p}^T \underbrace{\left( \sum_i \mathbf{z}_i \mathbf{G}_i^T \mathbf{G}_i \right)}_{\mathbf{Q}_L(\mathbf{z})} \mathbf{p} - 2 \underbrace{\left( \sum_i \mathbf{z}_i \mathbf{r}_i \mathbf{G}_i \right)}_{\mathbf{w}_L^T(\mathbf{z})} \mathbf{p} + \sum_i \mathbf{z}_i \mathbf{r}_i^2 \\
 &= \mathbf{p}^T \mathbf{Q}_L(\mathbf{z}) \mathbf{p} - 2\mathbf{w}_L^T(\mathbf{z}) \mathbf{p} + C_L,
 \end{aligned}$$

Using similar manipulations we obtain

$$\sum_i (1 - \mathbf{z}_i) c_{\mathbf{q}}(\mathbf{x}_i) = \mathbf{q}^T \mathbf{Q}_R(\mathbf{z}) \mathbf{q} - 2\mathbf{w}_R^T(\mathbf{z}) \mathbf{q} + C_R.$$

The matrix  $\mathbf{Q}_L(\mathbf{z})$  and the vector  $\mathbf{w}_L(\mathbf{z})$  are thus based on all those rows of  $\mathbf{G}$  for which  $\mathbf{z}_i = 1$ . Likewise, the matrix  $\mathbf{Q}_R(\mathbf{z})$  and the vector  $\mathbf{w}_R(\mathbf{z})$  are based on the rows for which  $\mathbf{z}_i = 0$ . We obtain Equation 3 by summing the above equations and omitting the constants  $C_L$  and  $C_R$ .

# Surprising Patterns for the Call Duration Distribution of Mobile Phone Users

Pedro O.S. Vaz de Melo<sup>1,4</sup>, Leman Akoglu<sup>2,3,4</sup>,  
Christos Faloutsos<sup>2,3,4</sup>, and Antonio A.F. Loureiro<sup>1</sup>

<sup>1</sup> Universidade Federal de Minas Gerais

<sup>2</sup> Carnegie Mellon University

<sup>3</sup> SCS, School of Computer Science

<sup>4</sup> iLab, Heinz College

olmo@dcc.ufmg.br, lakoglu@cs.cmu.edu,  
christos@cs.cmu.edu, loureiro@dcc.ufmg.br

**Abstract.** How long are the phone calls of mobile users? What are the chances of a call to end, given its current duration? Here we answer these questions by studying the call duration distributions (CDDs) of individual users in large mobile networks. We analyzed a large, real network of *3.1 million* users and more than one *billion* phone call records from a private mobile phone company of a large city, spanning *0.1TB*. Our first contribution is the TLAC distribution to fit the CDD of each user; TLAC is the truncated version of so-called *log-logistic* distribution, a skewed, power-law-like distribution. We show that the TLAC is an excellent fit for the overwhelming majority of our users (more than 96% of them), much better than exponential or lognormal. Our second contribution is the *MetaDist* to model the collective behavior of the users given their CDDs. We show that the *MetaDist* distribution accurately and succinctly describes the calls duration behavior of users in large mobile networks. All of our methods are fast, and scale linearly with the number of customers.

## 1 Introduction

In the study of phone calls databases [18,20,17], a common technique to ease the analysis of the data is the summarization of the phone calls records into aggregated attributes [10], such as the aggregate calls duration or the total number of phone calls. By doing that, the size of the database can be reduced by orders of magnitudes, allowing the execution of most well known data mining algorithms in a feasible time. However, we believe that such representation veils relevant temporal information inherent in a user or in a relationship between two people. When all the information about the phone calls records of a user is aggregated into single summarized attributes, we do not know anymore how often this user calls or for how long he talks per phone call. One may suggest, for instance, to use descriptive statistics such as mean and variance to describe the duration of the user's phone calls, but it is well known that the distribution of these values is highly skewed [20], what invalidate the use of such statistics.

In this paper, we tackle the following problem. Given a very large amount of phone records, what is the best way to summarize the calling behavior of a user? In order

to answer this question, we examine phone call records obtained from the network of a large mobile operator of a large city. More specifically, we analyze the duration of hundreds of million calls and we propose the *Truncated Lazy Contractor* (TLAC) model to describe how long are the durations of the phone calls of a single user. Thus, the TLAC models the Calls Duration Distribution (CDD) of a user and is parsimonious, having only two parameters, the *efficiency* coefficient  $\rho$  and the *weakness* coefficient  $\beta$ . We show that the TLAC model was the best alternative to model the CDD of the users of our dataset, mainly because it has a heavier tail and head than the log-normal distribution, that is the most commonly used distribution to model CDDs [7].

We also suggest the use of the TLAC parameters as a better way to summarize the calls duration behavior of a user. We propose the *MetaDist* to model the population of users that have a determined calls duration behavior. The *MetaDist* is the meta-distribution of the  $\rho_i$  and  $\beta_i$  parameters of each user's  $i$  CDD and, when its isocontours are visualized, its shape is surprisingly similar to a bivariate Gaussian distribution. This fascinating regularity, observed in a significantly noisy data, makes the *MetaDist* a potential distribution to be explored in the direction of better understanding the call behavior of mobile users.

Thus, in summary, the main contributions of this paper are:

- The proposal of the TLAC model to represent the individual phone calls durations of mobile customers;
- The *MetaDist* to model the group call behavior of the mobile phone users;
- The use of the *MetaDist* and the *Focal Point* to describe the collective temporal evolution of large groups of customers;

As an additional contribution, we show the usefulness of the TLAC model. We show that it can spot anomalies and it can succinctly verify correlations (or lack thereof) between the TLAC parameters of the users and their total number of phone calls, aggregate duration and distinct patterns. We also emphasize that the TLAC model can be used to generate synthetic datasets and to significantly summarize a very large number of phone calls records.

The rest of the paper is organized as follows. In Section 2, we provide a brief survey of other work that analyzed mobile phone records. In Section 3, we describe our proposed TLAC model and we show its goodness of fit. The *MetaDist* and the analysis on the temporal evolution of the collective call behavior of the customers of our dataset is shown in Section 4. In Section 5, we discuss the possible applications for our results and, finally, we show the conclusions and future research directions in Section 6.

## 2 Related Work

A natural use for a mobile phone dataset is to construct the social network from its records [10,8]. In [16,17], the authors construct a network from mobile phone calls records and, from it, they make a detailed analysis of its network properties. They identified relationships between node weights and network topology, finding that the weak ties are commonly responsible for linking communities, thus having a high betweenness centrality or low link overlap. Moreover, in [8], the authors verified that the persistence



of an edge is highly correlated to its reciprocity and to the topological overlap and, in [4], the authors explore communication networks in order to verify the patterns that occurs in its cliques. It is also common to analyze the networks from mobile companies in order to improve their services. For instance, in [9][3], the authors proposed a framework and data structures for identifying fraudulent consumers on telecommunication networks based on their degree distribution and dynamics and, in [15], the authors proposed metrics that can be employed by a business strategy planner involved in the telecom domain.

Another use for a mobile phone dataset is to study the individual attributes of the users. In [18], the authors proposed the DPLN distribution to model the distributions of the number of phone calls per customer, the total talk minutes per customer and the distinct number of calling partners per customer. In [7], the authors analyzed mobile phone calls that arrived in a mobile switch center in a GSM system of Qingdao, China, and they found that the duration of the phone calls is best modeled by a log-normal distribution. However, in [20], the authors studied the duration of mobile calls arriving at a base station during different periods and found that they are neither exponentially nor log-normally distributed, possessing significant deviations that make them hard to model. They verified that about 10% of calls have a duration of around 27 seconds, that correspond to calls which the called mobile users did not answer and the calls were redirected to voicemail. This makes the call durations distribution to be significantly skewed towards smaller durations due to nontechnical failures, e.g., failure to answer. Finally, the authors showed that the distribution has a “semi-heavy” tail, with the variance being more than three times the mean, which is significantly higher than that of exponential distributions. Comparing to a log-normal distribution, though the tails agree better, they too diverge at large values, what asks for a more heavy-tailed distribution.

### 3 Calls Duration Distribution

#### 3.1 Problem Definition

In this work, we analyze mobile phone records of 3.1 million customers during four months. In this period, more than 1 billion phone calls were registered and, for each phone call, we have information about the duration of the phone call, the date and time it occurred and encrypted values that represent the source and the destination of the call, that may be mobile or not. When not stated otherwise, the results shown in this work refer to the phone call records of the first month of our dataset. The results for the other 3 months are explicitly mentioned in Section 4.

The Call Duration Distribution (CDD) is the distribution of the call duration per user in a period of time, that in our case, is one month. In the literature, there is no consensus about what well known distribution should be used to model the CDD. There are researchers that claim that the PDD should be modeled by a log-normal distribution [7] and others that it should be modeled by the exponential distribution [19]. Thus, in this section, we tackle the following problem:

*Problem 1. CDD FITTING.* Given  $d_1, d_2, \dots, d_C$  durations of  $n_i$  phone calls made by a user  $i$  in a month, find the most suitable distribution for them and report its parameters.

As we mentioned before, there is no consensus about what well known distribution should be used to model the CDD, i.e., for some cases the log-normal fits well and for others, the exponential is the most appropriate distribution. Thus, finding another specific random distributions that could provide good fittings to a particular group of CDDs would just add another variable to Problem 1 without solving it. Therefore, we propose that the distribution that solves Problem 1 should necessarily obey to the following requirements:

- **R1**: Intuitively explain the intrinsic reasons behind the calls duration;
- **R2**: Provide good reliable fits for the great majority of the users.

In the following sections, we present a solution for Problem 1. In Section 3.2 we tackle Requirement R1 by presenting the TLAC model, that is a intuitive model to represent CDDs. Then, in Section 3.3 we tackle Requirement R2 by showing the goodness of fit of the TLAC model for our dataset.

### 3.2 TLAC Model

Given these constraints, we start solving Problem 1 by explaining the evolution of the calls duration by a survival analysis perspective. We consider that all the calls  $c_1, c_2, \dots, c_C$  made by a user in a month are individuals which are alive while they are active. When a phone call  $c_j$  starts, its initial lifetime  $l_j = 1$  and, as time goes by,  $l_j$  progressively increments until the call is over. It is obvious that the final lifetime of every  $c_j$  would be its duration  $d_j$ .

In the survival analysis literature, an interesting survival model that can intuitively explain the lifetime, i.e. duration, of the phone calls is the *log-logistic* distribution. And besides its use in survival analysis [12,11], there are examples in the literature of the use of the log-logistic distribution to model the distribution of wealth [5], flood frequency analysis [14] and software reliability [6]. All of these examples present a modified version of the well known “rich gets richer” phenomenon. First, for a variable to be “rich”, it has to face several risks of “dying” but, if it survives, it is more likely to get “richer” at every time. We propose that the same occurs for phone calls durations. After the initial risks of hanging up the call, e.g., wrong number calls, voice mail calls and short message calls such as “I am busy, talk to you later” or “I am here. Where are you?” type of calls, the call tends to get longer at every time. As an example, the lung cancer survival analysis case [1] parallels our environment if we substitute endurance to disease with propensity to talk: a patient/customer that has stayed alive/talking so far, will remain such, for more time, i.e., the longer is the duration of the call so far, the more the parties are enjoying the conversation and the more the call will survive.

Thus, to solve Problem 1 we propose the **Truncated Lazy Contractor (TLAC)** model, that is a truncated version of the log-logistic distribution, since it not contains the interval [01). Firstly we show, in Figure 1-a, the Probability Density Function (PDF) of the TLAC, the log-normal and exponential distributions, in order to emphasize the main differences between these models. The parameters were chosen accordingly to a median call duration of 2 minutes for all distributions. The TLAC and log-normal distributions are very similar, but the TLAC is less concentrated in the median than the

log-normal, i.e., it has power law increase ratios in its head and in its tail. We believe that this is another indication that the TLAC is suitable to model the users' CDD, since as it was verified by [20], CDDs have semi-heavy" tails. The basic formulas for the log-logistic distribution and, consequently, for the TLAC , are [11]:

$$PDF_{TLAC}(x) = \frac{\exp(z(1 + \sigma) - \mu)}{(\sigma(1 + e^z))^2}$$

$$CDF_{TLAC}(x) = \frac{1}{1 + \exp(-\frac{(\ln(x)-\mu)}{\sigma})}$$

$$z = (\ln(x) - \mu)/\sigma$$

where  $\mu$  is the location parameter and  $\sigma$  the shape parameter.

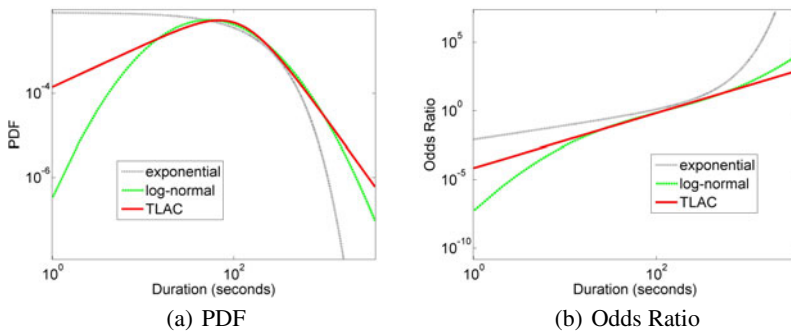


Fig. 1. Comparison among the shapes of the log-normal, exponential and TLAC distributions

Moreover, in finite sparse data that spans for several orders of magnitude, that is the case of CDDs when they are measured in seconds, it is very difficult to visualize the PDFs, since the distribution is considerably noisy at its tail. One option is to smooth the data by reducing its magnitude by aggregating data into buckets, with the cost of lost of information. Another option is to move away from the PDF and analyze the cumulative distributions, i.e., cumulative density function (CDF) and complementary cumulative density function (CCDF) [2]. These distributions veil the sparsity of the data and also the possible irregularities that may occur for any particular reason. However, by using the CDF (CCDF) you end up losing the information in the tail (head) of the distribution. In order to escape from this drawbacks, we propose the use of the Odds Ratio (OR) function, that is a cumulative function where we can clearly see the distribution behavior either in the head and in the tail. This  $OR(t)$  function is commonly used in the survival analysis and it measures the ratio between the number of individuals that have not survived by time  $t$  and the ones that survived. Its formula is given by:

$$OR(t) = \frac{CDF_{TLAC}(t)}{1 - CDF_{TLAC}(t)} \tag{1}$$

Therefore, in Figure 2-b, we plot the OR function for the TLAC , the log-normal and exponential distributions. The OR function of the exponential distribution is a power

law until  $t$  reaches the median, and then it grows exponentially. On the other hand, the OR function of the log-normal grows slowly in the head and then fastly in the tail. Finally, the OR function for the TLAC is the most interesting one. When plotted in log-log scales, is a straight line, i.e., it is a power law. Thus, as shown in [1], the  $OR(t)$  function can be summarized by the following linear regression model:

$$\ln(OR(t)) = \rho \ln(t) + \beta \quad (2)$$

$$OR(t) = e^{\beta t^{\rho}} \quad (3)$$

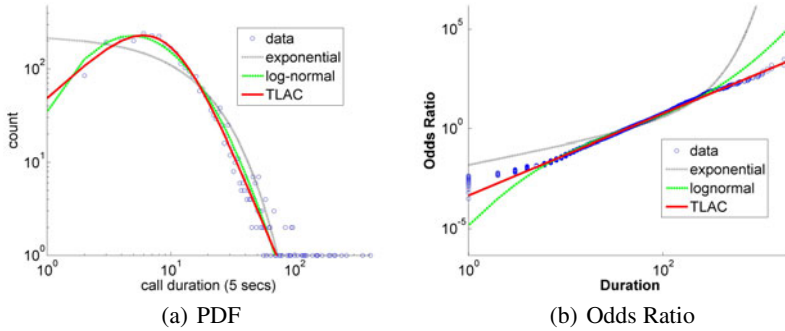
In our context, Equation 2 means that the ratio between the number of calls that will die by time  $t$  and the ones that will survive grows with a power of  $\rho$ . Moreover, given that the median  $\hat{t}$  of the CDD is given when  $OR(t) = 1$  and  $OR(t) < 1$  when  $t < \hat{t}$ , the probability of a call to end grows with  $t$  when  $t < \hat{t}$  and then decrease forever. We call this phenomenon the “lazy contractor” effect, which represents the time a lazy contractor takes to complete a job. If the job is easier and does require less effort than the ordinary regular job, he finishes it fastly. However, for jobs that are harder and that demand more work than the ordinary regular job, the contractor also gets more lazier and takes even more time to complete it, i.e., the longer a job is taking to be completed, the longer it will take. The  $\rho$  and the  $\beta$  are the parameters of the TLAC model, with  $\rho = 1/\sigma$ .

We conclude this section and, therefore, the first part of the solution to Problem 1 by explaining the intuition behind the parameters of the TLAC model. The parameter  $\rho$  is the *efficiency* coefficient, which measures how efficient is the contractor. The higher the  $\rho$ , the more efficient is the contractor and the faster he will complete the job. On the other hand, the location parameter  $\beta$  is the *weakness* coefficient, which gives the duration  $\hat{t}$  of the typical regular job a contractor with a determined efficiency coefficient  $\rho$  can take without being lazy, where  $\hat{t} = \exp(-\beta/\rho)$ . This means that the lower the  $\beta$ , the harder are the jobs that the contractor is used to handle.

### 3.3 Goodness of Fit

In this section, we tackle the second requirement of Problem 1 by showing the goodness of fit of our TLAC model. First, we show in Figure 2-a, the PDF of the CDD for a high talkative user, with 3091 calls, and with the values put in buckets of 5 seconds to ease the visualization. We also show the best fittings using Maximum Likelihood Estimation (MLE) for the exponential and the log-normal distributions and also for our proposed TLAC model. Visually, it is clear that the best fittings are the ones from the log-normal distribution and the TLAC distribution, with the exponential distribution not being able to explain either the head and the tail of the CDD.

However, by examining the OR plot in Figure 2-b, we clearly see the the TLAC model provide the best fitting for the real data. As verified for the exponential distribution in the PDF, in the OR case, the log-normal also could not explain either the head and the tail of the CDD. We also point out that we can see relevant differences between the TLAC model and the real data only for the first call durations, that happen because these regions represent only a very small fraction of the data. The results showed in Figure 2 once more validate our proposal that the TLAC is a good model for CDDs.

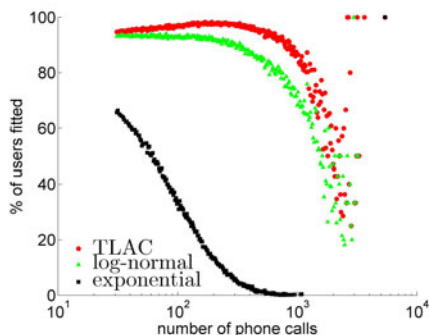


**Fig. 2.** Comparison of models for the distribution of the phone calls duration of a high talkative user, with 3091 calls. TLAC in red, log-normal in green and exponential in black. Visually, for the PDF both the TLAC and the log-normal distribution provide good fits to the CDD but, for the OR, the TLAC clearly provide the best fit.

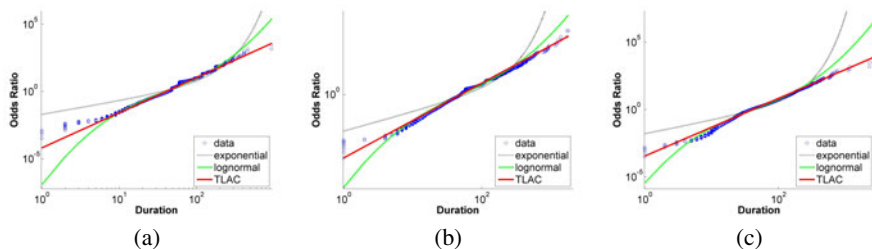
Given our initial analysis, we may state that the TLAC seems to be a good fit for the CDDs and also serve as an intuitively explanation for how the durations of the calls are generated. However, in order to conclude our answer for Problem 1, we must verify its generality power and also compare it to the log-normal and exponential generality power as well. Thus, we verify which one of the distributions can better fit the CDD of all the users of our dataset that have  $n > 30$  phone calls. We calculated, for every user, the best fit according to the MLE for the TLAC, the log-normal and exponential distributions and we performed a Kolmogorov-Smirnov goodness of fit test [13], with 5% of significance level, to verify if the user's CDD is either one of these distributions. For now on, every time we mention that a distribution was correctly fitted, we are implying that we successfully performed a Kolmogorov-Smirnov goodness of fit test.

In Figure 3, we show the percentage of CDDs that could be fitted by a log-normal, a TLAC and a exponential distribution. As we can see, the TLAC distribution can explain the highest fraction of the CDDs and the exponential distribution, the lowest. We observe that the TLAC distribution correctly fit almost 100% of the CDDs for users with  $n < 1000$ . From this point, the quality of the fittings starts to decay, but significantly later than the log-normal distribution. We emphasize that the great majority of users have  $n < 1000$ , what indicates that some of these talkative users' CDD are probably driven by non natural activities, such as spams, telemarketing or other strong comercial-driven intents. This result, allied to the fact that the TLAC distribution could model more than 96% of the users, make it reasonable to answer Problem 1 claiming that the TLAC distribution is the standard model for CDDs in our dataset.

Finally, we further explore Problem 1 by looking at the OR of the talkative users that were not correctly fitted by the TLAC model. In Figure 4, we show the OR for three of these users and, as we observe, even these customers have a visually good fitting to the TLAC model. These results corroborate even more with the generality power of TLAC. Despite of the fact that the irregularities of these customers' CDDs unable them to be correctly fitted by the TLAC model, it is clear that the TLAC can represent their CDDs significantly well.



**Fig. 3.** Percentage of users' CDDs that were correctly fitted vs. the user's number of calls  $c$ . The TLAC distribution is the one that provided better fittings for the whole population of customers with  $c > 30$ . It correctly fitted more than 96% of the users, only significantly failing to fit users with  $c > 10^3$ , probably spammers, telemarketers or other non-normal behavior user.



**Fig. 4.** Odds ratio of 3 talkative customers that were not correctly fitted by the TLAC model

## 4 TLAC over Time

We know it is trivial to visualize the distribution of users with a determined summarized attribute, such as number of phone calls per month or aggregate calls duration. However, if we want to visualize the distribution and evolution of a temporal feature of the user such as his CDD, things start to get more complicated. Thus, in this section, we tackle the following problem:

*Problem 2. EVOLUTION.* Given the  $\rho_i$  and  $\beta_i$  parameters of  $N$  customers ( $i = 1, 2, \dots, N$ ), describe how they collectively evolve over time.

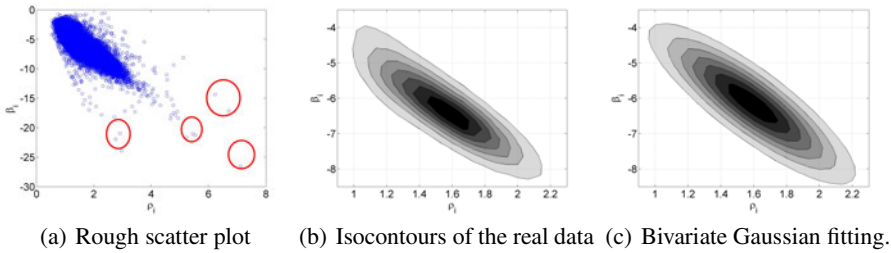
We propose two approaches to solve Problem 2. In Section 4.1 we describe the *MetaDist* solution and, in Section 4.2 we describe the *Focal Point* approach.

### 4.1 Group Behavior and Meta-fitting

Since we know that the great majority of users' CDD can be modeled by the TLAC model, in order to solve Problem 2, we need to figure out how each user  $i$  is distributed according to their parameters  $\rho_i$  and  $\beta_i$  of the TLAC model. If the meta-distribution

of the parameters  $\rho_i$  and  $\beta_i$  is well defined, then we can model the collective call behavior of the users and see its evolution over time. From now on, we will call the meta-distribution of the parameters  $\rho_i$  and  $\beta_i$  the *MetaDist* distribution.

In Figure 5-a, we show the scatter plot of the parameters  $\rho_i$  and  $\beta_i$  of the CDD of each user  $i$  for the first month of our dataset. We can not observe any latent pattern due to the overplotting but, however, we can spot outliers. Moreover, by plotting the  $\rho_i$  and  $\beta_i$  parameters using isocontours, as shown in Figure 5-b, we automatically smooth the visualization by desconsidering low populated regions. While darker colors mean a higher concentration of pairs  $\rho_i$  and  $\beta_i$ , white color mean that there are no users with CDDs with these values of  $\rho_i$  and  $\beta_i$ .



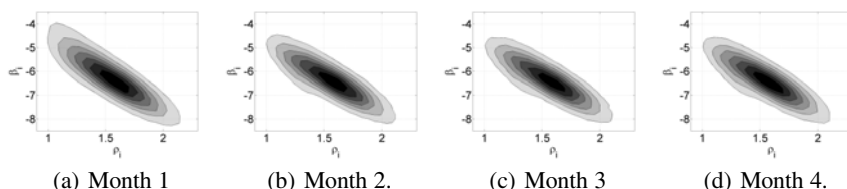
**Fig. 5.** Scatter plot of the parameters  $\rho_i$  and  $\beta_i$  of the CDD of each user  $i$  for the first month of our dataset. In (a) we can not see any particular pattern, but we can spot outliers. By plotting the isocontours (b), we can observe how well a bivariate Gaussian (c) fits the real distribution of the  $\rho_i$  and  $\beta_i$  of the CDDs ('meta-fitting').

Surprisingly, we observe that the isocontours of Figure 5-b are very similar to the ones of a bivariate Gaussian. In order to verify this, we extracted from the *MetaDist* distribution the means  $P$  and  $B$  of the parameters  $\rho_i$  and  $\beta_i$ , respectively, and also the covariance matrix  $\Sigma$ . We use these values to generate the isocontours of a bivariate Gaussian distribution and we plotted it in Figure 5-c. We observe that the isocontours of the generated bivariate Gaussian distribution are similar to the ones from the *MetaDist* distribution, which indicates that both distributions are also similar. Thus, we conjecture that a bivariate Gaussian distribution fits the real distribution of  $\rho$  and  $\beta$ s, making the *MetaDist* a good model to represent the population of users with a determined calls duration behavior.

Given that the *MetaDist* is a good model for the group behavior of the customers in our dataset, we can now visualize and measure how them evolve over time. In Figure 6 we show the evolution of the *MetaDist* over the four months of our dataset. The first observation we can make is that the bivariate Gaussian shape stands well during the whole analyzed period, what validates the robustness of the *MetaDist*. Moreover, a primarily view indicates that the meta-parameters also have not change significantly over the months. This can be confirmed by the first 5 rows of Table 1, which describes the value of the meta-parameters  $P$ ,  $B$  and  $\Sigma(\sigma_{\rho_i}^2, \sigma_{\beta_i}^2, cov(\rho_i, \beta_i))$  for the four analyzed months. This indicates that the phone company already reached a stable state before its customers concerning its prices, plans and services. In fact, the only noticeable

difference occurs between the first month and the others. We observe that the meta-parameters of the first month have a slightly higher variance than the others, what indicates that this is probably an atypical month for the residents of the country in which our phone records were collected. But in spite of that, in general, the meta-parameters do not change through time. Then, we can state the following observation:

**Observation 1. TYPICAL BEHAVIOR.** *The typical human behavior is to have a efficiency coefficient  $\rho \approx 1.59$  and a weakness coefficient  $\beta \approx -6.25$ . Thus, the median duration for a typical mobile phone user is 51 seconds and the mode is 20 seconds.*



**Fig. 6.** Evolution of the *MetaDist* over the four months of our dataset. Note that the collective behavior of the customers is practically stable over time.

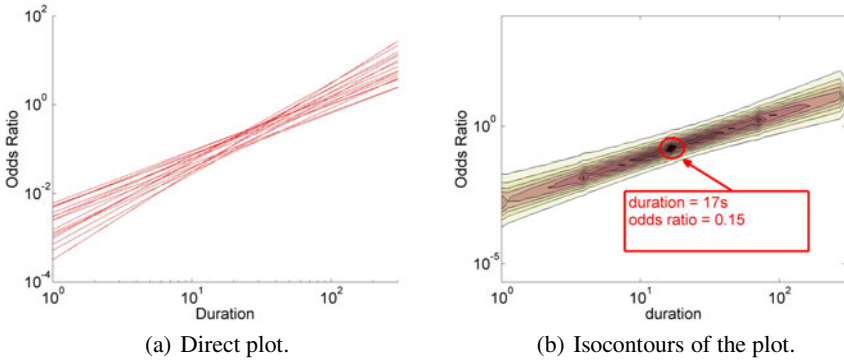
## 4.2 Focal Point

An interesting observation we can derive from the *MetaDist* showed in Figure 5 is that there exists a significant negative correlation between the parameters  $\rho_i$  and  $\beta_i$ . This negative correlation, more precisely of  $-0.86$ , lead us to the fact that the OR lines, i.e., the TLAC odds ratio plots of the customers of our dataset, when plotted together, should cross over a determined region. In order to verify this, we plotted in Figure 7-a the OR lines for some customers of our dataset. As we can observe, it appears that these lines are all crossing in the same region, when the duration is approximately 20 seconds and the odds ratio approximately 0.1. Then, in Figure 7-b, we plotted together the OR lines of 20,000 randomly picked customers and derived from them the isocontours to show the most populated areas. As we can observe, there is a highly populated point when the duration is 17 seconds and the OR is 0.15. By analyzing the whole month dataset, we verified that more than 50% of the users have OR lines that cross this point. From now on, we call this point the *Focal Point*.

Formally, the *Focal Point* is a point on the OR plot with two coordinates: a coordinate  $FP_{duration}$  in the duration axis and a coordinate  $FP_{OR}$  in the OR axis. When a set of customers have their OR plots crossing at a *Focal Point* with coordinates  $(FP_{duration}, FP_{OR})$ , it means that for all these customers the  $\frac{FP_{OR}}{1+FP_{OR}}$ -th percentile of their CDD is on  $FP_{duration}$  seconds. Thus, in the 2 bottom lines of Table 1, we describe the *Focal Point* coordinates for the four months of our analysis and, surprisingly, the *Focal Point* is stationary. Thus, we can make the following observation:

**Observation 2. UNIVERSAL PERCENTILE.** *The vast majority of mobile phone users has the same 10th percentile, that is on 17 seconds.*





**Fig. 7.** The TLAC lines of several customers plotted together. We can observe that, given the negative correlation of the parameters  $\rho_i$  and  $\beta_i$ , that the lines tend to cross in one point (a). We plot the isocontours of the lines together and approximately 50% of the customers have TLAC lines that pass on the high density point (duration=17s, OR=0.15) (b).

Observation 2 suggests that one of the risks for a call to end acts in the same way for everyone. We conjecture that, given the 17 seconds durations, this is the risk of a call to reach the voice mail of the destination’s mobile phone, i.e., the callee could not answer the call. The duration of this call involves listening to the voice mail record and leaving a message, what is coherent with the 17 seconds mark. It would be interesting to empirically verify the percentage of phone calls that reaches the voice mail and compare with the *Focal Point* result.

**Table 1.** Evolution of the meta-parameters (rows 1-5) and the *Focal Point* (rows 6-7) during the four months of our dataset

-	1st month	2nd month	3rd month	4th month
$P$	1.59	1.58	1.59	1.59
$B$	-6.16	-6.28	-6.32	-6.30
$\sigma_{\rho_i}^2$	0.095	0.086	0.084	0.083
$\sigma_{\beta_i}^2$	1.24	0.98	0.95	0.94
$cov(\rho_i, \beta_i)$	-0.30	-0.24	-0.24	-0.23
$FP_{duration}(s)$	17	17	17	17
$FP_{OR}$	0.15	0.12	0.11	0.11

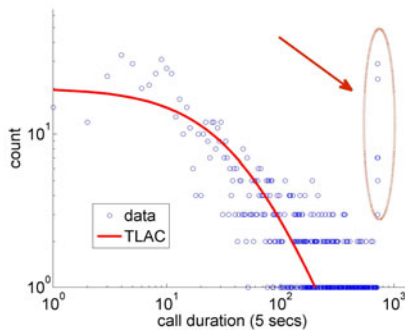
## 5 Discussion

### 5.1 Practical Use

In the previous section, we showed the collective behavior of millions of mobile phone users is stationary over time. We described two approaches to do that, one based on the *MetaDist* and the other based on the *Focal Point*. The initial conclusions of both approaches are same. First, the collective behavior of our dataset is stable, i.e., it does

not change significantly over time. Second, we could see a slight difference between the first month and the others, indicating that this month is an atypical month in the year. We believe that these two approaches can succinctly and accurately aid the mobile phone companies to monitor the collective behavior of their customers over time.

Moreover, since we could successfully model more than 96% of the CDDs as a TLAC, a natural application of our models would be for anomaly detection and user classification. A mobile phone user that does not have a CDD that can be explained by the TLAC distribution is a potential user to be observed, since he has a distinct call behavior from the majority of the other users. To illustrate this, we show in Figure 8 a talkative node with a CDD that can not be modeled by a TLAC distribution. We observe that this node, indeed, has an atypical behavior, with his CDD having a noisy behavior from 10 to 100 seconds and also an impressive number of phone calls with duration around 1 hour (or  $5 \times 700$  seconds). Moreover, another way to spot outliers is to check which users have a significant distance from the main cluster of the *MetaDist*. As we showed in Figure 5-c, this can be easily done even visually.



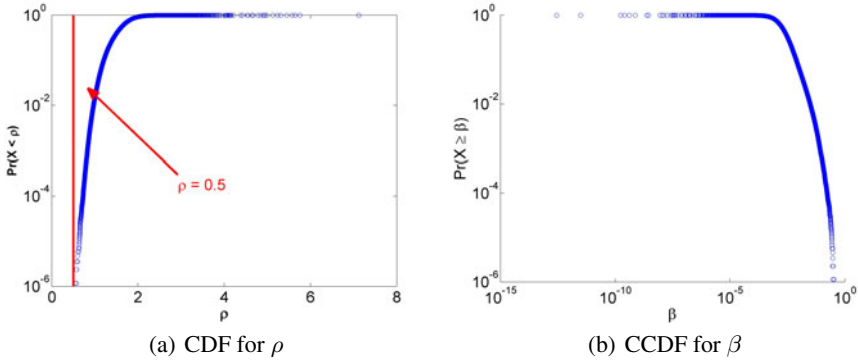
**Fig. 8.** Outlier whose CDD can not be modeled by the TLAC distribution

Another application that emerges naturally for our models is the summarization of data. By modeling the users' CDD into TLAC distributions, we are able to summarize, for each user  $i$ , hundreds or thousands of phone calls into just two values, the parameters  $\rho_i$  and  $\beta_i$  of the TLAC model. In our specific case, we could summarize over  $0.1TB$  of phone calls data into less than  $80MB$  of data. In this way, it is completely feasible to analyze several months, or even years of temporal phone calls data and verify how the behavior of the users is evolving through time. Also, all the proposed models in this work can be directly applied on the design of generators that produce synthetic data, allowing researchers that do not have access to real data to generate their own.

## 5.2 Generality of TLAC

As we mentioned earlier, one of the major strengths of the TLAC model is its generality power. We showed that even for distributions that oscillate between log-normal and log-logistic, or that have irregular spikes that unable them to be correctly fitted by TLAC, TLAC can represent them significantly well. Besides this, the simplicity of the TLAC

model allow us to directly understand its form when its parameters are changed and verify its boundaries. For instance, in the case of the CDD,  $e^\beta$  gives the odds ration when duration is 1 second. Thus, when  $e^\beta > 1$ , most of the calls have a lower duration than 1 second, which makes the CDD converges to a power law, i.e., the initial spike is truncated. Moreover, as  $\alpha \rightarrow 0$ , the odds ratio tends to be the constant  $e^\beta$ , what causes the variance to be infinity. By observing Figure 9 and concerning human calling behavior, we conjecture that  $\beta$  is upper bounded by 1 and  $\rho$  is lower bounded by 0.5. These values are coherent with the global intuition on human calling behavior.

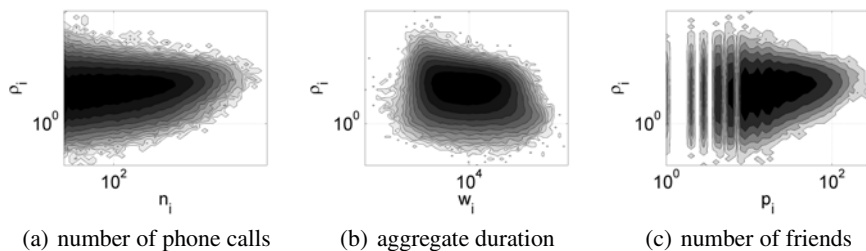


**Fig. 9.** Cumulative distributions for  $\rho$  and  $\beta$ . We can observe that  $\rho$  is lower bounded by 0.5 and  $\beta$  is upper bounded by 1. These values are coherent with the global intuition on human calling behavior.

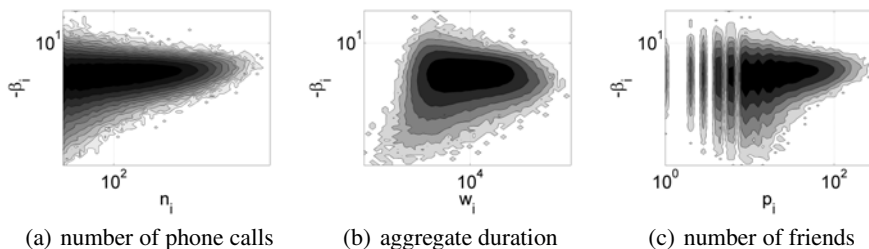
### 5.3 Additional Correlations

Given that the vast majority of users’ CDDs can be represented by the TLAC model, it would be interesting if we could predict their parameters  $\rho_i$  and  $\beta_i$  based on one of their summarized attributes. One could imagine that a user that makes a large number of phone calls per month might have a distinct CDD than a user that makes only a few. Moreover, we could also think that a user that has many friends and talk to them by the phone regularly may also have a distinct CDD from a user that only talks to his family on the phone. In Figures 10 and 11, we show, respectively, the the isocontours of the behavior of the  $\rho_i$  and  $\beta_i$  parameters for users with different values of number of phone calls  $n_i$ , aggregate duration  $w_i$  and number of partners  $p_i$ , i.e., the distinct number of persons that the user called in a month. With the exception made for the  $\rho_i$  against  $w_i$ , we observe that the variance decreases as the value of the summarized attribute increases. This suggests that the CDD of high or long talkative users, as well as users with many partners, is easier to predict. Moreover, as we can observe in the figures and also in Table 2, there is no significant correlation between the TLAC parameters and the summarized attributes of the users. Thus, we make the following observation:

**Observation 3. INVARIANT BEHAVIOR.** *The  $\rho_i$  and  $\beta_i$  parameters of user  $i$  behave as invariant with respect to (a) number of phone calls  $n_i$ , (b) aggregate duration  $w_i$  and (c) number of partners  $p_i$ .*



**Fig. 10.** Isocontours of the users' CDD efficiency coefficient  $\rho$  and their summarized attributes



**Fig. 11.** Isocontours of the users' CDD efficiency coefficient  $\beta$  and their summarized attributes

**Table 2.** Correlations between summarized attributes and  $\rho$  and  $\beta$

Attribute	Correlation with $\rho$	Correlation with $\beta$
number of phone calls	0.14	-0.18
aggregate duration	-0.21	0.01
number of partners	0.18	-0.18

Finally, since there is no significant correlation between the users' CDD parameters  $\rho_i$  and  $\beta_i$  with their summarized attributes, we emphasize that these parameters should be considered when characterizing user behavior in phone call networks. Moreover, besides characterizing individual customers, the TLAC model can also be directly applied to the relationship between users, analyzing how two persons call each other. One could use, for instance, the  $\rho$  parameter as the weight of the edges of the social network generated from phone call records.

## 6 Conclusions

In this paper, we explored the behavior of the calls' duration of the users of a large mobile company of a large city. We analyzed more than 3 million customers and 1 billion phone calls records. The main contributions of the paper are:

- The proposal of the TLAC distribution, which fits very well the vast majority of individual phone call durations, *much better* than log-normal and exponential;

- the introduction of *MetaDist*, which shows that the *collection* of TLAC parameters, and specifically the  $\rho$  and  $\beta$  ones, follow a striking bivariate Gaussian, with mean  $(P, B)$ ;
- Temporal evolution: the discovery that the *MetaDist* remains the same over time, with very small fluctuations;
- Usefulness of TLAC : it can spot anomalies (see Figure 8) and it can succinctly describe spot correlations (or lack thereof) between total phone call duration, number of calls, and number of distinct patterns, for a given user.

Moreover, we showed that TLAC has a very natural, intuitive explanation behind it (the more you waited so far, the even longer you will wait), and that it includes as special case the Pareto distribution.

Future work could focus on network effects, that is, if two people talk to each other, what is the relationship between their TLAC parameters? A second promising direction is to check whether TLAC also fits well other modes of human (or computer) communications, like length of SMS messages and length of postings on FaceBook “walls”.

**Acknowledgments.** We thank the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) for financial support. Research was also sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

## References

1. Bennett, S.: Log-logistic regression models for survival data. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 32(2), 165–171 (1983)
2. Clauset, A., Shalizi, C.R., Newman, M.E.J.: Power-law distributions in empirical data. *SIAM Review* 51(4), 661 (2009), <http://dx.doi.org/10.1137/070710111>
3. Cortes, C., Pregibon, D., Volinsky, C.: Communities of interest. In: Hoffmann, F., Hand, D.J., Adams, N.M., Fisher, D.H., Guimarães, G. (eds.) *IDA 2001*. LNCS, vol. 2189, pp. 105–114. Springer, Heidelberg (2001)
4. Du, N., Faloutsos, C., Wang, B., Akoglu, L.: Large human communication networks: patterns and a utility-driven generator. In: *KDD '09: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 269–278. ACM, New York (2009)
5. Fisk, P.R.: The graduation of income distributions. *Econometrica* 29(2), 171–185 (1961)
6. Gokhale, S.S., Trivedi, K.S.: Log-logistic software reliability growth model. In: *HASE '98: The 3rd IEEE International Symposium on High-Assurance Systems Engineering*, pp. 34–41. IEEE Computer Society Press, Washington (1998)
7. Guo, J., Liu, F., Zhu, Z.: Estimate the call duration distribution parameters in gsm system based on k-l divergence method. In: *International Conference on Wireless Communications, Networking and Mobile Computing, WiCom 2007*, pp. 2988–2991 (September 2007)

8. Hidalgo, C.A., Rodriguez-Sickert, C.: The dynamics of a mobile phone network. *Physica A: Statistical Mechanics and its Applications* 387(12), 3017–3024 (2008)
9. Hill, S., Nagle, A.: Social network signatures: A framework for re-identification in networked data and experimental results. In: *CASON '09: Proceedings of the 2009 International Conference on Computational Aspects of Social Networks*, pp. 88–97. IEEE Computer Society, Washington (2009)
10. Hill, S., Provost, F.J., Volinsky, C.: Learning and inference in massive social networks. In: Frasconi, P., Kersting, K., Tsuda, K. (eds.) *MLG (2007)*
11. Lawless, J.F.: *Statistical Models and Methods for Lifetime Data* (Wiley Series in Probability & Mathematical Statistics). John Wiley & Sons, Chichester (1982)
12. Mahmood, T.: Survival of newly founded businesses: A log-logistic model approach. *Journal Small Business Economics* 14(3), 223–237 (2000)
13. Massey, F.J.: The kolmogorov-smirnov test for goodness of fit. *Journal of the American Statistical Association* 46(253), 68–78 (1951), <http://dx.doi.org/10.2307/2280095>
14. Ahmad, M.I., Sinclair, C.D., Werritty, A.: Log-logistic flood frequency analysis. *Journal of Hydrology* 98, 205–224 (1988)
15. Nanavati, A.A., Gurumurthy, S., Das, G., Chakraborty, D., Dasgupta, K., Mukherjea, S., Joshi, A.: On the structural properties of massive telecom call graphs: findings and implications. In: *CIKM '06: Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, pp. 435–444. ACM, New York (2006)
16. Onnela, J.P., Saramäki, J., Hyvönen, J., Szabó, G., Lazer, D., Kaski, K., Kertész, J., Barabási, A.L.: Structure and tie strengths in mobile communication networks. *Proceedings of the National Academy of Sciences* 104(18), 7332–7336 (2007)
17. Onnela, J.P., Saramaki, J., Hyvonen, J., Szabo, G., de Menezes, M.A., Kaski, K., Barabasi, A.L., Kertesz, J.: Analysis of a large-scale weighted network of one-to-one human communication. *New Journal of Physics* 9(6), 179 (2007)
18. Seshadri, M., Machiraju, S., Sridharan, A., Bolot, J., Faloutsos, C., Leskove, J.: Mobile call graphs: beyond power-law and lognormal distributions. In: *KDD '08: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 596–604. ACM, New York (2008)
19. Tejinder, S., Randhawa, S.H.: *Network Management in Wired and Wireless Networks*. Springer, New York (2003)
20. Willkomm, D., Machiraju, S., Bolot, J., Wolisz, A.: Primary users in cellular networks: A large-scale measurement study. In: *3rd IEEE Symposium on New Frontiers in Dynamic Spectrum Access Networks, DySPAN 2008*, pp. 1–11 (2008)

# Variational Bayesian Mixture of Robust CCA Models

Jaakko Viinikanoja, Arto Klami, and Samuel Kaski

Aalto University School of Science and Technology  
Department of Information and Computer Science  
Helsinki Institute for Information Technology HIIT

<http://www.cis.hut.fi/projects/mi/>

**Abstract.** We study the problem of extracting statistical dependencies between multivariate signals, to be used for exploratory analysis of complicated natural phenomena. In particular, we develop generative models for extracting the dependencies, made possible by the probabilistic interpretation of canonical correlation analysis (CCA). We introduce a mixture of robust canonical correlation analyzers, using t-distribution to make the model robust to outliers and variational Bayesian inference for learning from noisy data. We demonstrate the improvements of the new model on artificial data, and further apply it for analyzing dependencies between MEG and measurements of autonomic nervous system to illustrate potential use scenarios.

**Keywords:** Bayesian data analysis, canonical correlation analysis, data fusion, latent variable models, robust models.

## 1 Introduction

Noisy estimates of human brain activity can be obtained with several measurement techniques, such as functional magnetic resonance imaging (fMRI) and magnetoencephalography (MEG). Given a controlled experiment, even relatively simple approaches can shed light on brain functions. For example, linear regression from brain activity to stimulus covariates can reveal which brain regions are related to the task at hand. For uncontrolled experiments, such as analysis of the brain functions in natural environments, the classical approaches, however, fall short since there are no simple covariates available and unsupervised analysis cannot separate the relevant variation from the rest.

A recent approach to tackling the problem is to consider correlations between the brain activity measurements and multivariate vectorial representations of the stimulus [6, 17]. This allows using the stimulus still as a supervision signal, despite the representation not being condensed as simple covariates. Instead, we need to assume the stimulus representation contains noise just like the brain activity measurements do. The actual signal is separated from the noise by making one simple assumption: Statistical dependencies between the brain activity and the stimulus must be related to processing the stimulus, while independent variation in either signal should be seen as structured noise.

Canonical correlation analysis (CCA) is a standard approach for finding correlations between two multivariate sources (see, *e.g.*, [9]), and is the method used in [17] for analysis of fMRI data under natural stimulation. CCA has, however, several limitations that make it suboptimal in practical applications. It assumes the signals are stationary, does not come with an easy way of estimating the number of correlated components, is not robust against outliers, and estimating the reliability of the components is difficult. Building on the probabilistic interpretation of CCA, we have earlier introduced a model that removes the stationarity assumption through mixture modeling and automatically learns the model complexity from data [10]. The earlier model, however, has practical limitations that prevent using it for neuroinformatics applications. In this article we improve the model further, by introducing more efficient and more easily interpretable inference procedure, and by making the model robust to outliers.

The model in [10] used posterior sampling for inference, and was formulated as a Dirichlet process mixture for estimating the model complexity, which makes the model suitable for small sample sizes but the inference becomes very inefficient for large data sets. In particular, the model does not have a fully conjugate prior, and hence less efficient sampling strategies need to be applied. Consequently, applying the model for analysis of MEG data would be beyond computationally feasible by some orders of magnitude. Furthermore, neuroscientific interpretation of the results of such a model would be difficult since the posterior sampler returns a set of results that need to be processed further for conclusive summaries. In this paper we solve both of these issues by switching to variational inference, which results in highly efficient optimization and also makes interpretation more straightforward since the approach is deterministic, while retaining the mixture capability and automatic relevance determination prior for inferring the number of correlating components.

The robustness to outliers is obtained by replacing the generative assumption of Gaussian noise by that of Student's *t*-distribution, modeled as a scale-mixture [11]. Similar representation was earlier used in the robust CCA variant of [2], but they only sought a maximum likelihood estimate for the model parameters instead of considering the full posterior distribution. Robust variational inference has earlier been presented only for simpler projection methods such as robust PCA [8,12] and robust factor analysis [5], using two different alternative approximations that have not been compared earlier. We show that there is no noticeable difference in accuracy or computational complexity between the two alternatives.

We illustrate the technical properties of the model using artificial data, showing the improvements in a set of experiments. We also demonstrate what kind of real analysis scenarios the model is useful for, by using it to learn dependencies between brain oscillations and autonomic nervous system (ANS) response under emotional sound stimuli. The emotions of the user are strongly visible in ANS but only vaguely in MEG, and correlations between these two pinpoint possible hypotheses on what part of the variation in the signals captured by MEG might be related to the emotions. In brief, the ANS measurements can be considered as



noisy descriptions of the stimulus itself. The model is shown to find interpretable clusters that capture much stronger correlations than stationary analysis could. The model also outperforms the alternative of first clustering the data and then applying CCA separately for each of the clusters [7].

## 2 Bayesian CCA

The probabilistic CCA (PCCA) [3] is a generative probabilistic model for two multi-dimensional data sources  $\mathbf{X}_1 = [\mathbf{x}_{11}, \dots, \mathbf{x}_{1N}]$  and  $\mathbf{X}_2 = [\mathbf{x}_{21}, \dots, \mathbf{x}_{2N}]$ . The model is written as

$$\begin{aligned} \mathbf{t}_n &\sim \mathcal{N}(\mathbf{t}_n | 0, \mathbf{I}_D) \\ \mathbf{x}_{1n} | \mathbf{t}_n &\sim \mathcal{N}(\mathbf{x}_{1n} | \mathbf{W}_1 \mathbf{t}_n + \boldsymbol{\mu}_1, \boldsymbol{\Psi}_1) \\ \mathbf{x}_{2n} | \mathbf{t}_n &\sim \mathcal{N}(\mathbf{x}_{2n} | \mathbf{W}_2 \mathbf{t}_n + \boldsymbol{\mu}_2, \boldsymbol{\Psi}_2), \end{aligned} \quad (1)$$

where the  $\boldsymbol{\Psi}$  denote the precision matrices of the normal distribution. The latent variables  $\mathbf{t}$  encode the low-dimensional statistically dependent part while projection matrices  $\mathbf{W}_1$  and  $\mathbf{W}_2$  specify how this dependency is manifested in each of the data sources. Bach and Jordan [3] established the connection between this probabilistic model and classical CCA by showing that the maximum-likelihood solution of the model coincides with the classical solution except for an arbitrary rotation in the latent space and projection matrices. PCCA as such does not solve any of the problems classical CCA has since it is merely an equivalent description, but the probabilistic formulation makes justified extensions possible. In the remainder of this section, we walk through the extensions and modifications required for creating a practically applicable dependency modeling tool for real-world signals.

The first step is to make the inference more reliable by switching from the maximum likelihood solution to full Bayesian analysis, by complementing the likelihood with priors for the model parameters. We adopt the formulation of [10,16] for Bayesian CCA (BCCA)

$$\begin{aligned} \mathbf{w}_{ij} | \alpha_i &\sim \mathcal{N}(\mathbf{w}_{ij} | 0, \text{diag}(\alpha_{i1}, \dots, \alpha_{iD})) \\ \alpha_{ij} &\sim \mathcal{G}(\alpha_{ij} | a_i, b_i) \\ \boldsymbol{\Psi}_i &\sim \mathcal{W}(\boldsymbol{\Psi}_i | \gamma_i, \boldsymbol{\Phi}_i) \\ \boldsymbol{\mu}_i &\sim \mathcal{N}(\boldsymbol{\mu}_i | 0, \beta_i \mathbf{I}), \end{aligned} \quad (2)$$

where  $\mathcal{G}$  is the gamma distribution,  $\mathcal{W}$  denotes the Wishart distribution, the subscript  $i$  is used to denote the data sources, and the distribution of data is given in (1). The rest of the symbols are hyper-priors of the model. The priors for the projection matrix row vectors  $p(\mathbf{w}_{ij} | \alpha_i)$  and the precision prior  $p(\alpha_{ij})$  implement the Automatic Relevance Determination (ARD) [14] which automatically controls the number of the components in the model by adjusting the precisions  $\alpha_{ij}$  – the precisions for unnecessary components are driven to infinity, and hence the posterior peaks around the zero vector. Both [10,16] experimentally verified

that the ARD mechanism detects the correct dimensionality of the latent space, which is the second necessary component for our practically usable model.

The Bayesian CCA model makes a strict assumption of Gaussian noise, which is problematic for many real life signals used as stimulus representations or brain activity measurements. This problem can be alleviated by replacing both the Gaussian noise and the Gaussian latent variables by Student’s t-distribution (with  $\nu$  degrees of freedom) that is more robust to outliers:

$$\begin{aligned} \mathbf{t}_n &\sim \mathcal{S}(\mathbf{t}_n|0, \mathbf{I}_D, \nu) \\ \mathbf{x}_{1n}|\mathbf{t}_n &\sim \mathcal{S}(\mathbf{x}_{1n}|\mathbf{W}_1\mathbf{t}_n + \boldsymbol{\mu}_1, \boldsymbol{\Psi}_1, \nu) \\ \mathbf{x}_{2n}|\mathbf{t}_n &\sim \mathcal{S}(\mathbf{x}_{2n}|\mathbf{W}_2\mathbf{t}_n + \boldsymbol{\mu}_2, \boldsymbol{\Psi}_2, \nu). \end{aligned}$$

For efficient inference, we exploit the latent infinite scale-mixture formulation of the t-distribution [11],

$$\mathcal{S}(\mathbf{t}|\boldsymbol{\mu}, \boldsymbol{\Lambda}, \nu) = \int_0^\infty du \mathcal{N}(\mathbf{t}|\boldsymbol{\mu}, \boldsymbol{\Lambda}) \mathcal{G}(u|\nu/2, \nu/2).$$

Using this formulation, we can write the robust CCA model by adding an extra level of hierarchy

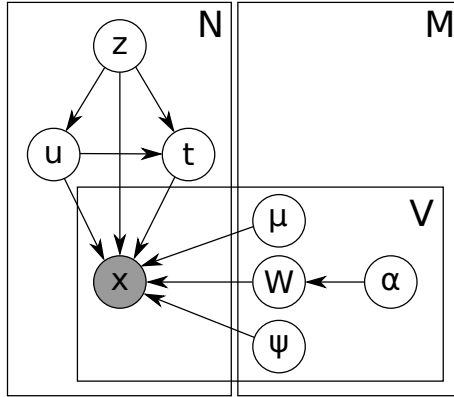
$$\begin{aligned} u_n &\sim \mathcal{G}(u_n|\nu/2, \nu/2) \\ \mathbf{t}_n|u_n &\sim \mathcal{N}(\mathbf{t}_n|0, u_n\mathbf{I}_D) \\ \mathbf{x}_{1n}|u_n, \mathbf{t}_n &\sim \mathcal{N}(\mathbf{x}_{1n}|\mathbf{W}_1\mathbf{t}_n + \boldsymbol{\mu}_1, u_n\boldsymbol{\Psi}_1) \\ \mathbf{x}_{2n}|u_n, \mathbf{t}_n &\sim \mathcal{N}(\mathbf{x}_{2n}|\mathbf{W}_2\mathbf{t}_n + \boldsymbol{\mu}_2, u_n\boldsymbol{\Psi}_2). \end{aligned}$$

This formulation has conjugate conditional distributions, which considerably simplifies inference. The above formulation for robust CCA has earlier been presented by [2], but they only considered the maximum likelihood estimate for the parameters. We couple the robust noise assumption with the priors for the Bayesian CCA model (2) to arrive at the novel model of Robust Bayesian CCA (RBCCA). It is a basic building block of our full model.

### 2.1 Mixture of Robust Bayesian CCAs

Next we turn our attention to removing the stationarity assumption, by replacing it with piecewise stationarity. In this work we follow our earlier model [10] and introduce a probabilistic mixture of robust Bayesian CCA models, letting each mixture cluster to model different kind of dependencies between the signals.

We formulate the probabilistic mixture by introducing an additional multinomial latent variable which generates the mixture assignment [13]. The robust mixture CCA model is therefore obtained by adding the latent variable  $z_n \sim \text{Multinomial}(z_n|\boldsymbol{\pi})$ , where  $\boldsymbol{\pi}$  denotes the probabilities of the clusters (we use point estimates for  $\boldsymbol{\pi}$ , but the extension to Dirichlet prior would be straightforward), and conditioning all the rest of the latent variables and parameters on the value of  $z_n$ .



**Fig. 1.** Plate diagram of the mixture of robust CCA models. N denotes the samples, M the mixture clusters and V the two data sources. The hyper-priors of the variables are excluded for clarity.

The full model including the mixture formulation for non-stationarity, ARD prior for choosing the number of correlated components within the clusters, and the t-distribution for handling outliers is represented in Figure 1. All of the models described above like RBCCA and mixture of Gaussian BCCAs are obtained as special cases of the full model, as are a number of other models. In particular, fixing  $1/\alpha_{ij} = 0$  results in (robust) Gaussian mixture model [1], setting  $\Psi$  diagonal gives (robust) factor analysis [5], and further restricting it to be spherical leads to (robust) Bayesian PCA [8,12].

## 2.2 Variational Inference

For analysis, the above model formulation needs to be coupled with an inference algorithm. In particular, we need to learn the posterior distribution of the model parameters, and be able to make predictions for future data. Since the goal is to be able to apply the model for analysis of potentially very large data sets, we steer away from the computationally heavy earlier alternatives like the Gibbs sampling approach of [10] for mixture of BCCA inference, and instead choose to use the deterministic variational approximation. The resulting algorithm is computationally as efficient as finding the maximum likelihood or maximum a posteriori estimate through the EM algorithm, but maintains the advantage of full Bayesian analysis in capturing the uncertainty in the results. Next, we briefly summarize the variational Bayesian (VB) approach for inference, and only explain in more detail the choices specific for the novel parts of the model. For more extensive introduction to variational inference see, *e.g.*, [4].

The core of the inference process is in learning the posterior distribution  $p(H|\mathbf{X}_1, \mathbf{X}_2, \Theta)$  of both the latent variables and the model parameters, denoted collectively as  $H = \{\mathbf{Z}, \mathbf{U}, \mathbf{T}, \mathbf{W}_1, \mathbf{W}_2, \mu_1, \mu_2, \Psi_1, \Psi_2\}$ , given the observed data and model hyper-parameters  $\Theta$ . Finding the true posterior is not feasible even

for the basic CCA model, let alone the full robust mixture, because evaluating the marginal log-likelihood

$$\ln p(\mathbf{X}_1, \mathbf{X}_2|\Theta) = \ln \left( \int dH p(H, \mathbf{X}_1, \mathbf{X}_2|\Theta) \right)$$

is not tractable.

With variational Bayesian inference the problem is solved by approximating the true posterior with a variational distribution  $q(H)$  from some limited class of distribution functions so that the inference remains tractable [4]. In practice, the class of distributions is limited by assuming that the full posterior factorizes into a number of (a priori) independent terms, and the optimal distribution in this class is chosen by minimizing the Kullback-Leibler divergence from the approximation to the true posterior  $D_{\text{KL}}(q(H)||p(H|\mathbf{X}_1, \mathbf{X}_2, \Theta))$ . The full posterior is then found by an iterative EM-style algorithm. The resulting update formulas, based on the factorization described below, are given in the Appendix.

Following the variational Bayesian CCA by Wang [16], we consider a variational distribution for which the parameter part is fully factorised as

$$\prod_{i=1}^2 \prod_{k=1}^M q(\Psi_i^k)q(\mu_i^k)q(\mathbf{W}_i^k)q(\alpha_i^k). \tag{3}$$

We then focus on the approximation used for the latent variables, naturally factorized over the data points as  $\prod_{n=1}^N q(z_n, u_n, \mathbf{t}_n)$ . It is clear that we need to consider separate terms for each cluster,  $q(z_n)q(u_n, \mathbf{t}_n|z_n)$ , but for the latter term two different tractable approximations are possible. For example [8,12] use the approximation  $q(u_n)q(\mathbf{t}_n)$ , assuming conditional independence between  $u_n$  and  $\mathbf{t}_n$ , whereas [5] chose  $q(u_n)q(\mathbf{t}_n|u_n)$ , not introducing any independence assumptions beyond those in the actual model. In our scenario both solutions are analytically tractable, conditioned on  $z_n$ .

To our knowledge, these two choices have not been compared before, nor the additional independence assumption justified. Since both are tractable and lead to implementations of comparable computational complexity, the relative accuracy of the two approximations is an interesting question for variational approximations of t-distribution models in general. Hence, we implemented both alternatives and empirically compare them in the experiments, showing that the difference in performance is negligible, making both alternatives valid. The formulas given in the Appendix assume the approximative  $q(u_n)q(\mathbf{t}_n)$  factorisation.

Besides learning the posterior distribution of the latent variables and model parameters, we are naturally interested in making predictions for new data. Both inferring  $\mathbf{x}_1$  given  $\mathbf{x}_2$  (or vice versa) and inferring the latent variable  $\mathbf{t}$  given  $\mathbf{x}_1$  and/or  $\mathbf{x}_2$  are useful for various application scenarios. Exact calculation of these distributions is again untractable due to the dependency on the hidden data posterior distributions. However, we can utilize the variational distributions to make the predictions tractable. For a new data point,  $q(z, u, \mathbf{t})$  is chosen as the distribution which maximizes the variational lower bound of  $\ln p(\mathbf{X}_{-i}|\Theta)$

where  $-i$  denotes the observed data source(s). As an example, we consider the predictive density  $p(\mathbf{x}_i|\mathbf{x}_{-i})$ . Using the conditional independence of  $\mathbf{x}_i$  and  $\mathbf{x}_{-i}$  when  $\mathbf{t}$  is known, and integrating out  $\mathbf{t}$  results in

$$\mathbf{x}_i|z_k = 1, u, \Psi_i, \boldsymbol{\mu}_i, \mathbf{W}_i \sim \mathcal{N}(\mathbf{x}_i|\mathbf{W}_i^k \boldsymbol{\mu}_{t_k} + \boldsymbol{\mu}_i^k, ((u\Psi_i^k)^{-1} + \mathbf{W}_i^k \boldsymbol{\Sigma}_{t_k} (\mathbf{W}_i^k)^\top)^{-1}),$$

where the information from the observed data source is encapsulated in the parameters  $\boldsymbol{\mu}_{t_k}$ ,  $\boldsymbol{\Sigma}_{t_k}$  and the gamma distribution of  $u$  (these parameters, however, are slightly different in comparison to the formulas in the Appendix as we observe only  $\mathbf{X}_{-i}$ ). The above predictive density assumes the factored approximation for the latent variables; with the non-factored alternative the density is of the same form but  $\boldsymbol{\Sigma}_{t_k}$  will explicitly depend on  $u$ .

The above expression does not yield a closed form expression for the distribution  $p(\mathbf{x}_i|\mathbf{x}_{-i})$  but at least the two first moments are analytically tractable. The most important quantity is the conditional mean

$$\mathbb{E}[\mathbf{x}_i|\mathbf{x}_{-i}] = \sum_{k=1}^M q(z_k) \langle \mathbf{W}_i^k \rangle_{q(\{\mathbf{W}_i\})} \boldsymbol{\mu}_{t_k} + \langle \boldsymbol{\mu}_i^k \rangle_{q(\{\boldsymbol{\mu}_i\})}.$$

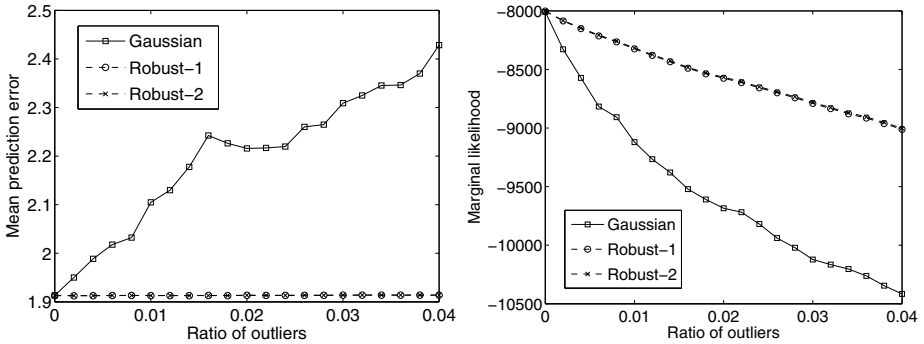
### 3 Model Validation

To validate that the model does what it promises, we performed two experiments using artificial data. First we show how replacing the Gaussian distribution with t-distribution considerably improves the accuracy of the model in presense of increasing amounts of outliers. At the same time we compare the two alternative variational factorizations for t-distributed latent variable models, showing that there is no noticeable difference in accuracy. Then we show how the model correctly captures non-stationarity with clusters and automatically extracts the correct number of correlated components.

In both of our artificial data experiments, we fix the hyperparameters to values corresponding to broad priors ( $a_i = b_i = 0.1$ ,  $\gamma_i = d_i + 1$ ,  $\Phi_i = 10^2 \mathbf{I}$ ,  $\beta_i = 1$ ) and consequently let the data determine the model parameters. The hyper-parameters  $\boldsymbol{\pi}$  and  $\nu$ :s are updated by maximizing the variational lower bound which leads to closed form and line-search update rules.

#### 3.1 Robustness against Outliers

We start by showing the importance of robust modeling in presense of outliers. We first generate data ( $N=500$ ) from the model (with single cluster), and then add a varying number of outlier data points drawn from the uniform distribution. We then compare the robust model with the two alternative variational approximations against the Gaussian model by measuring the variational lower bound and the mean error in predicting  $\mathbf{x}_1$  from  $\mathbf{x}_2$ . Figure 2 shows how the performance is identical for the case of no outliers, meaning that there is no harm in using the robust variant, and that already for fairly modest ratios of outliers the robust variant is considerably better.



**Fig. 2. Left:** The mean prediction error for the robust CCA model stays essentially constant irrespective of the number of outliers in the data, whereas the Gaussian CCA model starts losing accuracy already for a low number of outliers. Even below 1% of samples being outliers the difference in accuracy is already noticeable. **Right:** The same effect is visible also in the variational lower bound. For both measures the curves for the two alternative approximations for the robust variant are completely overlapping, showing no difference.

The results also indicate that there does not seem to be any difference between the two variational approximations for the t-distribution. To further compare the alternatives, we construct an experiment designed to emphasize potential differences. We generate the data from the t-distribution with just  $\nu = 2$  degrees of freedom, making the data very heavy-tailed (for high values of  $\nu$  the t-distribution approaches Gaussian). The model is trained for  $N = 10000$  data points, and 10 separate sets of 10000 data points are used for testing. We measure the error in predicting  $\mathbf{x}_1$  given  $\mathbf{x}_2$ , both with the mean prediction error and quantiles of the error distribution to emphasize potential tail effects. The results, collected in Table II, confirm that the accuracies are indeed comparable.

### 3.2 Model Selection

Next we show how the model comes with ready tools for choosing the model complexity. For any real analysis task both the number of clusters needed to correctly capture the non-stationary dependencies and the number of correlating components in each of the clusters are unknown. We show how the ARD prior for the projection matrices removes the need of explicitly specifying the number of components, by automatically ignoring unnecessary components, and how the marginal likelihood of the model reveals the correct number of clusters.

We created  $M = 3$  clusters each consisting of 2000 points from the model

$$\begin{aligned}
 \mathbf{t}_n &\sim \mathcal{N}(\mathbf{t}_n|0, \mathbf{I}) \\
 \mathbf{x}_{1n}|\mathbf{t}_n &\sim \mathcal{N}(\mathbf{x}_{1n}|\mathbf{W}_1^k \mathbf{t}_n + \boldsymbol{\mu}_1^k, (\mathbf{L}_1^k \mathbf{L}_1^{k\top})^{-1}) \\
 \mathbf{x}_{2n}|\mathbf{t}_n &\sim \mathcal{N}(\mathbf{x}_{2n}|\mathbf{W}_2^k \mathbf{t}_n + \boldsymbol{\mu}_2^k, (\mathbf{L}_2^k \mathbf{L}_2^{k\top})^{-1}),
 \end{aligned}$$

**Table 1.** Quantitative analysis of the prediction errors of the two alternative variational approximations for t-distributions. The table shows the mean prediction error over 10000 independent test samples, averaged over 10 different realizations of the data set, as well as different quantiles of the distribution of the errors. The two factorizations are equal with respect to all of the measures.

Approximation	Mean	5% quantile	50% quantile		95% quantile
			Predict $\mathbf{x}_2 \mathbf{x}_1$	Predict $\mathbf{x}_1 \mathbf{x}_2$	
$q(u)q(\mathbf{t})$	8.9180	2.1766	6.2896	6.2896	22.4195
$q(u)q(\mathbf{t} u)$	8.9180	2.1766	6.2895	6.2895	22.4177
$q(u)q(\mathbf{t})$	8.5027	2.0908	5.9635	5.9635	21.4362
$q(u)q(\mathbf{t} u)$	8.5028	2.0911	5.9636	5.9636	21.4384

where the mean vector entries are drawn randomly so that the cluster centers are well separated. The entries of the lower triangular matrices  $\mathbf{L}_1$  and  $\mathbf{L}_2$  are drawn from the uniform distribution between 0 and 0.5, with additional small positive entries added to the main diagonal, and the projection matrices  $\mathbf{W}_1$  and  $\mathbf{W}_2$  are generated as

$$\mathbf{W}_1^k = \sum_{j=1}^{d_k} \mathbf{w}_{1j}^k \mathbf{e}_j^\top$$

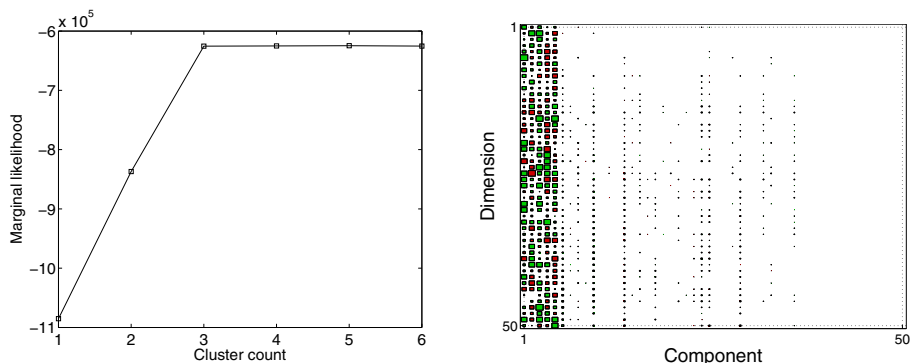
$$\mathbf{w}_{1j}^k \sim \mathcal{N}(\mathbf{w}_{1j}^k | \mathbf{1}, (10 * \mathbf{I})^{-1}),$$

where  $d_k = \{3, 5, 7\}$  encodes the dimensionality of the latent space in each of the clusters. For  $\mathbf{x}_2$  the procedure was the same.

Figure 3 shows two illustrations of the result, clearly demonstrating that the marginal likelihood grows until the correct number of clusters but does not improve further, indicating that coupling the likelihood with a reasonable prior on the number captures the correct complexity. The other sub-figure illustrates the projection matrix, revealing how only five components contain non-zero elements in the cluster that was created to have exactly five correlating components, even though the model was ran with maximal possible complexity (the number of dimensions that is here 50). Hence, the need for choosing the complexity is efficiently sidestepped. Note that the columns of the matrix have not been re-ordered for the illustration, but instead the approximation automatically learns the components roughly in the order of the magnitude.

## 4 MEG Analysis

To concretize the scenarios where searching for mutual dependencies is likely to be useful, we apply the model to analysis of brain response to natural stimulus. For natural stimuli the traditional approaches are not sufficient due to lack of repetition and control in the stimulus, and more data-driven approaches are needed. The primary purpose of the experiment is to illustrate potential uses for the model, and more detailed neuroscientific analysis is omitted.



**Fig. 3. Left:** Marginal likelihood as a function of the number of clusters reveals that the data has three clusters. The likelihood remains constant for larger number of potential clusters because the model discovers only three clusters, leaving the rest empty. **Right:** Hinton plot of the projection matrix  $\mathbf{W}_x^3$  of the three cluster model shows how the model correctly captures the correlating dimensions. The ARD prior automatically pushes the elements to zero for excess dimensions, keeping high values only for the five true components.

In particular, we demonstrate how statistical dependencies between brain activity measurements done with MEG and measurements of the autonomic nervous system (ANS) can be used to create hypotheses on where to look for emotional responses in MEG data. MEG measures the cortical brain activity while emotional stimuli mainly cause response in the deeper regions, and hence it is generally unknown to which degree emotional responses are visible in MEG data (see [15] for a analysis of a simple controlled experiment). Since ANS measurements are highly informative of emotional activity, correlations between the two sources provide a link between MEG and the emotions.

In this paper we present the results from the point of view of further validating the applicability of the model. In detail, we show how relaxing the stationarity assumption of the signal by mixture modeling reveals stronger correlations between the signals, and how the mixture components found by the model are interpretable and directly linked with the emotional stimuli labels not used in learning the model. These results complement the artificial experiments and show the model is directly applicable also for real scenarios, even for large sample sizes.

#### 4.1 Data

We apply the model for joint analysis of brain oscillations and autonomic nervous system (ANS) response to emotionally loaded auditory stimuli [18]. Emotional sounds obtained from the International Affective Digitized Sounds (IADS-2) library with varying arousal and valence values were played while the brain activity of the test subjects was measured with MEG, and pupil diameter measured



with iView X<sup>TM</sup> MEG eye tracker was used as an example signal for the ANS activity. A total of 48 stimuli, each lasting 6 seconds, were played with 10 second shade-in and shade-out periods.

We extract dependencies between a single MEG channel, chosen based on preliminary analysis, and pupil diameter, as a demonstration of what the model can achieve. The approach directly generalizes to more MEG channels and would also be applicable to other measurements of ANS, such as galvanic skin response or heart-rate variability, or combinations of those.

After basic signal pre-processing consisting of resampling, de-trending, filtering and renormalisation, we apply a sliding rectangular window function to both one-dimensional signals. The stimuli-evoked responses are time-localised in the MEG signal, and the resulting window-based feature representation is a natural choice for such short time analysis. In the context of the CCA-type models, this representation encodes the time-amplitude information through the mean vector and frequency-amplitude information in the CCA projections. In other words, projecting the signal windows to the canonical scores corresponds to filtering.

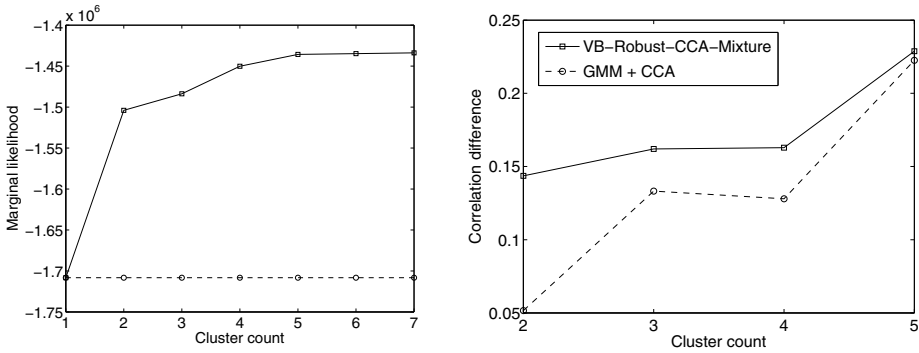
The model hyperparameters are set in exactly the same way as for the artificial data except for the prior precisions which are smaller (with the order of magnitude estimated from the empirical covariance matrices), because the biomedical signals are known to be very noisy.

## 4.2 Results

Figure 4 (left) shows the marginal likelihood as a function of the number of clusters, showing that the data strongly supports more than one cluster and hence that the signal is clearly non-stationary. Solutions between two and five clusters are all sensible, whereas using more than five clusters does not improve the likelihood anymore. In fact, the excess clusters become empty during the learning process, and hence play no role.

One of the main advantages of relaxing the stationarity assumption is that the regions of the data space showing strong dependency can be separated from the rest, to better capture the correlations. This should be manifested as some clusters having high correlations, while some other clusters learn to model the less dependent parts. We use this observation to construct a measure for comparing our model with the alternative solution of first clustering the data in the joint space and applying classical CCA for each of the clusters separately: For each model complexity we measure the difference between the highest correlations in the most and least dependent clusters. The comparison method also uses variational inference for learning the clusters, and the result is then turned into a hard clustering in order to compute the CCA. It hence follows the basic approach of 7, but the clustering model is replaced with a better one to compensate for the gain by our improved inference.

Figure 4 (right) shows how finding the clusters and the dependencies together improves compared to the alternative. The joint clustering is not able to separate the dependent parts from the independent ones, but instead merely divides the data into clusters of roughly equal size having correlations relatively close to



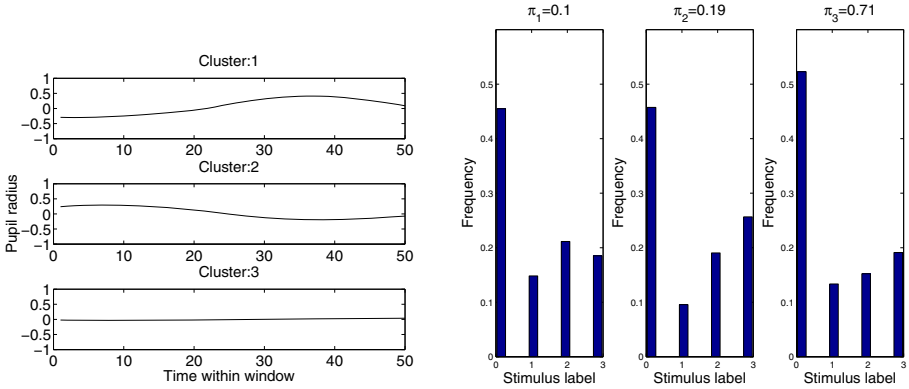
**Fig. 4. Left:** Marginal likelihood as a function of the number of clusters, showing how the data only supports at most five clusters. The baseline (dashed line) corresponds to the single-cluster solution, and the clear difference shows the improvement from relaxing the stationarity assumption. **Right:** Illustration of how the proposed model outperforms the alternative of first clustering the data and then applying classical CCA. The curves measure the ability of the model to separate different kinds of dependencies into different clusters, evaluated as the difference between the largest correlations in the most/least dependent cluster in the model.

each other. Increasing the number of clusters helps, since small enough clusters will start to capture the dependencies even when they were learned to model the joint distribution, but even then the joint clusters cannot be directly interpreted as capturing different kind of dependencies.

Next we take the three-cluster solution of our model for closer analysis in order to demonstrate that the clusters are interpretable. We do not proceed to analyze the projection vectors that would reveal the signal filters needed for full neuroscientific analysis, but instead identify the clusters based on the mean profiles of the pupil data (Figure 5, left) and show how the cluster identities are linked with the emotional stimuli labels (Figure 5, right). The labels were not used in learning the model, and hence this serves as an external validation. The mean vectors reveal that the largest cluster corresponds to no activity, while the other two clusters correspond to pupil dilation and contraction. The histogram of the stimuli labels in each of the clusters shows that the two smaller clusters are enriched with the positive and negative stimuli, respectively, proving that the model has learned not only a link between MEG and ANS, but that the link is indeed related to the underlying natural stimulus.

## 5 Discussion

Efficient and robust models for extracting statistical dependencies between multiple co-occurring data streams or signals are needed for exploratory analysis of complicated natural phenomena such as brain activity or cellular functions. We introduced a novel model that synthesises the latest advances in generative



**Fig. 5. Left:** Mean of the pupil diameter in each of the clusters. **Right:** Distribution of the stimuli labels in each of the clusters. The stimuli enumeration from 0 to 3 refers to no stimulus, emotionally neutral stimulus, positive stimulus and negative stimulus, respectively. Note how samples with positive stimulus labels are enriched in the first cluster and samples with negative stimulus label in the second cluster.

dependency modeling to create a practically applicable tool for large-scale analysis. The robust mixture of canonical correlation analyzers combines the mixture solution of [10] with the variational approximation of [16] and the robust CCA extension of [2] into a single model. The model is directly applicable to data sets of tens of thousands of observations, as demonstrated by the example application on the MEG data, and includes automatic solutions for model complexity selection. An open-source implementation of the model written in MATLAB is available at <http://www.cis.hut.fi/projects/mi/software/vbcc/>.

Furthermore, we studied alternative variational approximations for robust t-distribution models in general. Two different independence assumptions both lead to tractable approximations that have been used by earlier models [5, 8, 12]. We showed that the difference in the modeling accuracy between the two approximations is negligible, concluding that future variational approximations of scale-mixture models can choose either alternative based on the desired functional form for the predictive distributions, not needing to consider the modeling accuracy.

## Acknowledgments

The authors belong to the Adaptive Informatics Research Centre, a CoE of the Academy of Finland. The project was supported by the aivoAALTO project of the Aalto University and by the Academy of Finland decision number 133818, and was in part supported by the PASCAL2 EU NoE. We thank Siina Pamilo, Miika Koskinen, and Riitta Hari (Brain Research Unit, Low Temperature Laboratory, Aalto University School of Science and Technology) for providing the data used in the neuroscientific experiment.

### Appendix: Variational EM-Update Equations

The factorization  $q(u_n|z_n)q(\mathbf{t}_n|z_n)$  results in the variational distributions of the following form. The parameters of the approximation are implicitly defined as the symbols for which the right hand sides are conditioned on:

$$\begin{aligned}
 q(\{\Psi_i\}) &= \prod_{k=1}^M \mathcal{W}(\Psi_i^k | \tilde{\gamma}_i^k, \tilde{\Phi}_i^k) \\
 q(\{\mu_i\}) &= \prod_{k=1}^M \mathcal{N}(\mu_i^k | \mu_{\mu_i^k}, \Sigma_{\mu_i^k}) \\
 q(\{\mathbf{W}_i\}) &= \prod_{k=1}^M \prod_{j=1}^{d_i} \mathcal{N}(\mathbf{W}_{ij}^k | \mu_{W_{ij}^k}, \Sigma_{W_{ij}^k}) \\
 q(\{\alpha_i\}) &= \prod_{k=1}^M \prod_{j=1}^d \mathcal{G}(\alpha_{ij}^k | a_{ij}^k, b_{ij}^k) \\
 q(\mathbf{t}_n | z_{nk} = 1) &= \mathcal{N}(\mathbf{t}_n | \mu_{t_{nk}}, \Sigma_{t_{nk}}^{-1}) \\
 q(u_n | z_{nk} = 1) &= \mathcal{G}(u_n | \alpha_{q_{nk}}, \beta_{q_{nk}}) \\
 q(z_n) &= \text{Multinomial}(z_n | r_n).
 \end{aligned}$$

The update rules needed for learning the parameters of the approximation are then given by the following formulas, where  $\langle A \rangle_{q(\cdot)}$  denotes the expectation of  $A$  with respect to  $q(\cdot)$ . In addition, we denote the dimensionality of  $\mathbf{x}_{in}$  with  $d_i$  and the latent space dimensionality with  $D$ :

$$\begin{aligned}
 \mu_{t_{nk}} &= \Sigma_{t_k} \left( \sum_{i=1}^2 \langle (\mathbf{W}_i^k)^\top \Psi_i^k (\mathbf{x}_{in} - \mu_{ik}) \rangle_{q(\{\mu_i\})q(\{\Psi_i\})q(\{\mathbf{W}_i\})} \right) \\
 \Sigma_{t_{nk}}^{-1} &= \langle u_n \rangle_{q(u_n | z_{nk}=1)} \left( \sum_{i=1}^2 \langle (\mathbf{W}_i^k)^\top \Psi_i^k \mathbf{W}_i^k \rangle_{q(\{\Psi_i\})q(\{\mathbf{W}_i\})} + \mathbf{I}_D \right) \\
 &= \langle u_n \rangle_{q(u_n | z_{nk}=1)} \Sigma_{t_k} \\
 \alpha_{q_{nk}} &= \frac{\nu_k + \sum_{i=1}^2 d_i + D}{2} \\
 \beta_{q_{nk}} &= \nu_k / 2 + \langle \frac{1}{2} \mathbf{t}_n^\top \mathbf{t}_n \rangle_{q(\mathbf{t}_n | z_{nk}=1)} \\
 &\quad + \sum_{i=1}^2 \langle \frac{1}{2} (\mathbf{x}_{in} - \mathbf{W}_i^k \mathbf{t}_n - \mu_i^k)^\top \Psi_i^k (\mathbf{x}_{in} - \mathbf{W}_i^k \mathbf{t}_n - \mu_i^k) \rangle_{q(*)} \\
 &\text{where } q(*) = q(\mathbf{t}_n | z_{nk} = 1)q(\{\mu_i\})q(\{\Psi_i\})q(\{\mathbf{W}_i\})
 \end{aligned}$$

$$\begin{aligned}
\ln \rho_{nk} &= \ln \boldsymbol{\pi}_k - D_{\text{KL}}(q(u_n | z_{nk} = 1) || p(u_n | z_{nk} = 1)) \\
&\quad - \langle D_{\text{KL}}(q(\mathbf{t}_n | z_{nk} = 1) || p(\mathbf{t}_n | z_{nk} = 1, u_n)) \rangle_{q(u_n | z_{nk} = 1)} \\
&\quad + \sum_{i=1}^2 \langle \ln p(\mathbf{x}_{in} | z_{nk} = 1, u_n, \mathbf{t}_n) \rangle_{q(\{\boldsymbol{\mu}_i\})q(\{\boldsymbol{\Psi}_i\})q(\{\mathbf{W}_i\})q(u_n | z_{nk} = 1)q(\mathbf{t}_n | z_{nk} = 1)} \\
r_{nk} &= \frac{\rho_{nk}}{\sum_{j=1}^K \rho_{nk}} \\
\tilde{\gamma}_i &= \gamma_i + \sum_{n=1}^N q(z_{nk}) \\
(\tilde{\boldsymbol{\Phi}}_i^k)^{-1} &= (\boldsymbol{\Phi}_i)^{-1} + \sum_{n=1}^N q(z_{nk}) \langle (\mathbf{x}_{in} - \mathbf{W}_i^k \mathbf{t}_n - \boldsymbol{\mu}_i^k) \dots \\
&\quad \times (\mathbf{x}_{in} - \mathbf{W}_i^k \mathbf{t}_n - \boldsymbol{\mu}_i^k)^\top u_n \rangle_{q(u_n, \mathbf{t}_n | z_{nk} = 1)q(\{\mathbf{W}_i\})q(\{\boldsymbol{\mu}_i\})} \\
\boldsymbol{\Sigma}_{\boldsymbol{\mu}_i^k}^{-1} &= \beta_i \mathbf{I} + \sum_{n=1}^N q(z_{nk}) \langle u_n \boldsymbol{\Psi}_i^k \rangle_{q(u_n | z_{nk} = 1)q(\{\boldsymbol{\Psi}_i\})} \\
\boldsymbol{\mu}_{\boldsymbol{\mu}_i^k} &= \boldsymbol{\Sigma}_{\boldsymbol{\mu}_i^k} \left( \sum_{n=1}^N q(z_{nk}) \langle u_n \boldsymbol{\Psi}_i^k (\mathbf{x}_{in} - \mathbf{W}_i^k \mathbf{t}_n) \rangle_{q(u_n, \mathbf{t}_n | z_{nk} = 1)q(\{\mathbf{W}_i\})q(\{\boldsymbol{\Psi}_i\})} \right) \\
\boldsymbol{\Sigma}_{\mathbf{W}_{ij}^k}^{-1} &= \langle \text{diag}(\alpha_i^k) \rangle_{q(\{\alpha_i\})} + \sum_{n=1}^N q(z_{nk}) \langle u_n \mathbf{t}_n \mathbf{t}_n^\top (\boldsymbol{\Psi}_i^k)_{(j,j)} \rangle_{q(u_n, \mathbf{t}_n | z_{nk} = 1)q(\{\boldsymbol{\Psi}_i\})} \\
\boldsymbol{\mu}_{\mathbf{W}_{ij}^k} &= \boldsymbol{\Sigma}_{\mathbf{W}_{ij}^k} \left( \sum_{n=1}^N q(z_{nk}) \langle \mathbf{t}_n (\boldsymbol{\Psi}_i^k)_{(j,:)} u_n (\mathbf{x}_{in} - \boldsymbol{\mu}_i^k) \rangle_{q(u_n, \mathbf{t}_n | z_{nk} = 1)q(\{\boldsymbol{\mu}_i\})q(\{\boldsymbol{\Psi}_i\})} \right. \\
&\quad \left. - \sum_{n=1}^N \sum_{l \neq j}^{d_i} q(z_{nk}) \langle u_n \mathbf{t}_n \mathbf{t}_n^\top (\boldsymbol{\Psi}_i^k)_{(j,l)} \mathbf{W}_{il}^k \rangle_{q(u_n, \mathbf{t}_n | z_{nk} = 1)q(\{\boldsymbol{\mu}_i\})q(\{\boldsymbol{\Psi}_i\})} \right) \\
a_{ij}^k &= a_i + d_i / 2 \\
b_{ij}^k &= b_i + \langle \|\mathbf{W}_{i(:,j)}^k\|^2 \rangle_{q(\{\mathbf{W}_i\})} / 2
\end{aligned}$$

## References

1. Archambeau, C., Verleysen, M.: Robust Bayesian clustering. *Neural Networks* 20, 129–138 (2007)
2. Archambeau, C., Delannay, N., Verleysen, M.: Robust probabilistic projections. In: Cohen, W., Moore, A. (eds.) *Proceedings of ICML 2006, the 23rd International Conference on Machine Learning*, pp. 33–40. ACM, New York (2006)
3. Bach, F.R., Jordan, M.I.: A probabilistic interpretation of canonical correlation analysis. Tech. Rep. 688, Department of Statistics, University of California, Berkeley (2005)
4. Beal, M.: Variational algorithms for approximate Bayesian inference. Ph.D. thesis, Gatsby Computational Neuroscience Unit, University College London, UK (2003)

5. Chatzis, S., Kosmopoulos, D., Varvarigou, T.: Signal modeling and classification using a robust latent space model based on  $t$  distributions. *IEEE Transactions on Signal Processing* 56(3), 949–963 (2008)
6. Correa, N., Li, Y.O., Adali, T., Calhoun, V.D.: Canonical correlation analysis for feature-based fusion of biomedical imaging modalities to detect associative networks in schizophrenia. Special issue on fMRI analysis for Human brain mapping. *IEEE J. Selected Topics in Signal Processing* 2(6), 998–1007 (2008)
7. Fern, X., Brodley, C.E., Friedl, M.A.: Correlation clustering for learning mixtures of canonical correlation models. In: Kargupta, H., Kamath, C., Srivastava, J., Goodman, A. (eds.) *Proceedings of the Fifth SIAM International Conference on Data Mining*, pp. 439–448 (2005)
8. Gao, J.: Robust L1 principal component analysis and its Bayesian variational inference. *Neural Computation* 20(2), 555–572 (2008)
9. Haroon, D.R., Szedmak, S., Shawe-Taylor, J.: Canonical correlation analysis: An overview with application to learning methods. *Neural Computation* 16(12), 2639–2664 (2004)
10. Klami, A., Kaski, S.: Local dependent components. In: Ghahramani, Z. (ed.) *Proceedings of ICML 2007, the 24th International Conference on Machine Learning*, pp. 425–432. *Omnipress* (2007)
11. Liu, C., Rubin, D.: ML estimation of the  $t$  distribution using EM and its extensions, ECM and ECME. *Statistica Sinica* 5, 19–39 (1995)
12. Luttinen, J., Ilin, A., Karhunen, J.: Bayesian robust PCA for incomplete data. In: *Proceedings of the 8th International Conference on Independent Component Analysis and Signal Separation*, pp. 66–73 (2009)
13. McLachlan, G.J., Peel, D.: *Finite Mixture Models*. John Wiley & Sons, New York (2000)
14. Neal, R.: *Bayesian learning for neural networks*. *Lecture Notes in Statistics*, vol. 118. Springer, New York (1996)
15. Onoda, K., Okamoto, Y., Shishida, K., Hashizume, A., Ueda, K., Yamashita, H., Yamawaki, S.: Anticipation of affective images and event-related desynchronization (ERD) of alpha activity: An MEG study. *Brain Research* 1151, 134–141 (2007)
16. Wang, C.: Variational Bayesian approach to canonical correlation analysis. *IEEE Transactions on Neural Networks* 18, 905–910 (2007)
17. Ylipaavalniemi, J., Savia, E., Malinen, S., Hari, R., Vigário, R., Kaski, S.: Dependencies between stimuli and spatially independent fMRI sources: Towards brain correlates of natural stimuli. *NeuroImage* 48, 176–185 (2009)
18. Yokosawa, K., Pamilo, S., Hirvenkari, L., Ramkumar, P., Pihko, E., Hari, R.: Activation of auditory cortex by anticipating and hearing emotional sounds: a magnetoencephalographic study. In: *16th Annual Meeting of the Organization for Human Brain Mapping* (June 2010)

# Adverse Drug Reaction Mining in Pharmacovigilance Data Using Formal Concept Analysis

Jean Villerd<sup>1</sup>, Yannick Toussaint<sup>1</sup>, and Agnès Lillo-Le Louët<sup>2</sup>

<sup>1</sup> Loria – INRIA Nancy Grand Est, Nancy, France  
firstname.lastname@loria.fr

<sup>2</sup> Pharmacovigilance Regional Center, Hôpital Européen G. Pompidou, Paris, France  
agnes.lillo-lelouet@egp.aphp.fr

**Abstract.** In this paper we discuss the problem of extracting and evaluating associations between drugs and adverse effects in pharmacovigilance data. Approaches proposed by the medical informatics community for mining one drug - one effect pairs perform an exhaustive search strategy that precludes from mining high-order associations. Some specificities of pharmacovigilance data prevent from applying pattern mining approaches proposed by the data mining community for similar problems dealing with epidemiological studies. We argue that Formal Concept Analysis (FCA) and concept lattices constitute a suitable framework for both identifying relevant associations, and assisting experts in their evaluation task. Demographic attributes are handled so that the disproportionality of an association is computed w.r.t. the relevant population stratum to prevent confounding. We put the focus on the understandability of the results and provide evaluation facilities for experts. A real case study on a subset of the French spontaneous reporting system shows that the method identifies known adverse drug reactions and some unknown associations that has to be further investigated.

## 1 Introduction

Pharmacovigilance is the process of monitoring the safety of post-marketed drugs. The pharmacovigilance process starts with collecting *spontaneous case reports*: when suspecting an adverse drug reaction, health care practitioners send a case report to a spontaneous reporting system (SRS), mentioning the observed adverse effects, the drugs taken, and demographic data about the patient. These data are exploited by pharmacovigilance experts to detect *signals* of unexpected adverse drug reactions that require further clinical investigation. The size of these databases preclude their manual exploration: in 2008 more than 20,000 new cases were added to the French pharmacovigilance system while the WHO database contains more than 3 millions of reports.

The medical informatics community proposed some approaches that extract a set of *potential signals* for experts, i.e. a set of pairs  $(d, e)$  showing an unexpected correlation between an observed adverse effect  $e$  and the prescription

of a marketed drug  $d$  [112]. Disproportionality measures have been introduced to quantify this notion of *unexpectedness* [34]. However, the exhaustive search strategy performed by these approaches precludes from mining high-order associations between sets of drugs and adverse effects and from efficiently applying stratification on demographic attributes to prevent confounding.

In the meantime, the data mining community introduced statistical measures from epidemiology into the itemset and rule mining problems [5,6]. Considering exposures as items and a given outcome as a class label, [7,8] proposed efficient approaches that extract *risk patterns* (or *risk itemsets*) correlated with the given outcome. The relevance of a risk pattern is measured by statistical measures such as relative risk. Efficient pruning strategies have been proposed to reduce the search space and to provide concise representations of risk patterns. In particular, [9] considered optimal risk patterns where a risk pattern is said optimal if its relative risk is greater than the relative risk of all its subpatterns. This allows to reduce the number of extracted itemsets by discarding factors that do not increase the strength of shorter risk patterns.

However, some specificities of pharmacovigilance databases compared to epidemiological studies prevent from efficiently applying the above approaches. In contrary to epidemiological studies, pharmacovigilance databases are not designed to monitor one specific exposure to a drug or one specific outcome (adverse effect). Moreover, the database only contains situations "when things went wrong", leading to many potential biases that experts should take into account. In particular, demographic features may act as confounders and lead to extract spurious potential signals. Recent studies have shown that each demographic subpopulation should be separately investigated by performing stratification [10]. This paper deals with the following issues that are currently not addressed by available tools from the medical informatics community :

1. **Dealing with demographic factors.** Stratification is not performed on demographic factors such as age and gender because exhaustively generating measures on all strata has a prohibitive cost. The aim at dealing with demographic factors is twofold. Firstly, it provides insights into the distribution by demographic factors for a given pair  $(d, e)$  and enables a comparative study. Secondly, demographic factors are used to guide further investigation such as clinical trials, especially in patients selection.
2. **Handling complex associations.** A signal of the form  $(d_1, e)$  can be related with more complex associations involving several drugs and several adverse effects. For example, if  $(d_1 d_2, e)$  is recognised as a potential drug interactions, experts should be able to compare the respective strengths of  $(d_1 d_2, e)$ ,  $(d_1, e)$ , and  $(d_2, e)$ .
3. **Providing a complete information.** Since pharmacovigilance data contain many sources of bias, a potential signal  $(d, e)$  should be presented to the experts only if there is no hidden additional factor shared by the corresponding group of patients that took  $d$  and suffered from  $e$ . For instance if this subgroup only contains men,  $(d, e, M)$ , i.e.  $(d, e)$  on the male subpopulation, should be rather considered. Therefore, our aim is not to find the shortest



itemsets with the highest disproportionality, but to provide experts with potential associations  $(D, E, X)$  where the itemset  $DEX$  is the most complete description of the group of patients on which the potential association is observed.

In this paper, we propose a signal detection method based on Formal Concept Analysis that provides answers to these three points. Section 2 presents the issues concerning signal detection and introduces two constraints that define potential associations. Section 3 describes our method based on a concept lattice for identifying potential associations. Section 4 presents how the concept lattice provides features that help experts in evaluating potential interactions. An experiment on real data is analysed. Section 5 concludes the paper with a summary of contributions and future work.

## 2 Problem Setting

Meyboom *et al.* [11] gives a comprehensive definition of signal detection process as being "A set of data constituting a hypothesis that is relevant to the rational and safe use of a medicine. Such data are usually clinical, pharmacological, pathological or epidemiological in nature. A signal consists of a hypothesis together with data and arguments." A *potential signal* is then an hypothesis suggested by an automated signal detection system that has to be evaluated by an expert. More precisely, a signal consists in (i) a pair  $(d, e)$  where  $d$  is suspected to be the cause of  $e$  (hypothesis), (ii) a set of reports (data), and (iii) disproportionality measures (arguments).

Only a few studies extended this definition to *potential associations*, i.e. higher-order hypothesis  $(D, E)$  where  $D$  and  $E$  are sets, have been published on higher-order associations, mainly about drug-drug interactions [12].

The aim of signal detection methods is to identify, among all pairs  $(d, e)$ , those that occur more than expected when assuming the independence between  $d$  and  $e$ . However, although the number of reports for  $(d, e)$  is known in the database, the number of patients exposed to the drug  $d$  in the whole population is not, nor the number of patients suffering from  $e$ . Thus, the expected number of reports can not be reliably computed [13]. A solution consists in estimating the expected number of reports for  $(d, e)$  by considering the number of reports concerning other drugs and other adverse effects in the database. Therefore, contingency tables are central data structures. Table 1 depicts the contingency table for a pair  $(d, e)$ . Each cell contains the number of reports corresponding to a given combination in the database:  $n_{11}$  is the number of reports containing both  $d$  and  $e$ , i.e. the observed number of reports,  $n_{10}$  is the number of reports containing  $d$  but not  $e$ , and so on.  $N$  is the total number of reports. Several measures have been introduced to capture to what extent a pair is reported more than expected. The most widely used is the *Proportional Reporting Ratio (PRR)* [3], defined as

$$PRR(d, e) = \frac{P(e|d)}{P(e|\bar{d})} = \frac{\frac{n_{11}}{n_{11}+n_{10}}}{\frac{n_{01}}{n_{01}+n_{00}}}$$

The pair  $(d, e)$  is considered to be a potential signal if  $PRR \geq 2$  and  $\chi^2 \geq 4$  and  $n_{11} \geq 3$  [31]. This criterion is widely used, notably by the British Medicines and Healthcare products Regulatory Agency (MHRA). Intuitively, the first condition means that there must be twice as much probabilities to suffer from  $e$  while taking  $d$ , rather than while not taking  $d$ . The second one ensures that  $d$  and  $e$  are not independant. The third condition tells that there must be at least three reports containing  $d$  and  $e$  in the database. Other disproportionality measures such as the *Reporting Odds Ratio (ROR)* [4] are also used. More sophisticated methods implement disproportionality measures in a Bayesian framework [14].

**Table 1.** Contingency table for a signal  $(d, e)$

	$e$	$\bar{e}$	
$d$	$n_{11}$	$n_{10}$	$n_{11} + n_{10}$
$\bar{d}$	$n_{01}$	$n_{00}$	$n_{01} + n_{00}$
	$n_{11} + n_{01}$	$n_{10} + n_{00}$	$N$

**Table 2.** Contingency table on a subpopulation

	$eM$	$\bar{e}M$	
$dM$	$n_{11}$	$n_{10}$	$n_{11} + n_{10}$
$\bar{d}M$	$n_{01}$	$n_{00}$	$n_{01} + n_{00}$
	$n_{11} + n_{01}$	$n_{10} + n_{00}$	$supp(M)$

Demographic factors such as gender and age may help in identifying vulnerable subpopulations. Indeed, drugs may be administered differentially according to age (e.g. vaccines), gender, or both of them (e.g. contraceptive pills), and some adverse effects may only concern a specific subpopulation (e.g. sudden infant death syndrome). Therefore, disproportionality should be computed on groups of patients that belong to the same subpopulation. This stratification process leads to compute a  $PRR_{strat}$  value on each subpopulation, called *stratum*, for a given pair  $(d, e)$ . For instance, the  $PRR_{strat}$  of  $(d, e)$  on the male subpopulation is  $PRR_{strat}(d, e, M) = \frac{P(e|dM)}{P(e|\bar{d}M)}$  computed from a contingency table where each cell is restricted to the male subpopulation (see Table 2 where  $supp(M)$  is the number of male patients). Similarly,  $\chi^2_{strat}(d, e, M)$  denotes the  $\chi^2$  value computed from the restricted contingency table. Experts compare  $PRR_{strat}$  values between strata to evaluate if the strength of the association between  $d$  and  $e$  depends on a demographic factor. For instance, if  $(d, e)$  has the same  $PRR_{strat}$  value on both male and female strata, gender is not an increasing factor.

Stratification also allows to detect situations where demographic factors act as confounders [10]. Unbalanced subpopulations may lead to situations where  $PRR_{strat}(d, e, M)$  and  $PRR_{strat}(d, e, F)$  are equals while  $PRR_{strat}(d, e, \emptyset)$  (w.r.t. the whole population) has a different value. In such case,  $PRR_{strat}(d, e, \emptyset)$  is not reliable and is said to be counfounded by gender. Therefore both *crude*  $PRR_{strat}(d, e, \emptyset)$  and  $PRR_{strat}(d, e, x_i)$  on strata  $x_i$  are relevant for experts to evaluate the strength and the reliability of a signal  $(d, e)$ .

The three initial issues mentioned in introduction can be refined in extracting potential associations  $(D, E, X)$  such that:

1. the disproportionality of  $(D, E, X)$  is computed w.r.t. the subpopulation  $X$ , following the stratification strategy;

2. potential associations are presented to the experts in such a way that comparisons between an association  $(d_1, e, M)$  and its related associations (e.g.  $(d_1d_2, e, M)$ ,  $(d_1, e, \emptyset)$ ) is straightforward;
3. considering a potential association  $(D, E, X)$ , the corresponding group of patients do not share any additional attribute than those in  $DEX$ .

### 3 A FCA-Based Signal Detection Method

Let  $\mathcal{D}$  be a set of drugs,  $\mathcal{E}$  be a set of adverse effects and  $\mathcal{X}$  a set of binarized demographic attributes. We look for potential associations  $(D, E, X)$ ,  $(D \subseteq \mathcal{D}, E \subseteq \mathcal{E}, X \subseteq \mathcal{X}, D \neq \emptyset, E \neq \emptyset)$  that satisfy two types of constraints:

- a closure constraint: stating that patients that cover the itemset  $D \cup E \cup X$ , noted  $DEX$ , do not share any additional attribute,
- a strength constraint: stating that  $supp(DEX) \geq 3$ ,  $PRR_{strat}(D, E, X) \geq 2$  and  $\chi_{strat}^2(D, E, X) \geq 4$ .

The closure constraint clearly says that  $DEX$  must be a closed itemset. Thus, our search space for potential associations consists of closed itemsets that contain at least one element of  $\mathcal{D}$  and one element of  $\mathcal{E}$ . In the following we present basics on Formal Concept Analysis and concept lattices. We later show that the concept lattice is a suitable structure for extracting potential associations, in the sense that it covers our search space, and that it provides experts with efficient ways of comparing related associations.

#### 3.1 Basics on Formal Concept Analysis

Considering a binary relation between a set of objects  $\mathcal{O}$  and a set of binary attributes  $\mathcal{A}$ , FCA extracts a set of pairs  $(O, A)$  with  $O \subseteq \mathcal{O}$ ,  $A \subseteq \mathcal{A}$ , called formal concepts, such that each object in  $O$  owns all attributes in  $A$  and vice-versa. Formal concepts are partially ordered w.r.t. the inclusion of  $O$  and  $A$ , to form a lattice structure called concept lattice. In that way, the concept lattice can be seen as a conceptualization of the binary relation.

In the following, we present formal definitions from [15]. A *formal context* is a triple  $\mathbb{K} = (\mathcal{O}, \mathcal{A}, I)$  where  $\mathcal{O}$  is a set of *objects*,  $\mathcal{A}$  a set of *attributes*, and  $I \subseteq \mathcal{O} \times \mathcal{A}$  a binary relation such that  $oIa$  if the object  $o$  owns the attribute  $a$ . Figure 1 shows a formal context  $\mathbb{K}$  with  $\mathcal{O} = \{o_1 \dots o_7\}$  and  $\mathcal{A} = \{d_1 \dots d_3\} \cup \{e_1, e_2\} \cup \{M, F\}$ .

Two *derivation operators*, both denoted by  $(.)'$ , link objects and attributes. Considering a set of objects  $O \subseteq \mathcal{O}$ ,  $O' = \{a \in \mathcal{A} | oIa\}$ , i.e.  $O'$  is the set of attributes shared by all objects in  $O$ . Dually,  $A' = \{o \in \mathcal{O} | oIa\}$  is the set of objects that own all attributes in  $A$ .  $|A'|$  is called the *support* of  $A$ , noted  $\sigma(A)$ . For instance,  $\{d_1, d_2\}' = \{o_3, o_4\}$  and  $\{o_3, o_4\}' = \{d_1, d_2, e_1, M\}$ .

Two compound operators, both denoted by  $(.)''$ , composed of the two previous derivation operators, are *closure operators* on  $2^{\mathcal{O}}$  and  $2^{\mathcal{A}}$ . Therefore  $O''$  is the maximal set of objects that share the same attributes than the objects in  $O$ .

Dually,  $A''$  is the maximal set of attributes that are owned by the objects that share attributes in  $A$ . A set of attribute  $A$  is said to be *closed* if  $A = A''$ . The set of sets  $B$  such that  $B'' = A$  forms the *equivalence class* of  $A$ . All sets in the equivalence class of  $A$  have the same support  $\sigma(A)$ . For instance,  $\{d_1, d_2\}$  is not closed since  $\{d_1, d_2\}'' = \{o_3, o_4\}' = \{d_1, d_2, e_1, M\}$ , while  $\{o_3, o_4\}$  is closed since  $\{o_3, o_4\}'' = \{d_1, d_2, e_1, M\}' = \{o_3, o_4\}$ .

A *formal concept* is a pair  $(O, A)$  such that  $O = O''$  and  $A = O'$ . Each object in  $O$  owns all attributes in  $A$  and vice-versa. Both  $O$  and  $A$  are closed sets, which means that no object (resp. attribute) can be added to  $O$  (resp.  $A$ ) without changing  $A$  (resp.  $O$ ).  $O$  (resp.  $A$ ) is called the *extent* noted  $\text{Ext}(O, A)$  (resp. the *intent* noted  $\text{Int}(O, A)$ ) of the concept. The set of all formal concepts of the formal context  $\mathbb{K}$  is denoted  $\mathfrak{B}(\mathbb{K})$ . For instance,  $(\{o_3, o_4\}, \{d_1, d_2, e_1, M\})$  is a formal concept.

Formal concepts are partially ordered w.r.t. to the inclusion of their extents. Considering two concepts  $(O_1, A_1)$  and  $(O_2, A_2)$ ,  $(O_1, A_1) \leq (O_2, A_2)$  iff  $O_1 \subseteq O_2$  (which is equivalent to  $A_1 \supseteq A_2$ ). The set of all formal concepts ordered in this way is denoted by  $\underline{\mathfrak{B}}(\mathbb{K})$  and is called the *concept lattice* of the formal context  $\mathbb{K}$ . The maximal concept  $(\mathcal{O}, \mathcal{O}')$  is called the *top* concept, and the minimal concept  $(\mathcal{A}', \mathcal{A})$  is called the *bottom* concept.

The concept lattice  $\underline{\mathfrak{B}}(\mathbb{K})$ , built from  $\mathbb{K}$  is shown in Figure 1. Each box represents a formal concept with its intent in the upper part, and its extent in its lower part.

Considering an attribute  $a$ , its *attribute concept*, denoted  $\mu(a)$ , is the unique concept  $(a'', a')$ , i.e. the highest concept that contains  $a$  in its intent on Figure 1. For instance,  $\mu(e_2) = (\{o_1, o_7\}, \{e_2\})$ .

In the worst case, the number of concepts of  $\mathbb{K} = (\mathcal{O}, \mathcal{A}, I)$  is  $2^{\min(|\mathcal{O}|, |\mathcal{A}|)}$ . This occurs when each subset of  $\mathcal{O}$  or  $\mathcal{A}$  is closed, which is improbable in practice.

### 3.2 Our Approach

Our aim is to extract potential associations that satisfy a closure constraint and a strength constraint. We showed that only closed itemsets can satisfy these constraints. Moreover our aim is to provide an understandable representation of results. As said before, interpretation is a difficult task for experts since pharmacovigilance data may contain many biases. Since the content of the database is not the result of a sampling method, spurious potential associations may be extracted. Disproportionality measures can not make the difference between a spurious disproportion due to a selection bias and a real disproportion due to an adverse effect reaction. Only experts can make this difference w.r.t. the content of the database and their domain knowledge. Therefore, in order to evaluate a potential association  $(D, E, X)$ , experts need more information than disproportionality measures. They need to put back the association in its context of extraction, i.e. in the portion of the database where the disproportionality occurs.

The concept lattice is then a suitable structure for signal detection. It is built from the context  $(\mathcal{O}, \mathcal{A}, I)$ , where  $\mathcal{O}$  is the set of reports, and  $\mathcal{A} = \mathcal{D} \cup \mathcal{E} \cup \mathcal{X}$  is the set of attributes. Since concept intents are closed itemsets, the search

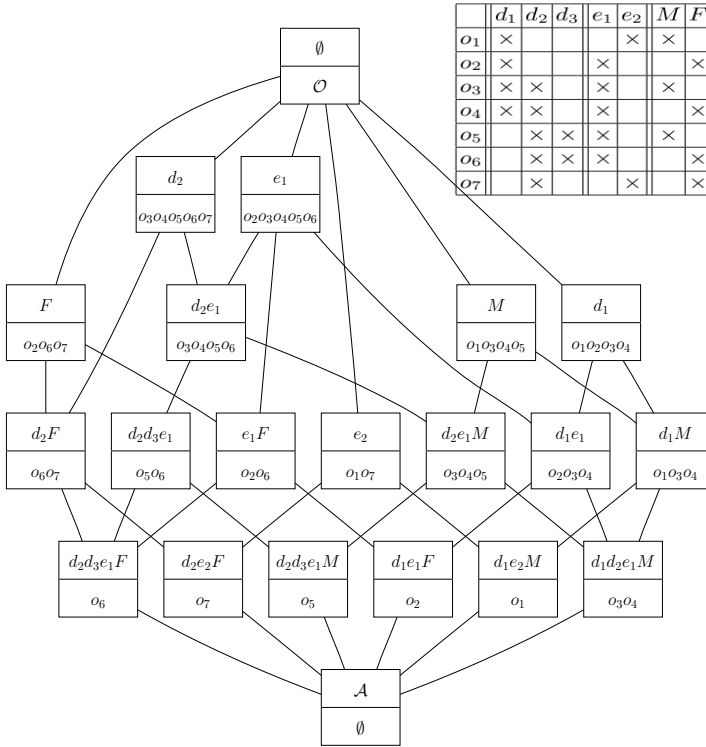


Fig. 1. A formal context and its associated concept lattice

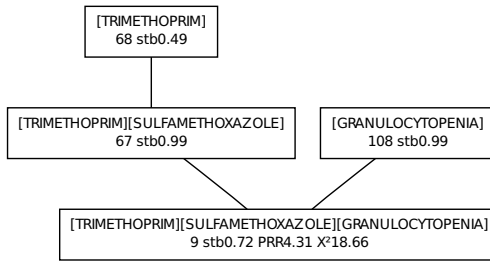


Fig. 2. An interaction example containing noise

space for potential associations is the set of concepts. Moreover, considering an association  $(d_1d_2, e_1, M)$ , the partial order between concepts allows to isolate relevant information for interpretation that will be presented to experts: more specific strata among descendants of the concept with intent  $\{d_1, d_2, e, M\}$ , more general strata among ascendants for instance. But also to compare strengths of related associations: more specific associations (e.g.  $(d_1d_2, e_1e_2, M)$ ) will be found among descendants and more general among ascendants.

Thus, our algorithm for extracting potential associations is straightforward. Concepts whose intent contains at least one drug and one adverse effect, and whose extent contains at least three reports are considered as candidate associations. Their contingency table is computed w.r.t. the demographic attributes in intent. If the MHRA criterion is satisfied, the intent is added to the set of potential associations.

```

Data: a concept lattice  $\mathcal{L}$ 
Result: a set of potential associations  $P$ 
foreach concept  $c \in \mathcal{L}$  do
  | if Int( $c$ ) contains at least one element of  $\mathcal{D}$  and one element of  $\mathcal{E}$  and
  | | Ext( $c$ )  $\geq 3$  then
  | | | compute the contingency table for Int( $c$ )
  | | | compute  $PRR_{strat}$  and  $\chi^2_{strat}$  values from the contingency table
  | | | if  $PRR_{strat} \geq 2$  and  $\chi^2_{strat} \geq 4$  then
  | | | | add Int( $c$ ) to the set of potential associations  $P$ 
  | | | end
  | | end
  | end
end
    
```

Algorithm for signal detection

Therefore, the number of candidate associations is bounded by  $2^{\min\{|\mathcal{O}|, |\mathcal{A}|\}}$ , which is the number of formal concepts in the worst case. In practice, the number of reports is larger than the number of attributes, and all subsets of  $\mathcal{A}$  are not closed.

**Computing contingency tables.** Contingency tables are built from the lattice, in order to compute  $PRR_{strat}$  and  $\chi^2_{strat}$  values.

Since each candidate association  $(D, E, X)$  is a closed itemset, there exists a unique formal concept  $c_{DEX}$  with  $\text{Int}(c_{DEX}) = D \cup E \cup X$ . We show in the following that the contingency table of any association can be computed knowing the support of  $c_{DEX}$  and the extent of the attribute-concepts  $\mu(a), a \in DEX$ .

In the general case of an association  $(D, E, X)$ , The cell values of its contingency table restricted to the subpopulation  $X$  are computed as follows.

$$\begin{aligned}
 n_{11} &= \sigma(DEX) = \left| \bigcap_{a \in DEX} \text{Ext}(\mu(a)) \right| = |\text{Ext}(c_{DEX})| \\
 n_{10} &= \sigma(D\bar{E}X) = \sigma(DX) - \sigma(DEX) = \left| \bigcap_{a \in DX} \text{Ext}(\mu(a)) \right| - n_{11} \\
 n_{01} &= \sigma(\bar{D}EX) = \sigma(EX) - \sigma(DEX) = \left| \bigcap_{a \in EX} \text{Ext}(\mu(a)) \right| - n_{11} \\
 n_{00} &= \sigma(\bar{D}\bar{E}X) = \left| \bigcap_{a \in X} \text{Ext}(\mu(a)) \right| - (n_{11} + n_{10} + n_{01})
 \end{aligned}$$

**Insights for noise detection.** The concept lattice provides an additional measure that helps in evaluating the reliability of a potential association. The *stability index* of a formal concept [16] quantifies the ability of the concept to remain existent after deletion of objects in its extent. In other words, the stability index of a concept  $c$  is low if  $\text{Int}(c)$  becomes non-closed after the removal of a few objects from  $\text{Ext}(c)$ . Then, an unstable concept  $c$  correspond to a *barely closed* itemset  $\text{Int}(c)$ . Therefore, stability can be presented to experts as an additional quality measure for potential association. A potential association  $(D, E, X)$  with a low stability index barely satisfies the closure constraint and should be considered with care by experts.

Moreover, stability can provide insights for detecting noisy reports. We illustrate this aspect on a real example. Trimethoprim ( $d_1$ ) and sulfamethoxazole ( $d_2$ ) come together in the dosage form of marketed drugs, thus a unique concept  $\mu(d_1) = \mu(d_2)$  should exist in the lattice. It is not the case (see Figure 2) since  $\mu(d_2) \leq \mu(d_1)$  and  $\sigma(\mu(d_1)) = 68$  while  $\sigma(\mu(d_2)) = 67$ . This means that, among all patients that took  $d_1$ , only one did not take  $d_2$ , which probably corresponds to a badly filled report. The stability index can capture such a situation. Here,  $\mu(d_1)$  has a low stability since the removal of the noisy report will lead  $d_1$  to become non-closed with  $d'_1 = \{d_1, d_2\}$  and then  $\mu(d_1)$  will become  $\mu(d_1) = \mu(d_2)$ . Thus, a low stability index for a given concept should draw experts' attention to the potentially noisy reports contained in its extent.

### 3.3 Related Work

Several works focused on finding *risk patterns* in epidemiological studies. Considering a set of patients described by a set of nominal attributes, and a target outcome  $e$  that partitions patients into two classes (presence/absence), a risk pattern is a set of attribute-value pairs  $D$  such that the pattern is locally frequent ( $\text{support}(De) \geq \text{min\_sup}$ ) and its *relative risk* is higher than a given threshold. Relative risk  $RR(D, e) = \frac{P(e|D)}{P(e|\bar{D})}$  is a widely used measure in epidemiological studies. Note that  $PRR$  and  $RR$  formula are identical when ignoring demographic factors. [9] proposed algorithms for efficiently mining risk patterns. A risk pattern is said *optimal* if its relative risk is greater than the relative risk of all its subpatterns. This allows to reduce the number of extracted patterns by discarding factors that do not increase the strength of more general risk patterns.

Although  $PRR$  and  $RR$  formula are identical for a given outcome  $e$  and a set of attributes  $D$ , this approach does not fit well our requirement for pharmacovigilance.

First there is no predefined outcome in pharmacovigilance data. Each combination of adverse effects may be considered as an outcome. Applying the precited approach would consist in generating the set of optimal risk patterns for each combination of adverse effects. This also prevents from applying other approaches such as subgroup discovery [17] and contrast set mining [18].

Secondly, in [9] demographic attributes play the same role as drugs in contingency tables. This means that the  $PRR$  of the pattern  $\{d, M\}$  is computed as

$PRR(d, e, M) = \frac{P(e|d, M)}{P(e|\bar{d}, M)}$ . In order to be consistent with the stratification recommendation about demographic factors, the  $PRR$  of  $\{d, M\}$  should be computed w.r.t. the male stratum. It should compare men that took  $d$  and suffered from  $e$  with men that did not take  $d$  and suffered from  $e$ :  $PRR_{strat}(d, e, M) = \frac{P(e|d, M)}{P(e|\bar{d}, M)}$ .

Thirdly, as defined in [9], risk patterns may not be closed itemsets and therefore may not satisfy our closure constraint.

Suppose that  $d_1 d_2 e$  is a closed itemset and that the closure of  $d_1$  is  $d_1'' = d_1 d_2$ , then  $PRR(d_1, e, \emptyset) = PRR(d_1 d_2, e, \emptyset)$  as well as  $PRR_{strat}(d_1, e, \emptyset) = PRR_{strat}(d_1 d_2, e, \emptyset)$ . The risk pattern  $d_1 d_2$  is not optimal since its  $PRR$  value is not higher than its subpattern  $d_1$  and is not retrieved, while following our constraints  $d_1 d_2$  has to be retrieved and not  $d_1$ . Moreover, the fact that risk patterns are extracted w.r.t. a given outcome would lead to generate risk patterns for  $e_1$  and then risk patterns for  $e_2$  without paying attention to situations where  $e_1'' = e_2$ . In this case, risk patterns w.r.t.  $e_1$  do not satisfy our closure constraint since all patients that suffer from  $e_1$  also suffer from  $e_2$ .

Moreover, considering non-closed itemsets prevents from computing  $PRR_{strat}$  in an accurate way. Consider the group of patients that took a drug  $d$ . Suppose that all patients that took  $d$  are men, i.e. the closure of  $\{d\}$  is  $\{d, M\}$ . Then  $PRR_{strat}(d, e, \emptyset) = \frac{P(e|d, \emptyset)}{P(e|\bar{d}, \emptyset)} = \frac{P(e|d, M)}{P(e|\bar{d}, \emptyset)}$ . The numerator group of patients actually belongs to a more specific stratum (men) than the denominator group (men and women).  $PRR_{strat}(d, e, \emptyset)$  can not be reliably computed w.r.t. the available data since only men took  $d$ . In this case, associations involving  $d$  are only reliable w.r.t. the male subpopulation. No reliable hypothesis can be made about  $d$  and  $e$  on the whole population since there is no female subpopulation that would allow to evaluate if  $(d, e)$  depends on gender or not.

Since signal detection aims at providing experts with hypothesis for further investigation, we claim that the reliability of an hypothesis is at least as important as its statistical strength. An hypothesis  $(D, E, X)$  is reliable if the corresponding set of patients do not share an additional attribute that may delude experts, i.e. if  $DEX$  is a closed itemset. This is true for demographic attribute as shown before but also for drugs and adverse effects.

## 4 Evaluation Facilities and Experimentation

This section shows how experts get a *contextualized* association using our approach. In addition to disproportionality measures, insights are given to help them in deciding whether a signal or an interaction should be further investigated or not.

### 4.1 Visualization and Navigation

The core idea is to use the concept lattice as a synthetic representation of the database. From the list of potential association, experts access to a detailed



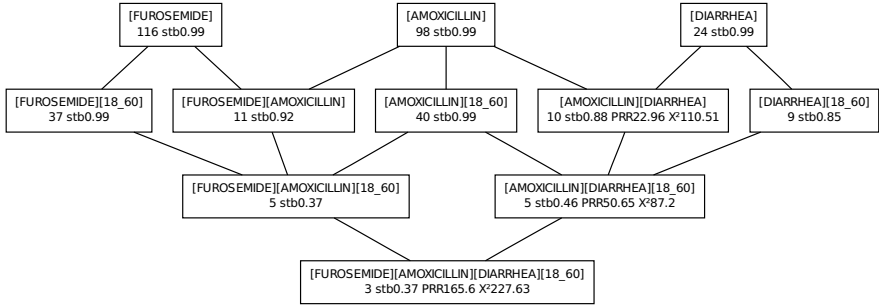


Fig. 3. Subpart of the lattice illustrating a potential interaction

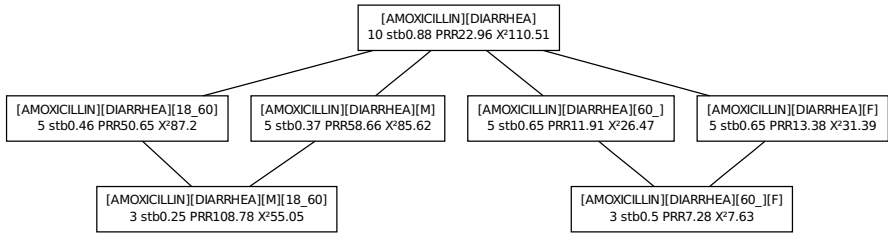


Fig. 4. Comparison of the different strata of a potential signal

view that shows a subpart of the concept lattice, revealing additional information compared to statistical measures and helping experts in their interpretation and evaluation task.

Figure 3 shows the user interface illustrating a potential interaction  $(d_1 d_2, e, X)$  where  $d_1$  is amoxicillin,  $d_2$  is furosemide,  $e$  is diarrhea and  $X$  is 18\_60, meaning age between 18 and 60.

A subpart of the lattice is shown, which contains the concept  $c_{d_1 d_2 e X}$ , corresponding to the interaction, at the bottom, the attribute-concepts  $\mu(d_1)$ ,  $\mu(d_2)$ ,  $\mu(e)$  at the top, and all concepts on the paths from  $c_{d_1 d_2 e X}$  to the attribute-concepts. Then the graph shows concepts that are more general than  $c_{d_1 d_2 e X}$ .

Concepts are labeled with their intent, support and stability. Concepts that own at least one drug and one adverse effect are also labelled with  $PRR_{strat}$  and  $\chi^2_{strat}$  values. Through this graph, experts can compare the  $PRR_{strat}$  values of the interaction  $(d_1 d_2, e, X)$  with those of the signals  $(d_1, e, X)$  and  $(d_1, e, \emptyset)$ , and observe that there are no concepts representing the signals  $(d_2, e, X)$  and  $(d_2, e, \emptyset)$ . This gives the information that no patient took furosemide and suffered from diarrhea without amoxicillin. Concepts that do not correspond to associations are also relevant. For instance, experts can observe that among the 24 patients that suffered from diarrhea, 10 took amoxicillin and state whether this ratio is realistic or is due to a selection bias.

Another graph (cf. Figure 4) shows a given association as root and those of its subconcepts that correspond to its demographic strata with a least 3 reports. Experts can compare their respective  $PRR_{strat}$  values and observe that, in this example, age distribution is different in male and female strata.

## 4.2 Experimentation

We applied our method on a subset of the French national SRS database. This subset contains 3249 cases, 976 drugs, 573 adverse effects. Two demographic attributes, gender and age are binarized into 6 binary attributes (2 for gender and 4 for age). The resulting lattice contains 13178 concepts, among which 6788 with support  $\geq 3$ . Since only signals (one drug, one adverse effects), and interactions (two drugs, one adverse effect) are currently considered by pharmacovigilance experts, we only showed potential signals and interactions to experts. The 2812 candidate signals led to 786 potential signals and the 836 candidate interactions to 183 potential interactions.

**Review of potential signals.** Potential signals were reviewed by an expert who classified them into 5 categories (see Table 3). Categories (1),(2) contain true positives, (3),(4) false positives and (5) unknown potential signals. 27 signals were classified as unknown, i.e. not reported in the literature, but interesting enough for further investigation by experts.

True positives are consistent with results of previous studies [19] and no known true-positive is missing. In the majority of cases, the demographic attributes associated to the couple drug/effect constitute a known risk factor or probable risk factor. For example, cases of **Pulmonary Hypertension** associated with the use of appetite suppressants amphetamine-like were observed in women, between the ages of 18 and 60.

False positives (contained in categories (3) and (4)) are common in signal detection and some of them are well-known. The signal (**hydrochlorothiazide, cough**) is detected because these drug and adverse effects often appear together. However in these cases, cough is actually caused by ACE inhibitors taken concomitantly with **hydrochlorothiazide**. Since there are several ACE inhibitors  $d_i$ , each association  $(d_i, \text{cough})$  appears with a lower support than the association (**hydrochlorothiazide, cough**), which may delude experts. A solution would be to introduce drug therapeutic families, such as ACE, as attributes, with  $(o, \text{ACE}) \in I$  for each case  $o$  containing an ACE inhibitor. Then signals of the form (**ACE, cough**) would be detected, where ACE is a drug family, even if each signal  $(d, e)$  where  $d$  is an ACE inhibitor is too rare to be detected. Current improvements of our method aim at solving this problem.

**Review of potential interactions.** The evaluation of interactions is more difficult since it involves complex pharmacokinetics aspects. Moreover there is no consensus on whether  $(d_1 d_2, e, X)$  should be considered as an interaction when both  $d_1$  and  $d_2$  are known to be the cause of  $e$ . Thus, we are not able to separate

**Table 3.** Potential signals

category	count	
1. known (in reference documents)	720 (91.6%)	true positives
2. known (in a similar form)	24 (3.1%)	
3. the effect is the origin of the medication	3 (0.4%)	false positives
4. due to concomitant drug	11 (1.4%)	
5. unknown potential signal	28 (3.5%)	further investigations needed

**Table 4.** Potential interactions

category	count
either $d_1$ or $d_2$ is a known cause of $e$	64(35.0%)
both $d_1$ or $d_2$ are known causes of $e$	66(36.0%)
$d_1$ and $d_2$ in the same dosage form	34(18.6%)
neither $d_1$ or $d_2$ are known causes	19(10.4%)

true and false positives. Experts classified the 183 potential interactions into 4 categories (see Table 4). The last category corresponds to cases where further investigations are needed.

We noted that, in some cases, the  $PRR_{strat}$  value of an interaction  $(d_1d_2, e, X)$  where only  $d_1$  is a known cause of  $e$  was greater than  $PRR_{strat}(d_1, e, X)$ . In such cases, it is not clear if the focus should be put on  $(d_1d_2, e, X)$  or on  $(d_1, e, X)$ . To our knowledge, there has been no pharmacovigilance study on defining preferences between an interaction  $(d_1d_2, e, X)$  and a signal  $(d_1, e, X)$  w.r.t.  $PRR$  value. Therefore, we can not discard  $(d_1d_2, e, X)$  when  $PRR_{strat}(d_1d_2, e, X) < PRR_{strat}(d_1, e, X)$ . This prevents from using the pruning strategy of the optimal risk patterns approach [9], that would discard  $(d_1d_2, e, X)$ .

**Detection of noisy reports.** In a previous section, we showed that the stability index of a concept may be a clue for noisy reports detection. However, we faced the difficulty of defining a threshold on stability that defines *unstable* concepts. Frequent unstable concepts are interesting. They can be seen as concepts that gather a high number reports, but that actually exist because of only a few of them, which may be noisy reports. Frequent unstable concepts should be found in the upper left hand corner of the Figure 5. We empirically decided to investigate the 20 concepts with a minimum support of 20 reports and a stability index below 0.5. All of these concepts were in the same configuration than in Figure 2, i.e. among the  $n$  reports gathered by the unstable concept,  $n - 1$  also share another attribute. For instance, among the 20 reports gathered by the unstable concept with intent  $\{\text{tacrine, M}\}$ , 19 also own the attribute  $age > 60$ . Since tacrine is used in the treatment of Alzheimer’s disease, the report that does not own  $age > 60$  is suspect and should be verified. The expert considered that the  $n^{\text{th}}$  report was actually suspect in 19 of the 20 unstable concepts under review.

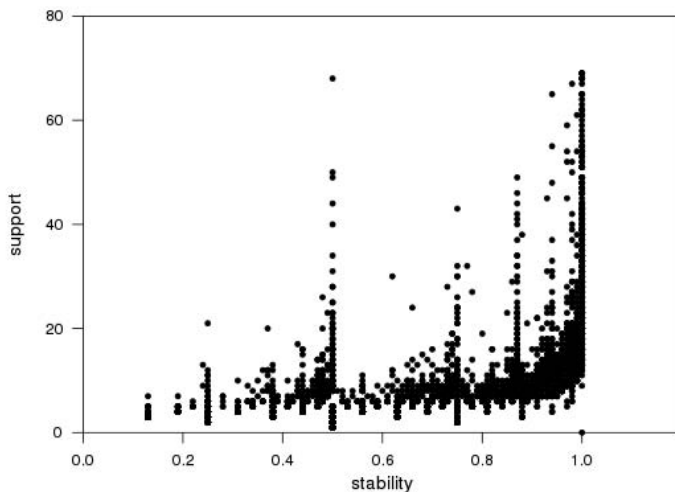


Fig. 5. Stability and support

## 5 Conclusion

In this paper, we presented an automated signal detection method, based on concept lattices, that provides a framework for extracting potential associations and performing qualitative analysis of the extracted associations.

We claim that only associations that are closed itemsets should be presented to experts, since non-closed associations do not fully describe the set of factors shared by a subgroup of patients. Demographic attributes are taken into account in the *PRR* computation so that the disproportionality of an association is computed w.r.t. the subpopulation in which the association is observed. The closure constraint allows to identify the accurate subpopulations and prevents from exhaustively evaluating each population stratum.

Our method is thought for extracting complex associations, i.e. extracting associations where there are one or more drugs, one or more adverse effects and several demographic factors. Nowadays, if signals have been quite well studied, little work has been done on interactions, and practically none on syndromes (1 drug, several effects) or protocols (several drugs, several effects) which justifies the facts that our evaluation has only been performed on signal and interactions.

When evaluating extracted associations, experts have access to subparts of the lattice for visualizing related associations, for example, an interaction is displayed with its related signals as well as its different "strengths" on subpopulations. This visualization is of particular interest when both a signal and an interaction pass the MHRA criterion. Only experts – no automated process – are able to decide which of signals and interactions should be validated, mostly because of pharmacokinetics complexity. The interface is designed to facilitate a qualitative analysis by experts and guides exploration, interpretation and validation of associations.

Our approach is closely related to *domain driven data mining* [20]: starting from a domain-specific problem, our goal is to discover actionable knowledge to satisfy user needs. Here actionable knowledge consists of unexpected associations that need further investigations. These *actionable* associations are identified by experts among *interesting* associations that satisfy strength and closure constraints. The interface supports experts in finding actionable associations among interesting ones. This approach could be used for other applications where a synthetic graphical view is needed by experts to evaluate the actionability of extracted patterns. Finally, we are currently investigating solutions that include domain knowledge such as families of drugs and adverse effects.

## References

1. Hauben, M., Madigan, D., Gerrits, C.M., Walsh, L., Puijtenbroek, E.P.V.: The role of data mining in pharmacovigilance. *Expert Opinion on Drug Safety* 4(5), 929–948 (2005)
2. Bate, A., Lindquist, M., Edwards, I.R.: The application of knowledge discovery in databases to post-marketing drug safety: example of the WHO database. *Fundamental & Clinical Pharmacology* 22(2), 127–140 (2008)
3. Evans, S.J.W., Waller, P.C., Davis, S.: Use of proportional reporting ratios for signal generation from spontaneous adverse drug reaction reports. *Pharmacoepidemiology and Drug Safety* 10(6), 483–486 (2001)
4. van der Heijden, P.G.M., van Puijtenbroek, E.P., van Buuren, S., van der Hofstede, J.W.: On the assessment of adverse drug reactions from spontaneous reporting systems: the influence of under-reporting on odds ratios. *Statistics in Medicine* 21(14), 2027–2044 (2002)
5. Morishita, S., Sese, J.: Transversing itemset lattices with statistical metric pruning. In: *Proc. of the 19th ACM Symposium on Principles of Database Systems*, pp. 226–236. ACM, New York (2000)
6. Gu, L., Li, J., He, H., Williams, G., Hawkins, S., Kelman, C.: Association rule discovery with unbalanced class distributions. In: Gedeon, T(T.) D., Fung, L.C.C. (eds.) *AI 2003. LNCS (LNAI)*, vol. 2903, pp. 221–232. Springer, Heidelberg (2003)
7. Li, H., Li, J., Wong, L., Feng, M., Tan, Y.: Relative risk and odds ratio: A data mining perspective. In: *Proc. of the 24th ACM Symposium on Principles of Database Systems*, p. 377. ACM, New York (2005)
8. Li, J., Fu, A., He, H., Chen, J., Jin, H., McAullay, D., Williams, G., Sparks, R., Kelman, C.: Mining risk patterns in medical data. In: *11th ACM International Conference on Knowledge Discovery in Data Mining*, pp. 770–775. ACM Press, New York (2005)
9. Li, J., Fu, A., Fahey, P.: Efficient discovery of risk patterns in medical data. *Artificial Intelligence in Medicine* 45(1), 77–89 (2009)
10. Woo, E., Ball, R., Burwen, D., Braun, M.: Effects of stratification on data mining in the US Vaccine Adverse Event Reporting System. *Drug safety* 31(8), 667–674 (2008)
11. Meyboom, R.H., Egberts, A.C., Edwards, I.R., Hekster, Y.A., De Koning, F.H.P., Gribnau, F.W.J.: Principles of signal detection in pharmacovigilance. *Drug Safety* 16(6), 335–365 (1997)

12. Almenoff, J., DuMouchel, W., Kindman, L., Yang, X., Fram, D.: Disproportionality analysis using empirical Bayes data mining: a tool for the evaluation of drug interactions in the post-marketing setting. *Pharmacoepidemiology and Drug Safety* 12(6), 517–521 (2003)
13. Roux, E., Thiessard, F., Fourrier, A., Bégaud, B., Tubert-Bitter, P.: Evaluation of statistical association measures for the automatic signal detection generation in pharmacovigilance. *IEEE Transactions on Information Technology in Biomedicine* 9(4), 518–527 (2005)
14. DuMouchel, W.: Bayesian data mining in large frequency tables, with an application to the FDA spontaneous reporting system. *The American Statistician* 53(3), 177–190 (1999)
15. Ganter, B., Wille, R.: *Formal concept analysis: Mathematical Foundations*. Springer, Berlin (1999)
16. Kuznetsov, S.: On stability of a formal concept. *Annals of Mathematics and Artificial Intelligence* 49(1-4), 101–115 (2007)
17. Boley, M., Grosskreutz, H.: Non-redundant Subgroup Discovery Using a Closure System. In: *Proc. of ECML/PKDD*, p. 194. Springer, Heidelberg (2009)
18. Bay, S., Pazzani, M.: Detecting group differences: Mining contrast sets. *Data Mining and Knowledge Discovery* 5(3), 213–246 (2001)
19. Bousquet, C., Sadakhom, C., Le Beller, C., Jaulen, M., Lillo-Le Louët, A.: Revue des signaux générés par une méthode automatisée sur 3324 cas de pharmacovigilance. *Thérapie* 61(1), 39–47 (2006)
20. Cao, L., Zhang, C., Yu, P.S., Zhao, Y.: *Domain Driven Data Mining*. Springer, New York (2010)

# Topic Models Conditioned on Relations

Mirwaes Wahabzada, Zhao Xu, and Kristian Kersting

Knowledge Discovery Department

Fraunhofer IAIS, Schloss Birlinghoven, 53754 Sankt Augustin, Germany

`firstname.lastname@iais.fraunhofer.de`

**Abstract.** Latent Dirichlet allocation is a fully generative statistical language model that has been proven to be successful in capturing both the content and the topics of a corpus of documents. Recently, it was even shown that relations among documents such as hyper-links or citations allow one to share information between documents and in turn to improve topic generation. Although fully generative, in many situations we are actually not interested in predicting relations among documents. In this paper, we therefore present a Dirichlet-multinomial nonparametric regression topic model that includes a Gaussian process prior on joint document and topic distributions that is a function of document relations. On networks of scientific abstracts and of Wikipedia documents we show that this approach meets or exceeds the performance of several baseline topic models.

## 1 Introduction

One of the most fundamental problems in information retrieval is the extraction of meaningful, low-dimensional representations of data. In computer vision, where it is natural to represent images as vectors in a high-dimensional space, they represent e.g. visual words and have been used for face and object recognition or color classification. Social networks such as Flickr, Facebook and Myspace, allow for a diverse range of interactions amongst their members, resulting in temporal datasets relating users, media objects and actions. Here, low-dimensional representations may be used to identify and summarize social activities. If the data are words of documents, low-dimensional representations yield topic models representing each document as a mixture of a small number of topics and each word is attributable to one of the topics.

Topic models, originally explored by Deerwester *et al.* [9], Hofmann [13], Blei *et al.* [5], Griffiths and Steyvers [11], Buntine and Jakulin [6], and many others, have received a lot of attention due to their simplicity, usefulness in reducing the dimensionality of the data, and ability to produce interpretable and semantically coherent topics. They are typically fully generative, probabilistic models that uncover the underlying semantic structure of a document collection based on an hierarchical Bayesian analysis, see e.g. [3], and have been proven successful in a number of applications such as analyzing emails [19], classifying natural scenes [16], detecting topic evolutions of a document corpus [431], analyzing the human semantic memory [30], and many more.

The starting point for our analysis here is an often perceived limitation of latent Dirichlet allocation (LDA), which is one of the most popular and commonly used topic models today [5]: *it fails to make use of relations among documents*. Nowadays, however, networks of documents such as citation networks of scientific papers, hyperlinked networks of web pages, and social networks of friends are becoming pervasive in machine learning applications. Consider a collection of hyperlinked webpages. LDA uses the word distribution within the body of each page only to determine a per-document topic distribution. More precisely, LDA models each document as a mixture over topics, where each vector of per-document mixture proportions is assumed to have been drawn from a Dirichlet distribution with the hyperparameters  $\alpha$  shared among all documents. Webpages, however, do not exist in isolation: there are links connecting them. Two pages having a common set of links are evidence for similarity between such pages. For instance, if W1 and W2 both link to W3, this is commonly considered to be evidence for W1 and W2 having similar topic distributions. In other words, relational knowledge can further reveal additional correlations between variables of interest such as topics. Therefore, it is not surprising that several fully generative relational topic models have been proposed recently, see e.g. [2,12,20,23,7], that take correlations among inter-related documents into account. They have been proven to be successful for modeling networks of documents and even for predicting relations among documents. However, adding additional complexity such as relations to a fully generative model generally results in a larger number of variables to sample and in turn in a more complicated sampling distribution. Thus, the flexibility of relational topic models comes at the cost of increasingly intractable inference. If we are actually not interested in predicting relations, this is an unnecessary complication.

Our main contribution is a novel relational topic model, called xLDA. At the expense of not being able to predict relations among documents anymore, we condition topic models on the metadata such as the relations among the documents (citations, hyperlinks, and so on) as well as attributes describing each document  $d$  (authors, year, venue, and so on) provided in the data. Specifically, xLDA is a Dirichlet-multinomial (nonparametric) regression topic model that includes a Gaussian process prior on joint document and topic distributions that is a function of document attributes and relations. That is, given metadata such as relations we generate a per-document  $\alpha_d$ , the (hyper-)parameters of a Dirichlet distribution. Then, we model each document using LDA with the generated  $\alpha_d$ . Intuitively, documents from the same authors, or published in the same conference, or being related by citations are stronger correlated than other documents. The more correlated two documents are, the more likely they have similar topics. Because all the relational information is accounted for in the document-specific Dirichlet hyperparameters  $\alpha_d$ , the sampling phase of xLDA is no more complicated than a simple LDA sampler. In other words, we sacrifice flexibility for a relatively simple inference. Moreover, we can extend the basic xLDA model through topic meta-information that allows us to express or even to learn conditional independencies that cannot be explained well by the document



meta-information only. On networks of scientific abstracts and of Wikipedia documents we show that xLDA meets or exceeds the performance of several baseline topic models.

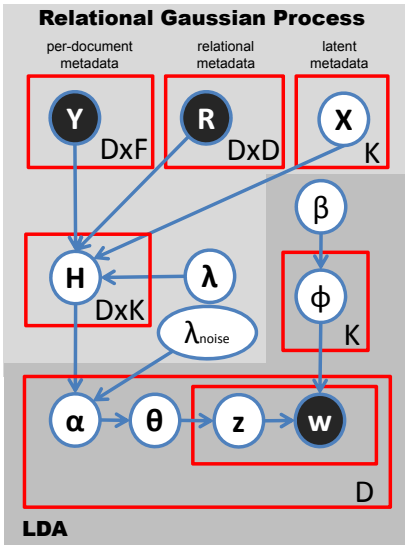
We proceed as follows. After touching upon further related work, we will introduce the xLDA model in Section 3. In Section 4, we then discuss its approximate inference and learning methods. Before concluding, we present our experimental evaluation.

## 2 Related Work

Network data is currently receiving a lot of attention. Several latent variable models that decompose the network according to hidden patterns of connections between its nodes have been proposed, see e.g. [33,15,11]. Indeed quite powerful, these models mainly account for the structure of the network, essentially ignoring the observed attributes of the nodes. Relational matrix factorization approaches such as [28,17,14] are not tailored towards discovering topics.

Recently, several relational topic models have been proposed that also take the observed attributes of nodes, i.e., documents into account [2,12,20,23,7]. They are all fully generative models and due to the additional relations modeled have a more complicated sampling distribution. If we are not interested in predicting relations, this is an unnecessary complication and conditioning on the relations is an attractive alternative.

The idea of conditioning topic models on metadata is not new. Several models have been proposed in which a hidden variable selects one of several topic models conditioned on some metadata. For instance, Rosen-Zvi *et al.*'s author-topic models [26] generates words by first selecting an author uniformly from an observed author list and then selecting a topic from a distribution over topics that is specific to that author. Mimno and McCallum [21] extend this author-topic model to the author-persona topic model that associates multiple topical mixtures with each individual author. McCallum *et al.* [18] employ the "conditioning" idea to model authors and recipients of email, and Dietz *et al.* [10] use it for inferring the influence of individual references on citing papers. Recently, Mimno and McCallum [22] introduced the Dirichlet-multinomial regression topic model. It includes a log-linear prior on document-topic distributions that is a function of observed features of the document, such as author, publication venue, references, and dates. An investigation of this model was the seed that grew into the current paper. It is important, however, to distinguish xLDA from Mimno and McCallum's model. Whereas Mimno and McCallum proposed to model relational information such as citations as per-document attributes of a log-normal prior with diagonal covariance, xLDA employs Silva *et al.*'s [27] *directed mixed graph* Gaussian process framework to incorporate relational information into the topic model. A directed mixed graph model propagates training data information through other training points. Reconsider our webpage domain where each page may have links to several other pages. A chain of intermediated pages between two pages  $W_1$  and  $W_2$  is likely to be more informative if we know the



Symbol	Description
$y_d \in Y$	observed attribute vector of a document $d$
$r_d \in R$	observed relation vector of a document $d$
$x_k \in X$	latent metainformation vector
$\eta_{dk} \in H$	noise-free topic concentration for document $d$
$\lambda, \lambda_{noise}$	hyperparameters of covariance
$\beta$	prior belief on the distribution over the vocabulary
$\alpha_d$	prior belief on topic proportions, $\alpha_d = \exp(\eta_d + \epsilon_{noise}) = \exp(\tau_d)$
$\phi_k \in \Phi$	preference of a topic $k$ over the vocabulary with $\sum_n \phi_{k,n} = 1$
$\theta_d$	topic proportions of a document
$W_{d,n} \in W$	$n$ 'th word in the document $d$
$Z_{d,n} \in Z$	topic assignment of a word $W_{d,n}$
$D$	number of documents
$K$	number of topics

**Fig. 1.** The xLDA topic model and the notation used in the paper. Unlike all previous models, the hyperparameters  $\alpha$  of the Dirichlet distribution over topics are a function of observed document features  $Y$ , relations  $R$ , and hidden topic features  $X$ , and is therefore specific to each distinct combination of document feature values and relations among the documents.

$\alpha$  values of the pages in this chain. In contrast, a relational probabilistic model such as a Markov logic network would — without additional modeling effort — ignore all training pages in this chain besides the endpoints due to the Markov assumption, see [27] for more details. In addition, most state-of-the-art probabilistic relational models focus on discrete quantities and not continuous ones such as  $\alpha$ .

### 3 Modeling the Influence of Document Relations with Dirichlet-multinomial Regression

Nowadays, networks of  $D$  many documents, such as citation networks of scientific papers, hyperlinked networks of web pages, and social networks of friends, are becoming pervasive in machine learning applications. For each document  $d$ , we observe  $N_d$  words, each of which is an element in a  $V$ -term Vocabulary. To capture the per-document meta-information, let  $y_d$  be a vector containing  $F$  many features that encode metadata values for the document  $d$ . For example, if the observed features are indicators for the type of venue the paper was published in, then  $y_d$  would include a 1 in the positions for venue the document  $d$  has published in, and a 0 otherwise. The network among the documents (citations,

hyperlinks, friends relationships, and so on), form a graph and are captured by the adjacency matrix  $R$ . For instance, if documents  $d_i$  cites  $d_j$ , there is a 1 in position  $R_{ij}$ .

The topic model we propose, called xLDA, is a model of data composed of documents, which are collections of words, and relations among them. It is graphically depicted in Fig. 1 and consists essentially of two phases: (1) a relational Gaussian process (GP) phase, and (2) a document-specific LDA phase. In the relational GP phase, given the per-document metadata  $Y$ , the relations  $R$  among the documents and some optional meta-information for topics  $X$ , we generate the per-document  $\alpha_d$  Dirichlet hyperparameters for each document. To do so, we use Silva *et al.*'s [27] *directed mixed graph* Gaussian process framework as it propagates training  $\alpha_d$  through other training documents'  $\alpha_d$  (as discussed in the related work section). Then, in the LDA phase, we run standard LDA using the generated  $\alpha_d$  for each document  $d$ . We assume that each latent  $\alpha_{d,k}$  is a function value of document metadata  $y_d$ , document relations  $r_d$  and topic metadata  $x_k$ . The (optional) topic-metainformation  $x_k$  allows one to accommodate for correlations not well explained by document metainformation only. All the function values are drawn from a GP prior with mean zero and covariance function  $c((y_d, r_d, x_k), (y_{d'}, r_{d'}, x_{k'}))$ . Then, the topic proportion  $\theta_d$  of document  $d$  is a sample of  $\text{Dir}(\alpha_d)$ . That is, we use a distinct Dirichlet distribution  $\text{Dir}(\alpha_d)$  with hyperparameters  $\alpha_d$  as predicted by the Gaussian process. However, we have to be a little bit more careful: we have to ensure that the  $\alpha_d$ s are positive. We do so by predicting a noisy  $\tau_d \in \tau$ , the logarithms  $\tau_d = \eta_d + \epsilon_{noise} = \log(\alpha_d)$  of the concentration parameters  $\alpha_d$ .

The xLDA topic model integrates heterogeneous information, namely the per-document metadata and the relations among documents, into a single probabilistic framework. The dependencies from different sources are captured in a natural and elegant way. Moreover, it can directly be used in combination with various existing relational GPs to realize multiple relations, relation prediction using fully generative relational models [8,32] — of course to the expense of a more complicated GP inference step — or even transfer learning among topic models [34].

Let us now discuss the prior distribution and how to generate words and documents in more details.

**Prior Distribution:** For each document, we introduce a  $K$ -dimensional vector  $\alpha_d$  where each value  $\alpha_{d,k}$  denotes the preference of a document  $d$  on a topic  $k$ . It is a function of the document's metadata  $y_d$ , its relations  $r_d$ , and the (optional) latent topic meta-information  $x_k$ . Additionally, to meet the constraint on Dirichlet parameters, i.e.  $\alpha_{d,k} > 0$ , we assume  $\alpha_{d,k} = \exp(\tau_{d,k})$ , where  $\tau_{d,k} = f(y_d, r_d, x_k)$ . Now, we assume that an infinite number of latent function values  $\{\tau_{1,1}, \tau_{1,2}, \dots\}$  follows a GP prior with mean function  $m(y_d, r_d, x_k)$  and covariance function  $c((y_d, r_d, x_k), (y_{d'}, r_{d'}, x_{k'}))$ . Consequently, any finite set of function values  $\{\tau_{d,k} : d = 1 \dots D; k = 1 \dots K\}$  has a multivariate Gaussian distribution with mean and covariance matrix defined in terms of the mean and

covariance functions of the GP, see e.g. [25]. Without loss of generality, we assume zero mean so that the GP is completely specified by the covariance function only.

In other words, to define the GP prior it is enough to specify the covariance function  $c((y_d, r_d, x_k), (y_{d'}, r_{d'}, x_{k'}))$ . How does it look like in our case? The input features of  $\tau_{d,k}$  include document-oriented information, namely  $y_d$  and  $r_d$ , and topic-oriented information, namely  $x_k$ . Therefore, we decompose the overall covariance as a product of two types of covariances, i.e.,  $c_d((y_d, r_d), (y_{d'}, r_{d'})) \times c_x(x_k, x_{k'})$ . Then, we notice that the document covariance component  $c_d$  involves per-document metadata and relational information. Unfortunately, it is difficult — if not impossible — to represent both jointly using a single kernel only. Consequently, we borrow the underlying assumption in Silva et al.’s relational GP model [27]:  $c_d((y_d, r_d), (y_{d'}, r_{d'}))$  is further decomposed into a sum of two kernels  $c_y(y_d, y_{d'}) + c_r(r_d, r_{d'})$ . Putting everything together, the covariance function of the GP prior is defined as  $[c_y(y_d, y_{d'}) + c_r(r_d, r_{d'})] \times c_x(x_k, x_{k'})$ . The decomposition of the covariance matrix is based on the direct sum and tensor product of kernels [25].

For the per-document respectively per-topic covariance functions  $c_y(y_d, y_{d'})$  respectively  $c_x(x_k, x_{k'})$ , we can select any Mercer kernel. A typical choice is the squared exponential covariance function with isotropic distance measure:

$$c_y(y_d, y_{d'}) = \kappa^2 \exp\left(-\frac{\rho^2}{2} \sum_s^S (y_{d,s} - y_{d',s})^2\right), \quad (1)$$

where  $\kappa$  and  $\rho$  are parameters of the covariance function, and  $y_{d,s}$  denotes the  $s$ -th dimension of the attribute vector  $y_d$ .

For the relation-wise covariance function  $c_r(r_d, r_{d'})$ , any graph kernel is a natural candidate [29,35,27]. Here, we used the  $p$ -steps random walk kernel:

$$(1 - \gamma^{-1} \Delta)^p = \left[ (1 - \gamma^{-1})I + \gamma^{-1} G^{-1/2} W G^{-1/2} \right]^p \quad (2)$$

where  $\gamma$  (with  $\gamma \geq 2$ ) and  $p$  are the two parameters of the graph kernel and  $\Delta = I - G^{-1/2} W G^{-1/2}$ . The matrix  $W$  denotes the adjacency matrix of a weighted, undirected graph, i.e.,  $W_{i,j}$  is taken to be the weight associated with the edge between  $i$  and  $j$ .  $G$  is a diagonal matrix with entries  $g_{i,i} = \sum_j w_{i,j}$ . Here, multiple relations could be encoded by weighted sum of graph kernels, kernels over weighted graphs (also to incorporate link counts) [24] or multi-relational GP’s [32].

The overall covariance matrix  $\Sigma$  (a  $DK \times DK$  matrix) computed with the covariance function  $c((y_d, r_d, x_k), (y_{d'}, r_{d'}, x_{k'}))$  can be represented as  $\Sigma = (\Sigma_Y + \Sigma_R) \otimes \Sigma_T$ , where  $\otimes$  denotes the Kronecker product between two matrices. The matrix  $\Sigma_Y$  is a  $D \times D$  matrix that represents the per-document metadata covariances between documents. It is computed using (1). The matrix  $\Sigma_R$  is also a  $D \times D$  matrix that represents the relation-wise covariances between documents’ metadata. It is computed using (2). The sum  $\Sigma_D = \Sigma_Y + \Sigma_R$  represents the document-oriented covariances. Finally,  $\Sigma_T$  is a  $K \times K$  matrix that represents the covariances between (optional/latent) topic metadata. Every element  $(k, k')$

of  $\Sigma_T$  is computed based on the latent attributes  $x_k$  and  $x_{k'}$  of topics  $k$  and  $k'$  using (II). Together, this leads to the following prior distribution:

$$P(\tau|X, R) = \mathcal{N}(0, \Sigma) = \frac{1}{(2\pi)^{DK} |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{\tau^T \Sigma^{-1} \tau}{2}\right), \quad (3)$$

where  $\tau$  denotes the logarithmic level of the  $\alpha = (\alpha_{1,1} \dots \alpha_{d,k} \dots \alpha_{D,K})$ .

**Generating Documents and Words:** Given the prior  $\mathcal{N}(0, \Sigma)$  and hyperparameters  $\beta$ , the generative process for documents and their words is as follows:

1. Draw  $\tau \sim \mathcal{N}(0, \Sigma)$ .
2. For each topic  $k$ , draw  $\phi_k \sim \text{Dir}(\beta)$ .
3. For each document  $d$ ,
  - (a) Draw  $\theta_d \sim \text{Dir}(\alpha_d) = \text{Dir}(\exp(\tau_d))$  with  $\tau_d \in \tau$ .
  - (b) For each word  $n$ ,
    - Draw  $Z_{d,n} \sim \text{Mult}(\theta_d)$ .
    - Draw  $W_{d,n} \sim \text{Mult}(\phi_{Z_{d,n}})$

The model therefore includes the following fixed parameters: the hyperparameters of the covariance matrix  $\Sigma$ ;  $\beta$ , the Dirichlet prior on the topic-word distributions; and  $K$ , the number of topics.

## 4 Inference and Learning

With the xLDA model defined, we now turn to approximate posterior inference and parameter estimation. The main insight for both is that knowing  $\alpha$  d-separates the relational Gaussian process phase and the LDA phase.

**Inference:** We predict  $\alpha$  given the metadata and the relations and then run any LDA sampler.

**Learning:** Given  $\alpha$ , (1) the GP phase of xLDA is no more complicated than a standard XGP, and (2) the sampling phase of xLDA is no more complicated than a simple LDA sampler. Thus, we can train xLDA using a stochastic EM sampling scheme. That is we alternate between sampling topic assignments from the current prior distribution conditioned on the observed words, features and relations, and numerically optimizing the parameters of the relational Gaussian process given the topic assignments. For that we need the gradients of the (log)likelihood for parts of the model that contain the GP prior respectively the topics  $Z$ .

The likelihood can be found to be  $P(\tau, z|GP) = P(\tau|GP)P(z|\tau)$  with  $\tau = \log(\alpha)$ . Due to (3), the first term on the right-hand side is

$$P(\tau|GP) = \mathcal{N}(0, \Sigma) = \frac{1}{(2\pi)^{DK} |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{\tau^T \Sigma^{-1} \tau}{2}\right)$$

The second term can be written as (see [11] for details)

$$P(z|\tau) = \prod_d^D \frac{\Gamma(\sum_k^K \exp(\tau_{d,k}))}{\Gamma(\sum_k^K \exp(\tau_{d,k}) + n_{d,k})} \prod_k^K \frac{\Gamma(\exp(\tau_{d,k}) + n_{d,k})}{\Gamma(\exp(\tau_{d,k}))}$$

Consequently, the log likelihood is  $LL = \log P(\tau|GP) + \log P(z|\tau) =$

$$\begin{aligned} &= \underbrace{-\frac{1}{2}\tau^T \Sigma^{-1} \tau - \frac{1}{2} \log |\Sigma| - \frac{DK}{2} \log(2\pi)}_{=: \Delta ll_{gp}} \\ &+ \underbrace{\sum_d^D \left( \log \frac{\Gamma(\sum_k^K \exp(\tau_{d,k}))}{\Gamma(\sum_k^K \exp(\tau_{d,k}) + n_{d,k})} + \sum_k^K \log \frac{\Gamma(\exp(\tau_{d,k}) + n_{d,k})}{\Gamma(\exp(\tau_{d,k}))} \right)}_{=: \Delta ll_d} \end{aligned}$$

The derivative of  $LL$  with respect to  $\tau$  can be found to be:

$$\begin{aligned} \frac{\partial LL}{\partial \tau} &= \frac{\partial \Delta ll_{gp}}{\partial \tau} + \frac{\partial \Delta ll_d}{\partial \tau} = -\tau^T \Sigma^{-1} + \frac{\partial \Delta ll_d}{\partial \exp(\tau)} * \frac{\partial \exp(\tau)}{\partial \tau} \\ &= -\tau^T \Sigma^{-1} + \frac{\partial \Delta ll_d}{\partial \exp(\tau)} * \exp(\tau). \end{aligned}$$

With respect to each element  $\tau_{d,k} \in \tau$ , we can find  $(\partial \Delta ll_d)/(\partial \exp(\tau_{d,k})) =$

$$\begin{aligned} &= \frac{\partial}{\partial \exp(\tau_{d,k})} \sum_d^D \left( \log \frac{\Gamma(\sum_k^K \exp(\tau_{d,k}))}{\Gamma(\sum_k^K \exp(\tau_{d,k}) + n_{d,k})} + \sum_k^K \log \frac{\Gamma(\exp(\tau_{d,k}) + n_{d,k})}{\Gamma(\exp(\tau_{d,k}))} \right) \\ &= \Psi\left(\sum_k^K \exp(\tau_{d,k})\right) - \Psi\left(\sum_k^K (\exp(\tau_{d,k}) + n_{d,k})\right) + \Psi(\exp(\tau_{d,k}) + n_{d,k}) - \Psi(\exp(\tau_{d,k})) \end{aligned}$$

where  $\Psi(\cdot)$  is the logarithmic derivative of the Gamma function. This completes the partial derivative of  $LL$  w.r.t  $\tau_{d,k} = \log(\alpha_{d,k})$ . In other words, we can numerically optimize the  $\alpha_{d,k}$  respectively  $\tau_{d,k}$  values given topic assignments.

The partial derivatives of the GP with respect to (hyper)parameters are essentially the same as for standard Gaussian processes; they only appear in  $\Delta ll_{gp}$ , which is the standard data log-likelihood of GPs. Only due to the use of the Kronecker product, the derivatives look slightly different than the standard ones. Let us exemplify this for the (optional/latent) topic metadata; the other ones can be found in a similar fashion. We note that  $\frac{\partial LL}{\partial x} = \frac{\partial \Delta ll_{gp}}{\partial x} + \frac{\partial \Delta ll_d}{\partial x}$ . First,

$$\begin{aligned} \frac{\partial \Delta ll_{gp}}{\partial x} &= -\frac{1}{2} \tau^T \frac{\partial \Sigma^{-1}}{\partial x} \tau - \frac{1}{2} \text{tr}(\Sigma^{-1} \frac{\partial \Sigma}{\partial x}) \\ &= \frac{1}{2} \tau^T \Sigma^{-1} \frac{\partial \Sigma}{\partial x} \Sigma^{-1} \tau - \frac{1}{2} \text{tr}(\Sigma^{-1} \frac{\partial \Sigma}{\partial x}) \\ &= \frac{1}{2} \tau^T \Sigma^{-1} (I_D \otimes (\frac{\partial \Sigma_T}{\partial x} \Sigma_T^{-1})) \tau - \frac{D}{2} \text{tr}(\Sigma_T^{-1} \frac{\partial \Sigma_T}{\partial x}) \end{aligned}$$

with  $\Sigma = \Sigma_D \otimes \Sigma_T$  where  $\otimes$  denotes the Kronecker product. Then, we can find

$$\frac{\partial \Delta lld}{\partial x} = \underbrace{\frac{\partial \Delta lld}{\partial \exp(\tau)} * \frac{\partial \exp(\tau)}{\partial \tau}}_{\text{as above}} * \frac{\partial \tau}{\partial x},$$

where we have  $\frac{\partial \tau}{\partial x} = \frac{\partial}{\partial x} \Sigma^* (\Sigma + \lambda_{noise}^2 I)^{-1} \tau =$

$$= \Sigma^* \frac{\partial \Sigma^{-1}}{\partial x} \tau = -\Sigma^* \Sigma^{-1} \frac{\partial \Sigma}{\partial x} \Sigma^{-1} \tau = -I(I_D \otimes (\frac{\partial \Sigma_T}{\partial x} \Sigma_T^{-1})) \tau.$$

$\Sigma^*$  is the covariance of the training input. In our current setting, it coincides with  $\Sigma$ . For sparse extensions, however, it might be different. Now, we have all gradients together required to implement the stochastic EM approach.

## 5 Experimental Evaluation

Our intention here is to explore the relationship between the latent space computed by xLDA and the underlying link structure. More precisely, we investigated the following question:

(Q) Does the latent space computed by xLDA capture the underlying link structure better than LDA respectively xLDA without relational information?

To do so, we implemented LDA and xLDA in Python and C/C++. We used a standard conjugate-gradient optimizer and a collapsed Gibbs sampling-based LDA trainer. We also compared xLDA with Chang and Blei’s recent relational topic model (RTM) [7]. All experiments ran on a standard Intel(R) Core(TM)2 Duo CPU with 3 GHz and 4GB main memory. All LDA and xLDA models were initialized with Dirichlet hyperparameters set to 5. The parameters of the  $p$ -steps graph kernel were set to  $\gamma = 2.0$  and  $p = 3.0$ . (While we omit a full sensitivity study here, we observed that the performance of the models was similar for  $p = 1, 2, \dots, 8$ ).

**Description of the Datasets:** For the experiments, we used two datasets: a small dataset<sup>1</sup> of Wikipedia web pages used by Gruber *et al.* [12] and the Cora dataset<sup>2</sup> (abstracts with citations) used by Chang and Blei [7].

The Wikipedia dataset is a collection of 105 web pages with in total 89349 words and 790 links between the pages. Gruber *et al.* downloaded the web pages from Wikipedia by crawling within the Wikipedia domain, starting from the NIPS Wikipedia page. The vocabulary consists of 2247 words. The Cora dataset is a collection of 2410 abstracts from the Cora computer science research paper search engine, with in total 126394 words and 4356 links between documents that cite each other. The vocabulary consists of 2961 words. Directed links were

<sup>1</sup> <http://www.cs.huji.ac.il/~amitg/lthm.html>

<sup>2</sup> <http://cran.r-project.org/web/packages/lda/>

converted to undirected links, and documents with no links were removed. For Wikipedia, we also excluded the links of the "Machine Learning" Wikipedia page as it essentially linked to all pages. Furthermore, we turned the link structure into the co-citation link structure. That is if two documents link to another common document, we added an undirected link between these two documents.

**Experimental Protocol:** Due to the transductive nature of our datasets, we considered how well the models predict the remaining words of a document after observing a portion of it. Specifically, we observe  $p$  words from a document and are interested in which model provides a better predictive distribution of the remaining words  $P(w|w_1, w_2, \dots, w_p)$ . To compare these distributions, we use perplexity, which can be thought of as the effective number of equally likely words according to the model:

$$\text{Perp}(\Theta) = \left( \prod_{d=1}^D \prod_{i=p+1}^{N_d} P(w_i|w_1, w_2, \dots, w_p) \right)^{-1/(\sum_{d=1}^D (N_d - p))}$$

where  $\Theta$  denotes the model (hyper-)parameters. Specifically, for each dataset, we created  $p\%$  /  $(100 - p)\%$  train / test splits of the words per document for  $p = 10, 20, 30, \dots, 90$ . We trained the models on each training set and evaluated the perplexity on the corresponding test set. We compared **LDA**, xLDA without relational information, which is identical to DMR with identity matrix (**DMR id**) and to LDA with hyperparameter optimization, DMR with relations as document attributes (**DMR ra**, this is essentially Mimno and McCallum's original DMR model [22] but now using a GP and treating the relations as per-document attributes), and xLDA using the co-citing information (**xLDA**). Both xLDAs estimated no topic metadata; its correlation matrix was set to the identity matrix. Each experiment was repeated 5 times, each time using a different random order of the words per document.

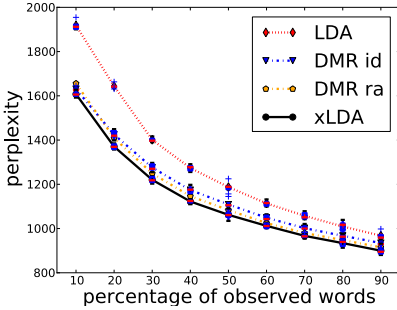
As noted by Gruber *et al.* [12], relational and non-relational LDA models can very well be of comparable quality in terms of perplexity on a dataset. The assignment of topics to documents, however, can be quite different. To measure this effect, we also report the Hellinger distances among related documents, i.e., documents are co-linked. Consider two documents  $d_i$  and  $d_j$

$$\text{dist}(d_i, d_j) = \sum_k \left( \sqrt{\theta_{ik}} - \sqrt{\theta_{jk}} \right)^2 .$$

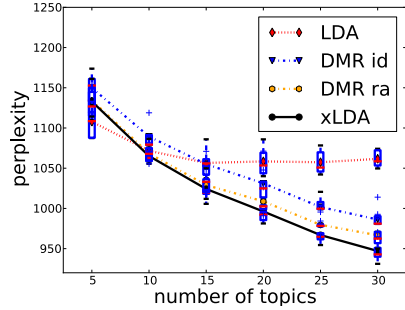
If a model captures the link structure well, we expect the Hellinger distance smaller between co-linked documents.

Additionally, although we are not interested in link prediction per se, we followed Chang and Blei [7] and evaluated the predictive link-likelihood of our models by first fitting the LDA models to the documents (on the full dataset) and then fitting a logistic regression model to the observed links, with input given by the Hadamard (element-wise) product of the latent class distributions of each pair of documents. That is, we first perform unsupervised dimensionality reduction, and then regression to understand the relationship between the latent space and underlying link structure. Here, we additionally compare to **RTM** [7].

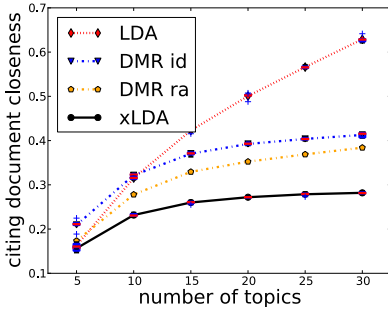




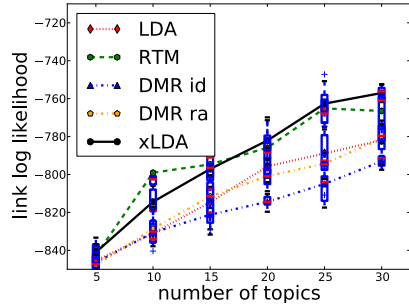
(a) Perplexity (the lower, the better) for different percentages of observed words per document.



(b) Perplexity using 70% / 30% training/test splits for different numbers of topics.



(c) Hellinger distance (the lower, the better) of linked documents for different number of topics.

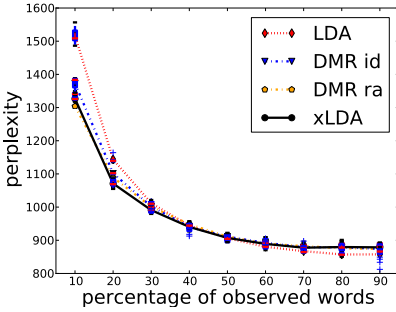


(d) Average link log likelihood (the higher, the better) for different number of topics.

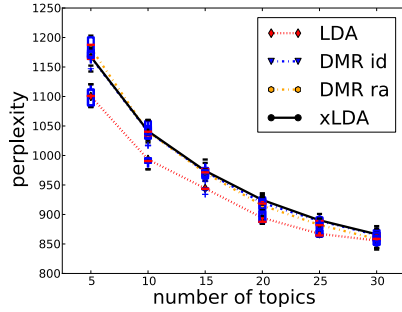
**Fig. 2.** Results on the Cora: Perplexity, Hellinger distance, and average link log-likelihood for **LDA**, **DMR id**, **DMR ra**, and **xLDA** using co-citation. For the average link log-likelihood, we also compare to **RTM**. (Best viewed in color.)

Finally, we investigated the benefit of latent topic meta-information. We ran xLDA estimating latent topic metadata (**xLDAm<sub>tic</sub>**) on the Wikipedia dataset with and without co-citation relations assuming 10 topics. We show the estimated covariance matrices and qualitatively compare the correlations found with the topics found.

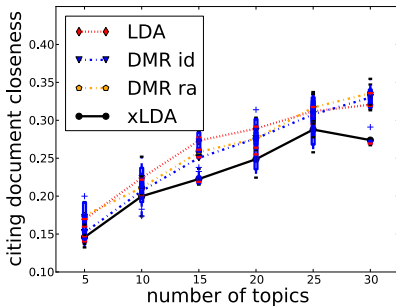
**Results:** The perplexity results on Cora, Fig. 2(a), clearly show that **xLDA** can significantly be less uncertain about the remaining words than **LDA** and **DMR** ( $K = 25$ ). The reason is that after seeing a few words in one topic, **xLDA** uses the link structure to infer that words in a related topic may also be probable. In contrast, **LDA** cannot predict the remaining words as well until a large portion of the document has been observed so that all of its topics are represented. Only when a very small number of words have been observed, the difference starts to vanish. This performance gain was also very stable when varying the number of



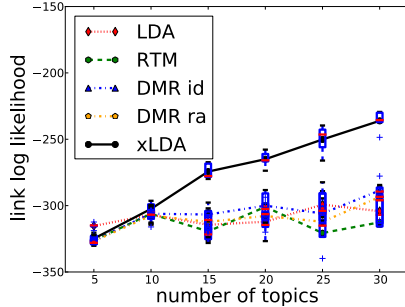
(a) Perplexity (the lower, the better) for different percentages of observed words per document.



(b) Perplexity using 70% / 30% training/test splits for different numbers of topics.



(c) Hellinger distance (the lower, the better) of linked documents for different number of topics.



(d) Average link log likelihood (the higher, the better) for different number of topics.

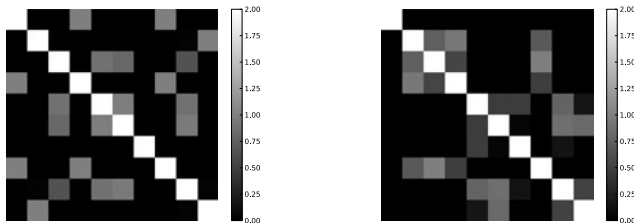
**Fig. 3.** Results on the Wikipedia: Perplexity, Hellinger distance, and average link log-likelihood for **LDA**, **DMR id**, **DMR ra**, and **xLDA** using co-citation. For the average link log-likelihood, we also compare to **RTM**. (Best viewed in color.)

topics as shown in Fig. 2(b). For larger numbers of topics, **LDA** starts to break down compared to **DMR** and **xLDA**. This effect can be broken when optimizing the Dirichlet hyperparameters for each document separately as essentially done by **DMR id**. Again, however, **xLDA** can make use of the link structure to infer that words in a related topic may also be probable. That **xLDA** captures the link structure better is best seen when considering the Hellinger distances between co-linked documents as shown in Fig. 2(c). Most surprisingly, however, in predicting links based on the topics proportions only, **xLDA**'s performance is even comparable with **RTM**'s, a recent fully-generative model.

The perplexity results on Wikipedia, Fig. 3(a), show a similar result. When a small number of words have been observed, there is less uncertainty about the remaining words under **DMR** and **xLDA** than under **LDA** ( $K = 25$ ). Given that this dataset is much smaller, we can better observe that **LDA** cannot

**Table 1.** The 5 nearest pages for two example Wikipedia pages ("Academic Conference", "Bayesian network") according to Hellinger distances (shown next to the page names) learned by **xLDA**, **DMR id**, and **LDA**. Bold pages denote that there is co-citation link between the two pages, italic ones that is a directed link.

Wikipedia: Academic conference					
<i>Proceedings</i>	0.0839	<i>Proceedings</i>	0.0974	<i>Proceedings</i>	0.1634
<b>NIPS</b>	0.1109	<b>MLMTA</b>	0.1865	<b>MLMTA</b>	0.1641
<i>Neural Information Processing Systems</i>	0.1192	Morgan Kaufmann	0.1982	<i>NIPS</i>	0.1853
<b>MLMTA</b>	0.1323	<b>Taxonomy</b>	0.2145	<i>Neural Information Processing Systems</i>	0.1948
Morgan Kaufmann	0.1397	<i>NIPS</i>	0.2318	Inductive transfer	0.2172
Wikipedia: Bayesian network					
<b>Bayes net</b>	0.0015	<b>Bayes net</b>	0.005	<b>Bayes net</b>	0.0022
<i>Markov network</i>	0.0892	<i>Markov network</i>	0.0991	<i>Graphical model</i>	0.1522
<i>Graphical model</i>	0.0932	Random forest	0.1449	<i>Loopy belief propagation</i>	0.1628
<i>Bayesian statistics</i>	0.1153	Minimum message length	0.1467	<i>Variational Bayes</i>	0.1922
<i>Conditional probability</i>	0.1351	<i>Graphical model</i>	0.1498	<i>Markov network</i>	0.1963
<b>xLDA</b>		<b>DMR id</b>		<b>LDA</b>	



No Relations		Co-Citations
theory problems logic called	<b>Topic 0</b>	theory problems logic called
information network time use	<b>Topic 1</b>	information network time use
brain systems vision processing	<b>Topic 2</b>	brain human systems processing
learning fixes import skins	<b>Topic 3</b>	learning fixes import skins
science amp intelligence press	<b>Topic 4</b>	science amp intelligence press
data neural search networks	<b>Topic 5</b>	data neural search networks
city retrieved colorado canada	<b>Topic 6</b>	city retrieved vancouver colorado
probability example recognition new	<b>Topic 7</b>	probability new example recognition
function algorithm model models	<b>Topic 8</b>	function algorithm model method
psychology used study field image	<b>Topic 9</b>	used models analysis psychology

**Fig. 4.** Correlations among latent topic metadata found by **xLDA** on the Wikipedia dataset

predict the remaining words as well until a large portion of the document has been observed so that all of its topics are represented. Zooming in, we found the **LDA** topics on this dataset were of comparable quality, even slightly better, cf. Fig. 3(b). The assignments of topics to documents, however, are very different. **xLDA**'s Hellinger distances between co-linked documents, as shown in Fig. 3(c), is significantly lower for larger number of topics. Table 1 additionally shows for two Wiki pages the 5 nearest Wiki pages. As one can see, **xLDA** gets more related pages closer together. Again kind of surprising, in predicting links based on the topics proportions only, **xLDA**'s performance is even comparable with **RTM**'s performance.

Finally, Figure 4 shows the latent topic metadata correlations estimated by **xLDA** with and without co-citations on the Wikipedia dataset. Without link information, Topic 6 is unrelated to any other topic. This is not surprising as it is about cities (Denver and Vancouver, the current and previous venues of the NIPS conference). When we make uses of the link structure, however, it gets correlated to the meta information of topic 4, science and intelligence. Also the

metadata of topics 1,2,3 of the NIPS conference get more correlated. Note that we used only one-dimensional topic metadata  $x$ . To model richer correlations, one should move to higher dimensions.

To summarize, our experimental results clearly affirmatively answer our question (**Q**): the latent space computed by xLDA captures the underlying link structure better than LDA respectively xLDA without relational information.

## 6 Conclusions

The xLDA model is a new topic model of networks of documents. It can be used to analyze linked corpora such as citation networks, linked web pages, and social networks with user profiles. We have demonstrated qualitatively and quantitatively that the xLDA model provides an effective and useful mechanism for analyzing and using such data. It significantly improves on non-relational topic models, integrating both node-specific information and link structure to give better predictions.

The xLDA model provides a useful complement to fully generative relational topic models such as hyper-linked LDA [12] and the RTM [7], which can make predictions on relations. More importantly, it opens the door to statistical relational reasoning and learning techniques in general. It is a very attractive avenue for future work to explore this connection and to build knowledge rich topic models using probabilistic relational models such as Markov logic network or ProbLog.

**Acknowledgements.** The authors would like to thank Jonathan Chang and David Blei as well as Amit Gruber, Michal Rosen-Zvi and Yair Weiss for making their document networks publically available. This work was partly supported by the Fraunhofer ATTRACT fellowship STREAM and by the German Federal Ministry of Economy and Technology (BMW) under the THESEUS project.

## References

1. Airoldi, E.M., Blei, D.M., Fienberg, S.E., Xing, E.P.: Mixed membership stochastic blockmodels. *Journal of Machine Learning Research* 9, 1981–2014 (2008)
2. Bhattacharya, I., Getoor, L.: A latent dirichlet model for unsupervised entity resolution. In: *Proceeding of SIAM Conference on Data Mining, SDM* (2006)
3. Blei, D., Lafferty, J.: Topic models. In: Srivastava, A., Sahami, M. (eds.) *Text Mining: Theory and Applications*. Taylor & Francis, Abington (2009)
4. Blei, D.M., Lafferty, J.D.: Dynamic topic models. In: *Proceedings of the 23rd International Conference on Machine Learning*, pp. 113–120. ACM, New York (2006)
5. Blei, D.M., Ng, A., Jordan, M.: Latent dirichlet allocation. *Journal of Machine Learning Research* 3, 993–1022 (2003)
6. Buntine, W., Jakulin, A.: Applying discrete pca in data analysis. In: *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 59–66 (2004)
7. Chang, J., Blei, D.: Relational topic models for document networks. In: *Proceeding of the International Conference on Artificial Intelligence and Statistics, AISTATS* (2009)

8. Chu, W., Sindhvani, V., Ghahramani, Z., Keerthi, S.: Relational learning with gaussian processes. In: Neural Information Processing Systems (2006)
9. Deerwester, S., Dumais, S., Landauer, T., Furnas, G., Harshman, R.: Indexing by latent semantic analysis. *Journal of the American Society of Information Science* 41(6), 391–407 (1990)
10. Dietz, L., Bickel, S., Scheffer, T.: Unsupervised prediction of citation influence. In: Proceeding of the International Conference on Machine Learning, ICML (2007)
11. Griffiths, T.L., Steyvers, M.: Finding scientific topics. *PNAS* 101(suppl. 1), 5228–5235 (2004)
12. Gruber, A., Rosen-Zvi, M., Weiss, Y.: Latent topic models for hypertext. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence, UAI (2008)
13. Hofmann, T.: Probabilistic latent semantic indexing. *Research and Development in Information Retrieval*, 50–57 (1999)
14. Tenenbaum, J., Sutskever, I., Salakhutdinov, R.: Modelling relational data using bayesian clustered tensor factorization. *Neural Information Processing Systems* (2009)
15. Kemp, C., Tenenbaum, J.B., Griffiths, T.L., Yamada, T., Ueda, N.: Learning systems of concepts with an infinite relational model. In: Proc. 21st AAAI (2006)
16. Li, F.-F., Perona, P.: A bayesian hierarchical model for learning natural scene categories. In: Proceeding of IEEE CVPR (2005)
17. Li, W., Yeung, D., Zhang, Z.: Probabilistic relational pca. In: Neural Information Processing Systems (2009)
18. McCallum, A., Corrada-Emmanuel, A., Wang, X.: Topic and role discovery in social networks. In: Proceeding of the International Joint Conference on Artificial Intelligence, IJCAI (2005)
19. McCallum, A., Corrada-Emmanuel, A., Wang, X.: Topic and role discovery in social networks. In: Proceedings of International Joint Conference on Artificial Intelligence (2005)
20. Mei, Q., Cai, D., Zhang, D., Zhai, C.: Topic modeling with network regularization. In: Proceeding of the 17th International Conference on World Wide Web (2008)
21. Mimno, D., McCallum, A.: Expertise modeling for matching papers with reviewers. In: Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD (2007)
22. Mimno, D., McCallum, A.: Topic models conditioned on arbitrary features with dirichlet-multinomial regression. In: Proceedings of the Conference on Uncertainty in Artificial Intelligence, UAI (2008)
23. Nallapati, R., Cohen, W.: Link-plsa-lda: A new unsupervised model for topics and influence of blogs. In: Proceedings of the International Conference on Weblogs and Social Media, ICWSM (2008)
24. Neumann, M., Kersting, K., Xu, Z., Schulz, D.: Stacked gaussian process learning. In: Kargupta, W.W.H. (ed.) Proceedings of the 9th IEEE International Conference on Data Mining (ICDM-09), Miami, FL, USA (December 6-9, 2009)
25. Rasmussen, C.E., Williams, C.K.I.: *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge (2006)
26. Rosen-Zvi, M., Griffiths, T., Steyvers, M., Smyth, P.: The author-topic model for authors and documents. In: Proceeding of UAI (2004)
27. Silva, R., Chu, W., Ghahramani, Z.: Hidden common cause relations in relational learning. In: Neural Information Processing Systems (2007)
28. Singh, A.P., Gordon, G.J.: Relational learning via collective matrix factorization. In: Proc. 14th Intl. Conf. on Knowledge Discovery and Data Mining (2008)

29. Smola, A.J., Kondor, I.R.: Kernels and regularization on graphs. In: Annual Conference on Computational Learning Theory (2003)
30. Steyvers, M., Griffiths, T.L., Dennis, S.: Probabilistic inference in human semantic memory. *Trends in Cognitive Science* 10, 327–334 (2006)
31. Wang, C., Blei, D.M., Heckerman, D.: Continuous time dynamic topic models. In: Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (2008)
32. Xu, Z., Kersting, K., Tresp, V.: Multi-relational learning with gaussian processes. In: Boutilier, C. (ed.) Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI-09 (2009)
33. Xu, Z., Tresp, V., Yu, K., Kriegel, H.-P.: Infinite hidden relational models. In: Proc. 22nd UAI (2006)
34. Yu, K., Chu, W.: Gaussian process models for link analysis and transfer learning. In: Neural Information Processing Systems (2007)
35. Zhu, X., Kandola, J., Lafferty, J., Ghahramani, Z.: Graph kernels by spectral transforms. In: Chapelle, O., Schoelkopf, B., Zien, A. (eds.) *Semi-Supervised Learning*. MIT Press, Cambridge (2005)

# Shift-Invariant Grouped Multi-task Learning for Gaussian Processes

Yuyang Wang<sup>1</sup>, Roni Khardon<sup>1</sup>, and Pavlos Protopoulos<sup>2</sup>

<sup>1</sup> Tufts University, Medford, MA USA  
{[ywang02](mailto:ywang02@cs.tufts.edu), [roni](mailto:roni@cs.tufts.edu)}@cs.tufts.edu

<sup>2</sup> Harvard-Smithsonian Center for Astrophysics, Cambridge, MA USA  
[pprotopoulos@cfa.harvard.edu](mailto:pprotopoulos@cfa.harvard.edu)

**Abstract.** Multi-task learning leverages shared information among data sets to improve the learning performance of individual tasks. The paper applies this framework for data where each task is a phase-shifted periodic time series. In particular, we develop a novel Bayesian nonparametric model capturing a mixture of Gaussian processes where each task is a sum of a group-specific function and a component capturing individual variation, in addition to each task being phase shifted. We develop an efficient EM algorithm to learn the parameters of the model. As a special case we obtain the Gaussian mixture model and EM algorithm for phased-shifted periodic time series. Experiments in regression, classification and class discovery demonstrate the performance of the proposed model using both synthetic data and real-world time series data from astrophysics. Our methods are particularly useful when the time series are sparsely and non-synchronously sampled.

## 1 Introduction

In many real world problems we are interested in learning multiple related tasks where the training set for each task is quite small. For example, in pharmacological studies, we may be attempting to predict the concentration of some drug at different times across multiple patients. Finding a good regression function for an individual patient based only on his or her measurements can be difficult due to insufficient training data for the patient. Instead, by using measurements across all the patients, we may be able to leverage common patterns across patients to obtain better estimates for the population and for each patient individually. Multi-task learning captures this intuition aiming to learn multiple correlated tasks simultaneously. This idea has attracted much interest in the literature and several approaches have been applied to a wide range of domains, including medical diagnosis [1], recommendation systems [2] and HIV Therapy Screening [3]. Building on the theoretical framework for single-task learning, multi-task learning has recently been formulated as a multi-task regularization problem in vector-valued Reproducing Kernel Hilbert space (RKHS) [4].

Several approaches to multi-task learning exist in the context of Bayesian statistics. Considering hierarchical Bayesian models, one can view the parameter

sharing of the prior among tasks as a form of multi-task learning [5]. Recently, Bayesian models for multi-task learning were formalized using Gaussian processes [6,7,8]. In this nonparametric mixed-effect model, information is shared among tasks by having each task combine a common (fixed effect) portion and a task specific portion, each of which is generated by an independent Gaussian process. Our work builds on this formulation extending it and the associated algorithms in several ways.

We introduce a multi-task learning model with two novel aspects. First, we allow the fixed effect to be multi-modal so that each task may draw its fixed effect from a different cluster. Second, we extend the model so that each task may be an arbitrarily phase-shifted image of the original time series. This yields our model the *Shift-invariant Grouped Mixed-effect model*.

Our main technical contribution is the inference algorithm for the proposed model. We develop details for the EM algorithm optimizing the *Maximum A Posteriori* (MAP) estimates for the parameters of the model. Technically, the two main insights are in estimating the expectation for the coupled hidden variables (the cluster identities and the task specific portion of the time series) and in solving the regularized least squares problem for a set of phase-shifted observations. As a special case our algorithm yields the Gaussian mixture model (GMM) for phase shifted time series.

Seen from this perspective the paper provides a probabilistic extension of the Phased K-means algorithm [9] that performs clustering for phase-shifted time series data, and a nonparametric Bayesian extension of mixtures of random effects regressions [10] that was recently used for curve clustering.

Our model primarily captures regression of time series but because it is a generative model it can be used for class discovery, clustering and classification. We demonstrate the utility of the model for such applications with both synthetic data and real-world time series data from astrophysics. The experiments show that our model can yield superior results when compared to single task learning and Gaussian mixture models, especially when each individual task is sparsely and non-synchronously sampled.

## 2 Shift-Invariant Grouped Mixed-Effect Model

### 2.1 Preliminaries

Throughout the paper, scalars are denoted using italics, as in  $x, y \in \mathbb{R}$ ; vectors use bold typeface, as in  $\mathbf{x}, \mathbf{y}$ , and  $x_i$  denotes the  $i$ th entry of  $\mathbf{x}$ . For a vector  $\mathbf{x}$  and real valued function  $f : \mathbb{R} \rightarrow \mathbb{R}$ , we extend the notation for  $f$  to vectors so that  $f(\mathbf{x}) = [f(x_1), \dots, f(x_n)]^T$  where superscript the T stands for transposition (and the result is a column vector).  $\mathbb{I}$  is the identity matrix.

A Gaussian process (GP) is a functional extension for Multivariate Gaussian distributions. In the Bayesian literature, it has been widely used in statistical models by substituting a parametric latent function with a stochastic process with Gaussian prior [11]. More precisely, under the single-task setting a simple



Gaussian regression model is given by  $y = f(\mathbf{x}) + \epsilon_i$ , where  $f$ 's prior is a zero-mean GP with covariance function  $\mathcal{K}$  and  $\epsilon_i$  is independent zero mean white noise with variance  $\sigma^2$ . Given data set  $\mathcal{D} = \{\mathbf{x}_i, y_i\}, i = 1, \dots, N$ , let  $\mathbf{K} = (\mathcal{K}(\mathbf{x}_i, \mathbf{x}_j))_{i,j}$ , then  $\mathbf{f} \triangleq [f(\mathbf{x}_1), \dots, f(\mathbf{x}_N)]^T \sim \mathcal{N}(\mathbf{0}, \mathbf{K})$  where  $\mathcal{N}$  denotes the normal distribution and the posterior on  $\mathbf{f}$  is given by

$$\Pr(\mathbf{f}|\mathcal{D}) = \mathcal{N}(\mathbf{K}(\sigma^2\mathbb{I} + \mathbf{K})^{-1}\mathbf{y}, \sigma^2(\sigma^2\mathbb{I} + \mathbf{K})^{-1}\mathbf{K}).$$

The predictive distribution for some test point  $\mathbf{x}_*$  is

$$\begin{aligned} \Pr(f(\mathbf{x}_*)|\mathbf{x}_*, \mathcal{D}) &= \int \Pr(f(\mathbf{x}_*)|\mathbf{x}_*, f) \Pr(f|\mathcal{D}) df \\ &= \mathcal{N}(\mathbf{k}(\mathbf{x}_*)^T(\sigma^2\mathbb{I} + \mathbf{K})^{-1}\mathbf{y}, \mathcal{K}(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}(\mathbf{x}_*)^T(\sigma^2\mathbb{I} + \mathbf{K})^{-1}\mathbf{k}(\mathbf{x}_*)) \end{aligned}$$

where  $\mathbf{k}(\mathbf{x}_*) = [\mathcal{K}(\mathbf{x}_1, \mathbf{x}_*), \dots, \mathcal{K}(\mathbf{x}_N, \mathbf{x}_*)]^T$ . Furthermore, a Gaussian process  $f$  corresponds to a RKHS  $\mathcal{H}$  with kernel  $\mathcal{K}$  such that  $\text{cov}[f(\mathbf{x}), f(\mathbf{y})] = \mathcal{K}(\mathbf{x}, \mathbf{y})$  for any  $\mathbf{x}, \mathbf{y} \in \mathcal{X}$ . In this way, we can express a zero mean Gaussian process as a distribution on functions  $f$  with the following probability [12]<sup>1</sup>

$$f \sim \exp\left\{-\frac{1}{2}\|f\|_{\mathcal{H}}^2\right\}. \quad (1)$$

## 2.2 The GMT Model

Recently, GPs have been used for multi-task learning [6,8,12]. Given data  $\{\mathcal{D}^j\}$ , the Bayesian nonparametric mixed-effect model captures each task  $f^j$  with respect to  $\mathcal{D}^j$  using a sum of an average effect function and an individual variation for each task,  $f^j(x) = \bar{f}(x) + \tilde{f}^j(x)$ ,  $j = 1, \dots, M$  where  $\bar{f}$  and  $\{\tilde{f}^j\}$  are zero mean Gaussian processes. This assumes that the fixed-effect (mean function)  $\bar{f}$  is sufficient to capture the behavior of the data, an assumption that is problematic for distributions with several modes. The model is also not suitable for shifted time series. To address these deficiencies we model our data as follows. For each  $j$  and  $x \in [0, T)$ ,

$$f^j(x) = [\bar{f}_{z_j} * \delta_{t_j}](x) + \tilde{f}^j(x), j = 1, \dots, M \quad (2)$$

where  $z_j \in \{1, \dots, k\}$ ,  $\{\bar{f}_s\}, s = 1, \dots, k$  and  $\tilde{f}^j$  are zero mean Gaussian processes. We use  $*$  to denote circular convolution, and  $\delta_{t_j}$  is the Dirac  $\delta$  function with support at  $t_j \in [0, T)$ . Therefore,  $[\bar{f}_{z_j} * \delta_{t_j}](x) = \bar{f}_{z_j}(x - t_j \bmod T)$ . The underlying GPs, i.e.  $\{\bar{f}_s\}, \tilde{f}^j$ , are assumed to be mutually independent.

We next define the generative model which we call *Shift-invariant Grouped Mixed-effect model* (GMT). In this model,  $k$  group effect functions are assumed to share the same Gaussian prior characterized by  $\mathcal{K}_0$ . The individual effect

<sup>1</sup> In general, a Gaussian process does not induce a probability distribution on the corresponding RKHS but such use has been previously advocated; for further details see [13].

functions are Gaussian processes with covariance function  $\mathcal{K}$ . The model is characterized by parameter set  $\{\mathcal{K}_0, \mathcal{K}, \boldsymbol{\alpha}, \{t_j\}, \sigma^2\}$  and summarized as follows:

$$\begin{aligned} \bar{f}_s | \mathcal{K}_0 &\sim \exp \left\{ -\frac{1}{2} \|\bar{f}_s\|_{\mathcal{H}_0}^2 \right\}, \quad s = 1, 2, \dots, k \\ \tilde{f}^j | \mathcal{K} &\sim \exp \left\{ -\frac{1}{2} \|\tilde{f}^j\|_{\mathcal{H}}^2 \right\}, \\ z_j | \boldsymbol{\alpha} &\sim \text{Discrete}(\boldsymbol{\alpha}), \\ \mathbf{y}^j &\sim \mathcal{N}(f^j(\mathbf{x}^j), \sigma^2 \mathbb{I}), \quad \text{where } f^j = \bar{f}_{z_j} * \delta_{t_j} + \tilde{f}^j, \quad j = 1, 2, \dots, M \end{aligned} \quad (3)$$

where  $\boldsymbol{\alpha}$  specifies the mixture proportions. Let  $\mathcal{X} = \{\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^M\}$  and  $\mathcal{Y} = \{\mathbf{y}^1, \mathbf{y}^2, \dots, \mathbf{y}^M\}$ , where  $\mathbf{x}^j$  are the time points when each time series is sampled and  $\mathbf{y}^j$  are the corresponding observations.

We assume that the group effect kernel  $\mathcal{K}_0$  and the number of centers  $k$  are known. The assumption on  $\mathcal{K}_0$  is reasonable, in that normally we can get more information on the shape of the mean waveforms, thereby making it possible to design kernel for  $\mathcal{H}_0$  but the individual variations are more arbitrary. The assumption that  $k$  is known requires model selection. An extension using a non-parametric model like the *Dirichlet process* that does not limit  $k$  is possible but we leave this to future work. The group effect  $\{\bar{f}_s\}$ , individual shifts  $\{t_j\}$ , noise variance  $\sigma^2$  and the kernel for individual variations  $\mathcal{K}$  are unknown and need to be estimated.

The model above is a standard model for regression. We propose to use it for classification by learning a mixture model for each class and using the MAP probability for the class for classification. In particular, consider a training set that has  $L$  classes, where the  $j$ th instance is given by  $\mathcal{D}^j = (\mathbf{x}^j, \mathbf{y}^j, o^j) \in \mathbb{R}^{n_j} \times \mathbb{R}^{n_j} \times \{1, 2, \dots, L\}$ . Each observation  $(\mathbf{x}^j, \mathbf{y}^j)$  is given a label from  $\{1, 2, \dots, L\}$ . The problem is to learn the model  $M_\ell$  for each class ( $L$  in total) separately and the classification rule for a new instance  $(\mathbf{x}, \mathbf{y})$  is given by  $o = \operatorname{argmax}[\Pr(\mathbf{y}|\mathbf{x}; M_\ell) \Pr(\ell)]$ . As we show in our experiments, the proposed generative model can provide explanatory power for the application while giving excellent classification performance.

### 2.3 Parameter Learning

Given data set  $\mathcal{D} = \{\mathbf{x}^j, \mathbf{y}^j\} = \{x_i^j, y_i^j\}$ ,  $i = 1, \dots, n_j$ ,  $j = 1, \dots, M$ , the learning process aims to find the MAP estimates of the parameter set  $\mathcal{M}^* = \{\boldsymbol{\alpha}, \{\bar{f}_s\}, \{t_j\}, \sigma^2, \mathcal{K}\}$  such that

$$\mathcal{M}^* = \operatorname{argmax}_{\mathcal{M}} (\Pr(\mathcal{Y}|\mathcal{X}; \mathcal{M}) \times \Pr[\{\bar{f}_s\}; \mathcal{K}_0]). \quad (4)$$

The direct optimization of Eq. (4) is analytically intractable because of coupled sums that come from the mixture distribution. To solve this problem, we resort to the EM algorithm. The EM algorithm is an iterative method for optimizing the maximum likelihood (ML) or MAP estimates in the context of hidden variables.

In our case, the hidden variables are  $\mathbf{z} = \{z_j\}$  (which is the same as in standard GMM), and  $\mathbf{f} = \{\mathbf{f}_j \triangleq \tilde{f}^j(\mathbf{x}^j)\}, j = 1, \dots, M$ . The algorithm iterates between the following expectation and maximization steps until it converges to a local maximum.

## 2.4 Expectation Step

In the **E**-step, we calculate

$$Q(\mathcal{M}, \mathcal{M}^g) = \mathbb{E}_{\{\mathbf{z}, \mathbf{f} | \mathcal{X}, \mathcal{Y}; \mathcal{M}^g\}} [\log \{ \Pr(\mathcal{Y}, \mathbf{f}, \mathbf{z} | \mathcal{X}; \mathcal{M}) \times \Pr\{\{\bar{f}_s\}; \mathcal{K}_0\} \}] \quad (5)$$

where  $\mathcal{M}^g$  stands for estimated parameters from the last iteration. For our model, the difficulty comes from estimating the expectation with respect to the coupled latent variables  $\{\mathbf{z}, \mathbf{f}\}$ . We next show how this can be done. First notice that,  $\Pr(\mathbf{z}, \mathbf{f} | \mathcal{X}, \mathcal{Y}; \mathcal{M}^g) = \prod_j \Pr(z_j, \mathbf{f}_j | \mathcal{X}, \mathcal{Y}; \mathcal{M}^g)$  and further that

$$\Pr(z_j, \mathbf{f}_j | \mathcal{X}, \mathcal{Y}; \mathcal{M}^g) = \Pr(z_j | \mathbf{x}^j, \mathbf{y}^j; \mathcal{M}^g) \times \Pr(\mathbf{f}_j | z_j, \mathbf{x}^j, \mathbf{y}^j; \mathcal{M}^g). \quad (6)$$

The first term in Eq. (6) can be further written as

$$\Pr(z_j | \mathbf{x}^j, \mathbf{y}^j; \mathcal{M}^g) \propto \Pr(z_j; \mathcal{M}^g) \Pr(\mathbf{y}^j | z_j, \mathbf{x}^j; \mathcal{M}^g) \quad (7)$$

where  $\Pr(z_j; \mathcal{M}^g)$  is specified by the parameters estimated from last iteration. Since  $z_j$  is given,  $\Pr(\mathbf{y}^j | z_j, \mathbf{x}^j; \mathcal{M}^g)$  is the marginal distribution that can be calculated using a GP regression model, that is,  $\mathbf{y}^j | z_j \sim \mathcal{N}([\bar{f}_{z_j} * \delta_{t_j}](\mathbf{x}^j), \mathbf{K}_j^g + \sigma^2 \mathbb{I})$ . As a result  $\Pr(z_j | \mathbf{x}^j, \mathbf{y}^j; \mathcal{M}^g)$  can be calculated explicitly. Next consider the second term in Eq. (6). Given  $z_j$ , we know that  $f^j = \bar{f}_{z_j} + \tilde{f}^j$ , i.e. there is no uncertainty about the identity of  $\bar{f}_{z_j}$  and therefore the calculation amounts to estimating the posterior distribution under standard GP regression. The conditional distribution is given by  $\mathbf{f}_j | z_j, \mathbf{x}^j, \mathbf{y}^j \sim \mathcal{N}(\mu_j^g, \mathbf{C}_j^g)$  where  $\mu_j^g$  and  $\mathbf{C}_j^g$  are the posterior mean and covariance matrix given by

$$\mu_j^g = \mathbf{K}_j^g (\mathbf{K}_j^g + \sigma^2 \mathbb{I})^{-1} (\mathbf{y}^j - \bar{\mathbf{f}}^j), \quad \mathbf{C}_j^g = \mathbf{K}_j^g - \mathbf{K}_j^g (\mathbf{K}_j^g + \sigma^2 \mathbb{I})^{-1} \mathbf{K}_j^g \quad (8)$$

where  $\mathbf{K}_j^g$  is the kernel matrix for the  $j$ th task using parameters from the last iteration. To derive the concrete form of  $Q(\mathcal{M}, \mathcal{M}^g)$ , denote  $z_{il} = 1$  iff  $z_i = l$ . Then the complete data likelihood can be reformulated as

$$\mathcal{L} = \Pr(\mathcal{Y}, \mathbf{f}, \mathbf{z}; \mathcal{X}, \mathcal{M}) = \prod_j \prod_s [\alpha_s \Pr(\mathbf{y}^j | \mathbf{f}_j, z_j = s; \mathcal{M}) \Pr(\mathbf{f}_j; \mathcal{M})]^{z_{js}} \quad (9)$$

where we have used the fact that exactly one  $z_{js}$  is 1 for each  $j$  and thus included the last term inside the product over  $s$ . Then Eq. (5) can be written as

$$Q(\mathcal{M}, \mathcal{M}^g) = -\frac{1}{2} \sum_s \|f_s\|_{\tilde{\gamma}_{t_0}}^2 + \mathbb{E}_{\{\mathbf{z}, \mathbf{f} | \mathcal{X}, \mathcal{Y}; \mathcal{M}^g\}} [\log \mathcal{L}]. \quad (10)$$

Denote the second term by  $\tilde{Q}$ . By a version of Fubini's theorem we have

$$\begin{aligned} \tilde{Q} &= \mathbb{E}_{\{\mathbf{z}|\mathcal{X}, \mathcal{Y}; \mathcal{M}^g\}} \mathbb{E}_{\{\mathbf{f}|\mathbf{z}, \mathcal{X}, \mathcal{Y}; \mathcal{M}^g\}} [\log \mathcal{L}] \\ &= \sum_{\mathbf{z}} \Pr(\mathbf{z}|\mathcal{X}, \mathcal{Y}; \mathcal{M}^g) \left\{ \sum_j \sum_s z_{js} \right. \\ &\quad \left. \times \int d\Pr(\mathbf{f}_j|z_j = s) \log [\alpha_s \Pr(\mathbf{y}^j|\mathbf{f}_j, z_j = s; \mathcal{M}) \Pr(\mathbf{f}_j; \mathcal{M})] \right\}. \end{aligned} \quad (11)$$

Since the last term in Eq. (11) does not include any  $z_i$ , the equation can be further decomposed as

$$\begin{aligned} \tilde{Q} &= \sum_j \sum_s \left( \sum_{\mathbf{z}} \Pr(\mathbf{z}|\mathcal{X}, \mathcal{Y}; \mathcal{M}^g) z_{js} \right) \\ &\quad \times \left\{ \int d\Pr(\mathbf{f}_j|z_j = s) \log [\alpha_s \Pr(\mathbf{y}^j|\mathbf{f}_j, z_j = s; \mathcal{M}) \Pr(\mathbf{f}_j; \mathcal{M})] \right\} \\ &= \sum_j \sum_s \gamma_{js} \int d\Pr(\mathbf{f}_j|z_j = s) \log [\alpha_s \Pr(\mathbf{y}^j|\mathbf{f}_j, z_j = s; \mathcal{M}) \Pr(\mathbf{f}_j; \mathcal{M})] \end{aligned} \quad (12)$$

where  $\gamma_{js} = \mathbb{E}[z_{js}|\mathbf{y}^j, \mathbf{x}^j; \mathcal{M}^g]$  can be calculated from Eq. (7) and it can be viewed as a fractional label for the  $j$ th task in the  $s$ th group.

Recall that  $\Pr(\mathbf{y}^j|\mathbf{f}_j, z_j = s)$  is a normal distribution given by  $\mathcal{N}(\bar{f}_{z_j} * \delta_{t_j}(\mathbf{x}^j) + \mathbf{f}_j, \sigma^2 \mathbb{I})$  and  $\Pr(\mathbf{f}_j; \mathcal{M})$  is a standard multivariate Gaussian distribution determined by its prior, that is

$$\Pr(\mathbf{f}_j; \mathcal{M}) = \frac{1}{\sqrt{(2\pi)^{n_j} |\mathbf{K}_j|}} \exp \left\{ -\frac{1}{2} \mathbf{f}_j^T \mathbf{K}_j^{-1} \mathbf{f}_j \right\}.$$

Using these facts and after some algebraic manipulation,  $Q(\mathcal{M}, \mathcal{M}^g)$  can be rewritten as

$$\begin{aligned} Q(\mathcal{M}, \mathcal{M}^g) &= -\frac{1}{2} \sum_s \|\bar{f}_s\|_{\mathcal{H}_0}^2 - \sum_j n_j \log \sigma + \sum_j \sum_s \gamma_{js} \log \alpha_s \\ &\quad - \frac{1}{2\sigma^2} \sum_j \sum_s \gamma_{js} \mathbb{E}_{\{\mathbf{f}_j|z_j=s, \mathbf{x}^j, \mathbf{y}^j; \mathcal{M}^g\}} [\|\mathbf{y}^j - [\bar{f}_s * \delta_{t_j}](\mathbf{x}^j) - \mathbf{f}_j\|^2] \\ &\quad + \sum_j \sum_s \gamma_{js} \mathbb{E}_{\{\mathbf{f}_j|\mathbf{x}^j, \mathbf{y}^j; \mathcal{M}^g\}} [\log \Pr(\mathbf{f}_j; \mathcal{M})]. \end{aligned} \quad (13)$$

## 2.5 Maximization Step

In this step, we aim to find

$$\mathcal{M}^* = \underset{\mathcal{M}}{\operatorname{argmax}} Q(\mathcal{M}, \mathcal{M}^g) \quad (14)$$

and use  $\mathcal{M}^*$  to update the model parameters. As we show next, this can be decomposed into three separate optimization problems as follows

$$\mathcal{M}^* = \operatorname{argmax}_{\mathcal{M}} Q_1(\{\bar{f}_s\}, \{\delta_{t_j}\}, \sigma) + Q_2(\mathcal{K}) + \left\{ \sum_j \sum_s \gamma_{js} \log \alpha_s \right\}.$$

That is,  $\alpha$  can be estimated easily using its separate term,  $Q_1$  is only a function of  $(\{\bar{f}_s\}, \{t_j\}, \sigma)$  and  $Q_2$  depends only on  $\mathcal{K}$ . We have

$$Q_1(\{\bar{f}_s\}, \{t_j\}, \sigma^2) = \frac{1}{2} \sum_s \|\bar{f}_s\|_{\mathcal{H}_0}^2 + \sum_j n_j \log \sigma + \frac{1}{2\sigma^2} \sum_j \sum_s \gamma_{js} \times \mathbb{E}_{\{\mathbf{f}_j|z_j=s, \mathbf{x}^j, \mathbf{y}^j; \mathcal{M}^g\}} [\|\mathbf{y}^j - [f_s * \delta_{t_j}](\mathbf{x}^j) - \mathbf{f}_j\|^2]. \tag{15}$$

The remaining part,  $Q_2$  is

$$\begin{aligned} Q_2(\mathcal{K}) &= \sum_j \sum_s \gamma_{js} \mathbb{E}_{\{\mathbf{f}_j|z_j=s, \mathbf{x}^j, \mathbf{y}^j; \mathcal{M}^g\}} [\log \Pr(\mathbf{f}_j; \mathcal{M})]. \\ &= -\frac{1}{2} \sum_j \log |\mathbf{K}_j| - \frac{1}{2} \sum_j \sum_s \gamma_{js} \operatorname{Tr}(\mathbf{K}_j^{-1}(\mathbf{C}_j^g + \mu_j^g(\mu_j^g)^T)). \end{aligned} \tag{16}$$

**Learning  $\{\bar{f}_s\}, \{t_j\}, \sigma^2$ :** To optimize Eq. (15), denote the residual  $\tilde{\mathbf{y}}^j = \mathbf{y}^j - \mu_{js}$ , where  $\mu_{js} = \mathbb{E}[\mathbf{f}_j|\mathbf{y}^j, z_j = s]$ . Given  $\sigma$ , optimizing  $(\{\bar{f}_s\}, \{t_j\})$  decouples into  $k$  independent sub-problems, where finding the  $s$ th group effect  $\bar{f}_s$  and its corresponding shift  $\{t_j\}$  amounts to solving

$$\operatorname{argmin}_{f \in \mathcal{H}_0, t_1, \dots, t_M \in [0, T]} \left\{ \frac{1}{2\sigma^2} \sum_j \gamma_{js} \sum_{i=1}^{n_j} (\tilde{\mathbf{y}}_i^j - [f * \delta_{t_j}](\mathbf{x}_i^j))^2 + \frac{1}{2} \|f\|_{\mathcal{H}_0}^2 \right\}. \tag{17}$$

Note that different  $\mathbf{x}^j, \mathbf{y}^j$  have different dimensions  $n_j$  and they are not assumed to be sampled at regular intervals. For further development, following [8], it is useful to introduce the closure vector  $\tilde{\mathbf{x}} \in \mathbb{R}^{\mathbb{N}}$  whose components are the distinct elements of  $\mathcal{X}$ . For example if  $\mathbf{x}^1 = [1, 2, 3]^T, \mathbf{x}^2 = [2, 3, 4, 5]^T$ , then  $\tilde{\mathbf{x}} = [1, 2, 3, 4, 5]^T$ . For the  $j$ th task, let the binary matrix  $C^k$  be such that  $\mathbf{x}^j = C^j \cdot \tilde{\mathbf{x}}$  and  $f(\mathbf{x}^j) = C^j \cdot f(\tilde{\mathbf{x}})$ . That is,  $C^j$  extracts the values corresponding to the  $j$ th task from the closure vector.

If  $\{t_j\}$  are fixed, then the optimization in Eq. (17) is standard and the representer theorem gives the form of the solution as  $f(\cdot) = \sum_{i=1}^{\mathbb{N}} c_i \mathcal{K}_0(\tilde{x}_i, \cdot)$ . Denoting the kernel matrix as  $\mathfrak{K} = \mathcal{K}_0(\tilde{x}_i, \tilde{x}_j), i, j = 1, \dots, \mathbb{N}$ , and  $\mathbf{c} = [c_1, \dots, c_{\mathbb{N}}]^T$  we get  $f(\tilde{\mathbf{x}}) = \mathfrak{K}\mathbf{c}$ . To simplify the optimization we assume that  $\{t_j\}$  can only take values in the discrete space  $\{\tilde{t}_1, \dots, \tilde{t}_L\}$ , that is,  $t_j = \tilde{t}_i$ , for some  $i \in 1, 2, \dots, L$  (e.g., a fixed finite fine grid), where we always choose  $\tilde{t}_1 = 0$ . Therefore, we can

write  $[f * \delta_{t_j}](\tilde{\mathbf{x}}) = \tilde{\mathcal{K}}_{t_j}^T \mathbf{c}$ , where  $\tilde{\mathcal{K}}_{t_j}$  is  $\mathcal{K}_0(\tilde{\mathbf{x}}, [(\tilde{\mathbf{x}} - \tilde{t}_j) \bmod T])$ . Accordingly, Eq. (17) is reduced to

$$\operatorname{argmin}_{\mathbf{c} \in \mathbb{R}^N, t_1, \dots, t_j \in \{\tilde{t}_i\}} \left\{ \sum_j \gamma_{j_s} \|\tilde{\mathbf{y}}^j - C^j \cdot \tilde{\mathcal{K}}_{t_j}^T \mathbf{c}\|^2 + \frac{1}{2} \mathbf{c}^T \mathfrak{K} \mathbf{c} \right\}. \tag{18}$$

To solve this optimization, we follow a cyclic optimization approach where we alternate between steps of optimizing  $f$  and  $\{t_j\}$  respectively.

At step  $\ell$ , we optimize equation (18) with respect to  $\{t_j\}$  given  $\mathbf{c}^{(\ell)}$ . Since  $\mathbf{c}^{(\ell)}$  is known, it follows immediately that Eq. (18) decomposes to  $M$  independent problems, where for the  $j$ th we need to find  $t_j^{(\ell)}$  such that  $C^j \tilde{\mathcal{K}}_{t_j^{(\ell)}}^T \mathbf{c}$  is closest to  $\tilde{\mathbf{y}}^j$  under Euclidean distance. A brute force search with time complexity  $\Theta(\text{INL})$  yields the optimal solution. If the time series are synchronously sampled (i.e.  $C^j = \mathbb{I}, j = 1, \dots, M$ ) and the allowed time shifts are at sample points, this reduces to finding the shift  $\tau$  corresponding the *cross-correlation*. In this case, one can use the convolution theorem to find the same value in  $\Theta(\text{IN log IN})$  time [14].

At step  $\ell+1$ , we optimize equation (18) with respect to  $\mathbf{c}^{(\ell+1)}$  given  $t_1^{(\ell)}, \dots, t_M^{(\ell)}$ . For the  $j$ th task, since  $t_j^{(\ell)}$  is known, denote  $C^j \tilde{\mathcal{K}}_{t_j^{(\ell)}}^T$  as  $\mathfrak{M}_j^{(\ell)}$ . Therefore,  $\mathbf{c}^{(\ell+1)}$  can be calculated by solving the following regularized least square problem

$$\operatorname{argmin}_{\mathbf{c} \in \mathbb{R}^N} \left\{ \sum_j \gamma_{j_s} \|\tilde{\mathbf{y}}^j - \mathfrak{M}_j^{(\ell)} \mathbf{c}\|^2 + \frac{1}{2} \mathbf{c}^T \mathfrak{K} \mathbf{c} \right\}. \tag{19}$$

Obviously, each step decreases the value of the objective function and therefore the algorithm will converge.

Given  $\{\tilde{f}_s\}, \{t_j\}$ , the value of  $\sigma^2$  is optimized by  $(\sigma^*)^2 = R / \sum_j n_j$ , where  $R = \sum_j \text{Tr}(\mathbf{C}_j^g) + \sum_j \sum_s \gamma_{j_s} (\|\mathbf{y}^j - [\tilde{f}_s^* * \delta_{t_j}](\mathbf{x}^j) - \boldsymbol{\mu}_{j_s}\|^2)$ . We use alternating optimization steps to optimize over the three parameters together.

**Learning the kernel for individual effects:** The work of [12] has already shown how to optimize the kernel in a similar context. Here we provide some of the details for completeness. If the kernel function  $\mathcal{K}$  admits a parametric form with parameter  $\theta$ , for example the RBF kernel

$$\mathcal{K}(x, y) = a \exp \left\{ -\frac{\|x - y\|^2}{2s^2} \right\} \tag{20}$$

then the optimization of  $\mathcal{K}$  amounts to finding  $\theta^*$  such that

$$\theta^* = \operatorname{argmax}_{\theta} \left\{ -\frac{1}{2} \sum_j \sum_s \gamma_{j_s} \text{Tr} \left( (\mathbf{K}_j; \theta)^{-1} (\mathbf{C}_j^g + \boldsymbol{\mu}_{j_s}^g (\boldsymbol{\mu}_{j_s}^g)^T) \right) \right\}.$$

The parametric form of the kernel is a prerequisite to perform the regression task when examples are not sampled synchronously as in our development above. If

the data is synchronously sampled, for classification tasks we only need to find the kernel matrix  $\mathbf{K}$  for the given sample points and the optimization problem can be rewritten as

$$\mathbf{K}^* = \operatorname{argmax}_{\mathbf{K}} \left\{ -\frac{1}{2} \sum_j \log |\mathbf{K}| - \frac{1}{2} \sum_j \sum_s \gamma_{js} \operatorname{Tr} (\mathbf{K}^{-1} (\mathbf{C}_j^g + \boldsymbol{\mu}_{js}^g (\boldsymbol{\mu}_{js}^g)^\top)) \right\}. \quad (21)$$

Similar to maximum likelihood estimation for the multivariate Gaussian distribution, the solution is  $\mathbf{K}^* = \frac{1}{M} \sum_j \sum_s \gamma_{js} (\mathbf{C}_j^g + \boldsymbol{\mu}_{js}^g (\boldsymbol{\mu}_{js}^g)^\top)$ . In our experiments, we use both approaches. For the parametric form we use Eq. (20).

### 3 Experiments

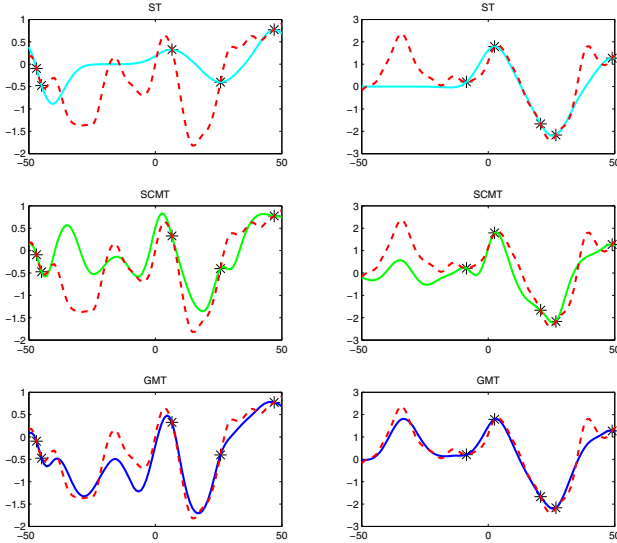
Our implementation of the algorithm makes use of the gpml package<sup>2</sup> [11] and extends it to implement the required functions. The EM algorithm is restarted 5 times and the function that best fits the data is chosen. The EM algorithm stops when the difference of the log-likelihood is less than 10e-5 or at a maximum of 200 iterations.

#### 3.1 Regression on Synthetic Data

In the first experiment, we demonstrate the performance of our algorithm on a regression task with artificial data. We generated data that is not phase shifted under a mixture of two Gaussian processes. More precisely, each  $\bar{f}_s(x)$ ,  $s = 1, 2$  is generated on interval  $[-50, 50]$  from a Gaussian process with covariance function  $\operatorname{cov}[\bar{f}_s(t_1), \bar{f}_s(t_2)] = e^{-(t_1-t_2)^2/25}$ ,  $s = 1, 2$ . The individual effect  $\tilde{f}_j$  is sampled via a Gaussian process with the covariance function  $\operatorname{cov}[\tilde{f}_j(t_1), \tilde{f}_j(t_2)] = 0.2e^{-(t_1-t_2)^2/16}$ . Then the hidden label  $z_j$  is sampled from a discrete distribution with  $\alpha = [0.5, 0.5]$ . The vector  $\check{\mathbf{x}}$  consists of 100 samples on  $[-50, 50]$ . We fix a sample size  $N$  and each  $\mathbf{x}^j$  includes  $N$  randomly chosen points from  $\{\check{x}_1, \dots, \check{x}_{100}\}$ . The observation  $f^j(\mathbf{x}^j)$  is obtained as  $(f_{z_j} + \tilde{f}_j)(\mathbf{x}^j)$ . In the experiment, we vary the individual sample length  $N$  from 5 to 50. Finally, we generated 50 random tasks with the observation  $\mathbf{y}^j$  for task  $j$  given by  $\mathbf{y}^j \sim \mathcal{N}(f^j(\mathbf{x}^j), 0.01 \times \mathbb{I})$ ,  $j = 1, \dots, 50$ . The methods compared here include:

1. **Single-task learning procedure (ST)**, where each  $\bar{f}^j$  is estimated only using  $\{\mathbf{x}_i^j, \mathbf{y}_i^j\}, i = 1, 2, \dots, N$ .
2. **Single center mixed-effect multi-task learning (SCMT)**, amounts to the mixed-effect model [8] where one average function  $\bar{f}$  is learned from  $\{\mathbf{x}^j, \mathbf{y}^j\}, j = 1, \dots, 50$  and  $f^j = \bar{f} + \tilde{f}^j, j = 1, \dots, 50$ .
3. **Grouped mixed-effect model (GMT)**, the proposed method.

<sup>2</sup> Available at <http://www.gaussianprocess.org/gpml/>



**Fig. 1.** Simulated data: Comparison of the estimated function between single, multi-task and grouped multi-task. The red dotted line is the reference true function.

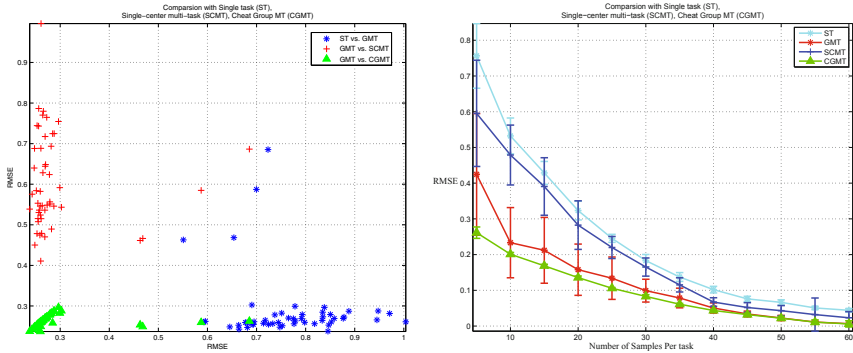
4. **“Cheating” grouped fixed-effect model (CGMT)**, which follows the same algorithm as the GMT but uses the true label  $z_j$  instead of their expectation for each task  $j$ . This serves as an upper bound for the performance of the proposed algorithm.

Except for ST, the other three algorithms use the same method to learn the kernel of the individual effects, which is assumed to be RBF. The Root Mean Square Error (RMSE) for the four approaches is reported. For task  $j$ , the RMSE is defined as  $\text{RMSE}_j = \sqrt{\|f(\tilde{\mathbf{x}}) - f^j(\tilde{\mathbf{x}})\|^2/100}$ , where  $f$  is the learned function and RMSE for the data set is the mean of  $\{\text{RMSE}_j\}, j = 1, \dots, 50$ .

To illustrate the results qualitatively, we first plot in Fig. (1) the true and learned functions in one trial. The left column illustrates some task that is sampled from group effect  $\bar{f}_1$  and the right column is for  $\bar{f}_2$ . It is easy to see that, as expected, the tasks are poorly estimated under ST due to the sparse sampling. The SCMT performs better than ST but its estimate is poor in areas where two centers disagree. The estimates of GMT are much closer to the true function.

The left plot of Fig. (2) shows a comparison of the algorithms for 50 random data sets under the above setting when  $N$  equals 5. We see that most of the time GMT performs as well as its upper bound, illustrating that it recovers the correct membership of each task. On a few data sets, our algorithm is trapped in a local maximum yielding performance similar to SCMT and ST. The right part of Fig. (2) shows the RMSE for increasing values of  $N$ . From the plot, we can draw the conclusion that the proposed method works much better than SCMT and ST when the number of samples is less than 30. As the number of samples





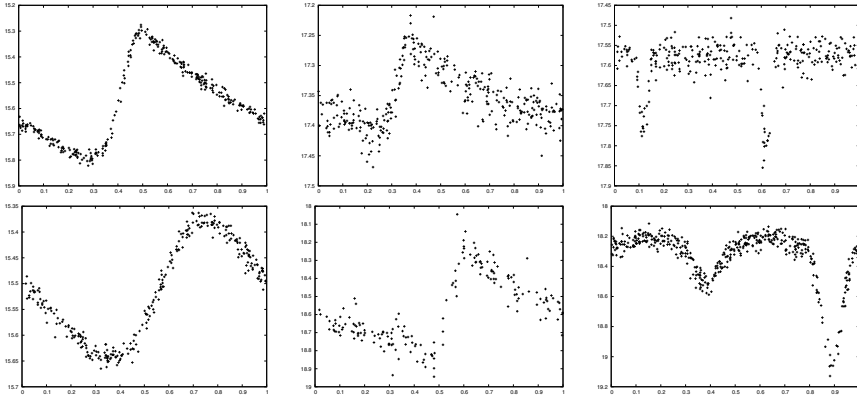
**Fig. 2.** Comparison between single, multi-task, and grouped multi-task learning on simulated data. Left: the figure gives 3 pairwise comparisons when sample size is 5. We can see that the GMT is better than ST since the blue stars are concentrated on the lower right. Similarly, the plot of red pluses demonstrates the advantage of GMT over SCMT and the plot of green triangles shows that the algorithm behaves almost as well as its upper bound. Right: the figure shows RMSE as a function of sample size.

for each task increases, all methods are improving, but the proposed method always outperforms SCMT and ST in our experiments. Finally, all algorithms converge to almost the same performance level where observations in each task are sufficient to recover the underlying function.

### 3.2 Classification on Astrophysics Data

The concrete application motivating this research is the classification of stars into several meaningful categories from the astronomy literature. Classification is an important step within astrophysics research, as evidenced by published catalogs such as OGLE [15] and MACHO [16,17]. However, the number of stars in such surveys is increasing dramatically. For example Pan-STARRS [18] will collect data on the order of hundreds of billions of stars. Therefore, it is desirable to apply state-of-art machine learning techniques to enable automatic processing for astrophysics data classification. The data from star surveys is normally represented by time series of brightness measurements, based on which they are classified into categories. Stars whose behavior is periodic are especially of interest in such studies. Fig. (3) shows several examples of such time series generated from the three major types of periodic variable stars: Cepheid, RR Lyrae, and Eclipsing Binary. In our experiments only stars of the types in Fig. (3) are present in the data, and the period of each star is given.

From Fig. (3), it can be noticed that there are two main characteristics of this domain. The time series are not phase aligned, meaning that the light curves in the same category share a similar shape but with some unknown shift. The time series are non-synchronously sampled and each light curve has a different number of samples and sampling times. We run our experiment on the OGLEII data set [19]. This data set consists of 14087 time series (light curves) with



**Fig. 3.** Examples of light curves of periodic variable stars. Each column shows two stars of the same type. Left: Cepheid, middle: RR Lyrae, right: Eclipsing Binary.

(3425,3390,7272) in the categories (CEPH, RRL, EB) respectively. We perform several experiments with this data set to explore the potential of the proposed method. Previous work using this data [20] developed a kernel for periodic time series and used it with SVM to obtain good classification performance. We use the results of [20] as our baseline.<sup>3</sup>

**Classification using dense-sampled time series:** In the first experiment, the time series are smoothed using a simple average filter, re-sampled to 50 points via linear-interpolation and normalized to have mean 0 and standard deviation of 1. Therefore, the time series are synchronously sampled in the pre-processing stage. We compare our method to GMM and 1-Nearest Neighbor (1-NN). These two approaches are performed on the time series processed by Universal phasing (UP), which uses the method from [14] to phase each time series according to the sliding window on the time series with the maximum mean. We use a sliding window size of 5% of the number of original points; the phasing takes place after the pre-processing explained above. We learn a model for each class separately and for each class the model order for the GMM and the GMT is set to 15.

We run 10-fold cross-validation (CV) over the entire data set and the results are shown in Tab. (1). We see that when the data is densely and synchronously sampled, the proposed method performs similar to the GMM, and they both outperform the kernel based results from [20]. The poor performance of SCMT shows that a single center is not sufficient for this data. The similarity of the GMM and the proposed method under these experimental conditions is not surprising. The reason is that when the time series are synchronously sampled,

<sup>3</sup> [20] used additional features, in addition to time series itself, to improve the classification performance. Here we focus on results using the time series only. Extensions to add such features to our model are orthogonal to the theme of the paper and we therefore leave them to future work in the context of the application.

**Table 1.** Accuracies with standard deviations reported on OGLEII dataset

	UP + GMM	SCMT	GMT	UP + 1-NN	K + SVM <sup>20</sup>
RESULTS	0.956 ± 0.006	0.874 ± 0.008	0.952 ± 0.005	0.865 ± 0.006	0.947 ± 0.005

aside from the difference of phasing, finding the group effect functions is reduced to estimating the mean vectors of the GMM. In addition, learning the kernel in the nonparametric approach is the same as estimating the covariance matrix of the GMM. More precisely, assuming all time series are phased and sampled at the same time points, the following results hold:

1. By placing a flat prior on the group effect function  $\bar{f}_s, s = 1, \dots, k$ , or equivalently setting  $\|\bar{f}_s\|_{\mathcal{H}_0}^2 = 0$ , Eq. (17) is reduced to finding a vector  $\mu_s \in \mathbb{N}$  that minimizes  $\sum_j \gamma_{js} \|\tilde{\mathbf{y}}^j - \mu_s\|^2$ . Therefore, we obtain  $\bar{f}_s = \mu_s = \sum_j \gamma_{js} \tilde{\mathbf{y}}^j / \sum_j \gamma_{js}$ , which is exactly the mean of the  $s$ th cluster during the iteration of EM algorithm under the GMM setting.

2. The kernel  $\mathbf{K}$  is learned in a nonparametric way. Instead of estimating  $\mathbf{K}$  and  $\sigma^2$ , it is convenient to put the two terms together, forming  $\hat{\mathbf{K}} = \mathbf{K} + \sigma^2 \mathbb{I}$ , that is  $\mathbf{y}^j$  is a deterministic function of  $f^j(\mathbf{x}^j)$  which in turn has an extra  $\sigma^2$  term on the diagonal of its covariance matrix. One can show that in this case the estimate of the kernel is  $\hat{\mathbf{K}} = \frac{1}{M} \sum_j \sum_s \gamma_{js} (\mathbf{y}^j - \mu_s)(\mathbf{y}^j - \mu_s)^T$ . In the standard EM algorithm for GMM, this is equal to the estimated covariance matrix when all  $k$  clusters are assumed to have the same covariance.

Accordingly, when time series are synchronously sampled, the proposed model can be viewed as an extension of the Phased K-means [9] and a variant of [21]. In experiments below, we extend the resulting Phased EM by allowing each cluster to have a separate covariance matrix.

**Classification using sparse-sampled time series:** The OGLEII data set is in some sense a “nice” subset of the data from its corresponding star survey. Stars with small number of samples are often removed in pre-processing steps (cf., [22]). The proposed method potentially provides a way to include these instances in the classification process. In the second experiment, we demonstrate the performance of the proposed method on times series with sparse samples. Similar to the synthetic data, we started from sub-sampled versions of the original time series to simulate the condition that we would encounter in further star surveys.<sup>4</sup> As in the previous experiment, each time series is universally phased, normalized and linearly-interpolated to length 50 to be plugged into GMM and 1-NN as well as the generalized Phased EM as discussed above. The RBF kernel is used for the proposed method and we used the same model order as above. Moreover, the performance for PKmeans is also presented, where the classification step is as follows: we learn the PKmeans model with  $k = 15$  for each class

<sup>4</sup> For the proposed method, we clip the samples to a fine grid of 200 equally spaced time points on  $[0, 1]$ , which is also the set of allowed time shifts. This avoids having a very high dimensional  $\tilde{\mathbf{x}}$ , e.g. over 18000 for OGLEII, which is not feasible for any kernel based regression method that relies on solving linear systems.

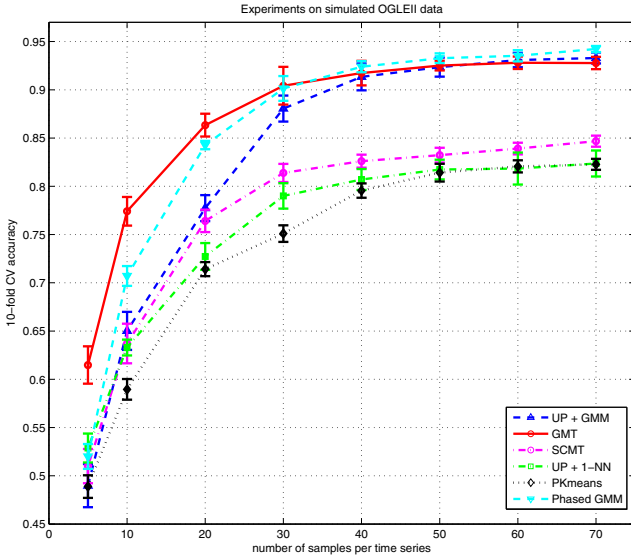


Fig. 4. OGLEII data: Comparison of algorithms with sparsely sampled data

and then the label of a new example is assigned to be the same as its closest centroid’s label. PKmeans is also restarted 5 times and the best clustering is used for classification.

The results are shown in Fig. (4). When each time series has sparse samples (the number of samples per task is less than 30), the proposed method has a significant advantage over the other methods. As the number of samples per task increases, the proposed method improves fast and performs close to its optimal performance given by previous experiment. When the number of samples increases, the performance of the Phased EM gradually catches up and becomes better than the proposed method when each task has more than 50 samples. GMM plus universal phasing (UP) also achieves better performance when time series are densely sampled. One reason for the performance difference is the parametric form of the kernel in the GMT in this experiment (which is less flexible). The difference can also be attributed to the sharing of the covariance function in our model where the GMM and the Phased EM do not apply this constraint. Notice that the Phased EM algorithm always outperforms the GMM plus UP demonstrating that re-phasing the time series inside the EM algorithm improves the results. We also notice that for all 3 methods, the performance with dense data is lower than the results in Tab. (1). This is caused because the data set obtained by the interpolation of the sub-sampled measurements contains less information than that interpolated from the original measurements. Finally, note that PKmeans performs much worse than Phased EM showing that the probabilistic model adds a significant advantage. To summarize, we conclude from this experiment that the proposed method should be used when data is sparsely and non-synchronously sampled.

### 3.3 Class Discovery on Astrophysics Data

We show the potential of our model for class discovery by running GMT on the joint data set of the three classes (not using the labels) with a model order of 45. Then, each cluster is labeled according to the majority class of the instances that belong to the center. For a new test point, we determine which cluster it belongs to via the MAP probability and its label is given by the cluster that it is assigned to. We run 10 trials with different random initialization. The accuracy and standard deviation obtained are [0.895, 0.010]. Given the size of the data set and the relatively small number of clusters this is a significant indication of the potential for class discovery in astrophysics.

## 4 Conclusion and Future Work

We developed a novel Bayesian nonparametric multi-task learning model where each task is modeled as a sum of a group-specific function and an individual task function with a Gaussian process prior. We gave an efficient EM algorithm to learn the parameters of the model and demonstrated its effectiveness using experiments in regression, classification and class discovery.

The literature includes a significant amount of work on time series classification [23] (space constraints preclude a lengthy discussion). These include approaches based on feature extraction followed by feature based learning model [24], approaches based on similarities for time series [20] and hidden Markov models [25,26]. Despite the similar name, our model is different from mixture of experts where different GPs are in control of sub-regions of the input space [27]. Our work is most closely related to the so-called *mixture of regressions* [10,28,29]. As discussed above our model can be seen as a nonparametric Bayesian extension of the model in [10] and includes [21] as a special case with shared covariance matrix when we use the diagonal kernel function on the time grid.

For application in the astronomy context it is important to consider all steps of processing and classification so as to provide an end to end system. Therefore, two important issues to be addressed in future work include incorporating the period estimation phase into the method and developing an appropriate method for abstention in the classification step. Further, to get a full generalization of the phased GMM, it will be interesting to generalize our model to allow individual variations to come from cluster dependent RKHS. It would also be interesting to develop a corresponding discriminative model extending [5] to the GP context.

## Acknowledgments

This research was partly supported by NSF grant IIS-0803409. The experiments in this paper were performed on the Odyssey cluster supported by the FAS Research Computing Group at Harvard and the Tufts Linux Research Cluster supported by Tufts UIT Research Computing.

## References

1. Bi, J., Xiong, T., Yu, S., Dundar, M., Rao, R.: An improved multi-task learning approach with applications in medical diagnosis. In: ECML/PKDD, pp. 117–132 (2008)
2. Dinuzzo, F., Pilonetto, G., De Nicolao, G.: Client-server multi-task learning from distributed datasets. Arxiv preprint arXiv:0812.4235 (2008)
3. Bickel, S., Bogojeska, J., Lengauer, T., Scheffer, T.: Multi-task learning for HIV therapy screening. In: ICML, pp. 56–63 (2008)
4. Evgeniou, T., Micchelli, C., Pontil, M.: Learning multiple tasks with kernel methods. *JMLR* 6(1), 615–637 (2006)
5. Xue, Y., Liao, X., Carin, L., Krishnapuram, B.: Multi-task learning for classification with Dirichlet process priors. *JMLR* 8, 35–63 (2007)
6. Yu, K., Tresp, V., Schwaighofer, A.: Learning Gaussian processes from multiple tasks. In: ICML, pp. 1012–1019 (2005)
7. Schwaighofer, A., Tresp, V., Yu, K.: Learning Gaussian process kernels via hierarchical Bayes. *NIPS* 17, 1209–1216 (2005)
8. Pilonetto, G., Dinuzzo, F., De Nicolao, G.: Bayesian Online Multitask Learning of Gaussian Processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32(2), 193–205 (2010)
9. Rebbapragada, U., Protopapas, P., Brodley, C.E., Alcock, C.: Finding anomalous periodic time series. *Machine Learning* 74(3), 281–313 (2009)
10. Gaffney, S.J., Smyth, P.: Curve clustering with random effects regression mixtures. In: Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics (2003)
11. Rasmussen, C.E., Williams, C.K.I.: *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge (2005)
12. Lu, Z., Leen, T., Huang, Y., Erdogmus, D.: A reproducing kernel Hilbert space framework for pairwise time series distances. In: ICML, pp. 624–631 (2008)
13. Seeger, M.: Gaussian processes for machine learning. *International Journal of Neural Systems* 14(2), 69–106 (2004)
14. Protopapas, P., Giammarco, J.M., Faccioli, L., Struble, M.F., Dave, R., Alcock, C.: Finding outlier light curves in catalogues of periodic variable stars. *Monthly Notices of the Royal Astronomical Society* 369, 677–696 (2006)
15. Udalski, A., Szymanski, M., Kubiak, M., Pietrzynski, G., Wozniak, P., Zebrun, Z.: Optical gravitational lensing experiment. photometry of the macho-smc-1 microlensing candidate. *Acta Astronomica* 47, 431–436 (1997)
16. Alcock, C., et al.: The MACHO Project - a Search for the Dark Matter in the Milky-Way. In: Soifer, B.T. (ed.) *Sky Surveys. Protostars to Protogalaxies*. Astronomical Society of the Pacific Conference Series, vol. 43, pp. 291–296 (1993)
17. Faccioli, L., Alcock, C., Cook, K., Prochter, G.E., Protopapas, P., Syphers, D.: Eclipsing Binary Stars in the Large and Small Magellanic Clouds from the MACHO Project: The Sample. *Astronomy Journal* 134, 1963–1993 (2007)
18. Hodapp, K.W., et al.: Design of the Pan-STARRS telescopes. *Astronomische Nachrichten* 325, 636–642 (2004)
19. Soszynski, I., Udalski, A., Szymanski, M.: The Optical Gravitational Lensing Experiment. Catalog of RR Lyr Stars in the Large Magellanic Cloud 06. *Acta Astronomica* 53, 93–116 (2003)
20. Wachman, G., Khardon, R., Protopapas, P., Alcock, C.: Kernels for Periodic Time Series Arising in Astronomy. In: ECML/PKDD, pp. 489–505 (2009)

21. Chudova, D., Gaffney, S.J., Mjolsness, E., Smyth, P.: Translation-invariant mixture models for curve clustering. In: SIGKDD, pp. 79–88 (2003)
22. Wachman, G.: Kernel Methods and Their Application to Structured Data. PhD thesis, Tufts University (2009)
23. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and mining of time series data: experimental comparison of representations and distance measures. VLDB 1(2), 1542–1552 (2008)
24. Osowski, S., Hoai, L., Markiewicz, T.: Support vector machine-based expert system for reliable heartbeat recognition. IEEE Transactions on Biomedical Engineering 51(4), 582–589 (2004)
25. Hughes, N., Tarassenko, L., Roberts, S.: Markov models for automated ECG interval analysis. NIPS 16 (2003)
26. Kim, S., Smyth, P.: Segmental Hidden Markov Models with Random Effects for Waveform Modeling. JMLR 7, 945–969 (2006)
27. Rasmussen, C.E., Ghahramani, Z.: Infinite mixtures of Gaussian process experts. NIPS 15, 881–888 (2002)
28. Gaffney, S.J., Smyth, P.: Trajectory clustering with mixtures of regression models. In: SIGKDD, pp. 63–72 (1999)
29. Gaffney, S.J., Smyth, P.: Joint probabilistic curve clustering and alignment. NIPS 17, 473–480 (2005)

# Nonparametric Bayesian Clustering Ensembles

Pu Wang<sup>1</sup>, Carlotta Domeniconi<sup>1</sup>, and Kathryn Blackmond Laskey<sup>2</sup>

<sup>1</sup> Department of Computer Science

<sup>2</sup> Department of Systems Engineering and Operations Research  
George Mason University  
4400 University Drive, Fairfax, VA 22030 USA

**Abstract.** Forming consensus clusters from multiple input clusterings can improve accuracy and robustness. Current clustering ensemble methods require specifying the number of consensus clusters. A poor choice can lead to under or over fitting. This paper proposes a nonparametric Bayesian clustering ensemble (NBCE) method, which can discover the number of clusters in the consensus clustering. Three inference methods are considered: collapsed Gibbs sampling, variational Bayesian inference, and collapsed variational Bayesian inference. Comparison of NBCE with several other algorithms demonstrates its versatility and superior stability.

## 1 Introduction

Clustering ensemble methods operate on the output of a set of base clustering algorithms to form a consensus clustering. Clustering ensemble methods tend to produce more robust and stable clusterings than the individual solutions [28]. Since these methods require only the base clustering results and not the raw data themselves, clustering ensembles provide a convenient approach to privacy preservation and knowledge reuse [31]. Such desirable aspects have generated intense interest in cluster ensemble methods.

A variety of approaches have been proposed to address the clustering ensemble problem. Our focus is on statistically oriented approaches. Topchy et al. [28] proposed a mixture-membership model for clustering ensembles. Wang et al. [31] applied a Bayesian approach to discovering clustering ensembles. The Bayesian clustering ensemble model has several desirable properties [31]: it can be adapted to handle missing values in the base clusterings; it can handle the requirement that the base clusterings reside on a distributed collection of hosts; and it can deal with partitioned base clusterings in which different partitions reside in different locations. Other clustering ensemble algorithms, such as the cluster-based similarity partitioning algorithm (CSPA) [25], the hypergraph partitioning algorithm (HGPA) [25], or  $k$ -means based algorithms [18] can handle one or two of these cases; however, none except the Bayesian method can address them all.

Most clustering ensemble methods have the disadvantage that the number of clusters in the consensus clustering must be specified *a priori*. A poor choice can lead to under- or over-fitting. Our approach, nonparametric Bayesian clustering ensembles (NBCE), can discover the number of clusters in the consensus



clustering from the observations. Because it is also a Bayesian approach, NBCE inherits the desirable properties of the Bayesian clustering ensembles model [31]. Similar to the mixture modeling approach [28] and the Bayesian approach [31], NBCE treats all base clustering results for each object as a feature vector with discrete feature values, and learns a mixed-membership model from this feature representation.

The NBCE model is adapted from the Dirichlet Process Mixture (DPM) model [22]. The following sections show how the DPM model can be adapted to the clustering ensemble problem, and examine three inference methods: collapsed Gibbs sampling, standard variational Bayesian inference, and collapsed variational Bayesian inference. These methods are compared in theory and practice. Our empirical evaluation demonstrates the versatility and superior stability and accuracy of NBCE.

## 2 Related Work

A clustering ensemble technique is characterized by two components: the mechanism to generate diverse partitions, and the consensus function to combine the input partitions into a final clustering. Diverse partitions are typically generated by using different clustering algorithms [1], or by applying a single algorithm with different parameter settings [10,16,17], possibly in combination with data or feature sampling [30,9,20,29].

One popular methodology to build a consensus function utilizes a co-association matrix [10,11,20,30]. Such a matrix can be seen as a similarity matrix, and thus can be used with any clustering algorithm that operates directly on similarities [30,1]. As an alternative to the co-association matrix, voting procedures have been considered to build consensus functions in [7]. Gondek et al. [11] derive a consensus function based on the Information Bottleneck principle: the mutual information between the consensus clustering and the individual input clusterings is maximized directly, without requiring approximation.

A different popular mechanism for constructing a consensus maps the problem onto a graph-based partitioning setting [25,3,12]. In particular, Strehl et al. [25] propose three graph-based approaches: Cluster-based Similarity Partitioning Algorithm (CSPA), HyperGraph Partitioning Algorithm (HGPA), and Meta-Clustering Algorithm (MCLA). The methods use METIS (or HMETIS) [15] to perform graph partitioning. The authors in [23] develop soft versions of CSPA, HGPA, and MCLA which can combine soft partitionings of data.

Another class of clustering ensemble algorithms is based on probabilistic mixture models [28,31]. Topchy et al. [28] model the clustering ensemble as a finite mixture of multinomial distributions in the space of base clusterings. A consensus result is found as a solution to the corresponding maximum likelihood problem using the EM algorithm. Wang et al. [31] proposed Bayesian Cluster Ensembles (BCE), a model that applies a Bayesian approach to protect against the overfitting to which the maximum likelihood method is prone [28]. The BCE model is applicable to some important variants of the basic clustering ensemble problem, including clustering ensembles with missing values, as well as row-distributed or

column-distributed clustering ensembles. Our work extends the BCE model to a nonparametric version, keeping all the advantages thereof, while allowing the number of clusters to adapt to the data.

### 3 Dirichlet Process Mixture Model

The Dirichlet process (DP) [8] is an infinite-dimensional generalization of the Dirichlet distribution. Formally, let  $S$  be a set,  $G_0$  a measure on  $S$ , and  $\alpha_0$  a positive real number. The random probability distribution  $G$  on  $S$  is distributed according to DP with the concentration parameter  $\alpha_0$  and the base measure  $G_0$ , if for any finite partition  $\{B_k\}_{1 \leq k \leq K}$  of  $S$ :

$$(G(B_1), G(B_2), \dots, G(B_K)) \sim \text{Dir}(\alpha_0 G_0(B_1), \alpha_0 G_0(B_2), \dots, \alpha_0 G_0(B_K))$$

Let  $G$  be a sample drawn from a DP. Then with probability 1,  $G$  is a discrete distribution [8]. In addition, if the first  $N - 1$  draws from  $G$  yield  $K$  distinct values  $\theta_{1:K}^*$  with multiplicities  $n_{1:K}$ , then the probability of the  $N^{\text{th}}$  draw conditioned on the previous  $N - 1$  draws is given by the Pólya urn scheme [5]:

$$\theta_N = \begin{cases} \theta_k^*, & \text{with prob } \frac{n_k}{N-1+\alpha_0}, k \in \{1, \dots, K\} \\ \theta_{K+1}^* \sim G_0, & \text{with prob } \frac{\alpha_0}{N-1+\alpha_0} \end{cases}$$

The DP is often used as a nonparametric prior in Bayesian mixture models [2]. Assume the data are generated from the following generative procedure:

$$\begin{aligned} G &\sim \text{Dir}(\alpha_0, G_0) \\ \theta_{1:N} &\sim G \\ x_{1:N} &\sim \prod_{n=1}^N F(\cdot | \theta_n) \end{aligned}$$

The  $\theta_{1:N}$  typically contains duplicates; thus, some data points are generated from the same mixture component. It is natural to define a cluster as those observations generated from a given mixture component. This model is known as the *Dirichlet process mixture* (DPM) model. Although any finite sample contains only finitely many clusters, there is no bound on the number of clusters and any new data point has non-zero probability of being drawn from a new cluster [22]. Therefore, DPM is known as an “infinite” mixture model.

The DP can be generated via the stick-breaking construction [24]. Stick-breaking draws two infinite sequences of independent random variables,  $v_k \sim \text{Beta}(1, \alpha_0)$  and  $\theta_k^* \sim G_0$  for  $k = \{1, 2, \dots\}$ . Let  $G$  be defined as:

$$\pi_k = v_k \prod_{j=1}^{k-1} (1 - v_j) \tag{1}$$

$$G(\theta) = \sum_{k=1}^{\infty} \pi_k \delta(\theta, \theta_k^*) \tag{2}$$

where  $\boldsymbol{\pi} = \langle \pi_k | k = 1, 2, \dots \rangle$  are the mixing proportions of the infinite number of components. Then  $G \sim Dir(\alpha_0, G_0)$ . It is helpful to use an indicator variable  $z_n$  to denote which mixture component is associated with  $x_n$ . The generative procedure for the DPM model using the stick-breaking construction becomes:

1. Draw  $v_k \sim Beta(1, \alpha_0)$ ,  $k = \{1, 2, \dots\}$  and calculate  $\boldsymbol{\pi}$  as in Eq (11).
2. Draw  $\theta_k^* \sim G_0$ ,  $k = \{1, 2, \dots\}$
3. For each data point:
  - Draw  $z_n \sim Discrete(\boldsymbol{\pi})$
  - Draw  $x_n \sim F(\cdot | \theta_{z_n}^*)$

In practice, the process is typically truncated at level  $K$  by setting  $v_{K-1} = 1$  [13]; Eq (11) then implies that all  $\pi_k$  for  $k > K$  are zero. The truncated process is called truncated stick-breaking (TSB). The resulting distribution, the truncated Dirichlet process (TDP), closely approximates the Dirichlet process when  $K$  is sufficiently large. The choice of the truncation level  $K$  is discussed in [13]. The joint probability over data items  $\mathbf{X} = \langle x_n | n \in \{1, \dots, N\} \rangle$ , component assignments  $\mathbf{Z} = \langle z_n | n \in \{1, \dots, N\} \rangle$ , stick-breaking weights  $\mathbf{v} = \langle v_k | k \in \{1, \dots, K\} \rangle$  and component parameters  $\boldsymbol{\theta}^* = \langle \theta_k^* | k \in \{1, \dots, K\} \rangle$  is:

$$p(\mathbf{X}, \mathbf{Z}, \mathbf{v}, \boldsymbol{\theta}^*) = \left[ \prod_{n=1}^N F(x_n | \theta_{z_n}^*) \pi_{z_n}(\mathbf{v}) \right] \left[ \prod_{k=1}^K G_0(\theta_k^*) Beta(v_k; 1, \alpha_0) \right]$$

Another approach to approximate the DP is to assume a finite but large  $K$ -dimensional symmetric Dirichlet prior (FSD) on the mixture proportion  $\boldsymbol{\pi}$  [14], which is  $\boldsymbol{\pi} \sim Dir(\alpha_0/K, \dots, \alpha_0/K)$ . This results in the joint distribution:

$$p(\mathbf{X}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\theta}^*) = \left[ \prod_{n=1}^N F(x_n | \theta_{z_n}^*) \pi_{z_n} \right] \left[ \prod_{k=1}^K G_0(\theta_k^*) \right] Dir(\boldsymbol{\pi}; \frac{\alpha_0}{K}, \dots, \frac{\alpha_0}{K})$$

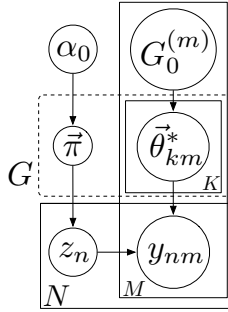
With TSB, the cluster weights differ in expected value, with lower-numbered cluster indices having higher probability. With FSD, the clusters are exchangeable. A detailed comparison of these DP approximations can be found in [19].

### 4 NBCE Generative Model

Following [28] and [31], we assume there are  $M$  base clustering algorithms, each generating a hard partition on the  $N$  data items to be clustered. Let  $J_m$  denote the number of clusters generated by the  $m^{th}$  clustering  $\varphi_m$ ,  $m \in \{1, \dots, M\}$ , and let  $y_{nm} \in \{1, \dots, J_m\}$  denote the cluster ID assigned to the  $n^{th}$  data item  $x_n$  by  $\varphi_m$ ,  $n \in \{1, \dots, N\}$ . The row  $\mathbf{y}_n = \langle y_{nm} | m \in \{1, \dots, M\} \rangle$  of the base clustering matrix  $\mathbf{Y}$  gives a new feature vector representation for the  $n^{th}$  data item.

Figure 1 depicts the generative model for  $\mathbf{Y}$ . We assume  $\mathbf{y}_n$  is generated from a truncated Dirichlet Process mixture model, where  $\alpha_0$  is the concentration parameter,  $G_0$  is the base measure, and  $K$  is the truncation level. The probability

of generating a cluster ID  $y_{nm} = j_m$  by  $\varphi_m$  for  $x_n$  is  $\theta_{nmj_m}$ ,  $j_m \in \{1, \dots, J_m\}$  and  $\sum_{j_m=1}^{J_m} \theta_{nmj_m} = 1$ . So  $\mathbf{y}_n = \langle y_{nm} = j_m | m \in \{1, \dots, M\} \rangle$  is generated with probability  $\prod_{m=1}^M \theta_{nmj_m}$ . We define  $\boldsymbol{\theta}_{nm} = \langle \theta_{nmj_m} | j_m \in \{1, \dots, J_m\} \rangle$ . We further assume a prior  $G_0^{(m)}$  for  $\boldsymbol{\theta}_{\cdot m} = \{\boldsymbol{\theta}_{nm} | n = 1, \dots, N\}$ , where  $G_0^{(m)}$  is a symmetric Dirichlet distribution of dimension  $J_m$  with hyperparameter  $\beta$ . The base measure  $G_0$  is defined as  $G_0 = G_0^{(1)} \times \dots \times G_0^{(M)}$ . We denote  $\boldsymbol{\theta}_n = \langle \boldsymbol{\theta}_{nm} | m \in \{1, \dots, M\} \rangle$ . Since the truncation level is  $K$ , there are  $K$  unique  $\boldsymbol{\theta}_n$ , denoted as  $\boldsymbol{\theta}_k^* = \langle \boldsymbol{\theta}_{km}^* | m \in \{1, \dots, M\} \rangle$ , where  $\boldsymbol{\theta}_{km}^* = \langle \theta_{kmj_m}^* | j_m \in \{1, \dots, J_m\} \rangle$ ,  $\sum_{j_m=1}^{J_m} \theta_{kmj_m}^* = 1$  and  $k \in \{1, \dots, K\}$ . We associate with each  $x_n$  an indicator variable  $z_n$  to indicate which  $\boldsymbol{\theta}_k^*$  is assigned to  $x_n$ ; if  $z_n = k$ , then  $\boldsymbol{\theta}_n = \boldsymbol{\theta}_k^*$ . A consensus cluster is defined as a set of data items associated with the same  $\boldsymbol{\theta}_k^*$ . That is,  $z_n$  indicates which consensus cluster  $x_n$  belongs to. There are at most  $K$  consensus clusters, but some consensus clusters may be empty; we define the total number of consensus clusters to be the number of distinct  $z_n$  in the sample.



**Fig. 1.** Nonparametric Bayesian Clustering Ensembles Model

The stick breaking generative process for  $\mathbf{Y}$  is:

1. Draw  $v_k \sim \text{Beta}(1, \alpha_0)$ , for  $k = \{1, \dots, K\}$  and calculate  $\boldsymbol{\pi}$  as in Eq (II)
2. Draw  $\boldsymbol{\theta}_k^* \sim G_0$ , for  $k = \{1, \dots, K\}$
3. For each  $x_n$ :
  - Draw  $z_n \sim \text{Discrete}(\boldsymbol{\pi})$
  - For each base clustering  $\varphi_m$ , draw  $y_{nm} \sim \text{Discrete}(\boldsymbol{\theta}_{z_n m}^*)$

Using the symmetric Dirichlet prior, step 1 becomes:

1. Draw  $\boldsymbol{\pi} \sim \text{Dir}(\frac{\alpha_0}{K}, \dots, \frac{\alpha_0}{K})$

## 5 Inference and Learning

This section considers three inference and learning methods: collapsed Gibbs sampling, standard variational Bayesian, and collapsed variational Bayesian inference. Table I gives the notation used throughout this section.

The joint probability of observed base clustering results  $\mathbf{Y} = \langle \mathbf{y}_n | n \in \{1, \dots, N\} \rangle$ , indicator variables  $\mathbf{Z} = \langle z_n | n \in \{1, \dots, N\} \rangle$ , component weights  $\boldsymbol{\pi} = \langle \pi_k | k \in \{1, \dots, K\} \rangle$ , and component parameters  $\boldsymbol{\theta}^* = \langle \theta_k^* | k \in \{1, \dots, K\} \rangle$  is given by:

$$\begin{aligned}
 p(\mathbf{Y}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\theta}^* | \alpha_0, G_0) &= \\
 &\left( \prod_{n=1}^N p(z_n | \boldsymbol{\pi}) p(\mathbf{y}_n | \boldsymbol{\theta}^*, z_n) \right) \cdot p(\boldsymbol{\pi} | \alpha_0) \left( \prod_{k=1}^K p(\boldsymbol{\theta}_k^* | G_0) \right) = \\
 &\left( \prod_{n=1}^N p(z_n | \boldsymbol{\pi}) \prod_{m=1}^M p(y_{nm} | \boldsymbol{\theta}_{z_n, m}^*) \right) \cdot p(\boldsymbol{\pi} | \alpha_0) \left( \prod_{k=1}^K \prod_{m=1}^M p(\boldsymbol{\theta}_{km}^* | G_0^{(m)}) \right) \quad (3)
 \end{aligned}$$

After marginalizing out the parameters  $\boldsymbol{\pi}$  and  $\boldsymbol{\theta}$ , the complete data likelihood is:

$$\begin{aligned}
 p(\mathbf{Y}, \mathbf{Z} | \alpha_0, G_0, ) &= p(\mathbf{Z} | \alpha_0) \quad (4) \\
 &\cdot \left( \prod_{m=1}^M \prod_{k=1}^K \frac{\Gamma(J_m \beta)}{\Gamma(J_m \beta + \mathcal{N}_{z.=k})} \prod_{j_m=1}^{J_m} \frac{\Gamma(\beta + \mathcal{N}_{z.=k}^{y_{.m}=j_m})}{\Gamma(\beta)} \right)
 \end{aligned}$$

where for the two DP approximations,  $p(\mathbf{Z} | \alpha_0)$  is different [19]:

$$\begin{aligned}
 p_{TSB}(\mathbf{Z} | \alpha_0) &= \prod_{k < K} \frac{\Gamma(1 + \mathcal{N}_{z.=k}) \Gamma(\alpha_0 + \mathcal{N}_{z.>k})}{\Gamma(1 + \alpha_0 + \mathcal{N}_{z. \geq k})} \\
 p_{FSD}(\mathbf{Z} | \alpha_0) &= \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_0 + N)} \prod_{k=1}^K \frac{\Gamma(\frac{\alpha_0}{K} + \mathcal{N}_{z.=k})}{\Gamma(\frac{\alpha_0}{K})}
 \end{aligned}$$

### 5.1 Collapsed Gibbs Sampling

Collapsed Gibbs sampling [21] speeds up the convergence of Gibbs sampling by marginalizing out the parameters  $\boldsymbol{\pi}$  and  $\boldsymbol{\theta}$ , sampling only the latent indicator variables  $\mathbf{Z}$  over the so-called collapsed space.

From Eq (4), we can derive the distribution for sampling components of  $\mathbf{Z}$ :

$$\begin{aligned}
 p(z_n = k | \mathbf{Z}_{-n}, \mathbf{Y}) &\propto \\
 p(z_n = k | \mathbf{Z}_{-n}) &\prod_{m=1}^M \left( \frac{\prod_{j_m=1}^{J_m} (\beta + \mathcal{N}_{z.=k, y_{.m}=j_m}^{-n})}{J_m \beta + \mathcal{N}_{z.=k}^{-n}} \right) \quad (5)
 \end{aligned}$$

where for the two different DP approximations,  $p(z_n = k | \mathbf{Z}_{-n})$  is different:

$$\begin{aligned}
 p_{TSB}(z_n = k | \mathbf{Z}_{-n}) &= \frac{1 + \mathcal{N}_{z.=k}^{-n}}{1 + \alpha_0 + \mathcal{N}_{z. \geq k}^{-n}} \prod_{h < k} \frac{\alpha_0 + \mathcal{N}_{z.>h}^{-n}}{1 + \alpha_0 + \mathcal{N}_{z. \geq h}^{-n}} \\
 p_{FSD}(z_n = k | \mathbf{Z}_{-n}) &= \frac{\frac{\alpha_0}{K} + \mathcal{N}_{z.=k}^{-n}}{\alpha_0 + N - 1}
 \end{aligned}$$

**Table 1.** Notation Description

Symbols	Description
$N$	the number of data
$x_n$	the $n^{th}$ data item
$M$	the number base clusterings
$\varphi_m$	the $m^{th}$ base clustering algorithm
$y_{nm}$	the cluster ID assigned to $x_n$ by $\varphi_m$
$K$	the number of consensus clusters (truncation level)
$J_m$	the number of clusters in the $m^{th}$ base clustering
$j_m$	the $j$ th cluster in the $m^{th}$ base clustering
$G_0^{(m)}$	the Dirichlet prior to $\{1, 2, \dots, J_m\}$ of $\varphi_m$
$\beta$	the hyperparameter of $G_0^{(m)}$
$G_0$	$\prod_{m=1}^M G_{0m}$
$z_n$	the indicator variable of $x_n$ to indicate which $\theta_k^*$ assigned to $x_n$
$\mathbf{Z}_{-n}$	the indicator variables except for $x_n$
$\theta_{nmj_m}$	the probability of $y_{nm} = j_m$
$\theta_{nm}$	$\langle \theta_{nmj_m}   j_m \in \{1, \dots, J_m\} \rangle$ and $\sum_{j_m=1}^{J_m} \theta_{nmj_m} = 1$
$\theta_n$	$\langle \theta_{nm}   m \in \{1, \dots, M\} \rangle$
$\theta_{kmj_m}^*$	the probability of $y_{nm} = j_m$ if $z_n = k$
$\theta_{km}^*$	$\langle \theta_{kmj_m}^*   j_m \in \{1, \dots, J_m\} \rangle$ and $\sum_{j_m=1}^{J_m} \theta_{kmj_m}^* = 1$
$\theta_k^*$	$\langle \theta_{km}^*   m \in \{1, \dots, M\} \rangle$ , unique parameter value of $\theta_n$
$\theta^*$	$\langle \theta_k^*   k \in \{1, \dots, K\} \rangle$
$\mathcal{N}_{z_n=k}$	$\sum_{n'=1}^N \delta(z_n, k)$
$\mathcal{N}_{z_n=k}^-$	$\sum_{n'=1, n' \neq n}^N \delta(z_{n'}, k)$
$\mathcal{N}_{z_n=y_m=j_m}$	$\sum_{n=1}^N \delta(z_n, k) \delta(y_{nm}, j_m)$
$\mathcal{N}_{z_n=y_m=j_m}^-$	$\sum_{n'=1, n' \neq n}^N \delta(z_{n'}, k) \delta(y_{n'm}, j_m)$
$\mathcal{N}_{z_n \geq k}$	$\sum_{n=1}^N \mathbf{1}_{\{z \geq k\}}(z_n)$
$\mathcal{N}_{z_n \geq k}^-$	$\sum_{n'=1, n' \neq n}^N \mathbf{1}_{\{z \geq k\}}(z_{n'})$

### 5.2 Standard Variational Bayesian Inference

Variational Bayesian inference [4] approximates the posterior distribution by adjusting free parameters of a tractable variational distribution to minimize the KL-divergence between the variational and true distributions. This is equivalent to maximizing a lower bound on the true log-likelihood.

We consider only the FSD prior as the DP approximation for standard variational Bayesian (VB) inference. VB assumes the following variational distributions:

$$\begin{aligned}
 q(\boldsymbol{\pi}, \boldsymbol{\theta}, \mathbf{Z} | \boldsymbol{\xi}, \boldsymbol{\rho}, \boldsymbol{\gamma}) &= q(\boldsymbol{\pi} | \boldsymbol{\xi}) \left( \prod_{k=1}^K p(\boldsymbol{\theta}_k^* | \boldsymbol{\rho}_k) \right) \left( \prod_{n=1}^N p(z_n | \gamma_n) \right) \\
 &= q(\boldsymbol{\pi} | \boldsymbol{\xi}) \left( \prod_{k=1}^K \prod_{m=1}^M p(\boldsymbol{\theta}_{km}^* | \boldsymbol{\rho}_{km}) \right) \left( \prod_{n=1}^N p(z_n | \gamma_n) \right) \quad (6)
 \end{aligned}$$

where  $\boldsymbol{\xi} = \langle \xi_k | k \in \{1, \dots, K\} \rangle$ ,  $\boldsymbol{\rho} = \langle \rho_k | k \in \{1, \dots, K\} \rangle = \langle \rho_{km} | k \in \{1, \dots, K\}, m \in \{1, \dots, M\} \rangle$  and  $\boldsymbol{\gamma} = \langle \gamma_n | n \in \{1, \dots, N\} \rangle$  are variational parameters, assumed to be independent. Further, given these variational parameters, the model parameters and indicator variables,  $\boldsymbol{\pi}$ ,  $\boldsymbol{\theta}$  and  $\mathbf{Z}$  are independent of each other [1]. In particular,  $\boldsymbol{\xi}$  specifies a  $K$ -dimensional Dirichlet distribution

<sup>1</sup> This is a strong assumption: note the dependences between  $\boldsymbol{\pi}$  and  $\mathbf{Z}$ ,  $\boldsymbol{\theta}$  and  $\mathbf{Z}$ ,  $\boldsymbol{\theta}$  and  $\boldsymbol{\pi}$  depicted in Figure 1.

for  $\boldsymbol{\pi}$ ,  $\rho_{km}$  specifies a  $J_m$ -dimensional Dirichlet distribution for  $\boldsymbol{\theta}_{km}^*$ , and  $\gamma_n$  specifies an  $N$ -dimensional multinomial distribution for the indicator  $z_n$  of  $x_n$ .

A lower bound  $\mathcal{L}_{VB}$  for the log-likelihood is given by:

$$\begin{aligned} \log p(\mathbf{Y}|\alpha_0, G_0) &\geq \tag{7} \\ E_q[\log p(\mathbf{Y}, \mathbf{Z}, \boldsymbol{\pi}, \boldsymbol{\theta}^*|\alpha_0, G_0)] - E_q[\log q(\boldsymbol{\pi}, \boldsymbol{\theta}^*, \mathbf{Z}|\boldsymbol{\xi}, \boldsymbol{\rho}, \boldsymbol{\gamma})] = \\ &\left( \sum_{n=1}^N \sum_{m=1}^M E_q[\log p(y_{nm}|\boldsymbol{\theta}_{z_n m}^*)] \right) - E_q[\log p(\boldsymbol{\pi}|\alpha_0)] + \left( \sum_{n=1}^N E_q[\log p(z_n|\boldsymbol{\pi})] \right) + \\ &\left( \sum_{k=1}^K E_q[\log p(\boldsymbol{\theta}_k^*|G_0)] \right) - E_q[\log q(\boldsymbol{\pi}|\boldsymbol{\xi})] - E_q[\log q(\boldsymbol{\theta}^*|\boldsymbol{\rho})] - E_q[\log q(\mathbf{Z}|\boldsymbol{\gamma})] \end{aligned}$$

See the Appendix for the expansion of Eq (7).

A local optimum is found by setting the partial derivatives of  $\mathcal{L}_{VB}$  with respect to each variational parameter to be zero. This gives rise to the following first-order conditions:

$$\begin{aligned} \gamma_{nk} &\propto \exp \left\{ \left( \sum_{m=1}^M \sum_{j_m=1}^{J_m} \delta(y_{nm}, j_m) \log \rho_{kmj_m} \right) + \Psi(\xi_k) - \Psi \left( \sum_{h=1}^K \xi_h \right) \right\} \\ \rho_{kmj_m} &= \beta + \sum_{n=1}^N \sum_{j_m=1}^{J_m} \gamma_{nk} \delta(y_{nm}, j_m) \\ \xi_k &= \frac{\alpha_0}{K} + \sum_{n=1}^N \gamma_{nk}. \end{aligned}$$

As for the remaining parameters  $\alpha_0$  and  $\beta$ , we first write the parts of  $\mathcal{L}_{VB}$  involving  $\alpha_0$  and  $\beta$  as:

$$\begin{aligned} \mathcal{L}_{VB}^{[\alpha_0]} &= \log \Gamma(\alpha_0) - K \log \Gamma\left(\frac{\alpha_0}{K}\right) + \left(\frac{\alpha_0}{K} - 1\right) \sum_{k=1}^K \left[ \Psi(\xi_k) - \Psi\left(\sum_{h=1}^K \xi_h\right) \right] \\ \mathcal{L}_{VB}^{[\beta]} &= \sum_{m=1}^M \left( K \log \Gamma(J_m \beta) - K J_m \log \Gamma(\beta) + \right. \\ &\quad \left. (\beta - 1) \sum_{k=1}^K \sum_{j_m=1}^{J_m} \left[ \Psi(\rho_{kmj_m}) - \Psi\left(\sum_{h=1}^{J_m} \rho_{kmh}\right) \right] \right) \end{aligned}$$

Estimates for  $\alpha_0$  and  $\beta$  are then obtained by maximization of  $\mathcal{L}_{VB}^{[\alpha_0]}$  and  $\mathcal{L}_{VB}^{[\beta]}$  using standard methods such as Newton-Raphson [6].

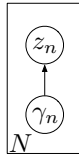
### 5.3 Collapsed Variational Bayesian Inference

Inspired by collapsed Gibbs sampling, collapsed variational Bayesian (CVB) inference for NBCE optimizes a lower bound  $\mathcal{L}_{CVB}$  for the log-likelihood in the collapsed space, in which the model parameters  $\boldsymbol{\theta}^*$  are marginalized out.

CVB assumes the following variational distribution:

$$q(\mathbf{Z}|\boldsymbol{\gamma}) = \prod_{n=1}^N q(z_n|\gamma_n) \tag{8}$$

where  $\boldsymbol{\gamma} = \langle \gamma_n | n \in \{1, \dots, N\} \rangle$  are variational parameters. Here,  $\gamma_n$  parameterizes an  $N$ -dimensional multinomial distribution for the indicator  $z_n$  of  $x_n$ . As shown in Figure 2, marginalizing out  $\boldsymbol{\theta}^*$  removes the need to specify variational parameters for  $\boldsymbol{\theta}^*$ . Thus, CVB searches for an optimum in a less restricted space than VB, which may lead to a better posterior approximation than VB.



**Fig. 2.** Graphical model representation of the collapsed variational distribution used to approximate the posterior in NBCE

The lower bound  $\mathcal{L}_{CVB}$  for the log-likelihood is:

$$\log p(\mathbf{Y}|\alpha_0, G_0, ) \geq E_{q(\mathbf{Z}|\boldsymbol{\gamma})}[\log p(\mathbf{Y}, \mathbf{Z}|\alpha_0, G_0, )] - E_{q(\mathbf{Z}|\boldsymbol{\gamma})}[\log q(\mathbf{Z}|\boldsymbol{\gamma})] \tag{9}$$

By taking the derivatives of  $\mathcal{L}_{CVB}$  with respect to  $q(z_n = k|\gamma_n)$ , we have:

$$q(z_n = k|\gamma_n) \propto \exp \left\{ E_{q(\mathbf{Z}_{-n}|\boldsymbol{\gamma})} \left[ \log p(z_n = k|\mathbf{Z}_{-n}) + \left( \sum_{m=1}^M \sum_{j_m=1}^{J_m} \log(\beta + \mathcal{N}_{z.=k, y.=j_m}^{-n}) \right) - \left( \sum_{m=1}^M \log(J_m \beta + \mathcal{N}_{z.=k}^{-n}) \right) \right] \right\} \tag{10}$$

where for the two DP approximations,  $\log p(z_n = k|\mathbf{Z}_{-n})$  is different:

$$\begin{aligned} \log p_{TSB}(z_n = k|\mathbf{Z}_{-n}) &= \log(1 + \mathcal{N}_{z.=k}^{-n}) - \log(1 + \alpha_0 + \mathcal{N}_{z. \geq k}^{-n}) + \\ &\quad \sum_{h < k} [\log(\alpha_0 + \mathcal{N}_{z. > h}^{-n}) - \log(1 + \alpha_0 + \mathcal{N}_{z. \geq h}^{-n})] \\ \log p_{FSD}(z_n = k|\mathbf{Z}_{-n}) &= \log\left(\frac{\alpha_0}{K} + \mathcal{N}_{z.=k}^{-n}\right) - \log(\alpha_0 + N - 1) \end{aligned}$$

Following [26], we apply the first-order latent-space variational Bayesian approximation to Eq (10). Applying the second-order latent-space variational Bayesian inference [27] will lead to a better approximation, but is more expensive. We plan to use it in our future work. Here we just illustrate how to calculate



$E_{q(\mathbf{Z}_{-n}|\gamma)}[\log(\beta + \mathcal{N}_{z.=k, y.=j_m}^{-n})]$  and  $E_{q(\mathbf{Z}_{-n}|\gamma)}[\log(J_m\beta + \mathcal{N}_{z.=k}^{-n})]$ . The calculation of other expectations is similar.

According to [26], we have:

$$\begin{aligned} E_{q(\mathbf{Z}_{-n}|\gamma)} \left[ \log(\beta + \mathcal{N}_{z.=k, y.=j_m}^{-n}) \right] &\approx \log \left( \beta + E_{q(\mathbf{Z}_{-n}|\gamma)} \left[ \mathcal{N}_{z.=k, y.=j_m}^{-n} \right] \right) \\ E_{q(\mathbf{Z}_{-n}|\gamma)} \left[ \log(J_m\beta + \mathcal{N}_{z.=k}^{-n}) \right] &\approx \log \left( J_m\beta + E_{q(\mathbf{Z}_{-n}|\gamma)} \left[ \mathcal{N}_{z.=k}^{-n} \right] \right) \end{aligned}$$

Denote  $\gamma_{nk} = q(z_n = k|\gamma_n)$ , then we get:

$$\begin{aligned} E_{q(\mathbf{Z}_{-n}|\gamma)} \left[ \mathcal{N}_{z.=k, y.=j_m}^{-n} \right] &= \sum_{n'=1, n' \neq n}^N \gamma_{n'k} \delta(y_{n'm} = j_m) \\ E_{q(\mathbf{Z}_{-n}|\gamma)} \left[ \mathcal{N}_{z.=k}^{-n} \right] &= \sum_{n'=1, n' \neq n}^N \gamma_{n'k} \end{aligned} \quad (11)$$

Calculating all the expectations and plugging them back into Eq (10) yields approximations to  $\gamma_{nk} = q(z_n = k|\gamma_n)$ . Repeating this process gives an EM-style iterative algorithm for estimating the  $\gamma_{nk}$ . The algorithm terminates when the change in  $\gamma_{nk}$  drops below a threshold.

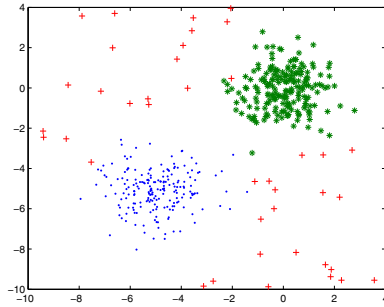
## 6 Empirical Evaluation

We compared several ensemble methods. We first used  $k$ -means with different initializations to obtain a set of base clusterings. Then we generated a consensus clustering using various clustering ensemble algorithms, including Bayesian clustering ensembles (BCE) [31], mixture model (MM) [28], CSPA, HGPA, and MCLA [25]. All of these are parametric methods. We also compared two different DP approximations, TSB and FSD, and the performance of NBCE estimated with collapsed Gibbs sampling, collapsed and standard variational approximation.

**Datasets.** We evaluated NBCE on both synthetic and real datasets. We generated a set of synthetic data with two clusters and some outliers to test the robustness of NBCE. The synthetic data are plotted in Figure 3. To generate the base clusterings on the synthetic data, following [31], we randomly added noise into the ground-truth labeling, e.g., we randomly modified the true labels of 5%, 10%, 15% and 20% of the data points. In each case, we generated 10 base noisy clusterings.

We also used five benchmark datasets from the UCI Machine Learning Repository<sup>2</sup>: *Glass*, *Ecoli*, *ImageSegmentation*, *ISOLET*, and *LetterRecognition*. *Glass* contains glass instances described by their chemical components. *Ecoli* contains

<sup>2</sup> <http://archive.ics.uci.edu/ml/>



**Fig. 3.** Synthetic Data: Two Clusters with Outliers

data on *E. Coli* bacteria. *ImageSegmentation* contains data from images that were hand-segmented classifying each pixel. *ISOLET* contains data representing spoken letters of the alphabet; we selected the letters A, B, C, D, E, and G. *LetterRecognition* contains character images corresponding to the capital letters in the English alphabet; we selected 700 samples of the letters A to J. We also used two time-series datasets from different application domains, namely *Tracedata* and *ControlChart*<sup>3</sup>. *Tracedata* simulates signals representing instrumentation failures. *ControlChart* contains synthetically generated control charts that are classified into one of the following: normal, cyclic, increasing trend, decreasing trend, upward shift, and downward shift.

To generate an ensemble on real data, we varied the number of output clusters of the base clustering algorithms. We computed clustering solutions obtained from multiple runs of  $k$ -means with different random initializations. The output clustering solutions were composed of a number of clusters equal to 50%, 75%, 100%, 150%, and 200% of the number of ideal classes of the specific dataset. We used 10 base clusterings for each dataset.

**Setting of Clustering Ensemble Methods.** For each parametric method and dataset, we set the number of output clusters equal to the actual number of classes, according to the ground truth. For the graph-partitioning-based methods (i.e., CSPA, HGPA, and MCLA), we set the METIS parameters as suggested in [15]. For NBCE, we set the truncation level  $K = 100$ . When comparing NBCE with other ensemble methods, we use Gibbs sampling for the inference of NBCE.

**Evaluation Criteria.** Since  $k$ -means, CSPA, HGPA, and MCLA are non-generative approaches, to compare the quality of their consensus partitions with NBCE, we evaluated their clustering accuracy using the  $F1$ -measure. The objective is to evaluate how well a consensus clustering fits the ground-truth partition. The  $F1$ -measure is defined as the harmonic average of precision and recall. Given a set  $D = \{x_1, \dots, x_n\}$  of  $n$  data objects, and  $A = \{A_1, \dots, A_h\}$  and

<sup>3</sup> For a description see: [http://www.cs.ucr.edu/~eamonn/time\\_series\\_data/](http://www.cs.ucr.edu/~eamonn/time_series_data/)

$B = \{B_1, \dots, B_k\}$  being two clustering solutions defined over  $D$ , the precision ( $P$ ) and recall ( $R$ ) are defined as:

$$P(A_i, B_j) = \frac{|A_i \cap B_j|}{|A_i|} \quad R(A_i, B_j) = \frac{|A_i \cap B_j|}{|B_j|}$$

$$P(A, B) = \frac{1}{h} \sum_{i=1}^k \max_{j \in \{1, \dots, k\}} P(A_i, B_j)$$

$$R(A, B) = \frac{1}{h} \sum_{i=1}^k \max_{j \in \{1, \dots, k\}} R(A_i, B_j)$$

The F1-measure is defined as:  $F1 = \frac{2P(A,B)R(A,B)}{P(A,B)+R(A,B)}$ .

Since MM, BCE and NBCE are generative models, we used perplexity to compare them. The perplexity of the observed base clusterings  $\mathbf{Y}$  is defined as [6]:

$$perp(\mathbf{Y}) = \exp\left(-\frac{\log p(\mathbf{Y})}{NM}\right) \tag{12}$$

Clearly, the perplexity monotonically decreases with the log-likelihood. Thus, a lower perplexity value on the training data means that the model fits the data better, and a lower value on the test data means that the model can better explain unseen data.

### 6.1 Results

**Evaluation of Clustering Ensemble Methods.** We held out 1/4 of the data to evaluate the predictive performance of MM, BCE and NBCE. Table 2 compares the clustering ensemble results for  $k$ -means, CSPA, HGPA, MCLA and NBCE in terms of the  $F1$ -measure on the real datasets excluding the hold-out set. We can see clearly that all ensemble methods outperform the baseline  $k$ -means algorithm, and NBCE gives the highest accuracy for each dataset. A paired t-test of NBCE against the next best accuracy is significant at the 0.002 level. Thus the comparison results of NBCE versus all competitors are statistically significant.

**Table 2.**  $F1$ -measure Results

	Base $k$ -means		CSPA	HGPA	MCLA	NBCE
	max	avg				
Glass	0.57	0.51	0.66	0.59	0.61	0.69
Ecoli	0.61	0.56	0.67	0.65	0.68	0.72
ImageSegmentation	0.52	0.42	0.53	0.44	0.59	0.65
ISOLET	0.53	0.41	0.59	0.50	0.65	0.66
LetterRecognition	0.48	0.40	0.49	0.50	0.53	0.62
Tracedata	0.49	0.44	0.51	0.62	0.61	0.66
ControlChart	0.62	0.56	0.73	0.70	0.67	0.77

**Table 3.** Perplexity Results on the Synthetic Dataset

	5%	10%	15%	20%
MM	10.04	13.27	17.36	21.20
BCE	7.92	9.76	14.22	18.98
NBCE	5.63	8.31	11.16	15.87

Table 4 compares MM, BCE and NBCE in terms of the perplexity on the synthetic datasets. It’s clear that NBCE fits the data better than BCE and MM. BCE and MM are parametric models, and thus fail to handle outliers. In contrast, NBCE is robust to outliers because it can find the number of clusters that fits the data best.

Tables 4 and 5 compare MM, BCE and NBCE in terms of the perplexity on training and test (i.e., hold-out) data for the real datasets. NBCE fits the data better than BCE, and BCE is better than MM.

**Table 4.** Perplexity Results on Training data for Real Datasets

	Glass	Ecoli	ImageSegmentation	ISOLET	LetterRecognition	Tracedata	ControlChart
MM	1.02	1.33	1.40	1.63	2.21	2.97	4.34
BCE	0.99	1.10	1.23	1.34	1.98	2.53	4.01
NBCE	0.77	0.92	1.03	1.24	1.76	2.38	3.63

**Table 5.** Perplexity Results on Test Data for Real Datasets

	Glass	Ecoli	ImageSegmentation	ISOLET	LetterRecognition	Tracedata	ControlChart
MM	1.15	1.51	1.49	1.72	2.51	3.22	5.56
BCE	1.07	1.39	1.37	1.60	2.33	2.94	4.88
NBCE	0.98	1.18	1.16	1.47	1.96	2.62	4.58

**Comparison of TSB and FSD.** In principle, TSB tends to produce larger clusters than FSD. The experimental results confirm this fact by showing that NBCE with a TSB prior gives a smaller number of singleton clusters than NBCE with FSD. Table 6 shows the percentage of outliers in singleton clusters for the five UCI datasets, when using collapsed Gibbs sampling with the two different priors.

**Comparison of CVB, VB and Gibbs.** Table 7 illustrates the perplexity of the three inference methods of NBCE on the UCI datasets excluding the hold-out set. Collapsed Gibbs sampling is asymptotically unbiased, so it gives lower perplexity than CVB and VB; CVB has less restricted assumption than VB, and CVB has lower perplexity than VB. The perplexity is calculated at convergence.

**Table 6.** Outlier Percentage

	TSB	FSD
Glass	3.2%	5.4%
Ecoli	4.3%	5.1%
ImageSegmentation	3.2%	3.5%
ISOLET	2.9%	3.1%
LetterRecognition	3.3%	3.6%

**Table 7.** Perplexity of Gibbs, CVB and VB

	Gibbs	CVB	VB
Glass	0.77	0.85	0.91
Ecoli	0.92	0.96	1.02
ImageSegmentation	1.03	1.06	1.11
ISOLET	1.24	1.28	1.30
LetterRecognition	1.76	1.80	1.88

## 7 Conclusion

A nonparametric Bayesian clustering ensemble model was proposed and three inference methods were considered: collapsed Gibbs sampling, variational Bayesian inference, and collapsed variational Bayesian inference. The versatility, and superior stability and accuracy of NBCE were demonstrated through empirical evaluation.

## Acknowledgements

This work is in part supported by NSF CAREER Award IIS-0447814.

## References

- Alexey, D.G., Tsymbal, A., Bolshakova, N., Cunningham, P.: Ensemble clustering in medical diagnostics. In: IEEE Symposium on Computer-Based Medical Systems, pp. 576–581. IEEE Computer Society, Los Alamitos (2004)
- Antoniak, C.E.: Mixtures of dirichlet processes with applications to bayesian non-parametric problems. *The Annals of Statistics* 2(6), 1152–1174 (1974)
- Ayad, H., Kamel, M.: Finding natural clusters using multi-clusterer combiner based on shared nearest neighbors. In: Windeatt, T., Roli, F. (eds.) MCS 2003. LNCS, vol. 2709, pp. 166–175. Springer, Heidelberg (2003)
- Beal, M.J.: Variational Algorithms for Approximate Bayesian Inference. PhD thesis, Gatsby Computational Neuroscience Unit, University College London (2003)
- Blackwell, D., Macqueen, J.B.: Ferguson distributions via pólya urn schemes. *The Annals of Statistics* 1, 353–355 (1973)
- Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *Journal of Machine Learning Research* 3(4-5), 993–1022 (2003)
- Dudoit, S., Fridlyand, J.: Bagging to improve the accuracy of a clustering procedure. *Bioinformatics* 19(9), 1090–1099 (2003)
- Ferguson, T.S.: A bayesian analysis of some nonparametric problems. *The Annals of Statistics* 1(2), 209–230 (1973)
- Fern, X.Z., Brodley, C.E.: Random projection for high-dimensional data clustering: A cluster ensemble approach. In: International Conference on Machine Learning, pp. 186–193 (2003)
- Fred, A.L.N., Jain, A.K.: Data clustering using evidence accumulation. In: International Conference on Pattern Recognition, Washington, DC, USA, vol. 4, pp. 276–280. IEEE Computer Society, Los Alamitos (2002)
- Gondek, D., Hofmann, T.: Non-redundant clustering with conditional ensembles. In: KDD '05: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pp. 70–77. ACM, New York (2005)

12. Hu, X.: Integration of cluster ensemble and text summarization for gene expression analysis. In: Fourth IEEE Symposium on Bioinformatics and Bioengineering, pp. 251–258 (2004)
13. Ishwaran, H., James, L.: Gibbs sampling methods for stick breaking priors. *Journal of the American Statistical Association* 96(453), 161–173 (2001)
14. Ishwaran, H., Zarepour, M.: Exact and approximate sum-representations for the dirichlet process. *The Canadian Journal of Statistics* 30(2), 269–283 (2002)
15. Karypis, G., Kumar, V.: A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing* 20(1), 359–392 (1998)
16. Kuncheva, L.I.: Experimental comparison of cluster ensemble methods. In: International Conference on Information Fusion, pp. 1–7 (2006)
17. Kuncheva, L.I., Hadjitodorov, S.T.: Using diversity in cluster ensembles. In: International Conference on Systems, Man and Cybernetics, vol. 2, pp. 1214–1219 (2004)
18. Kuncheva, L.I., Vetrov, D.: Evaluation of stability of k-means cluster ensembles with respect to random initialization. *PAMI* 28(11), 1798–1808 (2006)
19. Kurihara, K., Welling, M., Teh, Y.W.: Collapsed variational dirichlet process mixture models. In: IJCAI'07: Proceedings of the 20th international joint conference on Artificial intelligence, pp. 2796–2801. Morgan Kaufmann Publishers Inc., San Francisco (2007)
20. 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)
21. Neal, R.M.: Probabilistic inference using markov chain monte carlo methods. Technical Report CRG-TR-93-1, Dept. of Computer Science, University of Toronto (1993)
22. Neal, R.M.: Markov chain sampling methods for dirichlet process mixture models. *Journal of Computational and Graphical Statistics* 9(2), 249–265 (2000)
23. Punera, K., Ghosh, J.: Soft cluster ensembles. In: de Oliveira, J.V., Pedrycz, W. (eds.) *Advances in Fuzzy Clustering and its Applications*, ch. 4, pp. 69–90. John Wiley & Sons, Ltd., Chichester (2007)
24. Sethuraman, J.: A constructive definition of dirichlet priors. *Statistica Sinica* 4, 639–650 (1994)
25. Strehl, A., Ghosh, J.: Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research* 3, 583–617 (2003)
26. Sung, J., Ghahramani, Z., Bang, S.-Y.: Latent-space variational bayes. *IEEE Trans. Pattern Anal. Mach. Intell.* 30(12), 2236–2242 (2008)
27. Sung, J., Ghahramani, Z., Bang, S.-Y.: Second-order latent-space variational bayes for approximate bayesian inference. *IEEE Signal Processing Letters* 15, 918–921 (2008)
28. Topchy, A., Jain, A.K., Punch, W.: A mixture model for clustering ensembles. In: SIAM International Conference on Data Mining, pp. 379–390 (2004)
29. 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)
30. Topchy, A., Topchy, E., Jain, A.K., Punch, W.: Combining multiple weak clusterings. In: International Conference on Data Mining, pp. 331–338 (2003)
31. Wang, H., Shan, H., Banerjee, A.: Bayesian clustering ensembles. In: SIAM Data Mining (2009)

## Appendix

$\mathcal{L}_{VB}$ , Eq (7), has 7 terms. After the expansion,  $\mathcal{L}_{VB}$  can be rewritten as follows, where each line corresponds to a term of Eq (7):

$$\begin{aligned}
 \mathcal{L}_{VB} = & \left( \sum_{n=1}^N \sum_{m=1}^M \sum_{k=1}^K \sum_{j_m=1}^{J_m} \gamma_{nk} \delta(y_{nm}, j_m) \log \rho_{kmj_m} \right) + \\
 & \left( \log \Gamma(\alpha_0) - K \log \Gamma\left(\frac{\alpha_0}{K}\right) + \left(\frac{\alpha_0}{K} - 1\right) \sum_{k=1}^K \left[ \Psi(\xi_k) - \Psi\left(\sum_{h=1}^K \xi_h\right) \right] \right) + \\
 & \sum_{n=1}^N \sum_{k=1}^K \gamma_{nk} \left[ \Psi(\xi_k) - \Psi\left(\sum_{h=1}^K \xi_h\right) \right] + \\
 & \sum_{m=1}^M \left( K \log \Gamma(J_m \beta) - K J_m \log \Gamma(\beta) + (\beta - 1) \sum_{k=1}^K \sum_{j_m=1}^{J_m} \left[ \Psi(\rho_{kmj_m}) - \Psi\left(\sum_{h=1}^{J_m} \rho_{kmh}\right) \right] \right) - \\
 & \left( \log \Gamma\left(\sum_{k=1}^K \xi_k\right) - \sum_{k=1}^K \log \Gamma(\xi_k) + \sum_{k=1}^K (\xi_k - 1) \left[ \Psi(\xi_k) - \Psi\left(\sum_{h=1}^K \xi_h\right) \right] \right) - \\
 & \sum_{k=1}^K \sum_{m=1}^M \left( \log \Gamma\left(\sum_{j_m=1}^{J_m} \rho_{kmj_m}\right) - \sum_{j_m=1}^{J_m} \log \Gamma(\rho_{kmj_m}) + \right. \\
 & \qquad \qquad \qquad \left. \sum_{j_m=1}^{J_m} (\rho_{kmj_m} - 1) \left[ \Psi(\rho_{kmj_m}) - \Psi\left(\sum_{h=1}^{J_m} \rho_{kmh}\right) \right] \right) - \\
 & \sum_{n=1}^N \sum_{k=1}^K \gamma_{nk} \log \gamma_{nk}
 \end{aligned}$$

Here,  $\delta(\cdot, \cdot)$  is the Kronecker delta function;  $\Psi(\cdot)$  is the digamma function, the first derivative of the log Gamma function;  $\gamma_{nk} = q(z_n = k | \gamma_n)$ ;  $\rho_{kmj_m} = q(\theta_{kmj_m}^* | \rho_{km})$ ; and  $\gamma_{nk} = q(z_n = k | \gamma_n)$ .

# Directed Graph Learning via High-Order Co-linkage Analysis

Hua Wang, Chris Ding, and Heng Huang

Department of Computer Science and Engineering, University of Texas at Arlington,  
Arlington, TX 76019, USA

huawang2007@mavs.uta.edu, chqding@uta.edu, heng@uta.edu

**Abstract.** Many real world applications can be naturally formulated as a directed graph learning problem. How to extract the directed link structures of a graph and use labeled vertices are the key issues to infer labels of the remaining unlabeled vertices. However, directed graph learning is not well studied in data mining and machine learning areas. In this paper, we propose a novel Co-linkage Analysis (CA) method to process directed graphs in an undirected way with the directional information preserved. On the induced undirected graph, we use a Green's function approach to solve the semi-supervised learning problem. We present a new zero-mode free Laplacian which is invertible. This leads to an Improved Green's Function (IGF) method to solve the classification problem, which is also extended to deal with multi-label classification problems. Promising results in extensive experimental evaluations on real data sets have demonstrated the effectiveness of our approach.

## 1 Introduction

Different from undirected graphs, which only characterize symmetric pairwise similarity between data objects, directed graphs take into account edge directionality. This additional link structure usually brings useful information, though it makes learning on a directed graph more challenging. As a result, in contrast to a large number of classification methods devised for undirected graphs, classification on directed graphs has been much less studied [29]. In this work, we explore this area and solve the problem to classify unlabeled data on a directed graph by leveraging directed link structures when partially labeled data are given.

Directed graph appears extensively in diverse real world applications. Typical examples of classification on directed graphs include web page categorization [12] and spam host identification [1] on hyperlink networks, document classification or recommendation on citation graphs [10], and many practical problems in other domains such as computational biology [17, 15]. Besides these natural real world directed networks, asymmetric pairwise similarities between data objects also generate directed graphs, *e.g.*, the immediate outputs of widely used  $k$ -Nearest Neighbor ( $k$ -NN) graph construction method [11] and recently proposed sparse representation based graph construction methods [5, 25].



Because most existing graph-based semi-supervised classification algorithms only deal with undirected graphs, directed graphs are routinely converted to undirected ones via symmetrization in different ways prior to usage. For instance, when constructing a  $k$ -NN graph [11], an edge is placed between two data points  $\mathbf{x}_i$  and  $\mathbf{x}_j$  when one of them is among the  $k$  nearest neighbors of the other one. However, in reality,  $\mathbf{x}_i$  is not necessary to be among the  $k$  nearest neighbors of  $\mathbf{x}_j$ , when  $\mathbf{x}_j$  is among the  $k$  nearest neighbors of  $\mathbf{x}_i$ . Such symmetrization treatments [11,15,25] indeed simply discard the important structural information conveyed by edge directions, which inevitably impair the efficacy of subsequent classifications. For example, it is almost impossible to detect spam host without taking into consideration hyperlink direction — the main mechanism for web spam identification [1]: spam hosts frequently link to genuine hosts, while genuine hosts are rarely observed to link to spam ones. Therefore, there is a great need to develop directed graph based semi-supervised learning algorithms to make use of edge directionality of an input directed graph.

In this work, we focus on semi-supervised learning on a directed graph which classifies unlabeled vertices on a directed graph with partially labeled vertices. Our approach consists of two following steps.

Firstly, we provide an in-depth co-linkage analysis on co-citation and co-reference linkages at second, third and fourth orders. This leads to a novel Co-linkage Analysis (CA) similarity to process a directed graph in an undirected way with the directional information preserved. We also emphasize the importance of link normalization and refine CA similarity by symmetrically normalizing both in-links and out-links in a balanced manner. Once the symmetric pairwise similarity are obtained through this co-linkage analysis process, existing graph based semi-supervised learning methods can be employed.

Secondly, we further develop the Green’s function learning framework [8], and present an Improved Green’s Function (IGF) method to classify unlabeled data on the induced graph via CA similarity. Here we solve the problem caused by the zero-mode of the combinatorial Laplacian of an input graph. In addition, by incorporating label correlations through the kernel regularization framework derived from the theory of reproducing kernel Hilbert space (RKHS) [23], IGF method is extended to deal with multi-label data.

**Related works.** Due to the broad usage of directed graphs in numerous real applications, directed graph learning has attracted increasing attention in recent years. F. Chung [6] defined the combinatorial Laplacian of a directed graph, which laid foundation for label propagation on a directed graph. Zhou *et al.* [30] generalized their earlier work [28] for semi-supervised learning on undirected graphs to that on directed graphs by discriminatively normalizing in-links and out-links. They also proposed another method [29] upon the same intuition, in which the regularization on a directed graph has a similar form to the combinatorial Laplacian of a directed graph defined in [6]. Shin *et al.* considered learning on an artificial directed graph derived from an undirected graph through an interesting method — “graph sharpening” [18], which removes the direction from an unlabeled datum to a labeled one on all edges. Besides label propagation,

various other mechanisms have also been used to devise learning methods on a directed graph to take advantage of its asymmetric nature [17,15,31,127].

**Notations.** Pairwise similarities between data objects are usually described as an undirected graph  $\mathcal{G}^u$  with a *symmetric* weight matrix  $W \in \mathbb{R}^{n \times n}$ .  $D = \text{diag}(W\mathbf{e})$ , where  $\mathbf{e} = \{1, \dots, 1\}^T$ , and  $(D - W)$  is the graph Laplacian.

Suppose  $\mathcal{G}^d = (\mathcal{V}, \mathcal{E})$  is an unweighted directed graph with vertex set  $\mathcal{V}$  and edge set  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ .  $\mathcal{G}^d$  is described by an *asymmetric* adjacency matrix  $L = \{0, 1\}^{n \times n}$ , such that  $|\mathcal{V}| = n$ , and  $L_{ij} = 1$  if there exists an edge  $i \rightarrow j$  from vertex  $i$  to vertex  $j$ , and  $L_{ij} = 0$  otherwise. The edge  $i \rightarrow j$  is an ordered pair, and we say  $j$  is the *out-neighbor* of  $i$ , or  $i$  is the *in-neighbor* of  $j$ . The number of out-neighbors of  $i$  is the *out-degree* of  $i$ , given by  $d_{\text{out}}^i = \sum_k L_{ik}$ . Similarly, the number of in-neighbors of  $j$  is the *in-degree* of  $j$ , given by  $d_{\text{in}}^j = \sum_k L_{kj}$ . Let  $D_{\text{out}}$  be a diagonal matrix and  $D_{\text{out}}(i, i) = d_{\text{out}}^i$ , and  $D_{\text{in}}$  be a diagonal matrix and  $D_{\text{in}}(i, i) = d_{\text{in}}^i$ . When  $i \rightarrow i \in \mathcal{E}$ , the edge is called as a *loop*. A graph is *simple* if it has no loop. In this work, we only consider simple directed graphs, which are also strongly connected and aperiodic [2].

A weighted directed graph is described by a weight matrix  $R \in \mathbb{R}^{n \times n}$  when there exists a function  $r : \mathcal{E} \rightarrow \mathbb{R}^+$ , which associates a nonnegative value  $R_{i \rightarrow j}$  with every edge  $i \rightarrow j \in \mathcal{E}$ . Here we use  $R$  for directed graph to distinguish from  $W$  for undirected graph. An unweighted directed graph is a special case of weighted directed graphs when  $R = L$ . For a weighted directed graph, the out-degree is defined as  $d_{\text{out}}^i = \sum_k R_{ik}$ , and the in-degree is defined as  $d_{\text{in}}^i = \sum_k R_{ki}$ .

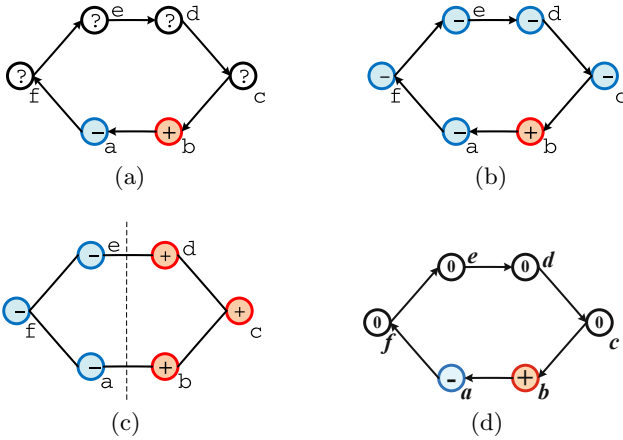
When it is clear from context, we use  $W$  and  $\mathcal{G}^u$  interchangeably, and the same for  $R$  (or  $L$ ) and  $\mathcal{G}^d$ .

## 2 Challenges of Semi-supervised Learning on A Directed Graph

The semi-supervised learning problem on a directed graph is as following. On a small subset of the vertices, the class labels are known. The task is to classify the rest vertices on the graph.

On an undirected graph, this problem is easy to understand. However, on a directed graph, this problem can be very intriguing. A semi-supervised learning problem on a simple unweighted directed graph is shown in Fig. 1(a). On this graph, the final class labels on the unlabeled vertices are not obvious. Fig. 1 illustrates three possible solutions.

**Using nearest neighbor classification.** If we use the nearest neighbor classification (NNC), the results are shown in Fig. 1(b). The NNC algorithm is the following iterative algorithm. It computes the label  $(y_1, \dots, y_n)$  on all unlabeled vertices with  $y_i$  fixed to their signs on all labeled vertices while  $y_j^{(t=0)} = 0$  for all unlabeled vertices. We iterate with  $y_j^{(t+1)} = \text{sign}\left(\sum_i L_{ij}y_i^{(t)}\right)$  until convergence. Vertex  $f$  will be labeled as “-” due to the the incoming neighbor  $a$ . Vertex  $e$  will be labeled as “-” due to the the incoming neighbor  $f$ . Repeating this, vertices  $d$  and  $c$  will be labeled as “-”.



**Fig. 1.** (a) Semi-supervised learning on a simple directed graph. Vertex  $a$  is positively labeled and vertex  $b$  is negatively labeled. The task is to classify the rest vertices. (b) Solution of the problem in (a) via nearest neighbor method. (c) Solution of the problem in (a) via symmetrization and label propagation method. (d) Solution of the problem in (a) via random walk method.

**Using symmetrization.** If we symmetrize the directed graph into an undirected graph by  $W = L + L^T$ , the results are shown in Fig. [1\(c\)](#). In this case, the problem becomes the semi-supervised learning on an undirected graph. It is now obvious that the final class labels are assigned as shown in Fig. [1\(c\)](#).

**Using random walk.** If we use information propagation via random walks, the results are shown in Fig. [1\(d\)](#), *i.e.*, class labels on the unlabeled vertices are undetermined. The reason is as following. A random walker starting from vertex  $a$  will carry negative class information. This walker will walk to vertex  $f$  with probability 1. It then will walk to vertex  $e$  with probability 1, *etc.* At time tends to infinity, this walker will reach all vertices with equal probability of  $1/6$ , passing on a negative label.

On the other hand, a random walker starting from vertex  $b$  will carry positive class information. It will visit each vertex with  $1/6$  probability as time tends to infinity, passing on a positive label. Thus on each unlabeled vertex, the probability of positive label is equal to the probability of negative label. Therefore, the final labeling is undetermined.

Note that the situation will be very different if the graph is **undirected** as shown in Fig. [1\(c\)](#). On the undirected graph, the random walker starting from vertex  $a$  (call it walker- $a$ ) will have a higher probability reaching  $f$  than reaching  $e$ , because after reaching  $f$ , instead of going to  $e$  (as required by the directed graph), it has the choice of **walking back** to  $a$ . Thus the farther-away from  $a$ , the smaller probability walker- $a$  will reach. The same holds for the random walker starting from vertex  $b$  (call it walker- $b$ ). Therefore, the probability for walker- $a$  reaching  $f$  is higher than the probability for walker- $b$  reaching  $f$ , leading to a “-” label for  $f$ .

**Challenges of learning on a directed graph** The above discussions show that semi-supervised learning on a directed graph is rather intriguing. Different approaches lead to very different results (while on an undirected graph, different approaches lead to the same results). Our analysis also shows that simple symmetrization of the adjacency matrix (link matrix  $L$ ), *i.e.*,  $W = L + L^T$ , loses critical information and results in very different outcomes.

We point out without elaboration that unsupervised learning such as clustering on a directed graph also has very similar intriguing problems. In general, research on directed graphs learning is lacking.

In this paper, we attempt to solve this learning problem by building a symmetric pairwise similarity from a directed graph. Once this symmetric similarity is constructed, the problem becomes learning on an undirected graph, and we may solve the problem using any existing algorithm for undirected graphs.

### 3 Co-linkage Analysis of A Directed Graph

In this section, we propose a novel Co-linkage Analysis (CA) method to process a directed graph in an undirected way. We first study the two fundamental co-linkages: co-citation and co-reference [9,7], and extend them to higher orders. Then we emphasize the importance of edge weight normalization. In our previous work [24], we use only second-order processes to describe a directed graph. In this work, we induce a symmetric similarity from a directed graph using both second-order co-linkages and their high-order extensions.

#### 3.1 Pairwise Similarity via Co-linkage Analysis

**Second-order co-citation and co-reference processes.** On a directed graph, we consider the following two second-order fundamental processes: *co-citation* [19] as shown in Fig. 2(a) and *co-reference* [13] as shown in Fig. 2(b).

If two vertices  $i$  and  $j$  are co-cited by many other vertices, such as vertex  $k$  in Fig. 2(a),  $i$  and  $j$  are likely to be related in some sense. Thus co-citation is a similarity measure and defined as the number of vertices that co-cite  $i$  and  $j$ :

$$W_{ij}^{(c)} = \sum_k L_{ki}L_{kj} = (L^T L)_{ij} . \tag{1}$$

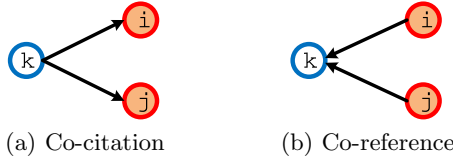
On the other hand, if two vertices  $i$  and  $j$  co-reference several other vertices, such as vertex  $k$  in Fig. 2(b),  $i$  and  $j$  are supposed to have certain commonality. Co-reference also measures similarity between vertices:

$$W_{ij}^{(r)} = \sum_k L_{ik}L_{jk} = (L L^T)_{ij} . \tag{2}$$

Combining  $W^{(c)}$  and  $W^{(r)}$ , we define the second-order similarity as:

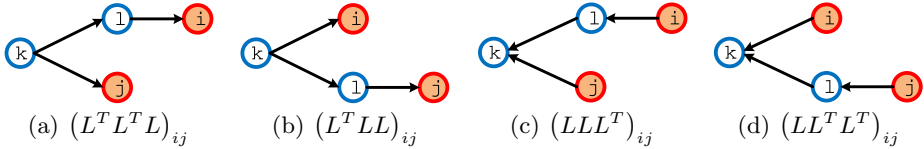
$$W^{(2nd)} = L^T L + L L^T , \tag{3}$$

where we assume co-citation and co-reference are equally important.



**Fig. 2.** Two fundamental second-order processes on a directed graph

**Third-order co-citation and co-reference processes.** Now we extend the co-citation and co-reference processes to the third-order. Specifically, for the co-citation between vertices  $i$  and  $j$  with respect to vertex  $k$  as in Fig. 2(a), an intermediate vertex can be inserted between  $k$  and  $i$  as in Fig. 3(a) or between  $k$  and  $j$  as in Fig. 3(b). We call them as *third-order co-citations*. Similarly, *third-order co-references* are defined as in Fig. 3(c) and Fig. 3(d). Same as the original second-order co-citation and co-reference, they also measure the similarities between vertices  $i$  and  $j$ .



**Fig. 3.** Third-order processes on a directed graph. (a)—(b): third-order co-citation; (c)—(d): third-order co-reference.

For the third-order co-citation in Fig. 3(a), the similarity between vertices  $i$  and  $j$  can be easily counted by  $\sum_k \sum_l L_{li} L_{kl} L_{kj} = (L^T L^T L)_{ij}$ . Following the same way for the rest three processes, the third-order similarity is defined as:

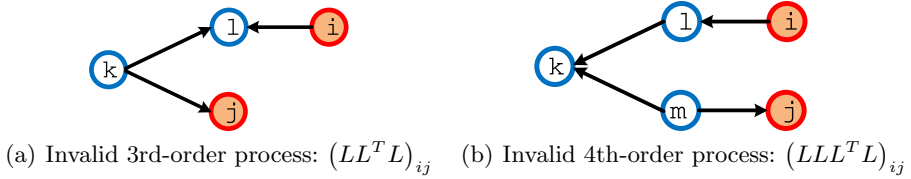
$$\begin{aligned}
 W^{(3rd)} &= L^T L^T L + L^T L L + L L L^T + L L^T L^T \\
 &= L (L + L^T) L^T + L^T (L + L^T) L,
 \end{aligned}
 \tag{4}$$

where we assume the four third-order processes in Fig. 3 are equally important.

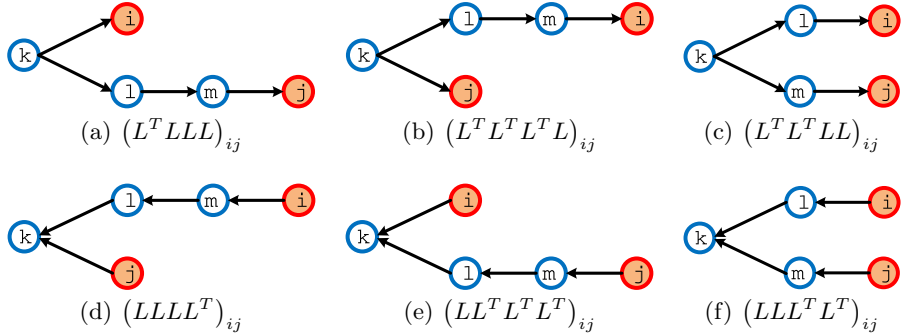
Note that, on a directed graph, other third-order processes also exist, such as the one shown in Fig. 4(a). However, because this process forms neither co-citation nor co-reference, it is not taken into account.

**Fourth-order co-citation and co-reference processes.** We further extend the co-citation and co-reference processes to the fourth-order, which are illustrated in Fig. 5. Again, we do not consider the processes not forming either co-citation or co-reference such as the one shown in Fig. 4(b). Thus, the fourth-order similarity is defined as:

$$\begin{aligned}
 W^{(4th)} &= L^T L L L + L^T L^T L^T L + L^T L^T L L + L L L L^T + L L^T L^T L^T + L L L^T L^T \\
 &= L (L L + L^T L^T + L L^T) L^T + L^T (L L + L^T L^T + L^T L) L .
 \end{aligned}
 \tag{5}$$



**Fig. 4.** Invalid third-order and fourth-order processes on a directed graph



**Fig. 5.** Fourth-order processes on a directed graph. (a)—(c): fourth-order co-citation; (d)—(e): fourth-order co-reference.

Combining  $W^{(2\text{nd})}$ ,  $W^{(3\text{rd})}$  and  $W^{(4\text{th})}$ , we obtain the proposed Co-linkage Analysis (CA) similarity as following:

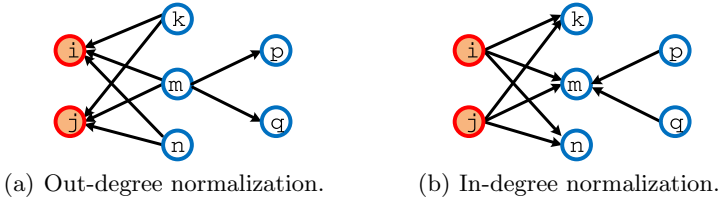
$$W = W^{(2\text{nd})} + \mu W^{(3\text{rd})} + \nu W^{(4\text{th})}, \quad (6)$$

where  $\mu$  and  $\nu$  are the parameters to balance the relative importance of the third-order and fourth-order similarities, which are empirically selected as  $\mu = \left( \sum_{i \neq j} W_{ij}^{(2\text{nd})} \right) / \left( \sum_{i \neq j} W_{ij}^{(3\text{rd})} \right)$  and  $\nu = \left( \sum_{i \neq j} W_{ij}^{(2\text{nd})} \right) / \left( \sum_{i \neq j} W_{ij}^{(4\text{th})} \right)$ .

### 3.2 Link Normalization

On the web, a vertex/web page with bigger out-degree has greater influence than another one with smaller out-degree. However, since these out-links can be arbitrarily added by the web page designer, and the importance of this web page can be arbitrarily increased.

In PageRank algorithm, every out-going hyperlinks from a vertex is inversely weighted by its out-degree, thereby every vertex has the same total out-going weight. This can be stated as *Internet Democracy*: every web site has a total of one vote. The hyperlink normalization and its importance are illustrated in Fig. 6(a). Basically, if a web page has a large out-degree, the significance/uniqueness of its co-citation is reduced. This points the necessity of out-degree normalization.



**Fig. 6.** Importance of link normalization. (a): vertices  $i$  and  $j$  are co-cited by vertices  $k$ ,  $m$  and  $n$ . However, since vertex  $m$  also cites vertices  $p$  and  $q$ , the co-citation of  $i$  and  $j$  by  $m$  is not as significant as that by either  $k$  or  $n$ . This fact can be compensated by normalizing the weights on the out-bound links of a vertex, *i.e.*, the co-citation of  $i$  and  $j$  by  $m$  is then  $2/4 = 50\%$  as important as that by either  $k$  or  $n$ . (b): vertices  $i$  and  $j$  co-reference vertices  $k$ ,  $m$  and  $n$ . However, since vertex  $m$  is also referenced by  $p$  and  $q$ , the co-reference of  $i$  and  $j$  by  $m$  is not as significant as that to either  $k$  or  $n$ . This fact can be similarly compensated by normalizing the in-bound links of a vertex.

Generally speaking, the in-degree of a document is not easily manipulated and is therefore a good indicator of the importance of the web page. But, when counting co-reference between two web pages as in Fig. 6(b) as similarity between the web pages, in-degree should also be normalized, because a web page  $i$  with large in-degree lose the specificity of the those web pages pointing to  $i$ .

With these discussions, the reasonable choices of link normalizations are:

$$L \rightarrow D_{\text{out}}^{-1}L, \tag{7}$$

$$L \rightarrow LD_{\text{in}}^{-1}, \tag{8}$$

$$L \rightarrow D_{\text{out}}^{-1/2}LD_{\text{in}}^{-1/2}. \tag{9}$$

Normalization of Eq. (7) uses the out-degree and is used in the PageRank algorithm [3,16], which is essentially the transition probability of a random walk. Normalization using out-degree is related to the concept of co-citation since co-citation uses out-links from those web pages/vertices pointing to them. Normalization using out-degree will balance the importance of each of these vertices.

Normalization of Eq. (8) uses the in-degree and can be viewed as the transition probability of a random walk on the inverse direction of the directed graph. Normalization using in-degree is related to the concept of co-reference since co-reference uses in-links from those web pages/vertices pointing to them. Normalization using in-degree will balance the importance of each of these vertices.

Normalization of Eq. (9) can be viewed as a compromise between the above two normalizations. This is also symmetric among the in-degree and out-degree. Considering the balance of in-degree and out-degree normalization and the balance among co-citation and co-reference, we adopt this symmetric normalization in our work.

Replacing  $L$  in Eq. (3), Eq. (4) and Eq. (5) by the symmetrically normalized  $D_{\text{out}}^{-1/2}LD_{\text{in}}^{-1/2}$  defined in Eq. (9), we can compute normalized CA through

Eq. (6), which is used in all our empirical evaluations. When a weighted directed graph is used,  $L$  is replaced by  $R$ .

## 4 Semi-supervised Learning via Improved Green's Function Method

With the symmetric CA similarity induced from a directed graph, we may use any existing graph-based semi-supervised learning algorithm for undirected graphs to classify the unlabeled data points. In this paper, we further develop the Green's function learning framework [8], and present a Improved Green's Function (IGF) method for classification. In this method, we solve the problem caused by the zero-mode of the combinatorial Laplacian of an input graph.

### 4.1 A Brief Review of the Green's Function Learning Framework

Suppose we have  $n = n_l + n_u$  data points  $\{\mathbf{x}_i\}_{i=1}^n$ , where the first  $n_l$  data points are labeled with  $\{\mathbf{y}_i\}_{i=1}^{n_l}$  for  $K$  target classes. Here,  $\mathbf{x}_i \in \mathbb{R}^p$  and  $\mathbf{y}_i \in \{-1, +1\}^K$ , such that  $\mathbf{y}_i(k) = +1$  if  $\mathbf{x}_i$  belongs to the  $k$ -th class, and  $-1$  otherwise. Our task is to learn the classification  $\{\mathbf{y}_i\}_{i=n_l+1}^n$  for the unlabeled data. For the unlabeled data points, we set  $\mathbf{y}_i(k) = 0$ . We write  $Y = [\mathbf{y}_1, \dots, \mathbf{y}_n]^T$ .

Given a graph with edge weight  $W$  among the data points  $\{\mathbf{x}_i\}_{i=1}^n$ , we wish to learn the mapping function  $F = \mathbb{R}^{n \times K}$  such that  $|F - Y|$  is minimized, where  $|\cdot|$  stands for the Frobenious norm of a matrix. Adding a penalty (regularization) term to ensure smoothness with respect to the underlying data manifold, the Green's function learning framework minimizes the following objective [8]:

$$J(F) = |F - Y| + \alpha F^T \mathcal{K}^{-1} F, \quad (10)$$

where  $\mathcal{K}$  is a kernel in RKHS, and  $\mathcal{K}^{-1} = (D - W)$ . Here  $\alpha$  is a parameter to balance the relative importance of the regularization term.

Taking the derivative of  $J$  with respect to  $F$  and set it as 0, we obtain  $F = [I + \alpha(D - W)]^{-1} Y$ . At large  $\alpha$  limit,  $F$  is computed as following:

$$F = GY = (D - W)^{-1} Y, \quad (11)$$

where  $G = (D - W)^{-1}$  is the Green's function of the input graph. However,  $G$  is not well defined due the existence of the zero-mode of  $(D - W)$ .

Let  $(D - W) \mathbf{v}_k = \lambda_k \mathbf{v}_k$ , where  $0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$  are the eigenvalues of  $(D - W)$  and  $\mathbf{v}_k$  are the corresponding eigenvectors. Because we consider connected graphs, the first eigenvector is a constant vector  $\mathbf{v}_1 = \mathbf{e}/\sqrt{n}$  with zero eigenvalue and multiplicity one. Thus,  $G$  is not well defined because  $\mathbf{v}_1 \mathbf{v}_1^T / \lambda_1 = \mathbf{e} \mathbf{e}^T / n \lambda_1$ . The analysis in [8] shows that this zero-mode of  $(D - W)$  is a consequence of the Von Neumann boundary condition (derivatives are continuous at the boundary) and thus the solution is undetermined up to an overall constant.



This overall constant is removed in [8] by explicitly discarding the zero-mode of  $(D - W)$  and the Green’s function is computed as follows:

$$G = \frac{1}{(D - W)_+} = \sum_{i=2}^n \frac{\mathbf{v}_i \mathbf{v}_i^T}{\lambda_i} . \tag{12}$$

### 4.2 Zero-Mode Free Laplacian

In this paper, we propose a zero-mode free Laplacian. The graph Laplacian is usually defined as the embedding of  $q_1, \dots, q_n$  by solving

$$\min_q \frac{1}{2} \sum_{ij} (q_i - q_j)^2 W_{ij}, \text{ s.t. } \sum_i q_i^2 = 1, \sum_i q_i = 0 . \tag{13}$$

Now, we propose to modify this to the following

$$\min_q \frac{1}{2} \sum_{ij} (q_i - q_j)^2 W_{ij} + \frac{W_{++}}{n^2} (\sum_i q_i)^2, \text{ s.t. } \sum_i q_i^2 = 1, \sum_i q_i = 0, \tag{14}$$

where  $W_{++} = \sum_{ij} W_{ij}$ . Clearly, the optimal solution for Eq. (14) is identical to that for Eq. (13). Note that

$$\frac{1}{2} \sum_{ij} (q_i - q_j)^2 W_{ij} + \frac{W_{++}}{n^2} (\sum_i q_i)^2 = \mathbf{q}^T L_+ \mathbf{q}, \tag{15}$$

where the **zero-mode free Laplacian**  $L_+$  is defined as

$$L_+ = D - W + \frac{W_{++}}{n^2} \mathbf{e}^T \mathbf{e} . \tag{16}$$

Some properties of  $L_+$  are:

- (1)  $\mathbf{v}_1 = \mathbf{e}/n^{1/2}$  is an eigenvector of  $L_+$  with eigenvalue  $\lambda_1(L_+) = W_{++}/n$ .
- (2)  $L_+$  and  $L = D - W$  have the same eigenvectors  $\mathbf{v}_2, \dots, \mathbf{v}_n$  with same eigenvalues.
- (3)  $L_+$  is **positive definite** and its inverse is well defined.

The new Green’s function becomes the following:

$$F = \frac{1}{D - W + \frac{W_{++}}{n^2} E} Y, \tag{17}$$

where  $E = \mathbf{e}^T \mathbf{e}$ . We call Eq. (17) as Improved Green’s Function (IGF) method.

### 4.3 Kernel Regularized Correlative Multi-label Classification

Multi-label data present a new opportunity to improve classification accuracy through label correlations, which is absent in single-label data. Typically, label correlations of a multi-label data set is captured by a correlation matrix  $C \in$

$\mathbb{R}^{K \times K}$ , which can be computed as in [23]. Adding a penalty for label correlations to impose smoothness, we minimize the following objective:

$$J(F) = \beta|F - Y|^2 + \text{tr} \left( F^T \mathcal{K}^{-1} F - \gamma \mathcal{K}^{-\frac{1}{2}} F C F^T \mathcal{K}^{-\frac{1}{2}} \right), \quad (18)$$

where  $\mathcal{K} = G = \left( D - W + \frac{W_{++}}{n^2} E \right)^{-1}$ ,  $\beta$  and  $\gamma$  are two small nonnegative constants to balance the two regularization terms.

When  $0 < \gamma < \min \{1, 1/\max(\zeta_k)\}$  where  $\zeta_k (0 < k < K)$  are the eigenvalues of  $C$ , following the same derivation as in [23], the solution to the optimization problem in Eq. (18) when  $\beta$  is small is obtained as:

$$F = GY (I - \gamma C)^{-1}. \quad (19)$$

We call Eq. (19) as Multi-Label Improved Green’s Function (ML-IGF) method, which solves multi-label classification problems.

## 5 Experiments

We evaluate the effectiveness of the proposed CA similarity, and the classification performances of IGF method on single-label data and ML-IGF method on multi-label data through classification tasks on directed graphs.

**Single-label data sets.** Because web data naturally generate directed graphs, we use the **WebKB** data set<sup>1</sup> for single-label classification. We consider a subset of the WebKB data set containing the pages from four universities, Cornell, Texas, Washington and Wisconsin, from which we remove the isolated pages, *i.e.*, those have no incoming and outgoing links, resulting in 858, 825, 1195 and 1238 pages respectively, for a total of 4116. These pages have been manually classified into the following seven categories: “student”, “faculty”, “staff”, “department”, “course”, “project” and “other”. We treat the extracted directed graphs as unweighted directed graphs and conduct classification on them.

**Multi-label data sets.** The following multi-label data sets are used to evaluate multi-label classification performance.

**MSRC**<sup>2</sup> has 591 images annotated by 22 classes. We divide each image into 64 blocks by a  $8 \times 8$  grid and compute the first and second moments (mean and variance) of each color band to obtain a 384-dimensional vector as features.

**Mediamill** [20] includes 43907 sub-shots with 101 classes, where each image is characterized by a 120-dimensional vector. Eliminating the classes containing less than 1000 samples, we have 27 classes. We randomly select 2609 sub-shots such that each class has at least 100 labeled data points.

**Music emotion** [21] comprises 593 songs with 6 emotions (labels). The dimensionality of the data points is 72.

<sup>1</sup> <http://www-2.cs.cmu.edu/~webkb/>

<sup>2</sup> <http://research.microsoft.com/en-us/projects/objectclassrecognition/default.htm>

**Yahoo** data described in [22] came from the “yahoo.com” domain. We use the “science” topic as it has maximum number of labels, which contains 6345 web pages with 22 labels.

Because these data sets are supplied in format of feature vectors, we construct directed graphs using  $k$ -NN graph construction method. Different from [11], we place a directed edge  $i \rightarrow j$  if vertex  $\mathbf{x}_j$  is a  $k$ -Nearest Neighbor of vertex  $\mathbf{x}_i$ . In our evaluations, we set  $k = 3$  ( $k = 1$  and  $k = 5$  lead to similar experimental results, which are not shown due to space limit).

### 5.1 Effectiveness of Co-linkage Analysis

We first evaluate the effectiveness of the proposed CA similarity defined in Eq. (6) in processing a directed graph in an undirected way.

A special benefit to use a separate graph construction step lies in that, existing graph-based semi-supervised learning methods can also benefit from the additional information contained in edge directions of a directed graph. Therefore we evaluate the effectiveness of the induced undirected graph by the proposed CA when it is used in the following three representative graph-based semi-supervised learning methods: (1) Gaussian fields and harmonic functions (GFHF) [32] method, (2) local and global consistency (LGC) [28] method, and (3) our previous work, *i.e.*, the Green’s function (GF) [8] method. Because these classification methods only work on undirected graphs, given a directed graph  $L$ , a simple symmetrization broadly used in existing works is as following:  $W_{ij} = 1$  if  $L(i \rightarrow j) = 1$  or  $L(j \rightarrow i) = 1$ . This graph is denoted as “Symmetrized graph” in Table 1, and compared against the undirected graph induced by the proposed CA which is denoted as “CA graph”.

We use the WebKB data set for evaluation. For each category of web pages from each university, a binary classification is conducted, *e.g.*, we classify “student” web pages *vs.* non-student web pages from Cornell university, denoted as “Cornell (student)”. Ignoring the “other” category, we perform  $4 \times 6 = 24$  binary classifications by every compared classification method. Because web pages within a same university are well-linked, and cross links between different universities are rare, we can imagine that a small number of training samples are sufficient to exactly classify web pages based on only link information. Therefore, in each binary classification, we randomly draw 4 pages as training examples, under the constraint that there is at least one labeled instance for each class. For each binary classification, we repeat 50 independent trials and the average test errors are reported in Table 1.

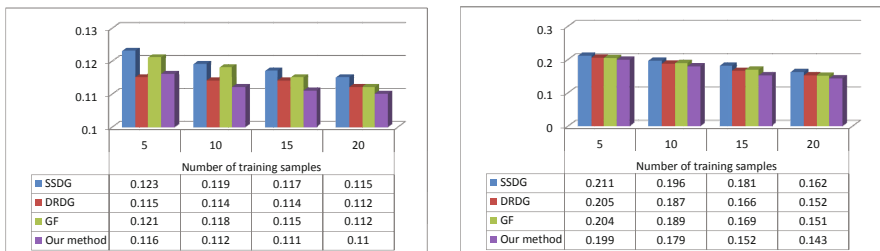
From Table 1 we can see that, the classification performances measured by “test error” on CA graphs always outperform those on symmetrized graphs. Due to space limit, we cannot list all classification results, and pick up one binary classification from each university as in Table 1, which are similar to those not shown. Therefore, we conclude that the proposed CA method is more effective to characterize a directed graph than the simple symmetrization methods that do not consider edge directions.

**Table 1.** Improved classification performance (test error) of three existing representative graph-based semi-supervised classification methods by using CA graph

	Cornell (student)			Wisconsin (student)		
	GFHF	LGC	GF	GFHF	LGC	GF
Symmetrized graph	0.246	0.238	0.225	0.207	0.205	0.196
CA graph	0.223	0.212	0.173	0.195	0.191	0.183
	Washington (course)			Texas (faculty)		
	GFHF	LGC	GF	GFHF	LGC	GF
Symmetrized graph	0.142	0.140	0.136	0.228	0.227	0.218
CA graph	0.137	0.135	0.121	0.221	0.215	0.204

## 5.2 Single-Label Classification Using IGF Method

We evaluate single-label classification performance of IGF method by conducting 2-class classification to distinguish “course” *vs.* non-course web pages in Washington University and “faculty” *vs.* non-faculty web pages in Texas University in WebKB data set. We compare the classification results of our method against two state-of-the-art classification algorithms on directed graphs: (1) Semi-Supervised learning on Directed Graph (SSDG) [30] method, and (2) Distribution Regularized classification on Directed Graph (DRDG) [29] method. We also report the results by the Green’s Function (GF) [8] method, where a simple symmetrization of  $W = (L + L^T) / 2$  is used to form the undirected graph. The classification performance comparison measured by average test error over 50 independent trials are listed in Fig. 7, which demonstrate the superiority of our method and thereby confirm its usefulness.

**Fig. 7.** Test errors to classify “course” *vs.* non-course web pages in Washington University and “faculty” *vs.* non-faculty web pages in Texas University in WebKB data set by four compared methods

**Table 2.** Performance evaluations of the compared methods by 5-fold cross validations

Data sets	Evaluation metrics		Compared methods					
			SSDG	DRDG	MLSI	SMSE	ML-IGF-S	ML-IGF
MSRC	Macro	Precision	0.215	0.224	0.252	0.248	0.281	<b>0.311</b>
		average	F1 score	0.223	0.238	0.287	0.279	0.288
	Micro	Precision	0.201	0.223	0.253	0.247	0.279	<b>0.317</b>
		average	F1 score	0.267	0.278	0.301	0.298	0.324
MediaMill	Macro	Precision	0.201	0.203	0.207	0.210	0.252	<b>0.274</b>
		average	F1 score	0.289	0.292	0.301	0.312	0.352
	Micro	Precision	0.203	0.206	0.207	0.215	0.259	<b>0.282</b>
		average	F1 score	0.332	0.334	0.341	0.347	0.368
Music emotion	Macro	Precision	0.313	0.317	0.329	0.331	0.392	<b>0.404</b>
		average	F1 score	0.305	0.308	0.323	0.331	0.399
	Micro	Precision	0.308	0.311	0.328	0.332	0.395	<b>0.412</b>
		average	F1 score	0.310	0.314	0.339	0.354	0.401
Yahoo (Science)	Macro	Precision	0.367	0.372	0.396	0.398	0.421	<b>0.443</b>
		average	F1 score	0.278	0.282	0.296	0.305	0.361
	Micro	Precision	0.369	0.375	0.395	0.402	0.448	<b>0.470</b>
		average	F1 score	0.202	0.203	0.209	0.215	0.236

### 5.3 Multi-label Classification Using Multi-label IGF Method

We use standard 5-fold cross validation to evaluate multi-label classification performance of ML-IGF method. We empirically selected  $\gamma = \min \{0.1, 1/\max(\zeta_k)\}$ . We compare our method with (1) SSDG method and (2) DRDG method as in Section 5.2, which, however, are designed for single label classifications. Therefore, for every class, we conduct a binary classification. We also compare our method to two recent multi-label classification methods: (3) Multi-label informed Latent Semantic Indexing (MLSI) [26] method, and (4) Semi-supervised learning by Sylvester Equation (SMSE) [4] method. The classification by these two methods are directly conducted on original data. Because, to our best knowledge, ML-IGF method presented in this work is the first one to exploit the information conveyed by both link directionality and label correlations, we cannot find a counterpart method for comparison.

We also evaluate the effectiveness of link normalization discussed in Section 3.2, and conduct classification using ML-IGF method on the induced graph when no normalization is used. We denote these results as ML-IGF-S in Table 2.

The widely used classification performance metrics in statistical learning, *precision* and *F1 score*, are used to evaluate the compared methods. Precision and F1 score are computed for every class following the standard definitions for a binary classification problem. To address multi-label classification, macro average and micro average are used to assess the overall performance across multiple labels [14].

Table 2 presents the classification performance comparisons by 5-fold cross validation, which show that ML-IGF method generally outperforms all other methods, sometimes significantly. These results quantitatively demonstrate the effectiveness of our method, and justify the utility of the CA similarity and label correlations. Besides, the classification performances of ML-IGF is always better than those of ML-IGF-S method, which provide a concrete evidence that link normalization is an indispensable part of the proposed CA similarity.

## 6 Conclusions

This paper explored the usage of directed graphs to solve semi-supervised learning problems. We proposed a novel Co-linkage Analysis (CA) method to transform a directed graph to an undirected one, which is built upon the co-linkage processes on directed graphs. With the induced symmetric CA similarity, a Improved Green's Function (IGF) method was presented to solve the classification problem, which is also generalized to deal with multi-label classification problems. Extensive experimental evaluations on real data sets have demonstrated that the performance of the proposed approach outperforms other related previous methods in literature.

**Acknowledgments.** This research is supported by NSF-CCF 0830780, NS-FCCF 0939187, NSF-CCF 0917274, NSF-DMS 0915228, NSF-CNS 0923494.

## References

1. Abernethy, J., Chapelle, O., Castillo, C.: Web spam identification through content and hyperlinks. In: Proc. of International Workshop on Adversarial Information Retrieval on the Web (2008)
2. Bang-Jensen, J.: Digraphs: theory, algorithms and applications. Springer, Heidelberg (2008)
3. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. In: WWW (1998)
4. Chen, G., Song, Y., Wang, F., Zhang, C.: Semi-supervised Multi-label Learning by Solving a Sylvester Equation. In: SDM (2008)
5. Cheng, H., Liu, Z., Yang, J.: Sparsity Induced Similarity Measure for Label Propagation. In: IEEE ICCV (2009)
6. Chung, F.: Laplacians and the Cheeger inequality for directed graphs. *Annals of Combinatorics* 9(1), 1–19 (2005)
7. Ding, C., He, X., Husbands, P., Zha, H., Simon, H.: PageRank, HITS and a unified framework for link analysis. In: ACM SIGIR (2002)
8. Ding, C., Simon, H., Jin, R., Li, T.: A learning framework using Green's function and kernel regularization with application to recommender system. In: ACM SIGKDD (2007)
9. Ding, C., Zha, H., He, X., Husbands, P., Simon, H.: Link analysis: hubs and authorities on the World Wide Web. *SIAM Review* 256 (2004)
10. Giles, C., Bollacker, K., Lawrence, S.: CiteSeer: An automatic citation indexing system. In: Proc. of ACM Conf. on Digital libraries (1998)

11. Hein, M., Maier, M.: Manifold denoising. In: NIPS (2007)
12. Joachims, T., Cristianini, N., Shawe-Taylor, J.: Composite kernels for hypertext categorisation. In: ICML (2001)
13. Kessler, M.: Bibliographic coupling between scientific papers. *American documentation* 14(1), 10–25 (1963)
14. Lewis, D., Yang, Y., Rose, T., Li, F.: Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research* (2004)
15. Meila, M., Pentney, W.: Clustering by weighted cuts in directed graphs. In: SDM (2007)
16. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Stanford Digital Library Technologies Project (1998)
17. Pentney, W., Meila, M.: Spectral clustering of biological sequence data. In: AAAI (2005)
18. Shin, H., Hill, N., Ratsch, G.: Graph based semi-supervised learning with sharper edges. In: ECML (2006)
19. Small, H.: Co-citation in the scientific literature: A new measure of the relationship between two documents. *J. Am. Soc. for Info. Sci. Tech.* 24(4), 265–269 (1973)
20. Snoek, C.G.M., Worring, M., van Gemert, J.C., Geusebroek, J.M., Smeulders, A.W.M.: The challenge problem for automated detection of 101 semantic concepts in multimedia. In: ACM Multimedia (2006)
21. Trohidis, K., Tsoumakas, G., Kalliris, G., Vlahavas, I.: Multilabel classification of music into emotions. In: ISMIR
22. Ueda, N., Saito, K.: Single-shot detection of multiple categories of text using parametric mixture models. In: ACM SIGKDD (2002)
23. Wang, H., Huang, H., Ding, C.: Image Annotation Using Multi-label Correlated Greens Function. In: IEEE ICCV (2009)
24. Wang, H., Huang, H., Ding, C.: Image Categorization Using Directed Graphs. In: ECCV (2010)
25. Yan, S., Wang, H.: Semi-supervised learning by sparse representation. In: SDM (2009)
26. Yu, K., Yu, S., Tresp, V.: Multi-label informed latent semantic indexing. In: ACM SIGIR (2005)
27. Zhang, D., Mao, R.: Classifying networked entities with modularity kernels. In: ACM CIKM (2008)
28. Zhou, D., Bousquet, O., Lal, T., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: NIPS (2004)
29. Zhou, D., Huang, J., Schölkopf, B.: Learning from labeled and unlabeled data on a directed graph. In: ICML (2005)
30. Zhou, D., Schölkopf, B., Hofmann, T.: Semi-supervised learning on directed graphs. In: NIPS (2005)
31. Zhu, S., Yu, K., Chi, Y., Gong, Y.: Combining content and link for classification using matrix factorization. In: ACM SIGIR (2007)
32. Zhu, X., Ghahramani, Z., Lafferty, J.: Semi-supervised learning using Gaussian fields and harmonic functions. In: ICML (2003)

# Incorporating Domain Models into Bayesian Optimization for RL

Aaron Wilson, Alan Fern, and Prasad Tadepalli

Oregon State University School of EECS

**Abstract.** In many Reinforcement Learning (RL) domains there is a high cost for generating experience in order to evaluate an agent's performance. An appealing approach to reducing the number of expensive evaluations is Bayesian Optimization (BO), which is a framework for global optimization of noisy and costly to evaluate functions. Prior work in a number of RL domains has demonstrated the effectiveness of BO for optimizing parametric policies. However, those approaches completely ignore the state-transition sequence of policy executions and only consider the total reward achieved. In this paper, we study how to more effectively incorporate all of the information observed during policy executions into the BO framework. In particular, our approach uses the observed data to learn approximate transitions models that allow for Monte-Carlo predictions of policy returns. The models are then incorporated into the BO framework as a type of prior on policy returns, which can better inform the BO process. The resulting algorithm provides a new approach for leveraging learned models in RL even when there is no planner available for exploiting those models. We demonstrate the effectiveness of our algorithm in four benchmark domains, which have dynamics of variable complexity. Results indicate that our algorithm effectively combines model based predictions to improve the data efficiency of model free BO methods, and is robust to modeling errors when parts of the domain cannot be modeled successfully.

## 1 Introduction

The advantages of direct policy search algorithms for solving the RL problem are well-understood. In contrast to model-based methods, policy search approaches dispense with the need to represent and employ a model for learning. Such models can be difficult to construct requiring significant engineering and domain specific knowledge. Instead, learning is based on Monte-Carlo samples of the expected return gathered directly from the environment of interest. These returned samples are used to improve the policy directly, removing the intermediate step of model learning. Unfortunately, by dispensing with learning a model, the policy search methods are far less data-efficient than the model-based alternatives. Many more samples are typically necessary before direct policy search algorithms find good policies.

Policy search algorithms based on Bayesian Optimization (BO) have been proposed as a method to improve the data efficiency of direct algorithms [1-3]. These methods improve data efficiency in two ways. First, they explicitly model the surface of the expected return. Samples of the expected return, generated by interaction with the real



environment, are not discarded when estimating the value of new policies. Second, they model the uncertainty in the return and use it to select which new policy should be executed, balancing the need for exploration with the benefits of exploitation. Results for direct RL methods based on these approaches indicate that such methods can substantially reduce the number of real world evaluations needed to find good solutions [1].

We propose to combine domain models with the BO framework to further improve data efficiency. Additionally, we do not require fully accurate domain models. Our approach is based on employing a (learned) *approximate* simulator to reduce the amount of real world experience needed to find good solutions. Crucially, we consider the setting where the simulator *is not* a replacement for the true domain in the sense that the domain model cannot be accurately learned no matter the amount of data available. We allow our simulator to have substantial errors in some regions of the state space which would prevent the direct application of standard model-based algorithms. Despite these significant errors the models may be partially accurate providing useful information about the performance of some policies. Taking advantage of this information is crucial to the success of our algorithm.

By extending the work on BO we place our efforts squarely within the growing literature on Bayesian RL. Most closely related is the extensive work by [2] which we extend by incorporating approximate domain models. Other related work includes [4] which models the value function as a Gaussian Process. This work, focused primarily on the problem of estimation, did not consider parameterized policies, and did not explore the use of domain models for improving data efficiency. Numerous model-based Bayesian approaches, including [5-7], take advantage of uncertainty in domain models to tackle the exploration-exploitation trade off. Like our algorithm these methods actively explore to reduce uncertainty. However, each of these methods requires the models to be accurate to insure eventual convergence. We do not have this stringent requirement.

We test our proposed algorithm on four benchmark RL environments including Cartpole, Mountain Car, 3-Link Planar Arm, and Acrobot tasks. We compare our algorithm to the standard Bayesian Optimization framework, to Least-Squares Policy Iteration [8], to Q-Learning with CMAC function approximation [9], to Dyna-Q with CMAC function approximation [9], and to OLPOMDP a policy gradient based algorithm for RL [10]. The empirical results show that the proposed algorithm outperforms all of these methods across our four benchmark tasks.

## 2 Bayesian Optimization

The general problem of maximizing a real valued function,

$$\theta^* = \max_{\theta} f(\theta), \quad (1)$$

has been studied at length in the literature. A subset of these approaches are considered global optimization algorithms guaranteed, given enough time, to find the true maximum of  $f$ . In some cases generating responses from  $f$  may have large costs. This occurs in many real world RL domains particularly in cases where the agent is physically embodied in a real robot. In these cases acquiring information from the objective

function may cost more than time (physical damage to the robot may occur). BO is a global method for tackling expensive objective functions by explicitly reducing the number of evaluations needed before the maximum is found.

As part of the BO approach to global optimization one must specify a prior distribution encoding uncertainty about the unknown objective function  $P(f(\theta))$ . In general the objective function is not inherently stochastic. Many objective functions are expectations of stochastic outcomes. However, uncertainty about the true form of the objective still justifies a Bayesian prior distribution. The goal is to combine this prior distribution with data to compute a posterior useful for deciding what new data to acquire.

When new data is accumulated, beliefs about the function space are updated. Given data in the form of tuples  $D = \{\langle \theta_i, f(\theta_i) \rangle\}_{i=1}^n$  the posterior distribution of the function space is computed:

$$P(f|D_{1:n}) \propto P(D_{1:n}|f)P(f). \quad (2)$$

The posterior distribution is typically called a response surface or surrogate function. It is a simplification of reality. Evaluation of the surrogate function has several advantages over evaluations of the true objective. The surrogate is less costly to compute than the true target function. It can be simulated internally by the agent preventing expensive computational and physical costs. This is a standard strategy in many learning problems where the true function may be approximated by e.g. regression trees, neural networks, polynomials, and other structures that match properties of the target function in some way. The surrogate function can be viewed as a function approximator from a specific class of functions which support Bayesian methods of analysis.

The key to BO techniques sits with using the surrogate function to select new points to evaluate. Ideally the selection should trade off improving the accuracy of the surrogate function (exploring the objective function) with taking advantage of points maximizing the mean of the surrogate (exploiting the information available so far). If the method of selecting new points has this property the system will select points to reduce its uncertainty about  $f$  until it is certain the true maximum  $f(\theta^*)$  has been discovered. The idea is to substitute a large number of surrogate function evaluations to construct an exploration strategy minimizing the number of costly objective function evaluations.

To proceed we can frame the Bayesian optimization problem as minimizing the following function,

$$\min_{\theta} \int \| f(\theta) - \max_{\theta'} f(\theta') \| dP(f) \quad (3)$$

specifying a search for a point  $\theta$  minimizing the expected difference in value between  $f(\theta)$  and the true maximum  $\max_{\theta'} f(\theta')$ . This is a problem of minimizing the expected risk (maximizing the value of  $f(\theta)$  minimizes this quantity). Equation 3 leads straightforwardly to an iterative process for selecting new points to evaluate,

$$\theta_{n+1} = \operatorname{argmin}_{\theta} \int \| f(\theta) - \max_{\theta'} f(\theta') \| dP(f|D_{1:n}), \quad (4)$$

where the expectation is computed in terms of the posterior distribution given the data accumulated so far. This conceptually simple procedure myopically selects the next new point,  $\theta_{n+1}$ , to evaluate (computing a non-myopic sample is hard). However, computing

the minimum point in this fashion may require substantial computational resources, making approximation necessary.

To keep the spirit of the optimization described above approximations can be made in terms of the maximal point found so far. Let the tuple  $\langle \theta_i, f^{max} \rangle$  be the data point with highest reported utility. Using this maximal value one can construct an *improvement* function,

$$I(\theta) = \max\{0, f(\theta) - f^{max}\}, \quad (5)$$

which is positive at all points where  $f(\theta)$  exceeds the current maximum and zero at all other points. New experiments are selected according to,

$$\theta_{n+1} = \operatorname{argmax}_{\theta} E [I(\theta) | D_{1:n}], \quad (6)$$

a Maximum Expected Improvement (MEI) criterion. Due to the uncertainty in  $f(\theta)$  the improvement function is a random variable. Crucially, the expected improvement function,  $E [I(\theta) | D_{1:n}]$ , takes into account uncertainty in the improvement at unseen points. When sufficient probability mass exists over values exceeding the current maximum the MEI for the unseen point will be positive. This nicely incorporates the posterior uncertainty into the optimization, encouraging exploration of the input space. This improvement function has persisted as the preferred method of selecting new points in Bayesian optimization because it can be computed efficiently, and leads to a good trade off between exploitation and exploration [11]. The experiments we report make use of this MEI criterion.

It is not satisfying to employ any optimization strategy without knowing of its effectiveness. In particular, a number of approximations have been introduced and it is of interest whether the proposed strategy of selecting points according to the MEI will find good solutions and whether it will converge to the global optimum. Recent work [12] gives positive convergence results when the prior function is a Gaussian Process (GP) with fixed mean and covariance [13]. These results hold under fairly general conditions which apply to the BO algorithm on which our work is based. However, please note that these results do not hold when either the mean or covariance function are changed as new data is acquired. This will be an important fact when we introduce our modifications below.

**Prior distribution.** Selecting an appropriate prior distribution for the function space is a non-trivial task. However, for this work we make use of the GP. A convenient property of the GP is that for a finite set of observations the distribution over  $f$  is represented entirely in terms of the data as a multi-variate normal distribution. The GP for function space  $f$ ,

$$f \sim GP(m(\theta), k(\theta, \theta)), \quad (7)$$

is represented by a mean function  $m(\theta)$  and kernel function  $k(\theta, \theta)$ . The mean function encodes base knowledge of the underlying function (frequently initialized to zero). The kernel function encodes relationships between inputs. Substantial engineering effort is often made to select appropriate mean functions and kernels because the impact on the performance of GP regression is strongly impacted by these selections.

For purposes of computing the improvement functions described above the posterior distribution at new points must be computed,  $P(f(\theta_{n+1}) | D_{1:n})$ . In the GP

model this posterior has a simple form. Given the data  $D_{1:n}$  let  $y$  be the vector of outputs,  $y = [f(\theta_1), \dots, f(\theta_n)]$  and let  $\theta = [\theta_1, \dots, \theta_n]$  be the matrix of observations. The posterior distribution is Gaussian with mean and variance,

$$\mu(\theta_{n+1}|D_{1:n}) = k(\theta_{n+1}, \theta)K(\theta, \theta)^{-1}(y - m(\theta)), \tag{8}$$

$$\sigma^2(\theta_{n+1}|D_{1:n}) = k(\theta_{n+1}, \theta_{n+1}) - k(\theta_{n+1}, \theta)K(\theta, \theta)^{-1}k(\theta, \theta_{n+1}). \tag{9}$$

The  $m(\theta)$  is a vector of mean function evaluations made at each data point,  $k(\theta_{n+1}, \theta)$  is the vector of similarities between the new point and all previously observed data, and the vector  $k(\theta, \theta_{n+1})$  is its transpose. The variable  $K(\theta, \theta)$  is the matrix of similarities between all observed points.

### 3 Bayesian Optimization for Reinforcement Learning

#### 3.1 Reinforcement Learning

We study the Reinforcement Learning (RL) problem in the context of Markov Decision Processes (MDP). An MDP is described by a tuple  $(S, A, P, P_0, R, \pi)$ . Each state  $s \in S$  encode all information about the world necessary to make a decision. An agent can execute any action  $a \in A$  from the set of all possible actions. The transition function  $P$  encodes a probability distribution over next states  $P(s_t|s_{t-1}, a_{t-1})$  given the current state and the action selected by the agent. The initial state distribution  $P_0$  is a distribution over starting states  $P_0(s)$ . The reward function  $R(s, a)$  returns a numeric value representing the immediate reward for the state action pair. Finally, the function  $\pi$  is a stochastic mapping from states to actions  $P_\pi(a|s, \theta)$ . It is a function of a vector of parameters  $\theta \in \mathbb{R}^n$ .

We express the expected return for a policy parameterized by  $\theta$  as,

$$\eta(\theta) = \int_{\xi} R(\xi)P(\xi; \theta)d\xi \tag{10}$$

where the variable  $\xi = [s_{1..n}, a_{1..n}]$  represents a trajectory of length  $n$  through the environment,  $R(\xi)$  denotes the reward along the trajectory,  $R(\xi) = \sum_{t=1}^n R(s_t, a_t)$ , and the conditional distribution  $P(\xi|\theta)$  is the probability density over trajectories given the policy parameters  $\theta$ ,  $P(\xi|\theta) = P_0(s_0) \prod_{t=1}^T P(s_t|s_{t-1}, a_{t-1})P_\pi(a_{t-1}|s_{t-1}, \theta)$ . The goal of learning in this setting is to find a set of policy parameters  $\theta^*$  maximizing Equation (10). For the purposes of our experiments we assume a fixed horizon MDP where every trajectory has a finite length.

If  $\eta$  was available in a closed form then the search for the optimal policy could, at least in principle, proceed directly. In this case the learning problem reduces to a problem of optimization for which a variety of algorithms are available. Unfortunately,  $\eta$  is hard to compute, and we are uncertain about the relationships between policies and returns.

---

**Algorithm 1.** Bayesian Optimization Algorithm (BOA)

---

- 1: Let  $D = \{\theta_i, \eta(\theta_i)\}_{i=1}^n$ .
  - 2: Select the next point in the policy space to evaluate:  $\theta_{n+1} = \operatorname{argmax}_{\theta} E(I(\theta)|D_{1:n})$ .
  - 3: Execute the policy with parameters  $\theta_{n+1}$  in the MDP.
  - 4: Update  $D_{1:n} = D_{1:n} \cup (\theta_{n+1}, \eta(\theta_{n+1}))$
  - 5: Return to step 2.
- 

### 3.2 Bayesian Optimization for Reinforcement Learning

The subject of our uncertainty is the expected return. It is a costly objective function for which we are uncertain of the location of its maximum. Therefore, to apply BO in this context we model the expected return using a GP and then proceed by using the sequential selection strategy discussed above. To make this strategy concrete please observe the BOA algorithm, Algorithm 1.

BOA is a sequential planning algorithm that selects a new policy to evaluate, based on the evaluations of all previous policies. BOA accumulates data,  $D_{1:n}$ , uses this data to estimate the posterior distribution for  $\eta$ , and then samples new policy parameters by maximizing the Expected Improvement. Importantly, by using BOA the problem of exploration in RL is addressed.

This simple algorithm, originally proposed by Mockus in the 70's [11], has already seen success in the RL literature. In [1] results are reported for a gait optimization problem on an AIBO robot. In this case the high cost of running the real physical robots motivates using BOA. They report a significant improvement in the time needed to train the robots, 2 hours using their BO approach, by comparison to the state of the art methods at that time, requiring 9 hours to achieve similar results. Additional RL results are reported in [3] for a car driving task in the TORCS simulator. The goal is to optimize the policy to guide a simulated car along a fixed trajectory. Good policies are found for the domain after 130 trials using BOA. Such examples illustrate the power of BOA for policy selection. Using the algorithm can lead to a significant reduction in the number of real trials needed to find good solutions.

However, when the transition and reward function of the domain can be approximated, BOA can have substandard performance by comparison to a model-based approach. As a result, we seek a principled integration of model-based ideas from RL with BOA. We formulate this integration below.

### 3.3 RL Domain Models for Bayesian Optimization

To improve BOA our goal is to make better use of the information present in the trajectories the agent generates while exploring its domain. The performance of BO algorithms based on GP priors depends on the selection of the mean function and the kernel function. Because of the well-understood impact on GP prediction, much work has been focused on the optimization of the kernel hyper parameters. However, instead of optimizing the kernel function we seek to improve the mean function using new information included in  $D$ . We augment the data vector  $D_{1:n}$  to include the trajectories experienced when acting in the domain  $D_{1:n} = \{\theta_i, \eta(\theta_i), \xi_i\}_{i=1}^n$ . This additional

information will be used in our model-based version of BOA. A carefully selected mean function has the advantage of improving the predictions for points far away from the current data; regardless of the quality of the kernel function an accurate mean function can lead to reliable predictions at unseen points.

To begin we assume a transition function,  $P(s_t|s_{t-1}, a_{t-1}, \phi)$ , a reward function,  $R(s_{t-1}, a_{t-1}, s_t)$ , and a function  $LearnModel(D_{1:n})$  which takes the data as input and returns updated parameters for the transition and reward functions.

Our mean function takes as input the augmented data  $D_{1:n}$  and a set of policy parameters  $\theta$  and computes a vector of expectations  $m(D_{1:n}, \theta)$ . To compute element  $m(D_{1:n}, \theta_i)$  we first call  $LearnModel(D_{1:n})$  with the current data, and then use the estimated models to compute a Monte-Carlo estimate of the expected return for policy  $\theta_i$ . To compute the Monte-Carlo estimate: 1. Sample a collection of initial states. 2. Simulate trajectories from the sampled initial states to terminal states using the approximate models. 3. Use the learned reward function to score the set of trajectories. 4. Compute the average value of the scored sample trajectories. The mean for each point  $\theta_i$  has the concise form,

$$m(D_{1:n}, \theta_i) = \hat{\eta}(\theta_i) = \frac{1}{N} \sum_{j=1}^N \hat{R}(\xi_j), \quad (11)$$

which is the average over  $N$  trajectories  $\xi_j$  generated using the approximate models. The function  $\hat{R}$  represents the application of the learned reward function to score the sampled trajectory.

To make use of this Monte-Carlo estimate we replace the zero mean function of the GP prior distribution with  $m(D_{1:n}, \theta)$ . The predictive distribution of the GP changes to be Gaussian with mean,

$$\mu(\theta_{n+1}|D_{1:n}) = m(D_{1:n}, \theta_{n+1}) + k(\theta_{n+1}, \theta)K(\theta, \theta)^{-1}(\eta(\theta) - m(D_{1:n})), \quad (12)$$

where  $m(D_{1:n}, \theta)$  is the vector of Monte-Carlo estimates for each policy in  $D$ . This vector must be recomputed whenever new trajectories are added to the data. The new predictive mean is a sum of the model-based estimate and the GPs prediction of the residual. Ultimately this mean function will incur additional computational cost during optimization of the expected improvement function, because for each point  $\theta_{n+1}$  considered during the optimization a Monte-Carlo estimate must be computed. The variance remains,

$$\sigma^2(\theta_{n+1}|D_{1:n}) = k(\theta_{n+1}, \theta_{n+1}) - k(\theta_{n+1}, \theta)K(\theta, \theta)^{-1}k(\theta, \theta_{n+1}). \quad (13)$$

The role of the GP has changed from directly modeling the surface of the expected return to modeling the disagreement between the Monte-Carlo estimates and the observed returns. Errors in the transition and reward functions will be compounded when generating long trajectories. Modeling the residuals using the GP corrects for these compounded errors in a principled way.

In the case where the single step transition models cannot be effectively approximated the model-based estimates of the expected return may badly skew the predictions.

For instance, if  $m(D_{1:n}, \theta)$  sufficiently underestimates the true mean across the space of policies then the expected improvement for unseen policies may be dangerously pessimistic. In this case, when the models are sufficiently wrong, the poor estimates can stifle search for additional points easily preventing the optimal solution being discovered. This is an atypical consideration for a model-based approach to RL. Typically it is assumed that given sufficient data the agents models will closely approximate the true transition and reward functions. For most model-based approaches, if this assumption is not met, optimizations using the model will lead to erratic results. We desire to have the performance of our model-based algorithm be no worse than the standard BO algorithm discussed above even in the case where the model is wrong. Already, the algorithm can correct for poor Monte-Carlo estimates so long as the residual function has a predictable errors. However, we have violated some of the requirements for convergence stated in [12] by allowing the mean function to change as new data is accumulated. Intuitively this would not be problematic if we were assuming convergence of the models. By allowing the models to have significant errors, errors that strongly impact the mean function, we do not enjoy the safety of the proofs. We need additional controls to prevent the model-based mean leading to degenerative performance when errors are large and difficult to predict.

To control the impact of the model-based estimates we propose to introduce a parameter  $\beta$  adjusting the impact of the model-based mean function on the predictive distribution. We model the true expected return by a sum,

$$\eta(\theta) = (1 - \beta)f(\theta) + \beta g(\theta), \quad (14)$$

of two unknown functions. We model the function  $f(\theta)$  with a zero mean GP, and we model function  $g(\theta)$  with a GP distribution that uses the model-based mean  $m(D_{1:n}, \theta)$ . The kernel of both GP priors is simply the squared exponential kernel discussed earlier. The resulting distribution for  $\eta(\theta)$  is a GP with mean  $\beta m(D, \theta)$  and covariance computed using the squared exponential kernel. This change impacts the predicted mean which now weights the model estimate by  $\beta$ ,  $\mu(\theta_{n+1}|D_{1:n}) = \beta m(D_{1:n}, \theta_{n+1}) + k(\theta_{n+1}, \theta)K(\theta, \theta)^{-1}(\eta(\theta) - \beta m(D_{1:n}, \theta))$ . The variance remains unchanged.

We cannot know how well the chosen models will approximate the true transition and reward functions a priori. However, as the system accumulates new data the disagreement between the vector of observed returns and the mean function can be computed. The magnitude of the residuals should impact the setting of  $\beta$ . To adjust the parameter  $\beta$  based on the agreement between the estimates and the observed data we maximize the log likelihood of the new GP prior. For a GP with mean  $\beta m$  and covariance matrix  $K$  the log likelihood of the data is,

$$P(\eta(\theta)|D_{1:n}) = -\frac{1}{2}(\eta(\theta) - \beta m)^t K^{-1}(\eta(\theta) - \beta m) - \frac{1}{2} \log |K + \sigma^2 I| - \frac{n}{2} \log(2\pi). \quad (15)$$

Taking the gradient of the log likelihood and setting the equation to zero we get a simple expression for  $\beta$ ,

$$\beta = \frac{\eta(\theta)^t K^{-1} m}{m^t K^{-1} m}. \quad (16)$$

By computing the value of  $\beta$  prior to maximizing the expected improvement our proposed algorithm can down weight the Monte-Carlo estimates when it is convinced of

**Algorithm 2.** model-based Bayesian Optimization Algorithm (MBOA)

- 
- 1: Let  $D_{1:n} = \{\theta_i, \eta(\theta_i), \xi\}_{i=1}^n$ .
  - 2: Run the  $LearnModel(D_{1:n})$  function.
  - 3: Compute the vector  $m(D_{1:n}, \theta)$  using the approximate models.
  - 4: Optimize  $\beta$  by maximizing the log likelihood.
  - 5: Select the next point in the policy space to evaluate:  $\theta_{n+1} = \operatorname{argmax}_{\theta} E(I(\theta)|D_{1:n})$ .
  - 6: Execute the policy with parameters  $\theta_{n+1}$  in the MDP.
  - 7: Update  $D_{1:n+1} = D_{1:n} \cup (\theta_{n+1}, \eta(\theta_{n+1}), \xi_{n+1})$
  - 8: Return to step 2.
- 

the models inaccuracy. Intuitively the algorithm can return to the performance of the unmodified BOA when errors in the transition function are large. In this case the algorithm should behave no worse than BOA and should always benefit from the model in those regions of the policy space where it performs well.

We show the additional steps added to BOA in Algorithm 2. MBOA has several nice properties: 1) When the model is accurate or is a reasonable approximation of the true model MBOA dramatically improves on the data efficiency of the BOA algorithm. 2) If the approximate model cannot capture the domain dynamics then the MBOA algorithm will quickly learn to ignore the model-based estimates resulting in performance comparable to BOA. MBOA does not assume that the approximated model represents the true domain. Furthermore, MBOA does not require a planning algorithm to take advantage of the models. This can be a significant detail in domains with continuous states and actions for which producing a planner can be a difficult problem. In the next section we examine the performance of the MBOA algorithm on four benchmark RL tasks.

## 4 Results

We examine the performance of MBOA in domains for which domain models can be learned successfully, and in cases where they cannot. To do so we examine the performance in four benchmark RL tasks including a mountain car task, a cart-pole balancing task, a 3-link planar arm task, and an acrobot swing up task. Additional details about the cart-pole, mountain car, and acrobot domains can be found in [9].

Accurate linear models can be learned for both the cart-pole task and for the planar arm task. The two other domains can be modeled with varying degrees of success. The mountain car task has some non-linear behavior in specific regions of the state space. The linear models we provide MBOA cannot model the data generated at these points. However, many of the policies do not visit these regions of the state space. The acrobot task cannot be modeled using a linear function. In fact, we found it difficult to find any good non-linear models of the acrobot transition function. Modeling the system with linear models leads to Monte-Carlo estimates of the expected return which differ substantially from the true values. In this case standard model-based algorithms which do not correct the model will fail to converge to a good solution. We show that MBOA does not suffer from this drawback.



Comparisons are made between MBOA, the model-based DYNA-Q algorithm, and three model-free approaches: 1. BOA. 2. OLPOMDP. and 3. Q-Learning with CMAC function approximation.

#### 4.1 Experiment Setup

We detail the special requirements necessary to implement each algorithm in this section.

**MBOA and BOA.** To use the GP prior in MBOA and BOA we must specify a kernel representing the relationship between policy parameters. The squared exponential kernel,

$$k(\theta_i, \theta_j) = \exp\left(-\frac{1}{2}\rho(\theta_i - \theta_j)^T(\theta_i - \theta_j)\right), \quad (17)$$

with a single scaling parameter  $\rho$ , was used in all experiments. Of course, more complex kernels can be selected and optimized to improve the results of this paper (i.e. introducing scale parameters for each dimension of the policy space.). However, we were able to get positive results with this simple kernel for both BO algorithms. The  $\rho$  parameter was tuned for each experiment, and the same value was used in both MBOA and BOA. The mean function of BOA is the zero function, and MBOA employs the model-based mean function discussed above.

Both MBOA and BOA require optimization of the expected improvement function after new data points are obtained. For purposes of optimizing Equation 6 we use a black box global optimization algorithm called DIRECT [14]. DIRECT does require upper and lower bounds on the input parameters passed to the objective function. We specify an upper bound of 1 and a lower bound of -1 for each dimension of the policy. These bounds hold for all experiments reported below.

MBOA also requires a model of the transition and reward dynamics. In all of the tasks discussed below we make use of a collection of linear models, one for each dimension of the state, and one for the reward function. The models are of the form,

$$s_i^t = \phi_i f(s^{t-1}, a^{t-1})', r^t = \phi_r f(s^{t-1}, a^{t-1}, s^t)', \quad (18)$$

where  $s_i^t$  is the  $i$ th state variable at time  $t$ ,  $w_i$  is the weight vector for the  $i$ th state variable, and  $f(s^{t-1}, a^{t-1}, s^t)'$  is the transpose of features computed from the states and actions (and next states in the case of the reward model). We define  $LearnModel(D)$  to be standard linear regression for all of the experiments reported below.

**DYNA-Q.** We make two slight modifications to the DYNA-Q algorithm. First, we provide the algorithm with the same linear models employed by MBOA. These are models of continuous transition functions which DYNA-Q is not normally suited to handle. The problem arises during the sampling of previously visited states during internal reasoning. To perform this sampling we simply maintain a dictionary of past observations and sample visited states from this dictionary. These continuous states are then mapped to a discrete value using a CMAC function approximator for the Q-function. The second change we make is to disallow internal reasoning until a trial is completed. Reasoning between steps does not happen. After each trial DYNA-Q is allowed 200000 internal samples to update its Q-function.

**Q-Learning with CMAC function approximation.** We use the standard algorithm with  $\epsilon$ -Greedy exploration. We optimize the number of discrete states used to approximate the Q-function and also optimize the parameters of the Q-Learning algorithm. We set the value of  $\epsilon$  to .1 and anneal it after each trial.

**OLPOMDP.** OLPOMDP is a simple gradient based policy search algorithm. It is worth noting that in each task OLPOMDP uses the same policy as the BOA and MBOA algorithms. OLPOMDP has two parameters one sets the discount factor for the gradient traces, set to .9 for all experiments, and the other sets the size of the gradient update which we optimize for each experiment individually.

**LSPI.** LSPI results are reported in the cart-pole and acrobot tasks. Despite our best efforts we were not able to get reasonable results in the mountain car and arm tasks. Our efforts included selecting combinations of basis functions and exploration strategies. We attempted radial basis, polynomial basis, and hybrid basis. Our exploration strategy uses policies returned from the LSPI optimization with added noise (including fully random exploration). No combination of basis functions and exploration strategies yielded reasonable results in two of the domains (addressing the exploration problem is a central issue for our algorithm and this issue is completely ignored by LSPI). For those domains with results radial basis functions were used with centers located according to the CMAC function approximator.

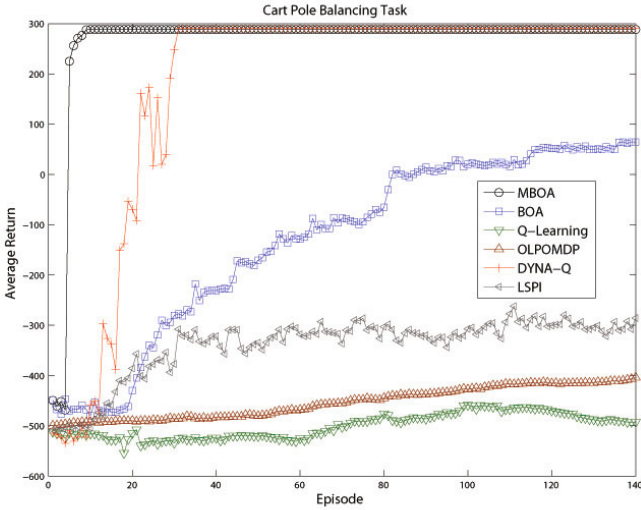
## 4.2 Cart-Pole Task

The agents goal in the cart-pole domain is to maintain the vertical position of the pole for 1000 steps while keeping the cart within a fixed boundary. The state of the system includes the location of the cart, velocity of the cart, the angle of the pole, and the rate of change of the poles angle. At each step the agent receives a positive reward of 30, plus a penalty proportional to the difference in angle of the pole and the desired upright position, plus a term penalizing distance away from the center of the boundary, plus a term penalizing large changes in the angle. A reward of 100 is received for keeping the pole upright for the complete duration. Essentially, the shaping reward encourages stable policies. MBOA, BOA, and OLPOMDP require a parameterized policy. We select a linear policy for each algorithm,

$$a = \theta s' + \epsilon, \quad (19)$$

where  $s$  includes the state variables described above, and the value of  $\epsilon$  is assumed to be Gaussian distributed. In this case  $\theta$  has four parameters.

Figure 1 shows the results for the Car Pole task. In this case MBOA can accurately model the dynamics of the system after a small number of observed trajectories. This results in fast convergence to an excellent policy which dominates the quality of the policies found by the other algorithms. Comparatively all other methods require much more data before a good policy is found. BOA always finds a policy which balances the pole for the full 1000 steps. However, some of the discovered policies have more erratic behavior decreasing the accumulated reward for each run. Q-Learning and OLPOMDP do not find comparable policies until after at least 250 additional episodes are experienced. LSPI converges faster but cannot match the performance of either the BOA or



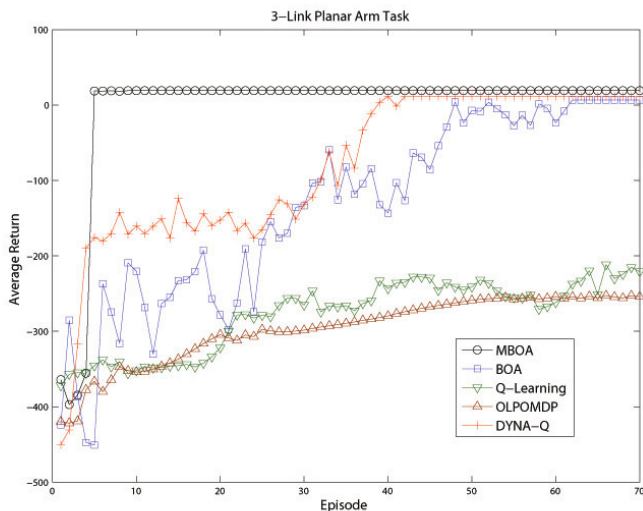
**Fig. 1.** cart-pole Balancing Task: We report the total return per episode averaged over multiple runs of each algorithm. MBOA, BOA, and DYNA-Q are averaged over 30 runs. Q-Learning and OLPOMDP are averaged over 300 runs to control for the erratic behavior of these algorithms.

MBOA algorithm (please note our problem is distinct from the original LSPI balancing task, we allow the agent to apply less force, include boundaries for the cart, and penalize unstable policies). After 30 episodes the DYNA-Q algorithm has found a solution comparable to that of MBOA. This is not surprising given that the shaping reward provides advice useful for the local updates performed by the DYNA-Q algorithm. We have extended the model-based results by extrapolating the best solution found after convergence in order to show the comparative performance of BOA to the other direct RL methods.

### 4.3 3-Link Planar Arm Task

In the 3-link Planar Arm task the goal is to direct an arm tip to a target region on a 2 dimensional plane. The arm is controlled by applying torques at each joint. The arm responds kinematically to the applied torques; only the torques at the joints influence the change of state. There are specified maximum joint angles, simulating the physical constraints of a real robot, preventing each joint from moving through more than  $180^{\circ}$  of rotation. The action space in this task is three dimensional, one dimension for each joint, each of which can apply a torque of -1 or 1. The reward signal for the agent is the squared distance between the tip of the arm and the center of the target region.

To handle the 3 dimensional action space a separate controller is learned for each joint. For MBOA, BOA, and OLPOMDP a logistic function controls each joint (6 total policy parameters). DYNA-Q and Q-Learning approximate separate Q-functions for each joint. The state space for each individual joint controller is simply the computed distance between the x and y coordinates of the arm tip and target locations respectively.



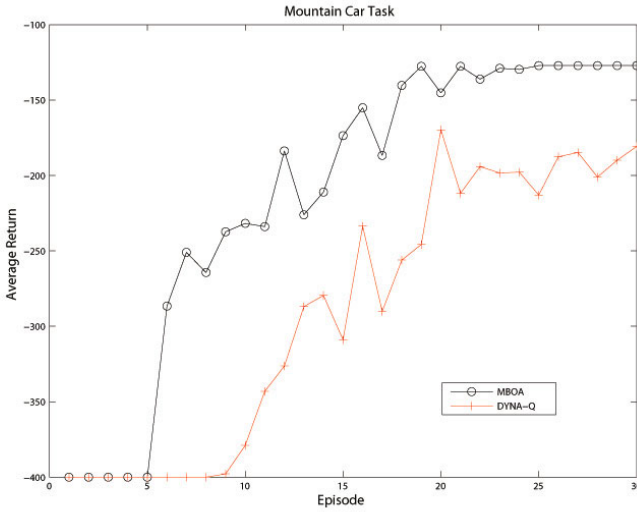
**Fig. 2.** Planar Arm Task: We report the total return per episode averaged over multiple runs of each algorithm. MBOA, BOA, and DYNA-Q are averaged over 30 runs. Q-Learning and OLPOMDP are averaged over 300 runs.

The relative performance of each algorithm is shown in Figure 2. Like the cart-pole task the transition and reward function can be captured by the linear model. Once the system is successfully modeled the optimization of the expected improvement quickly identifies the optimal policy. It takes several additional trials before the BOA algorithm begins finding policies of similar quality. The other model-free alternatives require 1000 episodes before converging to the same result. It is worth noting that given far more episodes the CMAC approximator for the Q-learning algorithm finds a policy superior, on average, to the BOA algorithm. The additional internal simulations clearly benefit the DYNA-Q algorithm. After 40 episodes we report the maximum average return achieved by the algorithm. DYNA-Q has found a policy comparable to the MBOA algorithm, but requires more experience.

#### 4.4 Mountain Car Task

In the mountain car domain the goal is to accelerate a car from a fixed position at the base of a hill to the hills apex. The state includes the location of the car on the hill and the car's velocity. At each step the agent receives a reward of -1. At the end of an episode if the agent has reached the apex of the hill it receives 100 reward. To control the car the agent can apply an acceleration of -1 or 1. It is important that the accelerations are not sufficient to simply force the car up one side of the hill. Doing so would make the task too trivial. Success is only achieved by using the force of gravity to augment the cars accelerations. MBOA, BOA, and OLPOMDP optimize a logistic function to control the car (8 policy parameters).

In Figure 3 we report the results for the MBOA and DYNA-Q algorithms. Due to the sparse reward signal, all of the model-free methods would appear at the base of this



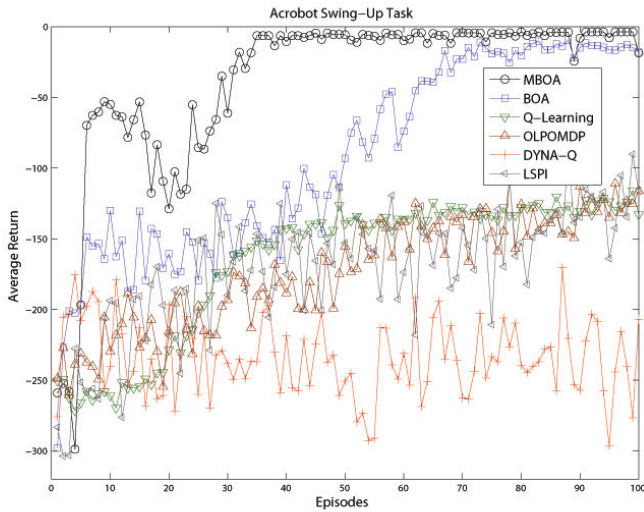
**Fig. 3.** Mountain Car Task: We report the average total return per episode in the mountain car task. The MBOA and DYNA-Q results were averaged over 30 runs.

graph. We omit these uninformative results. However, the important comparison between the model-based methods shows that the MBOA algorithm makes more effective use of the continuous domain models. In this case the linear models for the system cannot fully capture the transition function for the domain. MBOA more aptly corrects for modeling errors than the DYNA-Q algorithm, which cannot ignore or correct the model when it returns erroneous results. These errors accumulate when DYNA-Q performs internal updates impacting the quality of its solution.

#### 4.5 Acrobot Task

In the acrobot domain the goal is to swing the foot of the acrobot over a specified threshold by applying torque at the hip. The state of the system includes the angle of the torso of the acrobot, the angle between the torso and legs, and the rate of change of each angle. Like the planar arm task real world constraints are placed on the articulations of the joints which prevent the legs of the acrobot from crossing through the torso. The agent can only control the behavior of the acrobot at the hip by applying a torque of  $-1$  or  $1$ . At each step the agent receives  $+100$  bonus if the foot reached the goal height, or a  $-1$  penalty in other steps. Again, MBOA, BOA and OLPOMDP optimize a logistic function governing the probability of selecting an action (6 policy parameters).

This under-actuated task has a complicated transition between states. We were unable to identify even a non-linear model that predicts the controls with high accuracy. Instead the model-based systems use a linear approximation of the transition function, which when used for simulation reports incorrect returns for most of the policy space. Domains with these properties typically motivate the use of model-free methods. As indicated in Figure 4, MBOA can still make use of even this highly inaccurate model. In the initial stages of the learning process MBOA is uncertain of the quality of its



**Fig. 4.** Acrobot Task: We report the total return per episode averaged over multiple runs of each algorithm. MBOA, BOA, and DYNA-Q are averaged over 30 runs. Q-Learning and OLPOMDP are averaged over 300 runs.

model, but by modeling the residuals MBOA still benefits from the in regions where the predictions are accurate. This explains why the performance of MBOA improves over BOA’s sample complexity. Over time, as the difference in predictions increases, MBOA slowly begins ignoring the model estimates defaulting to the behavior of BOA. By contrast, because the DYNA-Q algorithm treats the model as a surrogate for the domain its estimates of the state action values are damaged during internal simulation. Regardless of the quantity of data available the assumptions made by the DYNA-Q algorithm (an assumption shared by most model-based algorithms) prevent it from learning in this context. It is worth noting that LSPI fails to improve on the performance of BOA (and barely improves on the performance of the model-free algorithms). This is additional evidence that expending some computational effort to select policies for exploration improves the quality of information observed. The reduction in exploratory episodes is considerable.

## 5 Conclusion

We have proposed extending the Bayesian Optimization approach to RL by augmenting the surrogate function representing the expected return to have a mean which is dependent on an approximate model. The MBOA algorithm which takes advantage of the improved surrogate function is designed to both improve data efficiency when the model reasonably approximates the domain, and be robust to extreme errors in the model when it is highly inaccurate. We demonstrate the effectiveness of the MBOA algorithm in four benchmark RL domains. Empirically the MBOA algorithm outperforms LSPI, OLPOMDP, Q-Learning with CMAC function approximation, BOA, and

DYNA-Q in the mountain car, cart-pole, and planar arm tasks. In the case of the acrobat swing up task, where we could not find an accurate model for the domain, MBOA still outperforms all other algorithms. Empirically MBOA outperforms all of the alternatives in terms of data efficiency. This efficiency is gained at the cost of additional computation time for the simulations, which generate sample trajectories of candidate policies during optimization of the expected improvement. Overall, MBOA appears to be a useful step toward combining model-based methods with Bayesian Optimization for purposes of handling inaccurate models and improving data efficiency.

## Acknowledgements

We gratefully acknowledge the support of the Army Research Office under grant number W911NF-09-1-0153 and the NSF under grant number IIS-0905678.

## References

1. Lizotte, D., Wang, T., Bowling, M., Schuurmans, D.: Automatic gait optimization with gaussian process regression. In: IJCAI'07: Proceedings of the 20th International Joint Conference on Artificial Intelligence, pp. 944–949. Morgan Kaufmann, San Francisco (2007)
2. Lizotte, D.: Practical Bayesian Optimization. PhD thesis, University of Alberta (2008)
3. Brochu, E., Cora, V., de Freitas, N.: A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. Technical Report TR-2009-023 (2009)
4. Engel, Y., Mannor, S., Meir, R.: Reinforcement learning with Gaussian processes. In: International Conference on Machine Learning, pp. 201–208 (2005)
5. Dearden, R., Friedman, N., Andre, D.: Model based Bayesian exploration. In: UAI (1999)
6. Strens, M.J.A.: A Bayesian framework for reinforcement learning. In: International Conference on Machine Learning, pp. 943–950 (2000)
7. Duff, M.: Design for an optimal probe. In: International Conference on Machine Learning (2003)
8. Lagoudakis, M.G., Parr, R., Bartlett, L.: Least-squares policy iteration. *Journal of Machine Learning Research* 4 (2003)
9. Sutton, R., Barto, A.G.: *Reinforcement Learning: An Introduction*. MIT Press, Cambridge (1998)
10. Baxter, J., Bartlett, P.L., Weaver, L.: Experiments with infinite-horizon, policy-gradient estimation. *Journal of Artificial Intelligence Research* 15(1), 351–381 (2001)
11. Mockus, J.: Application of bayesian approach to numerical methods of global and stochastic optimization. *Global Optimization* 4(4), 347–365 (1994)
12. Vazquez, E., Bect, J.: Convergence properties of the expected improvement algorithm with fixed mean and covariance functions. *Journal of Statistical Planning and Inference* 140(11), 3088–3095 (2010)
13. Rasmussen, C.E., Williams, C.K.I.: *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, Cambridge (2005)
14. Jones, D.R., Perttunen, C.D., Stuckman, B.E.: Lipschitzian optimization without the lipschitz constant. *J. Optim. Theory Appl.* 79(1), 157–181 (1993)

# Efficient and Numerically Stable Sparse Learning

Sihong Xie<sup>1</sup>, Wei Fan<sup>2</sup>, Olivier Verscheure<sup>2</sup>, and Jiangtao Ren<sup>1</sup>

<sup>1</sup> Sun Yat-Sen University, Guangzhou, China

{xiesihong1,issrjt}@gmail.com

<sup>2</sup> IBM T.J. Watson Research Center, New York, USA

{weifan,ov1}@us.ibm.com

**Abstract.** We consider the problem of numerical stability and model density growth when training a sparse linear model from massive data. We focus on scalable algorithms that optimize certain loss function using gradient descent, with either  $\ell_0$  or  $\ell_1$  regularization. We observed numerical stability problems in several existing methods, leading to divergence and low accuracy. In addition, these methods typically have weak controls over sparsity, such that model density grows faster than necessary. We propose a framework to address the above problems. First, the update rule is numerically stable with convergence guarantee and results in more reasonable models. Second, besides  $\ell_1$  regularization, it exploits the sparsity of data distribution and achieves a higher degree of sparsity with a PAC generalization error bound. Lastly, it is parallelizable and suitable for training large margin classifiers on huge datasets. Experiments show that the proposed method converges consistently and outperforms other baselines using 10% of features by as much as 6% reduction in error rate on average. Datasets and software are available from the authors.

## 1 Introduction

In this paper, we focus on training a sparse large margin model. Assume that we are given  $m$  labeled examples  $Z = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_m, y_m)\}$ , where  $\mathbf{x}_i \in \mathbb{R}^d, i = 1, \dots, m$ . We aim at solving the following optimization problem:

$$\min_{\mathbf{w} \in \mathbb{R}^d} \frac{1}{m} \sum_{i=1}^m L(\langle \mathbf{w}, \mathbf{x}_i \rangle, y_i) + \lambda \|\mathbf{w}\|_1 \quad (1)$$

$L$  is any smooth and differentiable loss function such as logistic or hinge loss.  $\lambda$  is the parameter for trading off between loss and  $\ell_1$  regularization. We are interested in scalable algorithm, with parallelizable data accesses and small communication cost. One can find such application in text mining and webspam detection, where the number of features could be in millions.

Sparse learning aims at accurate models using a small number of non-zero elements, with the advantages of efficiency and generalizability [13]. Some existing methods produce sparse models by forward-backward feature selection [13]



**Table 1.** Notations and definitions

Notation	definition	Notation	Definition
$\mathbf{x}$	Example in $\mathbb{R}^d$	$X$	Data matrix
$y$	Label of $\mathbf{x}$	$Z$	A series of $m$ labeled examples
$\mathbf{w}$	Linear model in $\mathbb{R}^d$	$\boldsymbol{\theta}$	Dual vector of $\mathbf{w}$ in Eq. (8)
$\mathbf{w}_Z$	Linear model learned from $Z$	$\mathbf{w}^*$	Optimal linear model
$\tilde{\mathbf{w}}$	Vector to be thresholded	$H(\cdot, s)$	Hard-thresholding function
$S(\cdot, \lambda)$	Soft-thresholding function	$\nabla f(\cdot)$	Gradient of $f(\cdot)$
$\lambda$	$\ell_1$ regularization parameter	$t$	Index of iterations
$\eta$	Learning rate	$[d]$	Set of indices $\{1, \dots, d\}$
$J$	Set of indices, $J \subseteq [d]$	$\ \mathbf{v}\ _0$	The support of $\mathbf{v}$
$\ \mathbf{v}\ _1$	The sum of $ v_i $ , $i \in [d]$	$\ \mathbf{v}\ _2$	The Euclidean length of $\mathbf{v}$
$\ \mathbf{v}\ _\infty$	The maximum of $ v_i $	$\sigma_i$	Eigenvalues

or boosting [3]. Though effective, these methods have to scan the training set at each iteration, which is expensive or even impossible for large datasets. Under some restrictive assumptions [14], one can achieve sparse models via convex programming with  $\ell_1$  regularization or constraint. Though scalable as methods in [8,12], there are two main drawbacks. First, numerical problems can lead to iteration divergence or summation cancellation, making the algorithm less useful. Second,  $\ell_1$  regularization only encourages sparsity and may not be enough for sparse learning. As the training proceeds, model complexity can grow faster than necessary and result in dense models, regardless of the regularization.

In this paper, we propose a perceptron-based algorithm to address the above problems. First, it is numerically stable with convergence guarantee. Second, in addition to  $\ell_1$  regularization, it further takes advantage of the sparsity of training examples to achieve a higher degree of model sparsity, and furthermore, a generalization error bound which is not provided in [8,12]. Experiments show that the proposed method converges with steadily reduced test error rates as the training proceeds. and consistently outperforms previous state-of-the-art algorithms by as much as 8.4% in accuracy using only 10% of all features in one of the tasks.

## 2 Numerical Challenges in Sparse Learning

Notations are summarized in Table 1. Note that an element of a vector is in normal font, for example,  $w_j$  is the  $j$ -th element of the vector  $\mathbf{w}$ . About norms, for any  $\mathbf{v} \in \mathbb{R}^d$ ,  $\|\mathbf{v}\|_{p_1} \leq \|\mathbf{v}\|_{p_2}$  for any  $p_1 \geq p_2$  [6]. If not otherwise specified,  $\|\cdot\|$  is equivalent to  $\|\cdot\|_2$

We have observed numerical problems in two sparse learning algorithms using either  $\ell_1$  or  $\ell_0$  regularization.

### 2.1 Numerical Problems of Direct Iterative Methods

The first problem arises when directly applying matrix iterative methods to solve a system of linear equations

$$\mathbf{y} = X\mathbf{w} \tag{2}$$

where  $X = (\mathbf{x}_1, \dots, \mathbf{x}_m)^\top$ .  $\ell_1$  or  $\ell_0$  regularization are used to obtain sparse model, as in compressive sensing or sparse learning. Suppose there exists an optimal solution  $\mathbf{w}^*$  satisfying Eq. (2), namely  $\mathbf{y} - X\mathbf{w}^* = \mathbf{0}$  ( $\mathbf{0}$  denotes the zero vector). The direct method tries to find  $\mathbf{w}^*$  by minimizing the potential function

$$\Psi(\mathbf{w}) = \frac{1}{2} \|\mathbf{y} - X\mathbf{w}\|^2 \tag{3}$$

Starting with an arbitrary vector  $\mathbf{w}^{(0)}$  (usually  $\mathbf{0}$ ), it follows the gradient direction at each iteration to reduce the potential function value. The gradient of Eq. (3) is  $\nabla\Psi(\mathbf{w}) = -X^\top(\mathbf{y} - X\mathbf{w})$  and we update  $\mathbf{w}^{(t)}$  by

$$\mathbf{w}^{(t)} = \mathbf{w}^{(t-1)} + \eta X^\top(\mathbf{y} - X\mathbf{w}^{(t-1)}) \tag{4}$$

where  $\eta$  is the learning rate. To see when the above iteration fails to converge, let  $\varepsilon^{(t)} = \mathbf{w}^{(t)} - \mathbf{w}^*$  be the error vector.

$$\begin{aligned} \varepsilon^{(t)} &= \mathbf{w}^{(t-1)} + \eta X^\top(\mathbf{y} - X\mathbf{w}^{(t-1)}) - \mathbf{w}^* - \eta X^\top(\mathbf{y} - X\mathbf{w}^*) \\ &= (I - \eta X^\top X)(\mathbf{w}^{(t-1)} - \mathbf{w}^*) = M\varepsilon^{(t-1)} \end{aligned}$$

where  $M = I - \eta X^\top X$  is the iteration matrix. Thus for  $t > 0$  we have

$$\|\varepsilon^{(t)}\| = \|\mathbf{w}^{(t)} - \mathbf{w}^*\| = \|M\varepsilon^{(t-1)}\| = \dots = \|M^t\varepsilon^{(0)}\|$$

The sufficient and necessary condition for the update rule (4) to converge is that the spectral radius of  $M$ ,  $\rho(M) = \max_i\{|\sigma_i|\}$  less than 1, where  $\sigma_i$  is the eigenvalues of  $M$ . Formally,

$$\lim_{t \rightarrow \infty} \|M^t\| = 0 \Leftrightarrow \rho(M) < 1 \tag{5}$$

The above analysis shows that if one cannot guarantee  $\rho(M) < 1$  at each iteration, then update rule (4) may not be applicable in reducing the potential function.

In addition, we would like  $\mathbf{w}^{(t)}$  to be sparse. It is shown in (5) that under the RIP condition (1), a truncated version of update rule (4) reduces the potential function at each iteration while maintaining a sparse solution. We show that the truncation does not automatically guarantee the convergence of  $\ell_0$  constrained gradient descent. Consider the following iteration

$$\mathbf{w}^{(t)} = H(\tilde{\mathbf{w}}^{(t)}, s) = D_{J^{(t)}}(\mathbf{w}^{(t-1)} + \eta X^\top(\mathbf{y} - X\mathbf{w}^{(t-1)})) \tag{6}$$

with  $\mathbf{w}^{(0)}$  being a zero vector.  $\tilde{\mathbf{w}}^{(t)}$  is the  $t$ -th solution before thresholding.  $H(\mathbf{v}, s) : \mathbb{R}^d \rightarrow \mathbb{R}^d$  keeps only  $s$  elements with largest absolute values in  $\mathbf{v}$

and set other elements to zero.  $H(\mathbf{v}, s)$  equals the row selection operation as shown in Eq. (6).  $J^{(t)} \subseteq [d]$  ( $|J^{(t)}| = s, \forall t = 1, 2, \dots$ ) represents the set of indices of  $s$  remaining elements in  $H(\tilde{\mathbf{w}}, s)$ .  $D_{J^{(t)}}$  is a diagonal matrix with  $D_{j,j} = 1, \forall j \in J^{(t)}$  and 0 otherwise. The error vector of iteration rule (6) can then be written as

$$\begin{aligned} \|\varepsilon^{(t)}\|^2 &= \|\mathbf{w}^{(t)} - \mathbf{w}^*\|^2 = \|D_{J^{(t)}}\tilde{\mathbf{w}}^{(t)} - \mathbf{w}^*\|^2 \\ &\geq \|D_{J^{(t)}}\tilde{\mathbf{w}}^{(t)} - D_{J^{(t)}}\mathbf{w}^*\|^2 \\ &= \|D_{J^{(t)}}[\mathbf{w}^{(t-1)} + \eta X^\top(y - X\mathbf{w}^{(t-1)})] \\ &\quad - D_{J^{(t)}}[\mathbf{w}^* + \eta X^\top(y - X\mathbf{w}^*)]\|^2 \\ &= \|D_{J^{(t)}}M(\mathbf{w}^{(t-1)} - \mathbf{w}^*)\|^2 \\ &= \|D_{J^{(t)}}M\varepsilon^{(t-1)}\|^2 \end{aligned} \tag{7}$$

Inequality (7) follows since  $w_j^*, j \notin J^{(t)}$  are held out of the sum of the  $\ell_2$ -norm. Therefore,  $\|\varepsilon^{(t)}\|^2$  is lower-bounded by  $\|D_{J^{(t)}}M\varepsilon^{(t-1)}\|^2$ . For the  $\ell_0$  projected gradient descent to converge,  $D_{J^{(t)}}M$  must satisfy  $\rho(D_{J^{(t)}}M) < 1$ , otherwise, update rule (6) diverges. Simple truncation using  $H(\mathbf{v}, s)$  doesn't guarantee such condition, as we demonstrated in experiments.

### 2.2 Numerical Problems of Mirror Descent

The mirror descent algorithm (MDA) has been recognized as an effective online learning algorithm. For example, MDA using Bregman divergence is proposed to approximate the exponentiated gradient (EG) algorithm, which have cumulative loss bound logarithmically to the number of irrelevant features in the target weight vector [6]. In convergence rate, it is proved that MDA is superior to the usual stochastic gradient descent methods [12], where MDA is adopted in online sparse learning. However, the price MDA pays for these advantages is numerical unstability, especially when the dimensionality of data is high, leading to less discriminative models. As proposed in [12], the main component of MDA for sparse learning is the alternative updates of two vectors  $\boldsymbol{\theta}, \mathbf{w} \in \mathbb{R}^n$  by the following rules:

$$w_j^{(t)} = f_j^{-1}(\boldsymbol{\theta}^{(t)}) = \frac{\text{sgn}(\theta_j^{(t)})|\theta_j^{(t)}|^{p-1}}{\|\boldsymbol{\theta}^{(t)}\|_p^{p-2}}, \forall j \tag{8}$$

$$\boldsymbol{\theta}^{(t+1)} = S(\boldsymbol{\theta}^{(t)} - \eta \nabla L(\mathbf{w}^{(t)}), \lambda) \tag{9}$$

where  $S(\cdot, \lambda)$  is the soft-thresholding operator [2]:

$$\begin{aligned} S(w_j, \lambda) &= \text{sgn}(w_j)(w_j - \lambda)_+ \\ &= \begin{cases} w_j - \lambda, & \text{if } w_j > 0 \text{ and } |w_j| > \lambda \\ w_j + \lambda, & \text{if } w_j < 0 \text{ and } |w_j| > \lambda \\ 0, & \text{if } |w_j| < \lambda \end{cases} \end{aligned} \tag{10}$$

$S(\mathbf{v}, \lambda)$  means applying Eq. (10) at each element of  $\mathbf{v}$ . As suggested in [6], the parameter  $p$  in Eq. (8) is set to  $O(\ln(d))$  (Note that different  $p$  lead to different

“algorithms”, for example, in binary classification, one obtain Perceptron when  $p = 2$  and Weighted Majority as  $p \rightarrow \infty$ . However, for MDA to approximate the EG algorithm,  $p$  should be sufficiently large such as logarithmic  $p$ ).  $\mathbf{w}$  and  $\boldsymbol{\theta}$  are called primal and dual vector, respectively. In the  $t$ -th iteration, MDA first computes  $\mathbf{w}^{(t)}$  using  $\mathbf{w}^{(t)} = f^{-1}(\boldsymbol{\theta}^{(t)})$ , with  $\boldsymbol{\theta}^{(0)} = \mathbf{0}$ . Then the stochastic gradient of the loss function  $\nabla L(\mathbf{w}^{(t)})$  is estimated using  $(\mathbf{x}_i, y_i)$  at  $\mathbf{w}^{(t)}$ . Finally gradient descent and soft-thresholding update  $\boldsymbol{\theta}^{(t)}$  to get  $\boldsymbol{\theta}^{(t+1)}$ . Converting  $\boldsymbol{\theta}$  to  $\mathbf{w}$  using Eq. (8) can cause numerical problem in MDA. Specifically, elements of  $\mathbf{w}$  could become very small and sensitive to difference of  $\theta_j$ 's scale. The following lemma reveals the numerical problem that update rules (8) and (9) can bring.

**Lemma 1.** *Assume that the values of features and  $\lambda$  in Eq. (9) are in the scale of  $O(1)$ . At the  $t$ -th iteration of MDA minimizing logistic loss, where  $t = O(p)$ ,  $\theta_j^{(t)}$  is also in  $O(p)$  and  $w_j^{(t)}$  is at most in the order of  $O(a^{-p})$  for some  $a > 1$ .*

*Proof.* Elements of the gradient of the logistic loss  $\nabla_j L(\cdot, \cdot)$  with respect to the first argument is  $L'(\langle \mathbf{w}, \mathbf{x} \rangle, y) \mathbf{x}$  and thus in the order of  $O(1)$ .  $\theta_j^{(t)}$  is the summation of  $t$  terms in  $O(1)$  and thus in the order of  $O(p)$ . Since  $\|\boldsymbol{\theta}\|_p > \|\boldsymbol{\theta}\|_\infty$ ,  $\exists \epsilon > 0$  s.t.  $\|\boldsymbol{\theta}\|_p > (1 + \epsilon)\|\boldsymbol{\theta}\|_\infty = (1 + \epsilon) \max_i |\theta_i|$ . By Eq. (8),

$$\begin{aligned} |w_j| &= \frac{|\theta_j|^{p-1}}{\|\boldsymbol{\theta}\|_p^{p-2}} < \frac{|\theta_j|^{p-1}}{(1 + \epsilon)^{p-2} (\max_i |\theta_i|)^{p-2}} \\ &= |\theta_j| \left( \frac{|\theta_j|}{(1 + \epsilon) \max_i |\theta_i|} \right)^{p-2} \\ &= O(p) O(a^{2-p}) = O(a^{-p}) \end{aligned}$$

where  $a = (1 + \epsilon) \max_i |\theta_i| / |\theta_j| > 1$ .

Particularly, because  $w_j$  is exponential in  $p = O(\ln(d))$ , small difference between exponents of dimensions could be greatly amplified in the resulting model. Consider two entries of  $\boldsymbol{\theta}$ :  $\theta_1$  and  $\theta_2$ . Without loss of generality, assume that  $\theta_1$  is only one order smaller than  $\theta_2$  in magnitude:  $|\theta_1| \approx 10^{-1} |\theta_2|$ . Reconstruct the primal vector  $\mathbf{w}$  from  $\boldsymbol{\theta}$ , we can see that  $w_1$  is  $p$  order smaller than  $w_2$  in magnitude:  $|w_1| \approx 10^{-p} |w_2|$ . When computing the inner product between  $\mathbf{w}$  and  $\mathbf{x}$ , if the difference between exponents of  $w_1$  and  $w_2$  is larger than the precision that the data type supports, then the 2nd element in  $\mathbf{x}$  would be totally lost. For example, double precision floating point number supported by C++ (64 bits in length according to IEEE 754-2008) typically has precision of  $10^{-16}$ . Therefore  $1.0 + 1.0^{-17} = 1.0$  in machine addition. In general, the larger  $p$  is, the more difference between  $w_j$ 's exponents. We'll show in experiment (Section 4.3) how the parameter  $p$  affects the performance of MDA.

### 3 Efficient and Numerically Stable Sparse Learning

We present the proposed sparse learning algorithm in Section 3.1. In Section 3.2 we show that the method is guaranteed to converge, with numerical errors taken into account. Finally we show generalization error bound in Section 3.3.

### 3.1 The Proposed Method

Given the labeled data  $Z = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_m, y_m)\}$ , drawn from some distribution  $D$  over  $\mathbb{R}^d \times \{-1, 1\}$ , we consider learning a sparse large margin linear classifier. This reduces to minimizing certain loss function. There are three concerns. First, the algorithm should be scalable. One of the best practical methods known is online learning or stochastic gradient descent, which iteratively minimizes the loss function and keeps only the footprint of the sparse model without storing any example in memory. Second, the algorithm should be numerically stable. We prefer algorithms which rely on practical assumption and are robust to small difference in the scale between dimensions of data. Finally, for efficiency consideration, algorithms achieve better classification performance using less features are desirable.

---

#### Algorithm 1. Numerically Stable Sparse Learning

---

```

1: Input: margin threshold  $\tau$ , learning rate  $\eta$ ,
      regularization parameter  $\lambda$ , density upper bound  $s$ 
2: initialize linear model  $\mathbf{w} = \mathbf{0}$ , model density=0
3: while model density less than  $s$  do
4:   Receive instance  $(\mathbf{x}_i, y_i)$  and compute  $z = y_i \mathbf{w} \mathbf{x}_i$ 
5:   if  $z \leq \tau$  then
6:     for Each non-zero element  $x_{ij}$  do
7:       Update  $w_j$  by  $w_j = w_j + \eta x_{ij} y_i$ 
8:       Soft threshold  $w_j$  using Eq. (10)
9:     end for
10:  end if
11: end while

```

---

We alter the perceptron algorithm such that it robustly produces sparse model with convergence and generalization error guarantees (see Section 3.2 and 3.3). The proposed method is described in Algorithm 1. It begins with a zero vector. During each iteration, it loads an example from secondary storage and updates the model as the original perceptron (line 4-7). The difference is that after each online update, the algorithm suppresses the weights in the model (line 8) using the soft-thresholding operator (Eq. (10)). Since we assume that the data is separable by a sparse linear classifier, by the fact that the data is in high dimensional space, it is expected that there would be many noisy or irrelevant features. For an example classified with margin higher than  $\tau$ , it is more likely that the features in the current model  $\mathbf{w}^{(t)}$  are sufficient to make correct decision, and the example is not used for update. This prevents unnecessarily introducing new features into the current model and maximally preserves the sparsity without increasing regret. In general, perceptron algorithm exploits the sparsity of the instance space, and further leads to a model with fewer features.

The above framework can be applied to very large datasets. The main task at each iteration is computing  $\langle \mathbf{w}^{(t)}, \mathbf{x} \rangle$ , which is in the form of summation

and can be distributed to  $r$  machines. One can partition the dataset on a per-feature basis into  $r$  partitions, with each assigned to a machine. Each machine computes  $O(\lceil d/r \rceil)$  components in  $\langle \mathbf{w}^{(t)}, \mathbf{x} \rangle$ . Adding these parts up obtains the inner product. There are  $O(\lceil d/r \rceil)$  cost on each machine for I/O and  $O(r)$  cost for communication. Thus the algorithm is scalable via parallelization on columns of the data matrix  $X$ .

### 3.2 Convergence of the Proposed Method

Without loss of generality, we assume that the learning rate equals 1 in the following analysis. We alter the convergence theorem in [10] and prove similar results of perceptron with soft-thresholding, and analyze conditions under which the soft-thresholding version perceptron converges. Assume that the  $t$ -th mistake happens on example  $(\mathbf{x}, y)$ . Let  $J_0 = \{j | x_j = 0\}$  and  $J_1 = \{j | x_j \neq 0\}$ . Then  $w_j^{(t)} = w_j^{(t-1)}$  for  $j \in J_0$  and we can only pay attention to the difference between  $w_j^{(t)}$  and  $w_j^{(t-1)}$  for  $j \in J_1$ . By Algorithm 1 and Eq. (10),

$$\mathbf{w}^{(t)} = S(\tilde{\mathbf{w}}^{(t)}, \lambda) = S(\mathbf{w}^{(t-1)} + y\mathbf{x}, \lambda)$$

Project  $\mathbf{w}^{(t-1)}$ ,  $\mathbf{w}^{(t)}$  and  $\mathbf{x}$  onto  $J_0$  and  $J_1$  respectively, and let the projection of a vector  $\mathbf{v}$  denoted by the vector with the index set as its subscript (for example,  $\mathbf{x}_{J_0}$  means keep the elements on indices in  $J_0$  unchanged and set the other elements to zero). By simple algebra,  $\mathbf{w}^{(t)}$  can be decomposed into the sum of two orthogonal vectors  $\mathbf{w}^{(t)} = \mathbf{w}_{J_0}^{(t-1)} + \mathbf{w}_{J_1}^{(t)}$ . We have the following lemma.

**Lemma 2.** Assume that  $\forall (\mathbf{x}, y) \in Z, \|\mathbf{x}\| < R$ . Let  $J_+^{(t)}$  denote the set  $\{j : |\tilde{w}_j^{(t)}| \geq \lambda\} \cap J_1$  and  $J_-^{(t)}$  be the set  $\{j : |\tilde{w}_j^{(t)}| < \lambda\} \cap J_1$ . Let  $\epsilon_\lambda^{(t)} = \lambda \sum_{j \in J_+^{(t)}} |\tilde{w}_j^{(t)}| + \sum_{j \in J_-^{(t)}} |\tilde{w}_j^{(t)}|^2$ . If  $\tau \leq \epsilon_\lambda^{(t)}/2$ ,  $\|\mathbf{w}^{(t)}\|^2 \leq tR^2$ .

*Proof.* Prove by induction,

$$\begin{aligned} \|\mathbf{w}^{(t)}\|^2 &= \|\mathbf{w}_{J_0}^{(t-1)} + \mathbf{w}_{J_1}^{(t)}\|^2 = \|\mathbf{w}_{J_0}^{(t-1)}\|^2 + \|\mathbf{w}_{J_1}^{(t)}\|^2 \\ &\leq \|\mathbf{w}_{J_0}^{(t-1)}\|^2 + \|\tilde{\mathbf{w}}_{J_1}^{(t)}\|^2 - \epsilon_\lambda^{(t)} \end{aligned} \tag{11}$$

$$= \|\mathbf{w}_{J_0}^{(t-1)}\|^2 + \|\mathbf{w}_{J_1}^{(t-1)} + y\mathbf{x}\|^2 - \epsilon_\lambda^{(t)} \tag{12}$$

$$\leq \|\mathbf{w}_{J_0}^{(t-1)}\|^2 - \epsilon_\lambda^{(t)} + \|\mathbf{w}_{J_1}^{(t-1)}\|^2 + \|y\mathbf{x}\|^2 + 2\tau \tag{13}$$

$$\leq \|\mathbf{w}^{(t-1)}\|^2 + R^2 - \epsilon_\lambda^{(t)} + 2\tau \tag{14}$$

Inequality (11) follows from the fact that at each iteration  $t$ ,  $\tilde{\mathbf{w}}^{(t)}$  suffered from at least  $\epsilon_\lambda^{(t)}$  of shrinkage. Inequality (13) holds because update occurs only when  $\langle \mathbf{w}, y\mathbf{x} \rangle \leq \tau$ . Since  $\tau \leq \epsilon_\lambda^{(t)}/2$ , then  $\|\mathbf{w}^{(t)}\|^2 \leq \|\mathbf{w}^{(t-1)}\|^2 + R^2$ .

**Discussion.** We assume the condition  $\tau \leq \epsilon_\lambda^{(t)}/2$  for the convenience of proof. If  $\tau = 0$ , then the condition is satisfied for  $\forall \lambda > 0$ . Particularly, if  $\lambda$  is set too small, then the algorithm asymptotically becomes Perceptron; By using a large  $\lambda$ , one can achieve a sparser model with sacrifice of accuracy. This is a more general problem of model selection and one need to trade off sparsity for accuracy [11]. In the experiment, we choose the parameters  $\lambda$  and  $\tau$  via validation, then they're fixed during iterations.

We proved the above lemma without considering numerical errors. Let the machine precision be  $eps = 2^{-r}$ . Soft-thresholding introduces errors if  $\lambda$  and the truncated elements are rounded-off. This gives an error up to  $|\epsilon_1| \leq 2eps$  which is minor and can be ignored. Note that  $\|\mathbf{w}^{(t)}\|$  usually grows with the iterations, thus more importantly, we must consider the numerical errors associated with this term. With  $|\epsilon_2| \leq eps$ , the exact term  $\|\mathbf{w}_{J_1}^{(t-1)} + y\mathbf{x}\|^2$  in Eq. (12) becomes  $\|(1 + \epsilon_2)(\mathbf{w}_{J_1}^{(t-1)} + y\mathbf{x})\|^2$  and can be approximately upper-bounded by  $(\|\mathbf{w}_{J_1}^{(t-1)}\|^2 + \|y\mathbf{x}\|^2 + 2\tau)(1 + 2\epsilon_2)$  by ignoring the higher order term  $\epsilon_2^2$  (an exact upper-bound should be  $(\|\mathbf{w}_{J_1}^{(t-1)}\|^2 + \|y\mathbf{x}\|^2 + 2\tau)(1 + 2\epsilon_2 + \epsilon_2^2)$ ). Substitute this error into Eq. (12), we can prove the following theorem, which is the key to the convergence proof.

**Theorem 1.** *Given  $\lambda \geq 0$ , let  $\epsilon_\lambda^* = \min_t \epsilon_\lambda^{(t)}$ , then  $\|\mathbf{w}^{(t)}\|^2 < tR^2$  holds,  $\forall t \in \mathbb{Z}^+$  such that*

$$t \leq \frac{\epsilon_\lambda^* - 2\tau}{2\epsilon_2 R^2} - \frac{2\tau}{R^2} \tag{15}$$

*Proof.* The inequality (14) becomes

$$\underbrace{\|\mathbf{w}^{(t-1)}\|^2 + R^2 - \epsilon_\lambda^* + 2\tau + 2\epsilon_2(\|\mathbf{w}_{J_1}^{(t-1)}\|^2 + \|y\mathbf{x}\|^2 + 2\tau)}_{\leq 0}$$

As in the proof of Lemma 2, we need to prove that the under-braced sum is smaller than  $R^2$ . Since by induction,  $\|\mathbf{w}_{J_1}^{(t-1)}\|^2 \leq \|\mathbf{w}^{(t-1)}\|^2 \leq (t-1)R^2$  and  $\|y\mathbf{x}\|^2 \leq R^2$  by assumption, we need only to prove that

$$-\epsilon_\lambda^* + 2\tau + 2\epsilon_2(tR^2 + 2\tau) \leq 0$$

Solving for  $t$  gets the conclusion.

In C++,  $\epsilon_2 \approx 10^{-16}$  in double-precision. If we set  $\lambda$  and  $\tau$  such that  $\epsilon_\lambda^* - 2\tau$  is not too small (for example, in the order of  $o(R^2)$ ), then  $\|\mathbf{w}^{(t)}\|^2 < tR^2$  holds for a sufficient large  $t$  in a reasonable training process.

**Theorem 2.** *Assume for all examples  $\mathbf{x}_i \in Z$ ,  $\|\mathbf{x}_i\| < R$ . If there exists a linear model  $\mathbf{u}$  such that  $\|\mathbf{u}\| = 1$  and  $0 < \epsilon_{\lambda,\gamma} < 1$  such that  $\langle (y_i\mathbf{x}_i - \mathbf{v}), \mathbf{u} \rangle \geq (1 - \epsilon_{\lambda,\gamma})\gamma$  for  $\forall (\mathbf{x}, y) \in Z$  and any  $\mathbf{v}$ ,  $\|\mathbf{v}\|_\infty < \lambda$ , then the number of mistakes made by the online perceptron algorithm on  $Z$  is at most  $k = (R/(1 - \epsilon_{\lambda,\gamma})\gamma)^2$ .*

*Proof.* For any  $t$  satisfying Eq. (15),

$$\begin{aligned} \mathbf{w}^{(t)} \cdot \mathbf{u} &= \mathbf{w}^{(t-1)} \cdot \mathbf{u} + (\Delta^{(t)} + y_i \mathbf{x}_i) \cdot \mathbf{u} \\ &\geq \mathbf{w}^{(t-1)} \cdot \mathbf{u} + (1 - \epsilon_{\lambda, \gamma})\gamma \end{aligned} \tag{16}$$

where  $\Delta^{(t)} = S(\tilde{\mathbf{w}}^{(t)}, \lambda) - \tilde{\mathbf{w}}^{(t)}$ . Thus  $\mathbf{w}^{(t)} \cdot \mathbf{u} \geq (1 - \epsilon_{\lambda, \gamma})t\gamma$ . By Lemma (2),  $\|\mathbf{w}^{(t)}\|^2 \leq tR^2$ .

$$(1 - \epsilon_{\lambda, \gamma})t\gamma \leq \mathbf{w}^{(t)} \cdot \mathbf{u} \leq \|\mathbf{w}^{(t)}\| \leq \sqrt{t}R$$

This theorem indicates that if data in  $Z$  can be separated with margin at least  $\gamma$ , and this margin does not shrink too much compared with  $\gamma$  (up to  $\epsilon_{\lambda, \gamma}$ ) given the examples are soft-thresholded, then the Algorithm 1 needs at most  $(R/(1 - \epsilon_{\lambda, \gamma})\gamma)^2$  examples among  $Z$  to learn a linear classifier consistent with all examples in  $Z$ . For the inseparable case, according to the method in [4], one can extend all the examples  $\mathbf{x}$  to  $\mathbf{x}'$  and the linear model  $\mathbf{u}$  to  $\mathbf{u}'$ . In this extended space,  $(\mathbf{x}', y)$  is linearly separable by  $\mathbf{u}'$ .

### 3.3 Generalization Error Bound of the Proposed Method

In Algorithm 1, features enter the model only when necessary. Such strategy also allow us to construct a consistent classifier using a subset of the training set. Combining these two properties, it is possible to learn a sparse model with generalization error guarantee. Algorithm 1 can be seen as a compression scheme [9] consisting of two mappings. The first mapping  $\kappa$  maps any training set  $Z$  of size  $m$  to  $Z_k \subset Z$ , called the “kernel” of  $Z$ , where  $k$  is the kernel size. The second mapping  $\pi$  uses  $Z_k$  to reconstruct the labels of all the examples in  $Z$ .  $\kappa$  can be seen as an algorithm, learning from  $m$  training examples and encoding the produced hypothesis  $h$  using only  $k$  of them. The mapping  $\pi$  requires that  $h$  is consistent with all the training examples. An algorithm with the data compression property is guaranteed with generalization error bound [9].

**Lemma 3.** *For any compression scheme with kernel size  $k$ , the probability that the generalization error of the learned hypothesis (with respect to distribution  $D$ ) being larger than  $\epsilon$  is less than  $\binom{m}{k}(1 - \epsilon)^{m-k}$*

With a smaller  $k$  where  $k < \lfloor m/2 \rfloor$ , the bound on generalization error becomes smaller. Combining Theorem 2 and Lemma 3, with high probability over the randomly drawn training set  $Z$ , the soft-thresholding perceptron algorithm can obtain a classifier  $\mathbf{w}_Z$  of good generalizability using only  $k$  examples out of the total  $m$  training examples.

**Theorem 3.** *With probability at least  $1 - \delta$  over the random draw of the training set  $Z$  of size  $m$ , given the conditions in Theorem 2, the generalization error of the classifier found by the proposed algorithm is less than*

$$\frac{1}{m - k} \left( \ln \binom{m}{k} + \ln(m) + \ln \frac{1}{\delta} \right) \tag{17}$$

where  $k = (R/(1 - \epsilon_{\lambda, \gamma})\gamma)^2$  for some  $0 < \epsilon_{\lambda, \gamma} < 1$ .



**Table 2.** Description of classification tasks. Columns sequentially show the task ids, the tasks’ names, the size of the training sets (#Tr), validation set (#Valid), test set (#Test) and number of total features (#Dim)

No.	Task	#Tr	#Valid	#Test	#Dim
1	comp_vs_rec	4278	1644	2948	21317
2	comp_vs_sci	4248	1766	2829	22944
3	comp_vs_talk	3960	1577	2607	24178
4	rec_vs_talk	3456	1423	2353	23253
5	sci_vs_rec	3772	1598	2561	22839
6	sci_vs_talk	3397	1445	2363	24777
7	rcv1	20242	3357	6854	47236
8	webspam	50000	4999	50000	16609143

*Proof.* The kernel size of the model learned from  $Z$  by the soft-thresholding algorithm is bounded by  $k = (R/(1 - \epsilon_{\lambda,\gamma})\gamma)^2$ , for some  $0 < \epsilon_{\lambda,\gamma} < 1$ . By Lemma 3,

$$\binom{m}{k} (1 - \epsilon)^{m-k} \leq \binom{m}{k} e^{(k-m)\epsilon} \quad (18)$$

Let the right hand side of the above inequality equal to  $\delta$  and solve for  $\epsilon$ , we obtain the conclusion.

This theorem is similar to the results given in [7] where the perceptron algorithm is run in the dual form and the output is a linear classifier in the kernel space. The difference is that the dual perceptron algorithm needs to store  $k$  training examples for prediction while the proposed method stores only a sparse vector.

## 4 Experiment

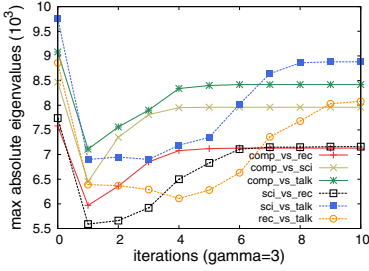
In this section, we present experiment results of the proposed and other methods, focusing on the numerical stability and efficiency of sparse models. After briefing the experiment settings, three subsections address the problems listed below:

- i Does it converge when applying gradient descent directly on real-world data.
- ii How numerical unstability affects MDA’s convergence and model accuracy.
- iii To what extent of sparsity can one achieve with performance guarantee.

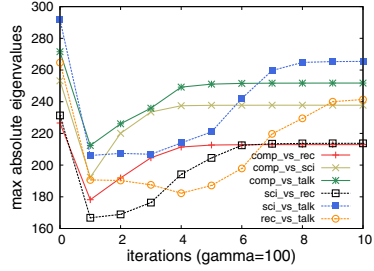
### 4.1 Experiment Settings

We conducted experiments on three datasets. The first dataset is 20newsgroups, from which we construct six binary classification tasks. We used word vector representation with TFIDF weighting. The seventh and eighth tasks are constructed from rcv1 and webspam datasets, respectively. Note that the webspam dataset is from *Pascal Large Scale Learning Challenge* [1]. See Table 2 for details

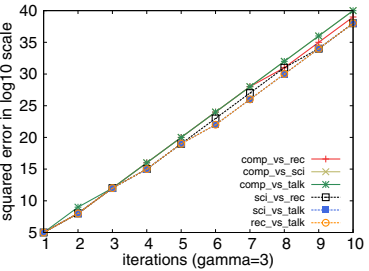
<sup>1</sup> <http://largescale.first.fraunhofer.de/instructions/>



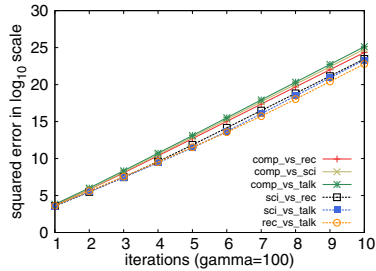
**Fig. 1.** Spectral Radius of Iteration Matrices of GraDes [5] ( $\gamma = 3$ )



**Fig. 2.** Spectral Radius of Iteration Matrices of GraDes ( $\gamma = 100$ )



**Fig. 3.** Potential function value of GraDes of the first 10 iterations ( $\gamma = 3$ )



**Fig. 4.** Potential function value of GraDes of the first 10 iterations ( $\gamma = 100$ )

of tasks. We compared 4 baselines with the proposed method, focusing on numerical stability, model sparsity and accuracy. The baselines are: GraDes [5], SMIDAS (Stochastic Mirror Descent Algorithm made Sparse), SCD (Stochastic Coordinate Descent) [12] and TG(Truncated Gradient) [8]. We next demonstrate the numerical problems of GraDes and SMIDAS, shown in Section 2.1 and 2.2.

### 4.2 Numerical Problems of Direct Gradient Descent

The GraDes algorithm attempts to find a sparse solution for the least squared error regression problem using  $\ell_0$  regularization. The accuracy of GraDes is not directly comparable to the proposed method due to the difference of the loss functions (GraDes uses squared loss while other three use logistic loss in our experiments), therefore we report the numerical problem of GraDes without comparing to other algorithms. As we analyzed in Section 2.1, for the direct gradient descent method to converge, one should keep the spectral radius of the iteration matrix  $M$  less than 1. For GraDes, though by sparsification using hard thresholding function (Eq. 6), the iteration can still diverge.

In practice, it is unrealistic to verify that the spectral radius of  $M$  is less than 1. This is restricted by the space and time complexity ( $M$  is a dense matrix in size of  $O(d^2)$ ). For instance, in one of the authors' machines with 2GB memory,

**Table 3.** Distribution and statistics of exponents

$p$ in Eq. (8)	$p = 2 \ln(d)$		$p = 0.5 \ln(d)$	
	range	within mean $\pm$ std	within	mean $\pm$ std
comp_vs_rec	0.095	-80.3 $\pm$ 23.6	1.000	-12.7 $\pm$ 7.2
comp_vs_sci	0.258	-69.8 $\pm$ 32.4	0.999	-15.4 $\pm$ 7.1
comp_vs_talk	0.282	-78.6 $\pm$ 43.1	0.996	-17.2 $\pm$ 9.1
sci_vs_rec	0.381	-63.6 $\pm$ 29.4	1.000	-14.3 $\pm$ 6.1
sci_vs_talk	0.273	-69.3 $\pm$ 26.9	1.000	-16.0 $\pm$ 5.9
rec_vs_talk	0.347	-68.4 $\pm$ 33.7	0.999	-15.8 $\pm$ 7.5
rcv1	0.012	-136.8 $\pm$ 32.6	0.998	-26.3 $\pm$ 6.5
webspam	NA	NA	0.976	-71.4 $\pm$ 22.1

MATLAB ran out of memory when computing the largest eigenvalue of a  $20000 \times 20000$  matrix. It costs even more if we have to compute the spectral radius of the thresholded iteration matrix  $D_{J^{(t)}}M$  to decide how to truncate  $\tilde{\mathbf{w}}^{(t)}$  to  $\mathbf{w}^{(t)}$ . For the above 20newsgroup tasks, we reduce the number of features to around 5000 during text preprocessing while keeping the number of examples the same. Then we compute the top eigenvalues of the iteration matrices  $D_{J^{(t)}}M$  at each iteration.  $|J^{(t)}|$  is set to 2000 and the elements of  $J^{(t)}$  are determined by the GraDes algorithm,  $D_{J^{(0)}}$  is the identity matrix. According to [1],  $\gamma$  is set to 3 and 100, respectively, Note that the learning rate is  $1/\gamma$ , thus we have two different learning rate settings. The spectral radii are plotted as a function of the number of iterations, under two learning rates, as shown in Fig. 1 and Fig. 2.

One can easily observe that the spectral radius is far more than 1, indicating a fast divergence speed of the solution. For example, in Fig. 1, before the first iteration, the iteration matrix  $M$  has a spectral radius at least  $7.5 \times 10^3$  (task “comp\_vs\_rec”, shown in the red line with plus “+”). Although in the first iteration, the spectral radii are reduced by the hard thresholding function, they are still in the scale of  $10^3$ . Moreover, the spectral radii increase as the algorithm proceeds. The radius of the iteration matrix for task “comp\_vs\_rec” increases from about  $6 \times 10^3$  at the first iteration to about  $7 \times 10^3$  at the 10-th iteration. All the other spectral radii increase to some extent and remain above  $10^3$ . Fig. 2 shows similar situations, the spectral radii are over 160. The large spectral radii of the iteration matrices indicate that the solution should go beyond the optimal solution and therefore increase the potential function. To show this, for each iteration of GraDes, we compute the the value of  $\Psi(\mathbf{w})$ , which is expected to be reduced by the algorithm. However, the potential function values also go up quickly. In Fig. 3 and Fig. 4, for the 6 20newsgroups tasks, we plot  $\Psi(\mathbf{w})$  in  $\log_{10}$  scale at each GraDes iteration under two settings of  $\gamma$ . The scales climb up linearly, indicating an exponential increase of  $\Psi(\mathbf{w})$ .

### 4.3 Numerical Problem of Mirror Descent Algorithm

We show the evidence of the numerical problem leading to the low accuracy of MDA. Typical *IEEE 745* double precision floating numbers have at most 52

bits in the mantissa. If  $p$  in Eq. (8) is set too large, then small difference in exponents of  $\theta_j$  would be greatly amplified in  $w_j$ . Adding two binary numerals with difference in exponent larger than 52 would cause the smaller one to be truncated. Usually, elements in one example of the data are normalized to be in the same scale, such as in the range of  $[0, 1]$ , the great disparity between exponents in the weights would cancel out parts of the data.

For each task, we trained a model with 40% of density (i.e. the percentage of non-zeros in the model, with 100% density we include all features in the model) using SMIDAS.  $p$  is set to  $2 \ln(d)$  and  $0.5 \ln(d)$ , respectively. For the last task, the density is set to 0.1% due to the large number of features (over 16 millions), and  $p \approx 33$  when  $p = 2 \ln(d)$ , this will obviously cause data truncation when computing inner products. Thus we only report results when  $p = 0.5 \ln(d)$  for this task. We calculated the exponents of  $w_j$  in 2-based numeric. Denote the largest exponent by  $e_m$ . We showed in Table 3 the ratios of exponents falling within  $[e_m - 51, e_m]$  (the column “within”). The columns “mean $\pm$ std” show the means and standard deviations of exponents. As we can see, for  $p = 2 \ln(d)$ , most of the  $w_j$  have their exponents out of  $[e_m - 51, e_m]$ . During prediction, the corresponding features of these elements in the data are totally truncated, therefore the models SMIDAS produce lack of sufficient discriminability and lead to poor performance (black lines with empty squares in Fig. (5)). The standard deviations are large, indicating a wide dynamic range of exponents. For  $p = 0.5 \ln(d)$ , SMIDAS approximates the normal gradient descent [6] and produces more reasonable models. Most of the exponents are in the range of  $[e_m - 51, e_m]$  and the dynamic ranges become much smaller. The performances go up dramatically, but still inferior to the proposed method (blue lines with filled squares in Fig. (5)).

#### 4.4 Sparsity and Performance Comparison

We focus on how the performance varies as a function of sparsity, and how numerical problem affects the MDA. The proposed method is compared with three scalable algorithms, SMIDAS, TG and SCD, all use logistic loss function. We randomly split all labeled data of each task into training, test and validation set (see Table 2). The training sets are further randomly shuffled (for task 1-7) or split (task 8) to create 10 copies of training data. For the proposed algorithm and TG, for each parameter setting, we trained 10 models using these 10 copies and calculated the average performance of the models on validation set. For SMIDAS and SCD, since they randomly shuffle the examples and features respectively, only one copy of training data of each task is needed. We chose the best parameters according to this average performance. Finally we reported the best models’ averaged accuracy on test sets. All algorithms require a regularization parameter  $\lambda$  to control the intensity of sparsification. With a larger  $\lambda$ , we expect more sparsity of the model. We tuned  $\lambda$  using 0.0001, 0.0005, 0.001, 0.01 and 0.1. For learning rate  $\eta$  needed in SMIDAS, TG and the proposed method, we varied it from 0.1 to 0.5 with step size 0.1. The proposed method requires

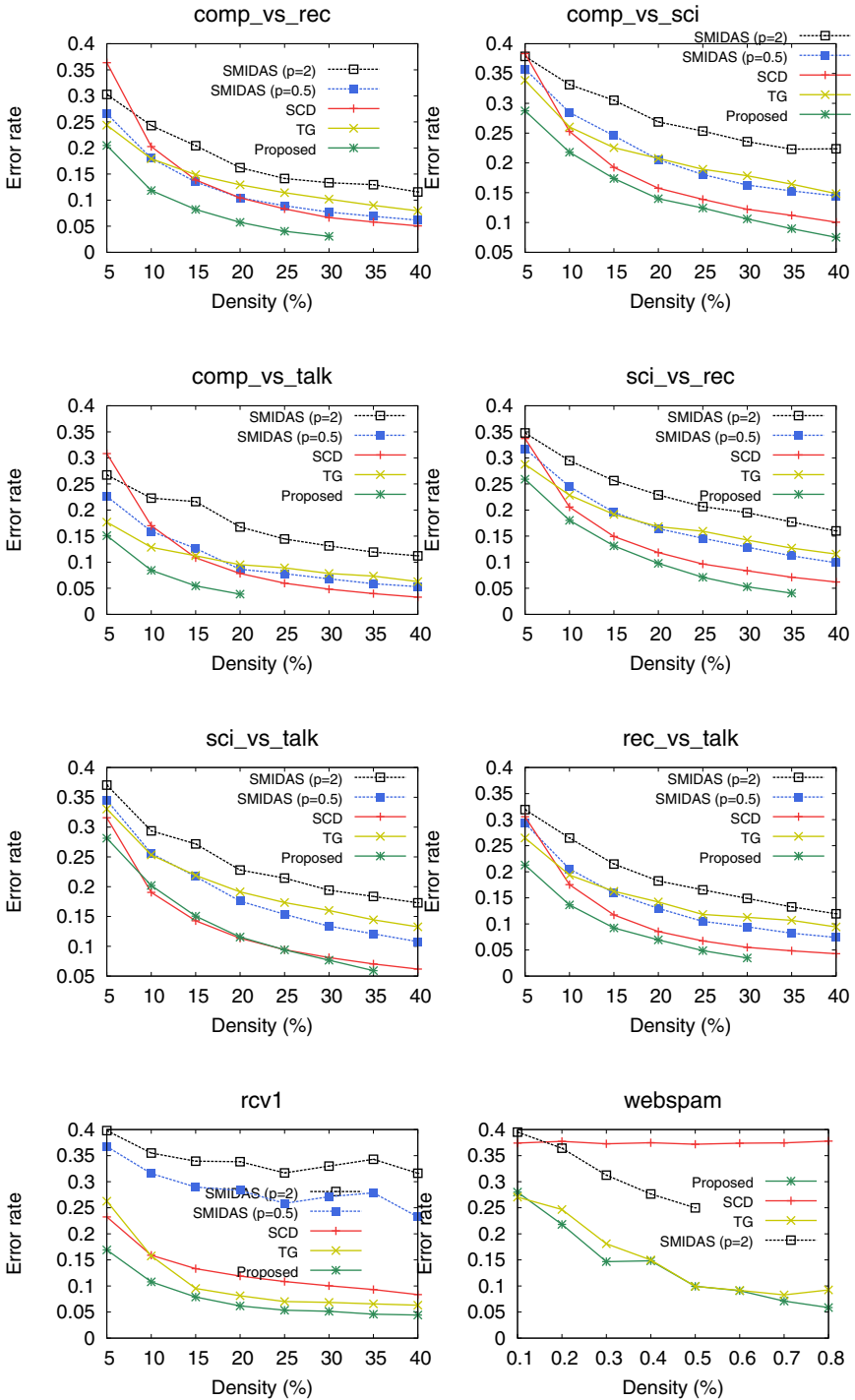


Fig. 5. Comparison of accuracy

one more parameter  $\tau$  determining the minimum margin above which to skip one instance.  $\tau$  was searched using 0.001, 0.01 and 0.1.

To compare the performance with sparsity restriction, we require the percentage of the non-zeros in the model (model density) to be at most 40% for task 1-7 and 0.1% for task 8. We recorded the error rates when the density reach pre-specified percentages. For all algorithms, we stopped running the program when they reach density upper bounds, or the maximal scans of training set (10 scans for all tasks). These error rates of each algorithms in 8 tasks are depicted in Fig. 5. In general, the proposed algorithm achieves lower error rate using fewer features compared with the other three. Particularly, in task 2, 7 and 8, though the density of the proposed algorithm went up to 40%, the errors are consistently lower than other methods. In the rest 5 tasks, the proposed method stop updating the model before the model is too dense, even when the maximum number of scans is reached. These not only demonstrated the convergence and generalizability of the proposed algorithm (see Section 3.2 and 3.3), but also the sparsity of the models it produces. Specifically, in task 1, the proposed algorithm converges before the density reaches 30%, with approximately 5% of error rate, which outperformed the remaining algorithms, even they used 40% of features. In task 3, the proposed method achieved 5% of error rate at density 20% which can only be obtained when SCD reached 40% of density, with other two have their error rates higher than that of SCD. Note that in task 1-7 SMIDAS with  $p = 2\ln(d)$  was beat by all other 3 methods and itself with  $p = 0.5\ln(d)$ . This showed that SMIDAS is not applicable for approximating the EG algorithm, especially when the dimensionality is high.

In task 1-6, when  $d$  and  $m$  differ by at most one order, TG converges slower than SCD. SCD adds features according to global information provided by all training examples, while TG works based on stochastic gradient. This is inaccurate compared with SCD. However, in task 8, when  $d$  are several order larger than  $m$  and sparser model are required, SCD fails to converge before reaching 40% of density. TG converges since it exploits more features than SCD. The proposed method takes the best of both. For any training example, it is always possible to reduce the loss following the gradient direction. As long as the current example does not increase the regret of online learning, we simply hold it out of the model. In this way, the proposed method achieves better generalization ability with a higher degree of sparsity.

## 5 Conclusions

We addressed several problems in existing sparse learning methods. Though RIP provides theoretical guarantee to recover the sparse solutions, it is mostly satisfied by designed matrices in compressive sensing, rather than data collected in the real world. Failing to meet RIP can cause matrix iteration-based gradient descent to diverge [5], while  $\ell_0$  constraint doesn't solve the problem. Second, though MDA using Bregman divergence enjoys fast convergence and approximates the EG algorithm, its update rules produce inaccurate models when dimensionality

is high. Lastly,  $\ell_1$  regularization is insufficient for finding an sparser solution, in several existing methods, density of model grows faster than necessary.

We have proposed to combine the perceptron algorithm with soft-thresholding. The algorithm converges with numerical error taken into account. We have also provided generalization error bound of the algorithm. Finally, the algorithm is highly scalable, data access can be distributed on a per-feature basis, making it more suitable for real-world applications. Experiments have shown that the proposed method outperformed 4 existing sparse learning algorithms in numerical stability, accuracy and model sparsity control. In one task with approximately 3.5GB of training data (16 million features, 50k examples), the proposed approach achieved 5.9% error rate using model with 0.8% density. But the best competing approach (TG for this dataset) can only get 8.2% error rate, using model with 0.7% density.

## References

1. Candes, E.J., Wakin, M.B.: An introduction to compressive sampling. *IEEE Signal Processing Magazine* 25(2), 21–30 (2008)
2. Donoho, D., Johnstone, I.M.: Adapting to unknown smoothness via wavelet shrinkage. *Journal of the American Statistical Association* 90, 1200–1224 (1995)
3. Duchi, J., Singer, Y.: Boosting with structural sparsity. In: *ICML*, p. 38 (2009)
4. Freund, Y., Schapire, R.E.: Large margin classification using the perceptron algorithm. In: *Machine Learning*, pp. 277–296 (1998)
5. Garg, R., Khandekar, R.: Gradient descent with sparsification: an iterative algorithm for sparse recovery with restricted isometry property. In: *ICML*, pp. 337–344 (2009)
6. Gentile, C., Littlestone, N.: The robustness of the p-norm algorithms. In: *Proceeding of 12th Annual Conference on Computer Learning Theory*, pp. 1–11. ACM Press, New York (1999)
7. Graepel, T., Herbrich, R.: From margin to sparsity. In: *Advances in Neural Information Processing Systems*, vol. 13, pp. 210–216. MIT Press, Cambridge (2001)
8. Langford, J., Li, L., Zhang, T.: Sparse online learning via truncated gradient. *Journal of Machine Learning Research* 10, 777–801 (2009)
9. Littlestone, N., Warmuth, M.: Relating data compression and learnability (1986)
10. Novikoff, A.B.: On convergence proofs for perceptrons. In: *Proceedings of the Symposium on the Mathematical Theory of Automata*, vol. 12, pp. 615–622 (1963)
11. Shalev-Shwartz, S., Srebro, N., Zhang, T.: Trading accuracy for sparsity. Technical report, Toyota Technological Institute at Chicago (2009)
12. Shwartz, S.S., Tewari, A.: Stochastic methods for  $\ell_1$  regularized loss minimization. In: *ICML*, pp. 929–936. ACM, New York (2009)
13. Zhang, T.: Adaptive forward-backward greedy algorithm for sparse learning with linear models. In: *NIPS*, pp. 1921–1928 (2008)
14. Zhao, P., Yu, B.: On model selection consistency of lasso. *Journal of Machine Learning Research* 7, 2541–2563 (2006)

# Fast Active Exploration for Link-Based Preference Learning Using Gaussian Processes

Zhao Xu<sup>1</sup>, Kristian Kersting<sup>1</sup>, and Thorsten Joachims<sup>2</sup>

<sup>1</sup> Fraunhofer IAIS, Schloss Birlinghoven, 53754 Sankt Augustin, Germany  
{zhao.xu,kristian.kersting}@iais.fraunhofer.de

<sup>2</sup> Department of Computer Science, Cornell University, Ithaca, NY 14853, USA  
tj@cs.cornell.edu

**Abstract.** In preference learning, the algorithm observes pairwise relative judgments (preference) between items as training data for learning an ordering of all items. This is an important learning problem for applications where absolute feedback is difficult to elicit, but pairwise judgments are readily available (e.g., via implicit feedback [13]). While it was already shown that active learning can effectively reduce the number of training pairs needed, the most successful existing algorithms cannot generalize over items or queries. Considering web search as an example, they would need to learn a separate relevance score for each document-query pair from scratch. To overcome this inefficiency, we propose a link-based active preference learning method based on Gaussian Processes (GPs) that incorporates dependency information from both feature-vector representations as well as relations. Specifically, to meet the requirement on computational efficiency of active exploration, we introduce a novel incremental update method that scales as well as the non-generalizing models. The proposed algorithm is evaluated on datasets for information retrieval, showing that it learns substantially faster than algorithms that cannot model dependencies.

## 1 Introduction

Preference learning is a natural and widely successful problem formulation for applications in search engines, information retrieval, and collaborative filtering [7,3,12,8]. The learning algorithm receives pairwise preferences that compare two entities, such as documents, webpages, products, songs etc. The goal is to learn a general ordering function that also ranks unobserved pairs correctly. Since collecting preference pairs is expensive (i.e. manual judgment effort, or presentation of inferior rankings for implicit feedback), designing learners that *actively* collect the most *informative* training preferences promises to reduce training cost. While some approaches for active learning with preferences exist (e.g., [21,6,19,27,26]), most assume the entities to be independent of each other and they do not take into account relations between the entities [1]. However, such link structures among entities are very informative [24,9,10,17]. For example, assume that there are  $n$  papers related to a query  $q$ , and these papers are linked together into



a network with co-author relations. Generally, the papers written by the same authors represent some similarity on research topics. If a user gives a judgement that a paper  $e_i$  is more relevant to the query  $q$  than another paper  $e_j$  ( $e_i \succ e_j$ ), then a co-authored paper  $e_{i'}$  of  $e_i$  is more likely to have similar relevance. In particular, it is likely to be preferred to  $e_j$  ( $e_{i'} \succ e_j$ ) as well. More generally, known preference information on papers propagates through the network providing useful evidence about unknown preferences and in turn decreasing our uncertainty about them. For active learning, this means that the search space is reduced.

Most of the active preference learning methods to date, however, have remained relatively agnostic to this rich structure. In fact, the most successful existing algorithms cannot model dependencies. For instance, Saar-Tsechansky and Provost [21] considered class-based ranking problems and proposed to use bootstrap samples of existing training data to examine the variance in the probability estimates for not-yet-labeled data. Brinker [2] proposed an SVM-based method which converts preference learning into a binary classification problem. [5,25] also ignore the link structure. In the model proposed by Radlinski and Joachims [19], the lack of dependencies manifests itself in a diagonal covariance matrix. Each entity is associated with a latent variable representing its utility (score) and all latent variables are independent of each other and follow Gaussian distributions. The entities are ranked according to these utilities: an entity  $e_i$  is ranked above another one  $e_j$  ( $e_i \succ e_j$ ), if and only if the utility of  $e_i$  is larger than that of  $e_j$ . For active exploration, Radlinski and Joachims propose to select an entity pair for which a preference label promises the largest expected reduction of uncertainty about the latent utilities. Specifically, for a user-defined loss function, the selection criterion optimizes the expected reduction of loss due to the variability of the utility estimates.

The method proposed in this paper overcomes the key limiting assumption of [19], namely that all document-query utilities have zero covariance. Specifically, we propose a link-based active preference learning method using Gaussian processes. It is based on the relational Gaussian process model for preference learning, called XPGP, recently introduced by Kersting and Xu [14]. Like in Radlinski and Joachims' model, associated with each entity is a continuous latent variable  $\xi_i$  that represents the latent utility (score) of the entity. The key difference from Radlinski and Joachims' model is that the latent utility (score) of each entity consists of two function values:  $f(x_i)$  and  $g(r_i)$ . With each entity  $e_i$  described by a vector of entity attributes  $x_i$ ,  $f(x_i)$  is a function value of these attributes. The  $\{f(x_1), f(x_2), \dots\}$  share a common Gaussian process prior  $GP_a$ . The term  $g(r_i)$  denotes a function value of entity relations  $r_i$  between  $e_i$  and other entities, and  $\{g(r_1), g(r_2), \dots\}$  share another common Gaussian process prior  $GP_r$ . The utility  $\xi_i$  of  $e_i$  is then modeled as the weighted sum of the two components:  $\xi_i = \omega_1 f(x_i) + \omega_2 g(r_i)$ . Unlike [19], it is obvious that this XPGP-based model exploits attribute and relational information in preference learning, enabling score propagate through the network.

---

**Algorithm 1.** The AXPGP algorithm for link-based active preference learning

---

**Input:**  $X$  (entity attributes),  $R$  (links),  $\mathcal{A}$  (the set of active pairs, one random pair at the beginning)

- 1  $m_0 = \mathbf{0}$ ;  $K_0$  is computed with Eq. (6).
- 2 **for**  $t = 1$  to  $T$  **do**
- 3     Approximate the likelihood distribution of the new active pair  $o_t$  (i.e., compute  $\tilde{\Sigma}_t$  and  $\tilde{\mu}_t$  with Eq. (22));
- 4     Update the posterior distribution of utilities for all entities given  $o_t$  (i.e., compute  $K_t$  and  $m_t$  with Eq. (24));
- 5     Compute expected loss for each candidate entity pair with Eq. (28) and pick the entity pair with the largest loss based on Eq. (29);
- 6     Label the chosen entity pair and add it to  $\mathcal{A}$ ;
- 7 **Output:**  $m_T$  (mean) and  $K_T$  (covariance matrix)

---

However, using XPGPs naively for active exploration would scale as  $\mathcal{O}(t^3)$ , where  $t$  is the number of actively selected points so far. Say we have  $t - 1$  active data points. After selecting an additional data point, we have to invert the new covariance matrix among all  $t$  data points, which takes  $\mathcal{O}(t^3)$  time. To meet the requirement on computational efficiency of active exploration, we propose an incremental update method for the XPGP model – the main contribution of the current paper. The incremental inference approach has scaling behavior comparable to the diagonal covariance method in [19]. We empirically evaluate the method using LETOR [18] benchmark datasets. The results show that AXPGP learns substantially faster than algorithms that cannot model dependencies.

The rest of the paper is organized as follows. We start off by introducing the link-based active preference learning method AXPGP in Sec. 2. We then discuss the model, incremental inference, and active learning. Before concluding, we present our experimental analysis.

## 2 Probabilistic Framework for Link-Based Active Preference Learning

The active exploration model we assume can be outlined as follows. Denote with  $\mathcal{T}$  the set of all possible entity preferences. Starting with an empty training set  $\mathcal{A} = \emptyset$ , we perform the following steps at each iteration: (1) a score (i.e. expected loss) is computed for all pairs in  $\mathcal{J} = \mathcal{T} \setminus \mathcal{A}$  based on the current distribution of the utilities; (2) the pair with the largest score is picked and added to the training set  $\mathcal{A}$  with its true preference; (3) the distribution of the utilities is updated based on the new  $\mathcal{A}$ . Note that the number of all possible entity preferences, i.e., the size of  $\mathcal{T}$  is  $\mathcal{O}(n^2)$  — much larger than the number  $n$  of entities.

The procedure simulates the interactions between users and information-retrieval systems and is summarized in Alg. 1. Before providing more details,

let us introduce some notations. We assume that there are (1) a set of  $n$  entities  $E = \{e_1, \dots, e_n\}$  with attributes  $X = \{x_i : x_i \in \mathbb{R}^D, i = 1, \dots, n\}$ , (2) relations  $R = \{r_{i,j} : i, j \in 1, \dots, n\}$  among the entities, and (3) a set of  $m$  observed preferences (i.e., pairwise rankings) among entities,  $O = \{e_{i_s} \succ e_{j_s} : s = 1, \dots, m; i_s, j_s \in 1, \dots, n\}$  ( $i_s$  and  $j_s$  are entities involved in the  $s$ -th observed preference). With  $r_i$ , we will denote all relations in which entity  $e_i$  participates.

## 2.1 Link-Based Preference Learning with Gaussian Processes

Gaussian process<sup>1</sup> (GP) models [20] are powerful nonparametric tools for Bayesian supervised learning. In contrast to other kernel machines such as support vector machines, GPs are probabilistic models, which means that they yield “error bars” on predictions and allow standard Bayesian model selection to be used. They generalize multivariate Gaussian distributions over finite dimensional vectors to infinite dimensionality. Specifically, a GP defines a distribution over functions, i.e. each draw from a Gaussian process is a function, and it is completely characterized by its mean function  $m(x) := [f(x)]$  and covariance (kernel) function  $C(x, x') := [(f(x) - m(x))(f(x') - m(x'))]$ . One attractive property of GPs is that any finite set of function values  $f = \{f(x_1), \dots, f(x_n)\}$  follow a multivariate Gaussian distribution so that mean and covariance can be computed based on the corresponding attribute vectors  $x = \{x_1, \dots, x_n\}$  with respect to the mean function and covariance functions.

Following the link-based GP (XPGP) model, the utility of each entity is modeled as a continuous latent variable, consisting of two latent function values  $f(x_i)$  and  $g(r_i)$  (shortened as  $f_i$  and  $g_i$ ).  $f(\cdot)$  and  $g(\cdot)$  are functions of attributes and links, respectively. We define GP priors for the attribute-wise and for the link-wise latent function values. Specifically, we assume an infinite number of latent function values  $\{f_1, f_2, \dots\}$  follow a Gaussian process prior with mean function  $m_a(x_i)$  and covariance function  $k_a(x_i, x_j)$ . Without loss of generality, we assume zero mean [20] so that the GP is completely specified by the covariance function only. Here we used the subscript  $a$  to emphasize that they are attribute-wise. In turn, any finite set of function values  $\{f_i : i = 1, \dots, n\}$  has a multivariate Gaussian distribution with mean zero and covariance matrix defined in terms of the covariance function of the GP. Formally, for the set of  $n$  entities, the prior distribution of the attribute-wise function values  $f = (f_1, \dots, f_n)^T$  can be represented as

$$P(f|X) = \frac{1}{(2\pi)^{\frac{n}{2}} |K|^{\frac{1}{2}}} \exp\left(-\frac{f^T K^{-1} f}{2}\right). \quad (1)$$

<sup>1</sup> Several GP models for preference learning have been proposed. For example Chu and Ghahramani [4] also considered the entity ranking problem (i.e. the setting discussed here) by introducing a novel likelihood function to express the entity-wise ordinal information. As another example, Guiver and Snelson [11] recently presented a sparse GP model for soft ranking problem for large-scale datasets. All previous GP models are reported to provide good performance on real-world datasets but they do not consider relational information.

Here,  $K$  denotes the  $n \times n$  covariance matrix whose  $ij$ -th entry is computed in terms of the covariance function with the corresponding attributes  $x_i$  and  $x_j$ . The covariance function can be any Mercer kernel function, e.g. squared exponential (SE)

$$k(x_i, x_j) = \kappa^2 \exp\left(-\frac{\rho^2}{2} \sum_d^D (x_{i,d} - x_{j,d})^2\right), \quad (2)$$

where  $\kappa$  and  $\rho$  are parameters of the covariance function, and  $x_{i,d}$  denotes the  $d$ -th dimension of the attribute  $x_i$ .

Similarly, we place a zero-mean GP over  $\{g_1, g_2, \dots\}$ . Again,  $\{g_i : i = 1, \dots, n\}$  follow a multivariate Gaussian distribution with mean zero and covariance matrix  $K_r$ . Since entities and links form a graph, we can naturally employ graph-based kernels to obtain the covariances, see e.g. [23]. The simplest graph kernel might be the regularized Laplacian

$$K_r = [\beta(\Delta + I/\iota^2)]^{-1}, \quad (3)$$

where  $\beta$  and  $\iota$  are two parameters of the graph kernel.  $\Delta$  denotes the combinatorial Laplacian, which is computed as  $\Delta = D - W$ , where  $W$  denotes the adjacency matrix of a weighted, undirected graph, i.e.,  $W_{i,j}$  is taken to be the weight associated with the edge between  $i$  and  $j$ .  $D$  is a diagonal matrix with entries  $d_{i,i} = \sum_j w_{i,j}$ . Formally, the prior distribution of the link-wise function values  $g = (g_1, \dots, g_n)^T$  is:  $P(g|R) = \mathcal{N}(0, K_r)$

$$= \frac{1}{(2\pi)^{\frac{n}{2}} |K_r|^{\frac{1}{2}}} \exp\left(-\frac{g^T K_r^{-1} g}{2}\right). \quad (4)$$

The utility  $\xi_i$  of an entity is a linear combination of the two function values:

$$\xi_i = \omega_1 f_i + \omega_2 g_i. \quad (5)$$

Since  $f_i$  and  $g_i$  have GP priors, their weighted sum  $\xi_i$  also follows a GP prior. For a finite number of entities, the prior is again reduced to a multivariate Gaussian distribution with mean zero and covariance matrix  $K$ , that is

$$P(\xi|X, R) = \mathcal{N}(0, K); \quad K = \omega_1^2 K_a + \omega_2^2 K_r. \quad (6)$$

After defining the prior of utilities, we now need to model how the utilities probabilistically decide the preferences, i.e. the likelihood distribution. For a preference  $o$  involving the entities  $e_i$  and  $e_j$ ,  $P(o_s|\xi_i, \xi_j)$  is defined as [4]

$$\int_{-\infty}^{\xi_i - \xi_j} \mathcal{N}(t|0, 1) dt \equiv \Phi(\xi_i - \xi_j). \quad (7)$$

The likelihood is a cumulative Gaussian. This encodes the natural assumption: *The larger the difference between the utilities of  $e_i$  and  $e_j$ , the more likely is that  $e_i$  is preferred to  $e_j$ .*

Finally, the joint probability of  $m$  observed preferences and  $n$  utilities  $\xi = \{\xi_1, \dots, \xi_n\}$  can be written as

$$P(\xi, O|X, R) = \mathcal{N}(\xi|0, K) \prod_{s=1}^m \Phi(\xi_{i_s} - \xi_{j_s}). \quad (8)$$

In this link-based GP ranking framework, the utilities are dependent on each other, meaning that knowing the preference of two entities can not only improve our utility predictions of the two entities, but also that of other entities which are not involved in the preference. Thus the ranking of a whole set of entities can be refined. How to diffuse the preference information among entities is decided by their dependencies. The probabilistic dependencies between them are captured with the covariance  $cov(\xi_i, \xi_j)$ , i.e. the entry  $(i, j)$  in the covariance matrix  $K$ , which measures to what extent the utilities change together. If two entities are inter-linked or have similar attributes, then their covariance  $cov(\xi_i, \xi_j)$  is high. Roughly speaking, if  $\xi_i$  is high, then so is  $\xi_j$ , and vice versa.

## 2.2 Incremental Inference Method for AXPGP

Assume that in the first  $t$  iterations, we collect  $t$  preferences  $\{o_1, \dots, o_t\}$  and have learned the posterior distribution  $P(\xi|X, R, o_1, \dots, o_t)$ . At iteration  $t + 1$ , a new preference  $o_{t+1}$  is collected, and we need to compute:

$$P(\xi|X, R, o_1, \dots, o_{t+1}) = \frac{P(\xi|X, R)P(o_1|\xi) \cdots P(o_t|\xi)P(o_{t+1}|\xi)}{P(o_1, \dots, o_t, o_{t+1}|X, R)}. \quad (9)$$

That means we have to completely re-compute the posterior based on all preferences collected so far. Then a question naturally arises: could we exploit the previous computation and only focus on the new preference. To meet the challenge, we develop an incremental inference method. Now the posterior distribution is represented as

$$P(\xi|X, R, o_1, \dots, o_{t+1}) = \frac{P(\xi|X, R, o_1, \dots, o_t)P(o_{t+1}|\xi)}{P(o_{t+1}|X, R, o_1, \dots, o_t)}, \quad (10)$$

where the first term in the numerator is obtained from the last iteration, and can be viewed as the *learned prior* of the current iteration. It is approximated with a Gaussian distribution with mean and covariance matrix denoted as  $m_t$  and  $K_t$ . Thus we have

$$P(\xi|X, R, o_1, \dots, o_t)P(o_{t+1}|\xi) \approx \mathcal{N}(\xi|m_t, K_t)\Phi(o_{t+1}|\xi).$$

Since the likelihood is not conjugate with the prior, an analytical solution is intractable. To solve the inference problem, the expectation propagation (EP) algorithm [16] is used. Namely, we use an unnormalized Gaussian distribution

$$\tilde{Z}_{t+1}\mathcal{N}(\xi_{i_{t+1}}, \xi_{j_{t+1}}|\tilde{\mu}_{t+1}, \tilde{\Sigma}_{t+1}) \quad (11)$$

to approximate the likelihood  $\Phi(o_{t+1}|\xi)$ . Note that (11) is a 2-dimensional Gaussian. Now the inference problem (10) is reduced to an optimization problem:

$$\begin{aligned} & \mathcal{N}(\xi_{i_{t+1}}, \xi_{j_{t+1}}|\mu_t, \Sigma_t)\Phi(o_{t+1}|\xi_{i_{t+1}}, \xi_{j_{t+1}}) \\ & \leftarrow \mathcal{N}(\xi_{i_{t+1}}, \xi_{j_{t+1}}|\mu_t, \Sigma_t)\tilde{Z}_{t+1}\mathcal{N}(\xi_{i_{t+1}}, \xi_{j_{t+1}}|\tilde{\mu}_{t+1}, \tilde{\Sigma}_{t+1}), \end{aligned} \tag{12}$$

where  $\mu_t$  and  $\Sigma_t$  denote the entries in  $m_t$  and  $K_t$  corresponding to the entities  $e_{i_{t+1}}$  and  $e_{j_{t+1}}$  involved in the preference  $o_{t+1}$ . The product on the right side equals to  $\mathcal{N}(\xi_{i_{t+1}}, \xi_{j_{t+1}}|\hat{\mu}, \hat{\Sigma}) \tilde{Z}_{t+1}/C$ , where

$$\begin{aligned} \hat{\mu} &= \hat{\Sigma}(\Sigma_t^{-1}\mu_t + \tilde{\Sigma}_{t+1}^{-1}\tilde{\mu}_{t+1}), \quad \hat{\Sigma} = (\Sigma_t^{-1} + \tilde{\Sigma}_{t+1}^{-1})^{-1}, \\ C &= 2\pi|\Sigma_t + \tilde{\Sigma}_{t+1}|^{\frac{1}{2}} \cdot \exp\left(\frac{1}{2}(\mu_t - \tilde{\mu}_{t+1})^T(\Sigma_t + \tilde{\Sigma}_{t+1})^{-1}(\mu_t - \tilde{\mu}_{t+1})\right). \end{aligned} \tag{13}$$

Thus the inference problem is reduced again: optimizing  $\hat{\mu}$  and  $\hat{\Sigma}$  to make the right side close to the left side, and then deriving  $\tilde{\mu}_{t+1}$  and  $\tilde{\Sigma}_{t+1}$  based on (13). To satisfy (12), we only need to match their first and second moments [20]. Since the product on the right side is an unnormalized Gaussian, we need to impose an additional condition, i.e. that the zero-th moments should also match. The zero-th moment of the left side is  $\int \mathcal{N}(\xi_{i_{t+1}}, \xi_{j_{t+1}}|\mu_t, \Sigma_t)\Phi(o_{t+1}|\xi_{i_{t+1}}, \xi_{j_{t+1}})$

$$= \int_{-\infty}^{\varrho^T \mu_t} \mathcal{N}(a|0, 1 + \varrho^T \Sigma_t \varrho) da = \Phi\left(\frac{\varrho^T \mu_t}{\sqrt{1 + \varrho^T \Sigma_t \varrho}}\right), \tag{14}$$

where  $\varrho$  denotes  $y_{t+1}[1, -1]^T$ . The term  $y_{t+1}$  is 1 if  $e_{i_{t+1}} \succ e_{j_{t+1}}$ , and  $-1$  otherwise. Since the zero-th moments of the two sides are equal, we have

$$\tilde{Z}_{t+1} = C \Phi\left(\frac{\varrho^T \mu_t}{\sqrt{1 + \varrho^T \Sigma_t \varrho}}\right). \tag{15}$$

The first moment of the left side is

$$\begin{aligned} & \frac{\partial}{\partial \mu_t} \int \mathcal{N}(\xi_{i_{t+1}}, \xi_{j_{t+1}}|\mu_t, \Sigma_t)\Phi(o_{t+1}|\xi_{i_{t+1}}, \xi_{j_{t+1}}) \\ & = \int \mathcal{N}(\xi_{i_{t+1}}, \xi_{j_{t+1}}|\mu_t, \Sigma_t)\Phi(\chi^T \varrho) (\chi - \mu_t)^T \Sigma_t^{-1}, \end{aligned} \tag{16}$$

where  $\chi^T$  denotes the vector  $[\xi_{i_{t+1}}, \xi_{j_{t+1}}]$ . The first moment of the right side is

$$\frac{\partial}{\partial \mu_t} \Phi\left(\frac{\varrho^T \mu_t}{\sqrt{1 + \varrho^T \Sigma_t \varrho}}\right) = \frac{\varrho^T \mu_t}{\sqrt{1 + \varrho^T \Sigma_t \varrho}} \mathcal{N}\left(\frac{\varrho^T \mu_t}{\sqrt{1 + \varrho^T \Sigma_t \varrho}}\right) \tag{17}$$

Making the two first moments equal, we obtain:

$$\begin{aligned} \hat{\mu} &= \mathbb{E}(\chi) = \mu_t + \frac{S}{Z\sqrt{1 + \varrho^T \Sigma_t \varrho}} \Sigma_t \varrho, \\ z &= \frac{\varrho^T \mu_t}{\sqrt{1 + \varrho^T \Sigma_t \varrho}}; \quad Z = \Phi(z); \quad S = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{z^2}{2}\right). \end{aligned} \tag{18}$$

Now we compute the second moment of the left side:

$$\begin{aligned} \frac{\partial^2}{\partial \mu_t^2} \int \mathcal{N}(\xi_{i_{t+1}}, \xi_{j_{t+1}} | \mu_t, \Sigma_t) \Phi(o_{t+1} | \xi_{i_{t+1}}, \xi_{j_{t+1}}) \\ = Z \mathbb{E} [\Sigma_t^{-1} (\chi - \mu_t) (\chi - \mu_t)^T \Sigma_t^{-1}] - Z \Sigma_t^{-1}. \end{aligned} \tag{19}$$

and that of the right side:

$$\frac{\partial^2}{\partial \mu_t^2} \Phi \left( \frac{\varrho^T \mu_t}{\sqrt{1 + \varrho^T \Sigma_t}} \right) = -\varrho \varrho^T \frac{zS}{1 + \varrho^T \Sigma_t \varrho}. \tag{20}$$

Since the two moments are equal, we obtain  $\mathbb{E}(\chi \chi^T) =$ :

$$\frac{zS}{(1 + \varrho^T \Sigma_t \varrho)Z} \Sigma_t \varrho \varrho^T \Sigma_t + \Sigma_t + \mathbb{E}(\chi) \mu_t^T + \mu_t \mathbb{E}(\chi^T) - \mu_t \mu_t^T. \tag{21}$$

Now, the second central moment  $\hat{\Sigma}$  can be computed with  $\mathbb{E}(\chi \chi^T) - \mathbb{E}(\chi) \mathbb{E}(\chi^T)$ . Substituting (18) and (21), we can get  $\hat{\Sigma}$ . In summary, the moment matching procedure yields the following equations to compute  $\tilde{\mu}_{t+1}$ ,  $\tilde{\Sigma}_{t+1}$ :

$$\begin{aligned} z &= \frac{\varrho^T \mu_t}{\sqrt{1 + \varrho^T \Sigma_t \varrho}}; & S &= \frac{1}{\sqrt{2\pi}} \exp(-\frac{z^2}{2}); & Z &= \Phi(z); \\ \hat{\mu} &= \mu_t + \frac{S \Sigma_t \varrho}{Z \sqrt{1 + \varrho^T \Sigma_t \varrho}}; & \hat{\Sigma} &= \Sigma_t - \frac{z S Z + S^2}{Z^2 (1 + \varrho^T \Sigma_t \varrho)} \Sigma_t \varrho \varrho^T \Sigma_t; \\ \tilde{\Sigma}_{t+1} &= (\hat{\Sigma}^{-1} - \Sigma_t^{-1})^{-1}; & \tilde{\mu}_{t+1} &= \tilde{\Sigma}_{t+1} (\hat{\Sigma}^{-1} \hat{\mu} - \Sigma_t^{-1} \mu_t). \end{aligned} \tag{22}$$

After getting the approximate likelihood, we can compute the posterior distribution of utilities. Since all entities are dependent on each other in the AXPGP algorithm, the update of the utility of one entity will change the utilities of all others. Thus, we need to compute the *new* distribution of all  $\xi_i$  at the iteration  $t + 1$ , i.e., the posterior distribution Eq. (10). The product of two Gaussian is an unnormalized Gaussian, so  $P(\xi | X, R, o_1, \dots, o_t, o_{t+1})$  is approximated with:

$$\begin{aligned} \frac{\mathcal{N}(\xi | m_t, K_t) \tilde{Z}_{t+1} \mathcal{N}(\xi_{i_{t+1}}, \xi_{j_{t+1}} | \tilde{\mu}_{t+1}, \tilde{\Sigma}_{t+1})}{P(o_{t+1} | X, R, o_1, \dots, o_t)} = \mathcal{N}(\xi | m_{t+1}, K_{t+1}) \\ K_{t+1} = (K_t^{-1} + [\hat{i}, \hat{j}] \tilde{\Sigma}_{t+1}^{-1} [\hat{i}, \hat{j}]^T)^{-1}; \\ m_{t+1} = K_{t+1} (K_t^{-1} m_t + [\hat{i}, \hat{j}] \tilde{\Sigma}_{t+1}^{-1} \tilde{\mu}_{t+1}). \end{aligned} \tag{23}$$

Here,  $[\hat{i}, \hat{j}]$  denotes the unit vectors, which is one at the entry  $i_{t+1}$  (resp.  $j_{t+1}$ ) and zero otherwise. Note that this approximate distribution is the learned prior of the next iteration  $t + 2$ . Given a new preference  $o_{t+2}$ , we can update the distribution  $P(\xi | X, R, o_1, \dots, o_t, o_{t+1}, o_{t+2})$  via (22) and (23) efficiently. Because the information from the historical data  $\{o_1, \dots, o_t\}$  propagates to the new iteration  $t + 1$  via the *learned prior*  $P(\xi | X, R, o_1, \dots, o_t)$ , the new iteration only needs to re-compute the distribution over utilities by one single new observation

$o_{t+1}$ , rather than to re-compute the distribution conditioned on all observations. Thus, it implements an incremental inference method.

The computation in (22) only involves operations on two-dimensional matrices which is cheap to compute. (23), however requires to compute the inverse of a  $n \times n$  matrix, which scales  $O(n^3)$ . The cost, however, can be reduced by using the Woodbury, Sherman & Morrison formula yielding a rank-two algorithm:

$$\begin{aligned} K_{t+1} &= K_t - ABA^T; \quad A = K_t[\hat{i}, \hat{j}]; \quad B^{-1} = \left( \tilde{\Sigma}_{t+1} + \Sigma_t \right) \\ m_{t+1} &= m_t - AB \left( \mu_t + \Sigma_t \tilde{\Sigma}_{t+1}^{-1} \tilde{\mu}_{t+1} \right) + A \tilde{\Sigma}_{t+1}^{-1} \tilde{\mu}_{t+1}, \end{aligned} \quad (24)$$

where  $A$  denotes the columns of  $K_t$  corresponding to the entities  $e_{i_{t+1}}$  and  $e_{j_{t+1}}$  involved in the preference  $o_{t+1}$ . In comparison with the Glicko [19] method, the information of the preference  $o_{t+1}$  is propagated to all the entities based on the dependencies between them. For example, the change of the covariance matrix is the product  $ABA^T$ , where  $B$  represents the change of covariances of the entities  $e_{i_{t+1}}$  and  $e_{j_{t+1}}$  introduced by the new observation.  $A$  represents the covariances between the two entities and all other unrelated entities to the new observation, by which the information propagates.

Here, we consider inference only in a transductive setting. However, the proposed method can also be extended to an inductive setting. The key challenge is getting the covariance between the new entities and the known ones based on relations. This can be addressed via several possible methods: Nyström [22] and similarity matching [28].

### 2.3 Active Exploration

There are two major characteristics, which distinguish the active exploration for preference learning from the ones for classification/regression: (1) preferences are pairwise (or list-wise) rather than entity-wise; (2) the entity property *utility* that decides the preference is latent and unobservable. Even if we observe the preferences of entity pairs, the utilities are still unknown. Now that we are armed with the posterior distribution of the latent utilities, we can use it to decide which is the next entity pair to query based on the expected loss.

Let  $\xi^*$  denote the current utility estimates used to produce the predicted ranking of items in the current iteration. Furthermore, let  $\mathcal{L}(\xi^*, \xi)$  be the loss incurred by the estimates  $\xi^*$  compared to the true utilities  $\xi$ . Since the true utilities are unknown and random, the value of the loss function is also random. We thus use the expectation of the loss functions with respect to the distribution of the true utilities. Suppose the loss function is pairwise decomposable (i.e. a sum of independent pairwise losses), then the total expected loss over all pairs can be factorized to

$$\mathbb{E}_{P(\xi|O)} [\mathcal{L}(\xi^*, \xi)] = \sum_{i=1}^N \sum_{j=i+1}^N \mathbb{E}_{P(\xi_i, \xi_j|O)} [\mathcal{L}(\xi_i^*, \xi_j^*, \xi_i, \xi_j)]. \quad (25)$$

Based on [19], the mode (i.e., mean  $\mu$ ) of their approximate posterior is used as the estimate  $\xi^*$ . [19] proves that for many loss functions, sorting entities by the



mode results in the ranking that minimizes the expected loss Eq. (25). Radlinski and Joachims [19] proposed the following loss function for information retrieval problems, since it (a) models misorderings, (b) takes into account that accuracy is more important at the top of the ranking, and (c) treats errors on pairs with almost equal utility different from pairs with large utility gap. In particular, for a pair of entities  $e_i$  and  $e_j$ ,

$$\mathcal{L}(\mu_i, \mu_j, \xi_i, \xi_j) = e^{-\gamma_{ij}} ((\mu_i - \mu_j) - (\xi_i - \xi_j))^2 \mathbf{1}_{\text{misordered}}, \quad (26)$$

where  $\gamma_{ij}$  denotes the minimum rank of  $e_i$  and  $e_j$  when all entities are ranked by their estimates. The term  $e^{-\gamma_{ij}}$  emphasizes the importance of entities ranked higher in the list. The higher the ranks of the two entities, the larger the loss if the estimated preference is wrong. The term  $\mathbf{1}_{\text{misordered}}$  is a function taking value one if  $(\mu_i - \mu_j)(\xi_i - \xi_j) < 0$ , and zero otherwise. Namely, we consider the difference between the estimations and the true values as loss only when it is substantial enough to influence the preference of the two entities. The expectation of the loss function (26) is computed as follows. Let  $\delta_{ij}$  denote  $\xi_i - \xi_j$ . Since  $\xi_i$  and  $\xi_j$  are Gaussian, their difference  $\delta_{ij}$  is still Gaussian,

$$\delta_{ij} \sim N(\delta_{ij} | \hat{\delta}_{ij}, \nu_{ij}^2), \quad (27)$$

with  $\hat{\delta}_{ij} = \mu_i - \mu_j$  and  $\nu_{ij}^2 = \text{var}(\xi_i) + \text{var}(\xi_j) - 2\text{cov}(\xi_i, \xi_j)$ , where  $\text{var}(x)$  and  $\text{cov}(x, y)$  denote the variance of  $x$  and the covariance between  $x$  and  $y$ . Permuting entities in each pair such that  $\hat{\delta}_{ij}$  is always negative, we have the expectation:

$$\mathbb{E}_{P(\xi_i, \xi_j | O)} [\mathcal{L}(\mu_i, \mu_j, \xi_i, \xi_j)] = e^{-\gamma_{ij}} \left[ \frac{\nu_{ij}^2}{2} \left( 1 + \text{erf} \left( \frac{\hat{\delta}_{ij}}{\sqrt{2\nu_{ij}^2}} \right) \right) - \frac{\hat{\delta}_{ij}\nu_{ij}}{\sqrt{2\pi}} \exp \left( \frac{-\hat{\delta}_{ij}^2}{2\nu_{ij}^2} \right) \right]. \quad (28)$$

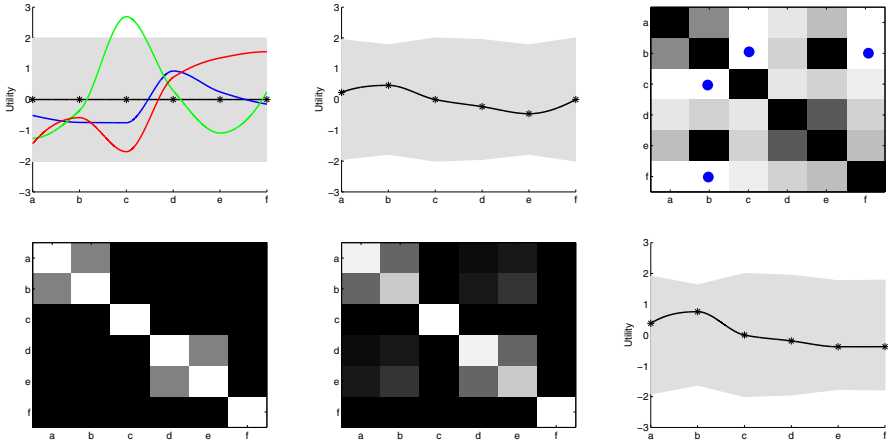
Radlinski and Joachims [19] introduced several exploration strategies, i.e., how to select the most informative entity pair based on the loss function. Here, we focus on the *largest expected loss pair (LELpair)*, which selects the entity pair  $e_i$  and  $e_j$  that has the largest pairwise expected loss contribution, i.e.,

$$\arg \max_{i \neq j} \mathbb{E}_{P(\xi_i, \xi_j | O)} [\mathcal{L}(\mu_i, \mu_j, \xi_i, \xi_j)] \quad (29)$$

We select the loss function (28) and the active strategy (29) due to their good performance and low computational cost. For more details on the theoretical and empirical analysis we refer to [19].

## 2.4 An Illustration

We have described the link-based active preference learning method AXPGP. To illustrate the principle behind AXPGP, we visualize the procedure with a simple example. Assume that there are six papers  $\{a, b, c, d, e, f\}$  ranked as  $a \succ b \succ c \succ d \succ e \succ f$ . We know that for  $(a, b)$  and  $(d, e)$  co-author relations exists. Intuitively, the utilities of  $a$  and  $b$  resp.  $d$  and  $e$  should be correlated.



**Fig. 1.** Illustration of the AXPGP algorithm. Top-left: Three utility functions drawn at random from a GP prior. The shaded area shows the 95% confidence region. Bottom-left: the prior covariance matrix. The smaller the covariance is, the darker the cell is. Top-middle: the posterior utilities learned with the prior and the observed preference  $b \succ e$ . The line is the posterior mean. Bottom-middle: The posterior covariance matrix. Top-right: The expected loss of each entity pair. The smaller the loss is, the darker the cell is. The blue points specify the pairs with the largest loss. Bottom-right: The updated posterior utilities given the actively collected preference  $b \succ f$ .

Formally, we compute the covariance matrix using the graph kernel (3), shown in Fig. 1 bottom-left, which just verifies our intuition. Since there is no preference information given, the expected utilities are the same for all entities (Fig. 1 top-left). Assume now that a user tells us that she prefers  $b$  to  $e$ . Our belief on the utilities changes, and we compute the posterior distribution based on (23). Fig. 1 top-middle clearly shows that the expected utilities of  $b$  and  $e$  reflect this. Additionally, the expected utilities of  $a$  and  $d$  also change. Since there is co-author relation between them, the preference information propagates through the network to other entities. Now, we compute the loss using (28) and select the next entity pair to ask for a label using (29). Fig. 1 top-right shows the loss for each entity pair. One can see that the pairs  $\{(a, d), (a, e), (b, d)\}$  have the smaller expected losses. The reason is again that  $a$  and  $d$  are related to  $b$  and  $e$ . The interesting fact, however, is the two isolated entities  $(c, f)$  do not give the largest loss. The reason is that, given loss function (26), it is more informative to find out whether  $c$  and  $f$  beat the current leader than learning about the relation between the two. Randomly selecting one pair  $(b, f)$  with the largest loss, the updated utilities are shown in Fig. 1 bottom-right. At this point, the ranking of the whole set of entities is already very close to the actual one except for the order of  $a$  and  $b$ . That is reasonable, because so far there is no explicit or implicit information available on their preference. The order between  $a$  and  $\{c, d, e, f\}$ , however, can already be accurately estimated.

### 3 Empirical Analysis

We evaluate the link-based active preference learning method AXPGP on two real-world datasets, namely OHSUMED and TREC in the LETOR repository [18]. The AXPGP method is compared with the Glicko [19] method to evaluate whether the relational model speeds up active learning. Furthermore, we compare computational efficiency, and evaluate in how far active learning using the AXPGP outperforms random sampling.

*Mean average precision (MAP)* is used as the performance measure [15]. MAP is defined as the average AP over multiple rankings,  $AP = \frac{1}{N^*} \sum_{n=1}^N prec(n)\delta(n)$ , where  $N^*$  denotes the number of relevant documents for the query.  $\delta(n)$  equals to one if the  $n$ -th document in the ranking is relevant, and zeros otherwise.  $prec(n) = \frac{1}{n} \sum_{i=1}^n \delta(i)$  is the precision after observing the first  $n$  documents in the ranking.

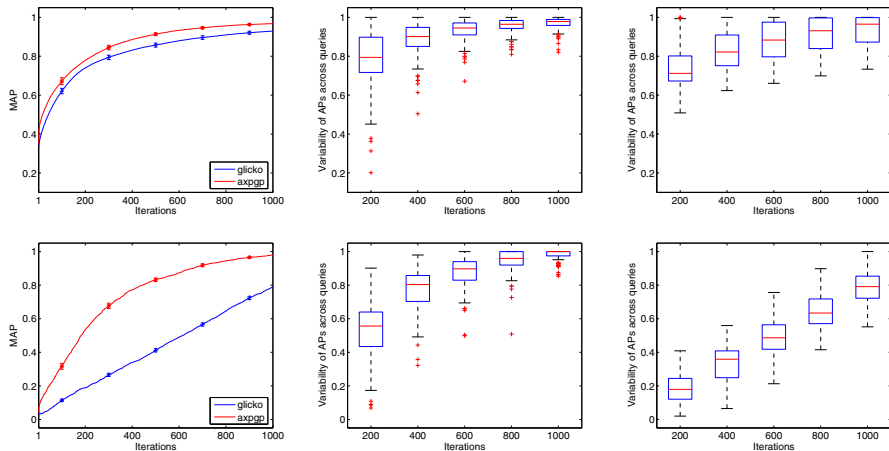
In each iteration of the following experiments, the algorithm asks for feedback on the preference of one entity pair,  $e_i$  and  $e_j$ . The feedback label is generated according to the order of the true utilities  $\xi_i$  and  $\xi_j$ . After observing a preference, the algorithm updates the model and we compute the MAP over all documents. To abstract from randomness in breaking ties of the LELpair criterion, each experiment is repeated 10 times. We used (2) and (3) to capture the feature and relational information.

#### 3.1 Data Description

LETOR is a benchmark dataset for learning to rank in information retrieval. We perform experiments on two datasets in LETOR: one is the OHSUMED dataset about medical articles, the other is the TREC dataset about .gov webpages.

In the OHSUMED dataset, each document consists of title, abstract, MeSH indexing terms, author, source, and publication type. There are 106 queries for which manual relevance judgments on the scale 'definitely relevant' (2), 'partially relevant' (1), and 'not relevant' (0) are available. We follow the experiment setup in [19]. In particular, since such coarse judgments are unrealistic in many real-world applications, we break ties by adding uniform noise in the range  $[-0.5, 0.5]$  to the true relevance degrees. Note that this preserves the relative order between definitely relevant (resp. partially relevant) documents and partially relevant (resp. not relevant) ones, but breaks ties within each relevance level. The dataset contains a feature vector for each query-document pair describing the match of the document to the query. Furthermore, we make use of the same relational information that was previously exploited in [17]. The relations are based on similarities, i.e. there is a weighted complete graph between documents. On average, there are about 152 documents per query.

The TREC dataset was originally collected for a special track on web information retrieval at TREC 2004. The goal of the track was to explore the performance of retrieval methods on large-scale data with hyperlinked structure such as the World Wide Web. The data was crawled from .gov domains in January, 2002. In total, there are 1,053,110 html documents with 11,164,829 hyperlinks.



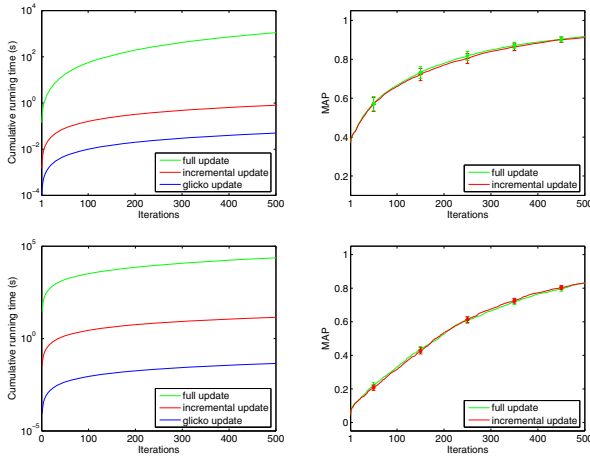
**Fig. 2.** MAP on the OHSUMED (top) and TREC (bottom) datasets when learning a separate model for each query. The average performance (with standard error) is given in the left-hand plot. The box plots show variability across queries for the AXPGP (middle) and Glicko (right).

To each query, documents were assigned labels by human experts. Each document has two possible states: relevant(1)/irrelevant(0). Qin *et al.* [18] processed the TREC dataset and turned it into a benchmark for information retrieval. Again, we add uniform noise in the range  $[-0.5, 0.5]$  to break ties and simulate a realistic situation in many real-world applications. On average, there were about 1000 documents per query which are linked with about 2387 hyperlinks.

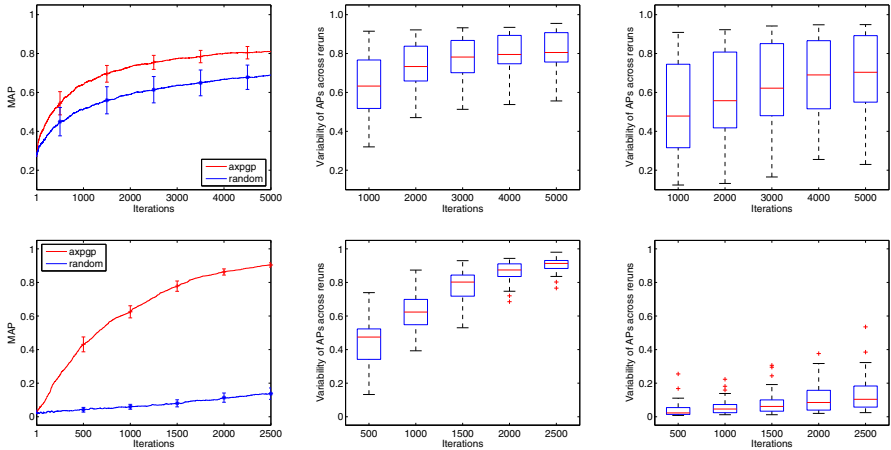
### 3.2 Experimental Results

*Can the AXPGP Exploit Cross-Document Information?* We first compare the AXPGP to Glicko following [19]. In particular, we evaluate whether the AXPGP can effectively transfer information between documents by using the features and the relations, while Glicko needs to learn each utility  $\xi_i$  individually. We run Glicko and AXPGP with the incremental updates separately for each individual query in the OSHUMED data. The MAP performance of the two methods averaged over all single-query experiments is given in the top-left plot of Fig. 2. The AXPGP does indeed learn significantly (according to both a Wilcoxon rank sum test and a t-test at each multiple of 100 iterations) faster than Glicko. Furthermore, the variability of the AP across multiple queries is substantially lower for the AXPGP (Fig. 2, top middle) than for Glicko (Fig. 2, right), especially for large numbers of iterations. The results on the TREC data, averaged over 10 randomly selected queries, show the similar trends.

*How Efficient is the AXPGP Compared to Glicko?* The left-hand plots of Fig. 3 show the average CPU time it takes to learn for a certain number of iterations. While Glicko is the fastest in absolute time, the incremental update shows the



**Fig. 3.** Computational efficiency of the AXPGP with full and incremental updates compared to Glicko (left). Comparison of MAP for AXPGP with full and incremental updates (right). The OSUMED (top) and TREC (bottom) results are respectively averaged over 30 and 10 randomly selected queries for efficiency reasons.



**Fig. 4.** Comparison of MAP (left), as well as variability when learning a single AXPGP model over multiple queries with active learning (middle) vs. random sampling (right). The top is the results on the OSUMED with a random sample of 10 queries. The bottom is that on the TREC data with a random sample of 5 queries.

same scaling and is slower only by a constant factor. The incremental update is substantially faster than the full update and shows better scaling behavior. It remains to investigate whether the incremental update is less accurate than the full update. The right-hand plots of Fig. 3 show that the MAP performance of the incremental update is not substantially lower than the MAP of the full update.

*How Large is the Benefit of Active Learning?* Finally, we evaluate in how far the active selection strategy improves on randomly selecting training pairs under the GP model. Here, we learn a single model over multiple queries using the feature-vector representation to define the covariance matrix of the GP. Fig. 4 (left) shows that the MAP grows significantly and substantially faster when using active learning. Furthermore, variability over multiple runs of the learning algorithms is much smaller for active learning (Fig. 4 middle) than for random sampling (Fig. 4 right).

## 4 Conclusions

In this paper, we explored how to integrate active learning into preference learning methods that can model dependencies from both feature-vector representations as well as relations. On real-world datasets for information retrieval, we have shown that actively learning a link-based Gaussian process ranking model is substantially faster than algorithms that cannot model dependencies. The key to the computational efficiency is a novel incremental update method that makes active exploration of link-based Gaussian process models essentially as fast as for the traditional models.

The most natural avenue for future work is the adaption of sparse Gaussian processes techniques to tackle large-scale datasets. In general, one should explore and develop active learning techniques for other relational models and tasks.

**Acknowledgments.** This work was funded in part by the Fraunhofer ATTRACT Fellowship "Statistical Relational Activity Mining" (STREAM), and by the German Federal Ministry of Economy and Technology (BMW) under the THESEUS project, as well as by the NSF Awards IIS-0905467 and IIS-0812091.

## References

1. Bilgic, M., Getoor, L.: Link-based active learning. In: Proceedings of NIPS Workshop on Analyzing Networks and Learning with Graphs (2009)
2. Brinker, K.: Active learning of label ranking functions. In: Proc. of ICML (2004)
3. Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., Hullender, G.: Learning to rank using gradient descent. In: ICML (2005)
4. Chu, W., Ghahramani, Z.: Preference learning with gaussian processes. In: ICML (2005)
5. Donmez, P., Carbonell, J.G.: Active sampling for rank learning via optimizing the area under the roc curve. In: Proc. ECIR (2009)
6. Brochu, N.d.E., Ghosh, A.: Active preference learning with discrete choice data. In: Advances in Neural Information Processing Systems, NIPS (2007)
7. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. JMLR 4, 933–969 (2003)
8. Fürnkranz, J., Hüllermeier, E.: Preference learning and ranking by pairwise comparison. In: Preference learning. Springer, Heidelberg (2010)
9. Geerts, F., Mannila, H., Terzi, E.: Relational link-based ranking. In: Proceedings of VLDB-04, pp. 552–563 (2004)

10. Getoor, L., Taskar, B. (eds.): An Introduction to Statistical Relational Learning. MIT Press, Cambridge (2007)
11. Guiver, J., Snelson, E.: Learning to rank with softrank and gaussian processes. In: SIGIR (2008)
12. Hüllermeier, E., Fürnkranz, J., Cheng, W., Brinker, K.: Label ranking by learning pairwise preferences. *Artificial Intelligence* 172(16-17), 1897–1916 (2008)
13. Joachims, T., Granka, L., Pan, B., Hembrooke, H., Radlinski, F., Gay, G.: Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Transactions on Information Systems (TOIS)* 25(2) (April 2007)
14. Kersting, K., Xu, Z.: Learning preferences with hidden common cause relations. In: *Proceeding of ECML09* (2009)
15. Manning, C.D., Raghavan, P., Schütze, H.: *Introduction to Information Retrieval*. Cambridge University Press, Cambridge (2008)
16. Minka, T.: A family of algorithms for approximate Bayesian inference. PhD thesis, MIT (2001)
17. Qin, T., Liu, T., Zhang, X., Wang, D., Xiong, W., Li, H.: Learning to rank relational objects and its application to web search. In: *WWW* (2008)
18. Qin, T., Liu, T.-Y., Xu, J., Li, H.: Letor: A benchmark collection for research on learning to rank for information retrieval. *Information Retrieval Journal* (2010)
19. Radlinski, F., Joachims, T.: Active exploration for learning rankings from click-through data. In: *Proc. SIGKDD*, pp. 570–579 (2007)
20. Rasmussen, C.E., Williams, C.K.I.: *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge (2006)
21. Saar-Tsechansky, M., Provost, F.: Active sampling for class probability estimation and ranking. *Machine Learning* 54(2), 153–178 (2004)
22. Schwaighofer, A., Tresp, V., Yu, K.: Learning gaussian process kernels via hierarchical bayes. In: *NIPS 17* (2005)
23. Smola, A.J., Kondor, I.: Kernels and regularization on graphs. In: *Annual Conference on Computational Learning Theory* (2003)
24. Taskar, B., Wong, M., Koller, D.: Learning on the test data: Leveraging unseen features. In: *ICML* (2003)
25. Yu, H.: Selective sampling for ranking with application to data retrieval. In: *Proc. SIGKDD* (2005)
26. Yue, Y., Broder, J., Kleinberg, R., Joachims, T.: The k-armed dueling bandits problem. In: *Conference on Learning Theory, COLT* (2009)
27. Yue, Y., Joachims, T.: Interactively optimizing information retrieval systems as a dueling bandits problem. In: *Proc. ICML* (2009)
28. Zhu, X., Lafferty, J., Ghahramani, Z.: Semi-supervised learning: From gaussian fields to gaussian processes. Technical Report CMU-CS-03-175, CMU (2003)

# Many-to-Many Graph Matching: A Continuous Relaxation Approach

Mikhail Zaslavskiy<sup>1,2,3,5,\*</sup>, Francis Bach<sup>4</sup>, and Jean-Philippe Vert<sup>1,2,3</sup>

<sup>1</sup> Center for Computational Biology, Mines ParisTech, Fontainebleau, France

<sup>2</sup> Institut Curie

<sup>3</sup> INSERM, U900, Paris, F-75248, France

<sup>4</sup> INRIA-WILLOW project, Ecole Normale Supérieure, Paris, France

<sup>5</sup> Center for Mathematical Morphology, Mines ParisTech, Fontainebleau, France

**Abstract.** Graphs provide an efficient tool for object representation in various machine learning applications. Once graph-based representations are constructed, an important question is how to compare graphs. This problem is often formulated as a graph matching problem where one seeks a mapping between vertices of two graphs which optimally aligns their structure. In the classical formulation of graph matching, only one-to-one correspondences between vertices are considered. However, in many applications, graphs cannot be matched perfectly and it is more interesting to consider many-to-many correspondences where clusters of vertices in one graph are matched to clusters of vertices in the other graph. In this paper, we formulate the many-to-many graph matching problem as a discrete optimization problem and propose two approximate algorithms based on alternative continuous relaxations of the combinatorial problem. We compare new methods with other existing methods on several benchmark datasets.

## 1 Introduction

The necessity to process data with complex structures has triggered the wide use of graph-based representation techniques in various applications domains. Graphs provide a flexible and efficient tools for data representation in computer vision (segmentation, contour and shock graphs), computational biology (biological networks), or chemoinformatics (representation of chemical compounds), to name just a few. A fundamental question when data are represented as graphs is to be able to *compare* graphs. In particular, it is important in many applications to be able to assess quantitatively the similarity between graphs (e.g., for applications in supervised or unsupervised classification), and to detect similar parts between graphs (e.g., for identification of interesting patterns in data).

Graph matching is one approach to perform these tasks. In graph matching, one tries to “align” two graphs by matching their vertices in such a way that most edges are conserved across matched vertices. Graph matching is useful both to assess the similarity between graphs (e.g., by checking how much the graphs differ after alignment), and to capture similar parts between graphs (e.g., by extracting connected sets of matched vertices).

---

\* Currently, Mikhail Zaslavskiy is with the bioinformatics group at Collectis S.A.



Classically, only one-to-one mappings are considered in graph matching. In other words, each vertex of the first graph can be matched to only one vertex of the second graph, and vice-versa<sup>1</sup>. This problem can be formulated as a discrete optimization problem, where one wishes to find a one-to-one matching which maximizes the number of conserved edges after alignment. This problem is NP-hard for general graphs, and remains impossible to solve *exactly* in practice for graphs with more than 30 vertices or so. Therefore much effort has been devoted to the development of approximate methods which are able to find a “good” solution in reasonable time. These methods can roughly be divided into two large classes. The first group consists of various local optimization algorithms on the set of permutation matrices, including  $A^*$ -Beam-search [2] and genetic algorithms. The second group consists in solving a continuous relaxation of the discrete optimization problem, such as the  $\ell_1$ -relaxation [3], the Path algorithm [4], various spectral relaxations [5,6,7,8,9] or power methods [10].

In practice, we are sometimes confronted with situations where the notion of one-to-one mapping is too restrictive, and where we would like to allow the possibility to match groups of vertices of the first graph to groups of vertices of the second graph. We call such a mapping *many-to-many*. For instance, in computer vision, the same parts of the same object may be represented by different numbers of vertices depending on the noise in the image or on the choice of object view, and it could be relevant to match together groups of vertices that represent the same part. From an algorithmic point of view, this problem has been much less investigated than the one-to-one matching problem. Some one-to-one matching methods based on local optimization over the set of permutation matrices have been extended to many-to-many matching, e.g., by considering the possibility to merge vertices and edges in the course of optimization [11,12]. Spectral methods have also been extended to deal with many-to-many matching by combining the idea of spectral decomposition of graph adjacency matrices with clustering methods [13,6]. However, while the spectral approach for one-to-one matching can be interpreted as a particular continuous relaxation of the discrete optimization problem [5], this interpretation is lost in the extension to many-to-many matching. In fact, we are not aware of a proper formulation of the many-to-many graph matching problem as an optimization problem solved by relaxation techniques.

Our main contribution is to propose such a formulation of the many-to-many graph matching problem as a discrete optimization problem, which generalizes the usual formulation for one-to-one graph matching (Section 2), and to present two approximate methods based on different continuous relaxations of the problem (Sections 3.1 and 3.2). In both cases, the relaxed problems are not convex, and we solve them approximately with a conditional gradient method. We also study different ways to map back the continuous solution of the relaxed problem into a many-to-many matching. We present experimental evidence in Section 5, both on simulated and simple real data, that this formulation provides a significant advantage over other one-to-one or many-to-many matching approaches.

---

<sup>1</sup> Note that with the introduction of dummy nodes, one may match a vertex of the first graph to *up to one* vertex of the second graph (see, e.g., [1]).

## 2 Many-to-Many Graph Matching as an Optimization Problem

In this section we derive a formulation of the many-to-many graph matching problem as a discrete optimization problem. We start by recalling the classical expression of the one-to-one matching problem as an optimization problem. We then show how to extend the one-to-one formulation to the case of many-to-one. Finally we describe how we can define many-to-many matchings via two many-to-one mappings.

**One-to-one graph matching.** Let  $G$  and  $H$  be two graphs with  $N$  vertices (if the graphs have different numbers of vertices, we can always add dummy nodes with no connection to the smallest graph). We also denote by  $G$  and  $H$  their respective adjacency matrices, i.e. square  $\{0, 1\}$ -valued matrices of size  $N \times N$  with element  $(i, j)$  equal to 1 if and only if there is an edge between vertex  $i$  and vertex  $j$ .

A one-to-one matching between  $G$  and  $H$  can formally be represented by a  $N \times N$  permutation matrix  $P$ , where  $P_{ij} = 1$  if the  $i$ -th vertex of graph  $G$  is matched to the  $j$ -th vertex of graph  $H$ , and  $P_{ij} = 0$  otherwise. Denoting by  $\|\cdot\|_F$  the Frobenius norm of matrices, defined as  $\|A\|_F^2 = \text{tr}A^T A = (\sum_i \sum_j A_{ij}^2)$ , we note that  $\|G - PHP^T\|_F^2$  is twice the number of edges which are not conserved in the matching defined by the permutation  $P$ . The one-to-one graph matching problem is therefore classically expressed as the following discrete optimization problem:

$$\begin{aligned} \min_P \|G - PHP^T\|_F^2 \text{ subject to } P \in \mathcal{P}_{oto}, \text{ with} \\ \mathcal{P}_{oto} = \{P \in \{0, 1\}^{N \times N}, P1_N = 1_N, P^T 1_N = 1_N\}, \end{aligned} \quad (1)$$

where  $1_N$  denotes the constant  $N$ -dimensional vector of all ones. We note that  $\mathcal{P}_{oto}$  simply represents the set of permutation matrices. The convex hull of this set is the set of doubly stochastic matrices, where the constraint  $P \in \{0, 1\}^{N \times N}$  is replaced by  $P \in [0, 1]^{N \times N}$ .

**From one-to-one to many-to-one.** Suppose now that  $G$  has more vertices than  $H$ , and that our goal is to find a matching that associates each vertex of  $H$  with one or more vertices of  $G$  in such a way that each vertex of  $G$  is matched to a vertex of  $H$ . We call such a matching *many-to-one* (or one-to-many if we invert the order of  $G$  and  $H$ ). The problem of finding an optimal many-to-one matching can be formulated as minimizing the same criterion as (1) but modifying the optimization set as follows:

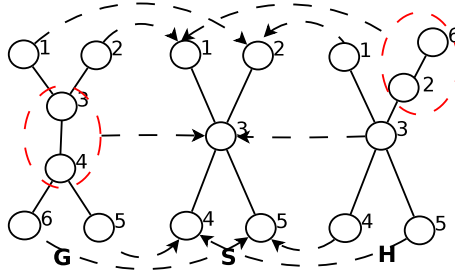
$$\begin{aligned} \mathcal{P}_{mto}(N_G, N_H) = \{P \in \{0, 1\}^{N_G \times N_H}, \\ P1_{N_H} = 1_{N_G}, P^T 1_{N_G} \leq k_{max} 1_{N_H}, P^T 1_{N_G} \geq 1_{N_H}\}, \end{aligned}$$

where  $N_G$  denotes the size of graph  $G$ ,  $N_H$  denotes the size of graph  $H$ , and  $k_{max}$  denotes an optional upper bound on the number of vertices that can be matched to a single vertex. As opposed to the one-to-one matching case, each column sum of  $P$  is allowed to be larger than one, and the non-zero elements of the  $j$ -th column of  $P$  corresponds to the vertices of graph  $G$  which are matched to the  $j$ -th vertex of  $H$ .

Most of existing continuous relaxation techniques may be adopted for many-to-one matching. For example, [8] describes how spectral relaxation methods may be used in the case of many-to-one matching. Other techniques like convex relaxation [4] may be

used as well since the convex hull of  $\mathcal{P}_{mto}$  is also obtained by relaxing the constraint  $P \in \{0, 1\}^{N_G \times N_H}$  to  $P \in [0, 1]^{N_G \times N_H}$ .

**From many-to-one to many-to-many.** Now to match two graphs  $G$  and  $H$  under many-to-many constraints we proceed as if we matched these two graphs to a virtual graph  $S$  under many-to-one constraints, minimizing the difference between the transformed graph obtained from  $G$  and the transformed graph obtained from  $H$ . The idea of many-to-many matching as a double many-to-one matching is illustrated in Figure 1. Graph  $S$  (assumed to have  $L$  vertices) represents the graph of matched vertex clusters.



**Fig. 1.** Many-to-many matching between  $G$  and  $H$  via many-to-one matching of both graphs to a virtual graph  $S$

Each vertex of  $S$  corresponds to a group of vertices of  $G$  and a group of vertices of  $H$  matched to each other. For example, in Figure 1 vertices  $g_3$  (vertex **3** of  $G$ ),  $g_4$  and  $h_3$  are matched to the same vertex  $s_3$ ; it means that in the final *many-to-many* matching between  $G$  and  $H$ ,  $g_3$  and  $g_4$  will be matched to  $h_3$ . Let  $P_1 \in \mathcal{P}_{mto}(L, N_G)$  denote a many-to-one matching  $G \rightarrow S$ , and  $P_2 \in \mathcal{P}_{mto}(L, N_H)$  a many-to one matching  $H \rightarrow S$ ; we propose to formulate the many-to-many graph matching problem as an optimization problem where we seek  $S$ ,  $P_1$  and  $P_2$  which minimize the difference between  $S$  and  $P_1GP_1^\top$  and between  $S$  and  $P_2HP_2^\top$

$$\min_{P_1, P_2, S} \|P_1GP_1^\top - S\|_F^2 + \|S - P_2HP_2^\top\|_F^2. \tag{2}$$

Note that if we know  $P_1$  and  $P_2$ , then the optimal  $S$  is just  $\frac{1}{2}(P_1GP_1^\top + P_2HP_2^\top)$  (point in the middle between  $P_1GP_1^\top$  and  $P_2HP_2^\top$ ). Plugging this expression in (2) we obtain the following objective function for the many-to-many graph matching problem

$$F(P_1, P_2) = \|P_1GP_1^\top - P_2HP_2^\top\|_F^2, \tag{3}$$

where  $P_1 \in \mathcal{P}_{mto}(L, N_G)$  and  $P_2 \in \mathcal{P}_{mto}(L, N_H)$  denote two many-to-one mappings. The objective function (3) is similar to the objective function for the one-to-one case (1). In (1), we seek a permutation which makes the second graph  $H$  as similar as possible to  $G$ . In (3), we seek *combinations of merges and permutations* which makes  $G$  and  $H$  as similar as possible to each other. The only difference between both formulations is that in the many-to-many case we add the merging operation. Given matrices  $P_1$  and  $P_2$ , it is easy to retrieve the many-to-many matching between  $G$  and  $H$  by considering

$$M = P_1 P_2^\top .$$

Indeed,  $M$  is a  $N_G \times N_H$  binary matrix such that  $M_{ij} = 1$  if and only if  $g_i$  is matched to  $h_j$ .

There are two slightly different ways of defining the set of matrices over which (3) is minimized. We can fix in advance the number of matching clusters  $L$ , which corresponds to the size of  $S$ , in which case the optimization set is  $P_1 \in \mathcal{P}_{mto}(L, N_G)$  and  $P_2 \in \mathcal{P}_{mto}(L, N_H)$ . An alternative way which we follow in the paper is to remove the constraint  $P_1 \mathbf{1}_{N_G} \geq \mathbf{1}_L$  from the definition of  $\mathcal{P}_{mto}(L, N_G)$ , in this case the method estimates itself the number of matching clusters (number of rows with non-zero sum). Finally, we thus formulate the many-to-many graph matching problem as follows:

$$\begin{aligned} & \min_{P_1, P_2} \|P_1 G P_1^\top - P_2 H P_2^\top\|_F^2 \quad \text{subject to} \\ & P_1 \in \{0, 1\}^{N_K \times N_G}, P_1 \mathbf{1}_{N_G} \leq k_{max} \mathbf{1}_{N_K}, P_1^\top \mathbf{1}_{N_K} = \mathbf{1}_{N_G}, \\ & P_2 \in \{0, 1\}^{N_K \times N_H}, P_2 \mathbf{1}_{N_H} \leq k_{max} \mathbf{1}_{N_K}, P_2^\top \mathbf{1}_{N_K} = \mathbf{1}_{N_H}, \end{aligned} \quad (4)$$

where  $N_K = \min(N_G, N_H)$  represents the maximal number of matching clusters. Note that  $N_K$  is only an upper bound on the number of matching clusters, since it can not exceed the size of the smallest graph. On the other hand some of the columns of  $P_1$  and  $P_2$  may be empty, meaning that the corresponding clusters do not contain vertices, i.e., that these clusters do not exist. This formulation is in fact valid for many kinds of graphs, in particular graphs may be directed (with asymmetric adjacency matrices), have edge weights (with real-valued adjacency matrices), and self-loops (with non-zero diagonal elements in the adjacency matrices). We also describe in Section 3.1 how this formulation can be modified to include information about vertex labels, which are important for machine learning applications.

### 3 Continuous Relaxations of the Many-to-Many Graph Matching Problem

The many-to-many graph matching problem (4) is a hard discrete optimization problem. We therefore need an approximate method to solve it in practice.

#### 3.1 Method 1: Gradient Descent

In this section we propose an algorithm based on a continuous relaxation of (4). For that purpose we propose to replace the binary constraints  $P_1 \in \{0, 1\}^{N_K \times N_G}$ ,  $P_2 \in \{0, 1\}^{N_K \times N_H}$  by continuous constraints  $P_1 \in [0, 1]^{N_K \times N_G}$ ,  $P_2 \in [0, 1]^{N_K \times N_H}$ . Let  $\mathcal{K}$  denote the new continuous optimization set

$$\begin{aligned} \mathcal{K} &= \mathcal{K}_1 \times \mathcal{K}_2, \text{ where} \\ \mathcal{K}_1 &= \{P_1 \in [0, \mathbf{1}]^{N_K \times N_G}, P_1 \mathbf{1}_{N_G} \leq k_{max} \mathbf{1}_{N_K}, P_1^\top \mathbf{1}_{N_K} = \mathbf{1}_{N_G}\}, \\ \mathcal{K}_2 &= \{P_2 \in [0, \mathbf{1}]^{N_K \times N_H}, P_2 \mathbf{1}_{N_H} \leq k_{max} \mathbf{1}_{N_K}, P_2^\top \mathbf{1}_{N_K} = \mathbf{1}_{N_H}\}, \end{aligned} \quad (5)$$

To solve the relaxed optimization problem we propose to use the following version of the conditional gradient (a.k.a. Frank-Wolfe method [14]):

- Input: initial values  $P_1^0$  and  $P_2^0$ ,  $t = 0$ ,
- Do
  1. compute  $\nabla F(P_1^t, P_2^t)$
  2. find the minimum of  $\nabla F(P_1^t, P_2^t)^\top (P_1, P_2)$  w.r.t.  $(P_1, P_2)$  i.e. over  $\mathcal{K}$
  3. perform line search in the direction of the optimum found in Step 2, assign the result to  $P_1^{t+1}, P_2^{t+1}$ ,  $t = t + 1$
- Until  $|\Delta F| + \|\Delta P_1\|_F + \|\Delta P_2\|_F < \varepsilon$
- Output:  $P_1^t, P_2^t$ .

Step 2 consists in the minimization of a linear function over a convex optimization set. This problem can be solved by a generic linear programming solver in  $O((N_G + N_H)N_K)^{3.5}$ . The minimum of the gradient function is one of the extreme points of the continuous optimization set. Since our ultimate objective is to minimize  $F(P_1, P_2)$  over the discrete optimization set (4) it would be better to move towards one of the points of (4) and not just any extreme point of  $\mathcal{K}$ . The good news is that due to the special structure of the optimization set, the gradient minimization can be solved in  $O(\max(k_{max}N_K)^3)$  by reformulating  $\min_P \nabla F(P_1^t, P_2^t)^\top (P_1, P_2)$  as a linear assignment problem (see Appendix A). We then have to solve a linear assignment problem for a  $(N_G + N_H) \times k_{max} \min(N_H, N_G)$  matrix, which can be done efficiently by the Hungarian algorithm [15].

The solution of the line search step can be found in closed form since the objective function is a polynomial of the fourth order.

The conditional descent algorithm converges to a stationary point of (4) [14]. Because of the non-convex nature of the objective function, we can only hope to reach a local minimum (or more generally a stationary point) and it is important to have a good initialization. In our experiments we found that a good choice is the fixed “uniform” initialization, where we initialize  $P_1$  by  $\frac{1}{N_K} 1_{N_G} 1_{N_H}^\top$  and  $P_2$  by the identity matrix  $I$ . Another option would be to use a convex relaxation of one-to-one matching [4].

Algorithm complexity is mainly defined by two parameters:

$$N = \max(k_{max} \min(N_G, N_H), N_G + N_H) \text{ and } \varepsilon.$$

In general the number of iterations of the gradient descent scales as  $O(\frac{N}{\varepsilon})$  where  $\kappa$  is the condition number of the Hessian matrix describing the objective function near a local minima [14].  $N$  has no direct influence on the number of iterations, but it defines the cost of one iteration, i.e., the complexity of the Hungarian algorithm  $O(N^3)$ .

## Projection

Once we have reached a local optimum of the relaxed optimization problem, we still need to project  $P_1$  and  $P_2$  to the set of matrices with values in  $\{0, 1\}$  rather than in  $[0, 1]$ . Several alternatives can be considered. A first idea is to use the columns of  $P_1$  and  $P_2$  to define a similarity measure between the vertices of both graphs, e.g., by computing the dot products between columns. Indeed, the more similar the columns

corresponding to two vertices, the more likely these vertices are to be matched if they are from different graphs, or merged if they are from the same graph. Therefore a first strategy is to run a clustering algorithm (e.g., K-means or spectral clustering) on the column vectors of the concatenated matrix  $(P_1, P_2)$  and then use the resulting clustering to construct the final many-to-many graph matching.

An alternative to clustering is an incremental projection or forward selection projection, which uses the matching objective function at every step. Once  $P_1$  and  $P_2$  are obtained from the continuous relaxation, we take the pair of vertices  $(g, h)$  from the union of the graphs having the most similar column vectors in  $(P_1, P_2)$ . We then re-run the continuous relaxation with the new (linear) constraint that these two vertices remain matched. We then go on and find the most similar pair of vertices from the constrained continuous solution. This greedy scheme can be iterated until all vertices are matched.

In our experiments, the second approach produced better results. This is mainly due to the fact that when we just run a clustering algorithm we do not use the objective function, while when we use incremental projection we adapt column vectors of unmatched vertices according to earlier established matchings.

**Neighbor merging.** In many cases, it can be interesting to favor the merging of neighboring vertices, as opposed to merging of any sets of vertices. To that end we propose the following modification to (4):

$$F_N(P_1, P_2) = F(P_1, P_2) - \text{tr}G^\top P_1^\top P_1 - \text{tr}H^\top P_2^\top P_2. \quad (6)$$

The matrix product  $P_1^\top P_1$  is a  $N_G \times N_G$  matrix, with  $(i, j)$ -th entry equal to 1 if  $i$  and  $j$  are merged into the same cluster. Therefore, the new components in the objective function represent the number of pairs of adjacent vertices merged together in  $G$  and  $H$ , respectively.

**Local similarities.** Like the one-to-one formulation, we can easily modify the many-to-many graph matching formulation to include information on vertex pairwise similarities by modifying the objective function as follows:

$$F_\lambda(P_1, P_2) = (1 - \lambda)F(P_1, P_2) + \lambda \text{tr}C^\top P_1^\top P_2, \quad (7)$$

where the matrix  $C \in \mathbb{R}^{N_G \times N_H}$  is a matrix of local dissimilarities between graph vertices, and parameter  $\lambda$  controls the relative impact of information on graph vertices and information on graph structures.

The new objective function is again a polynomial of the fourth order, so our algorithm may still be used directly without any additional modifications.

### 3.2 Method 2: SDP Relaxation

The second method consists in a relaxation of (4) to a quadratic semidefinite programming (SDP) problem.

First, we rewrite the objective function of (4) in an alternative form

$$\begin{aligned}
& \|P_1 G P_1^T - P_2 H P_2^T\|_F^2 = \\
& \text{tr} P_1 G^T \underbrace{P_1^T P_1}_{M_1} G P_1^T + \text{tr} P_2 H^T \underbrace{P_2^T P_2}_{M_2} H P_2^T - 2 \text{tr} P_1 G^T \underbrace{P_1^T P_2}_{M_{12}} H P_2^T = \\
& \text{tr} M_1 G^T M_1 G + \text{tr} M_2 H^T M_2 H - 2 \text{tr} M_{21} G^T M_{12} H = \\
& \text{tr} \left[ \underbrace{\begin{pmatrix} M_1 & M_{12} \\ M_{21} & M_2 \end{pmatrix}}_M \underbrace{\begin{pmatrix} G^T & 0 \\ 0 & -H^T \end{pmatrix}}_{A^T} \underbrace{\begin{pmatrix} M_1 & M_{12} \\ M_{21} & M_2 \end{pmatrix}}_M \underbrace{\begin{pmatrix} G & 0 \\ 0 & -H \end{pmatrix}}_A \right] = \\
& \text{tr} M A^T M A = \text{vec}(M)(A^T \otimes A)\text{vec}(M).
\end{aligned} \tag{8}$$

Let  $F(M)$  denote our new objective function  $\text{vec}(M)(A^T \otimes A)\text{vec}(M)$ , now we have to minimize the quadratic function  $F(M)$  over the discrete set  $\mathcal{M}$  of binary matrices with a special structure. Since matrix  $M$  is a positive-semidefinite matrix (indeed,  $M$  can be expressed as  $P^T P$  where  $P$  is the matrix made of  $P_1$  and  $P_2$  stacked one below another), we can relax the optimization problem  $\min_{M \in \mathcal{M}} F(M)$  to the minimization of a quadratic function over the convex set of positive-semidefinite matrices

$$\min_{M \succeq 0} F(M). \tag{9}$$

Therefore the second method consists in the running of the Frank-Wolfe algorithm with an SDP solver to compute conditional gradient and further projection of the produced solution on  $\mathcal{M}$ .

Here again we can run a clustering algorithm using the output of the Frank-Wolfe algorithm (a real-valued positive-semidefinite matrix  $M$ ) as a similarity matrix between vertices of two graphs, or use the incremental projection strategy fixing on each step the most probable matching and adjusting the optimum given the new constraint.

To favor the neighbor merging and to introduce local vertex similarities, we can use the same terms that we used before. Note,  $P_1 P_1^T = M_1$ ,  $P_2 P_2^T = M_2$  and  $P_1 P_2^T = M_{12}$ , therefore new terms in (6) and (7) can be expressed as linear functions of  $M$

$$\text{tr} G^T M_1 + \text{tr} H^T M_2 \text{ and } \text{tr} C^T M_{12}$$

The addition of these new terms to the objective function  $F(P)$  does not change its structure,  $F(P)$  stays a quadratic function, so we can use the same minimization procedure.

A serious drawback of the ‘‘SDP’’ approach lies in its complexity. The complexity of a generic SDP solver scales as  $O(m^{2.5})$  [16] where  $m$  is the number of variables of a given ‘‘SDP’’ problem. In our case,  $m$  is equal to  $(N_G + N_H)^2$  which means that the complexity of the gradient minimization step is at least  $O((N_G + N_H)^5)$ . Hence, it is almost impossible to run the ‘‘SDP’’ method on graphs with more than 30 vertices. To overcome this problem we propose to use an early stopping rule i.e. replace the exact SDP solution by an approximate one.

## 4 Related Methods

There exist two major groups of methods for many-to-many graph matching, which we briefly describe in this section. The first one consists of local search algorithms, generally used in the context of the graph edit distance, while the second one is composed of variants of the spectral approach.

**Local search algorithms.** Examples of this kind of approach are given in [11] and [12]. In the classical formulation of the graph edit distance, the set of graph edit operations consists of deletion, insertion and substitution of vertices and edges. Each operation has an associated cost, and the objective is to find a sequence of operations with the lowest total cost transforming one graph into another. In the case of many-to-many graph matching, this set of operations is completed by merging (and splitting if necessary) operations. Since the estimation of the optimal sequence is a hard combinatorial problem, approximate methods such as beam search [2] as well as other examples of best-first, breadth-first and depth-first searches are used.

**Spectral approach.** Caelli and Kosinov [6] discuss how spectral matching may be used for many-to-many graph matching. Their algorithm is similar to the Umeyama method [5] but instead of one-to-one correspondences, they search a many-to-many mapping by running a clustering algorithm. In the first step, the spectral decomposition of graph adjacency matrices is considered

$$G = V_G \Lambda_G V_G^\top, \quad H = V_H \Lambda_H V_H^\top. \quad (10)$$

Rows of eigenvector matrices  $V_G$  and  $V_H$  are interpreted as spectral coordinates of graph vertices. Then vertices having similar spectral coordinates are clustered together by a clustering algorithm, and vertices grouped in the same cluster are considered to be matched.

Another example of spectral approach is given in [13] where, roughly speaking, the adjacency matrix is replaced by the matrix of shortest path distances, and then spectral decomposition with further clustering is used.

## 5 Experiments

In this section we compare the new methods proposed in this paper with existing techniques (beam-search and spectral approach). We thus test four competitive approaches in several experiments: beam-search “Beam” (A\*-beam search from [2]), the spectral approach “Spec” [6] and two new approaches: the gradient descent method “Grad” (from Section 3.1), and the “SDP” relaxation (Section 3.2). All four algorithms are implemented in matlab and are available at <http://cbio.ensmp.fr/graphm/mtmgm.html>. We use SeDuMi (<http://sedumi.ie.lehigh.edu/>) to perform the gradient minimization in the “SDP” method.

### 5.1 Synthetic Examples

In this section, we compare the four many-to-many graph matching algorithms on pairs of randomly generated graphs with similar structures. We generate graphs according to



the following procedure: (1) generate a random graph  $G$  of size  $N$ , where each edge is present with probability  $p$ , (2) build a randomly permuted copy  $H$  of  $G$ , (3) randomly split the vertices in  $G$  (and in  $H$ ) by taking a random vertex in  $G$  (and in  $H$ ) and split it into two vertices (operation repeated  $M$  times), (4) introduce noise by adding/deleting  $\sigma \times p \times N^2$  random edges in both graphs.

As already mentioned, our principal interest here is to understand the behavior of graph matching algorithms as functions of the graph size  $N$ , and their ability to resist to structural noise. Indeed, in practice we never have identical graphs and it is important to have a robust algorithm which is able to deal with noise in graph structures. The objective function  $F(P_1, P_2)$  in (4) represents the quality of graph matching, so to compare different graph matching algorithms we plot  $F(P_1, P_2)$  as a function of  $N$  (Figure 2a), and  $F(P_1, P_2)$  as a function of  $\sigma$  (Figure 2b) for the four algorithms. In both cases, we observe that “Grad” and “SDP” significantly outperform both “Beam” and “Spec” with the “SDP” algorithm working slightly better than “Grad”. “Beam” was run with beam width equal to 3, which represents a good trade-off between quality and complexity, “Spec” was run with projection on the first two eigenvectors with the normalization presented in [6]. To maximally accelerate the “SDP” method, the number of iterations in SeDuMi was set to one, actually, one iteration is enough to have a good matching quality.

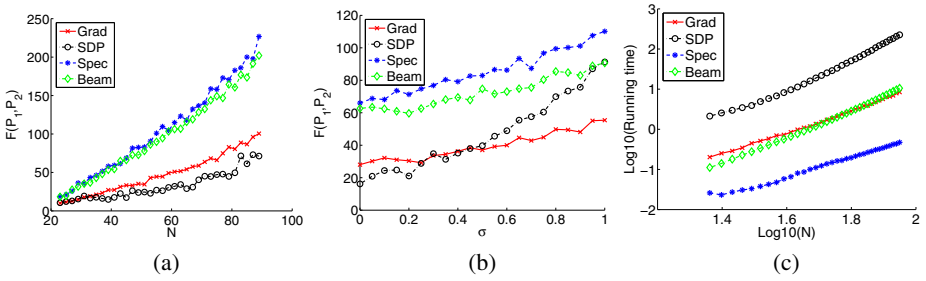
Figure 2c shows how algorithms scale in time with the graph size  $N$ . The “Spec” algorithm is the fastest one, but “Grad” has the same complexity order as “Spec” (corresponding curves are almost parallel lines in log-log scale, so both functions are polynomials with the same degree and different multiplication constants), these curves are coherent with theoretical values of algorithm complexity summarized in Section 3.1. The early stopping rule makes possible the use of the “SDP” algorithm on graphs with up to one hundred vertices, but it is still about 10 times slower than the “Grad” method, and almost 100 times slower than the “Spec” algorithm.

## 5.2 Chinese Characters

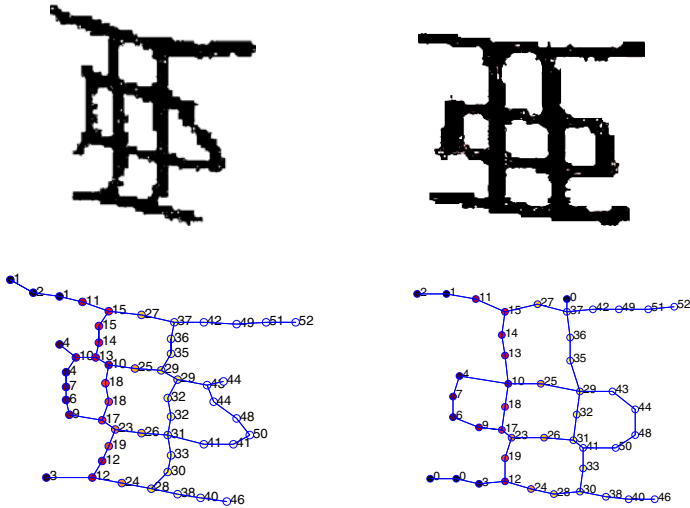
In this section we *quantitatively* compare many-to-many graph matching algorithms as parts of a classification framework. We use graph matching algorithms to compute similarity/distance between objects of interest on the basis of their graph-based representations. Our objective here is to see if with the new formulation of the many-to-many graph matching problem (4) and corresponding continuous relaxation algorithms improve the classification performance comparing to the existing state of the art algorithms. Since “Grad” and “Spec” are essentially two alternative approximate algorithms for the same discrete optimization problem with “Grad” working almost 10 times faster and providing the matching quality similar to that of the “SDP” method, we decided to test only the “Grad” algorithm. For example, to compute the similarity matrix of 600 graphs with in average 50 vertices, the running time of the “SDP” method would be about 280 hours ( $\approx 50$  seconds per pair).

---

<sup>2</sup> “Spec” variants with three and more eigenvectors were also tested, but two eigenvectors produced almost the same matching quality and worked faster.




**Fig. 2.** (a)  $F(P_1, P_2)$  (mean value over 30 repetitions) as a function of graph size  $N$ , simulation parameters:  $p = 0.1, \sigma = 0.05, M = 3$ . (b)  $F(P_1, P_2)$  (mean value over 30 repetitions) as a function of noise parameter  $\sigma$ , simulation parameters:  $N = 30, p = 0.1, M = 3$ . (c) Algorithm running time (mean value over 30 repetitions) as a function of  $N$  (log-log scale), other parameters are the same as in (a), “Beam” slope  $\approx 3.9$ ; “SDP” slope  $\approx 4.2$ , “Grad” slope  $\approx 2.9$ , “Spec” slope  $\approx 2.7$ .



**Fig. 3.** Different writings of the same Chinese character and the matching of the corresponding graphs made by “Grad”. Vertices having the same id’s are matched to each other.

As the classification problem, we chose the ETL9B dataset of Chinese characters. This dataset is well suited for our purposes, since Chinese characters may be naturally represented by graphs with variable non-trivial structures. Figure 3 illustrates how “Grad” works on graphs representing Chinese characters. We see that our algorithm produces a good matching, although not perfect, providing a correspondence between “crucial” vertices. The characters represented in Figure 3 are however very easy to recognize, and most classification algorithms show a good performance on them; for example, “Grad” produces a classification error rate below 0.2%. To test graph matching algorithms on more challenging situations, we chose three “hard to classify” Chinese

**Table 1.** Top: Chinese characters from three different classes. Bottom: classification results (mean and standard deviation of test error over cross-validation runs, with 50 repetitions of five folds)


Method	error	STD
Linear SVM	0.377	± 0.090
SVM with Gaussian kernel	0.359	± 0.076
k-NN (one-to-one, Path)	0.248	± 0.075
k-NN (shape context)	0.399	± 0.081
k-NN (shape context+tps)	0.435	± 0.092
k-NN (Spec)	0.254	± 0.071
k-NN (Beam)	0.283	± 0.079
k-NN (Grad)	<b>0.191</b>	± 0.063

characters, i.e., three characters sharing similar graph structures, as illustrated in Table 1. We ran k-nearest neighbor (k-NN) with graph matching algorithms used as distance measures. The dataset consists of 600 images, 200 images of each class.

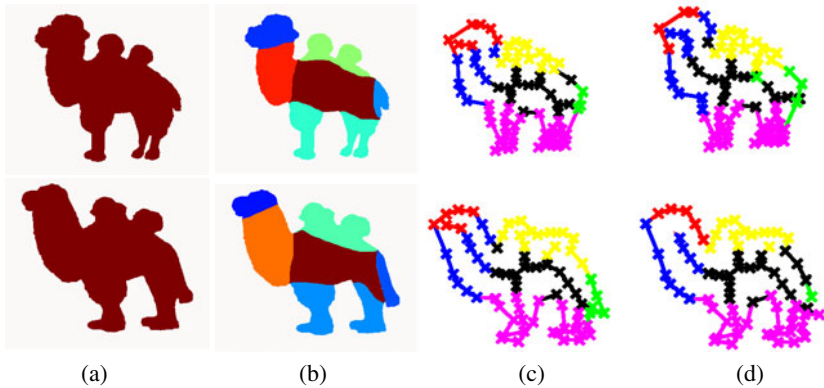
Table 1 shows classification results for the three many-to-many graph matching algorithms. In addition we report results for other popular approaches, namely, a SVM classifier with linear and Gaussian kernels, one-to-one matching with the Path algorithm (taken from [4]) and two versions of the shape context method [1], with or without thin plate spline smoothing. The version named “shape context” computes polar histograms with further bipartite graph matching. To run the “shape context+tps” method we used code available online<sup>3</sup>.

Graph matching algorithms are run using information on vertex coordinates through (7). The elements of the matrix  $C$  are defined as  $C_{ij} = e^{-(x_i - x_j)^2 - (y_i - y_j)^2}$ . The parameter  $\lambda$  in (7) as well as  $k$  (number of neighbors in k-NN classifier) are learned via cross-validation. We see that the “Grad” algorithm shows the best performance, outperforming other many-to-many graph matching algorithms as well as other competitive approaches.

### 5.3 Identification of Object Composite Parts

While the pattern recognition framework is interesting and important for the comparison of different graph matching algorithms, it evaluates only one aspect of these algorithms, namely, their ability to detect similar graphs. A second and important aspect is their ability to correctly align vertices corresponding to the same parts of two objects. To test this capability, we performed the following series of experiments. We chose ten camel images from the MPEG7 dataset and we divided by hand each image into 6 parts: head, neck, legs, back, tail and body (Figure 4). This image segmentation automatically defines a partitioning of the corresponding graph shown in the column (c) in Figure 4: all graph vertices are labeled according to the image part which they represent. Figure 4

<sup>3</sup> <http://www.eecs.berkeley.edu/vision/shape/>



**Fig. 4.** (a) Original images. (b) Manual segmentation (c) Graph-based representation (obtained automatically from subsampled contours and shock graphs) with induced vertex labels (d) Prediction of vertex labels on the basis of graph matching made by “Grad”. Best seen in color.

**Table 2.** Identification of object composite parts: mean and standard deviation of prediction error (see text for details). Note that standard deviations are not divided by the square root of the sample size (therefore differences are statistically significant).

	Grad	Spec	Beam	One-to-one
Error	<b>0.303</b>	0.351	0.432	0.342
STD	0.135	0.095	0.092	0.094

gives two illustrations of how this procedure works. A good graph matching algorithm should map vertices from corresponding image parts to each other, i.e., heads to heads, legs to legs, and so on. Therefore to evaluate the matching quality of the mapping, we use the following score. First, we match two graphs and then we try to predict vertex labels of one graph given the vertex labels of the second one. For instance, if vertex  $g_1$  of the first image is matched to vertices  $h_1$  and  $h_2$  representing the head of the second image, then we predict that  $g_1$  is of class “head”. The better the graph matching, the smaller the prediction error and vice-versa.

This experiment illustrates a promising application of graph matching algorithms. Usually segmentation algorithms extract image parts on the basis of different characteristics such as changing of color, narrowing of object form, etc. With our graph matching algorithm, we can extract segments which does not only have a specific appearance, but also have a semantic interpretation defined by a user (e.g., through the manual labelling of a particular instance).

Table 2 presents mean prediction error over 45 pairs of camel images (we exclude comparison of identical images). Each pair has two associated scores: prediction error of the first image given the second one and vice-versa. We thus have 90 scores for each algorithm, which are used to compute means and standard deviations. Like in the previous sections, graph matching algorithms are run using information on vertex coordinates (using Eq. (7)), with  $C_{ij} = e^{-(x_i-x_j)^2-(y_i-y_j)^2}$ . The parameter  $\lambda$  in (7)

as well as  $k$  (number of neighbors in  $k$ -NN classifier) are learned via cross-validation. Here, again we observe that the “Grad” algorithm works better than other methods.

## 6 Conclusion and Future Work

The main contribution of this paper is the new formulation of the many-to-many graph matching problem as a discrete optimization problem and new approximate algorithms “Grad” and “SDP” based on two alternative continuous relaxation. The success of the proposed methods compared to other competitive approaches may be explained by two reasons. First, methods based on continuous relaxations of discrete optimization problems often show a better performance than local search algorithm due to their ability to better explore the optimization set with potentially large moves. Second, “Grad” and “SDP” algorithms aim to optimize a clear objective function naturally representing the quality of graph matching instead of a sequence of unrelated steps. It is still difficult to run the “SDP” algorithm on real world datasets due to its time complexity, but we think a significant progress may be made by employing approximate SDP solvers. The hard limit on the number of iteration in SeDuMi is probably not the best way to find an approximate solution, we are planning to see other alternatives such as SDPA (<http://sdpa.indsys.chuo-u.ac.jp/sdpa/>) and CSDP (<https://projects.coin-or.org/Csdp/>). The reformulation of the gradient minimization as a linear assignment problem made possible the use of the “Grad” algorithm in large-scale graph matching problems. Probably, the special structure of the optimization set in the “SDP” relaxation may also represent an important clue for the further acceleration of the “SDP” algorithm. In particular, this direction seems to be interesting since “SDP” was able to find better matchings than “Grad” in numerical tests (Section 5.1). Another interesting direction is to try to construct a theoretical bound on the quality of the proposed approximate algorithms.

Besides a natural application of graph matching as a similarity measure between objects with complex structures, graph matching can also be used for object alignment. However, the structural noise usually encountered in graph-based representations have slightly hampered its application to natural images; but we believe that the many-to-many graph matching framework presented in this paper can provide an appropriate notion of robustness, which is necessary for many machine learning applications.

## Acknowledgments

This paper was supported in part by a grant from the Agence Nationale de la Recherche (MGA Project).

Mikhail Zaslavskiy thanks Collectis S.A.<sup>4</sup>, a genome engineering company, for kindly provided travel and registration grants.

## References

1. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.* 24(4), 509–522 (2002)

<sup>4</sup> [www.collectis.com](http://www.collectis.com)

2. Neuhaus, M., Riesen, K., Bunke, H.: Fast suboptimal algorithms for the computation of graph edit distance. In: Yeung, D.-Y., Kwok, J.T., Fred, A.L.N., Roli, F., de Ridder, D. (eds.) SSPR 2006 and SPR 2006. LNCS, vol. 4109, pp. 163–172. Springer, Heidelberg (2006)
3. Almohamad, H.A., Duffuaa, S.O.: A linear programming approach for the weighted graph matching problem. *IEEE Trans. Pattern Anal. Mach. Intell.* 15(5), 522–525 (1993)
4. Zaslavskiy, M., Bach, F., Vert, J.-P.: A path following algorithm for the graph matching problem. *IEEE Trans. Pattern Anal. Mach. Intell.* 31(12), 2227–2242 (2009)
5. Umeyama, S.: An eigendecomposition approach to weighted graph matching problems. *IEEE Trans. Pattern Anal. Mach. Intell.* 10(5), 695–703 (1988)
6. Caelli, T., Kosinov, S.: An eigenspace projection clustering method for inexact graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.* 26(4), 515–519 (2004)
7. Carcassoni, M., Hancock, E.: Spectral correspondence for point pattern matching. *Pattern Recogn.* 36(1), 193–204 (2003)
8. Cour, T., Srinivasan, P., Shi, J.: Balanced graph matching. In: *Advanced in Neural Information Processing Systems* (2006)
9. Leordeanu, M., Hebert, M.: A spectral technique for correspondence problems using pairwise constraints. In: *International Conference of Computer Vision (ICCV)*, vol. 2, pp. 1482–1489 (October 2005)
10. Duchenne, O., Bach, F., Kweon, I., Ponce, J.: A tensor-based algorithm for high-order graph matching. In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition CVPR 2009*, June 20–25, pp. 1980–1987 (2009)
11. Berretti, S., Del Bimbo, A., Pala, P.: A graph edit distance based on node merging. In: *Proc. of ACM International Conference on Image and Video Retrieval (CIVR)*, Dublin, Ireland, July 2004, pp. 464–472 (2004)
12. Ambauen, R., Fischer, S., Bunke, H.: Graph edit distance with node splitting and merging, and its application to diatom identification. In: *GbRPR*, pp. 95–106 (2003)
13. Keselman, Y., Shokoufandeh, A., Demirci, M.F., Dickinson, S.: Many-to-many graph matching via metric embedding. In: *CVPR*, pp. 850–857 (2003)
14. Bertsekas, D.: *Nonlinear programming*. Athena Scientific (1999)
15. Kuhn, H.W.: The Hungarian method for the assignment problem. *Naval Research* 2, 83–97 (1955)
16. Nesterov, Y., Nemirovsky, A.: *Interior point polynomial methods in convex programming: Theory and applications*. SIAM, Philadelphia (1994)

## Appendix A

Here we present how the gradient minimization step in the “Grad” method can be reformulated as a linear assignment problem. Let  $\nabla F_1$  and  $\nabla F_2$  denote the gradient of the function  $F(P_1, P_2)$  with respect to matrices  $P_1$  and  $P_2$  ( $\nabla F_1$  is a  $N_K \times N_G$  matrix and  $\nabla F_2$  is a  $N_K \times N_H$  matrix). Recall that our objective is to minimize  $\text{tr} F_1^T P_1 + \text{tr} F_2^T P_2$  over (4). Note that two terms of the gradient are independent linear functions and can be minimized one by one. Let us consider the first term (we drop the subscript 1 for simplicity)

$$\begin{aligned} \min_P \text{tr} \nabla F^T P \quad \text{subject to} \\ P \in \{0, 1\}^{N_K \times N_G}, P \mathbf{1}_{N_G} \leq k_{max} \mathbf{1}_{N_K}, P^T \mathbf{1}_{N_K} = \mathbf{1}_{N_G}. \end{aligned} \quad (11)$$

Matrix  $P$  is a  $N_K \times N_G$  binary matrix with up to  $k_{max}$  “ones” in each row, and one and only one “one” in each column. Let  $Q$  be a  $k_{max} N_K \times N_G$  binary matrix with up to one

“one” in each row (may be zero), and exactly one “one” in each column.  $Q$  may be seen as a splitted version of matrix  $P$ , the first  $k_{max}$  rows of the matrix  $Q$  correspond to the first row of  $P$ , then rows with indexes  $k_{max} + 1, \dots, 2k_{max}$  correspond to the second row of  $P$  and so on. We can always construct  $P$  from  $Q$  by merging corresponding rows, the reverse operation corresponds to splitting the rows of matrix  $P$ . We will write  $P \leftrightarrow Q$  to denote pairs  $(P, Q)$  which may be transformed to each other by merging/splitting operations, of course the same matrix  $P$  may correspond to many matrices  $Q$ 's. Now, let  $F_q$  denote a  $k_{max}N_K \times N_G$  real valued matrix constructed from the matrix  $F$  by duplicating every row  $k_{max}$  times i.e. first  $k_{max}$  rows of  $F_q$  are copies of the first row of  $F$  and so on.

This is easy to see, that if  $P \leftrightarrow Q$  then  $\text{tr}F_q^T Q = \text{tr}F^T P$  (the left side is just a splitted version of the right side) and therefore if  $P^* = \arg \min_P \text{tr}F^T P$  and  $Q^* \leftrightarrow P^*$  then  $Q^* = \arg \min_Q \text{tr}F_q^T Q$  and vice versa. Indeed, if  $Q^* \neq \arg \min_Q \text{tr}F_q^T Q$  then  $\exists Q^+$  such that  $\text{tr}F_q^T Q^+ < \text{tr}F_q^T Q^*$ , then we can construct  $P^+ \leftrightarrow Q^+$  such that  $\text{tr}F^T P^+ < \text{tr}F^T P^*$ .

We showed that  $\min_P F^T P$  is equivalent to  $\min_Q F_q^T Q$  which is nothing else than a linear assignment problem. Now, we can run the Hungarian algorithm to minimize  $F_q^T Q$  and then transform the optimal  $Q$ -solution to a  $P$ -solution by merging corresponding rows.

# Competitive Online Generalized Linear Regression under Square Loss

Fedor Zhdanov and Vladimir Vovk

Computer Learning Research Centre and Department of Computer Science,  
Royal Holloway, University of London, Egham, Surrey TW20 0EX, England  
{fedor,vovk}@cs.rhul.ac.uk

**Abstract.** We apply the Aggregating Algorithm to the problem of online regression under the square loss function. We develop an algorithm competitive with the benchmark class of generalized linear models (our “experts”), which are used in a wide range of practical tasks. This problem does not appear to be analytically tractable. Therefore, we develop a prediction algorithm using the Markov chain Monte Carlo method, which is shown to be fast and reliable in many cases. We prove upper bounds on the cumulative square loss of the algorithm. We also perform experiments with our algorithm on a toy data set and two real world ozone level data sets and give suggestions about choosing its parameters.

**Keywords:** online prediction, Aggregating Algorithm, square loss function, generalized linear models.

## 1 Introduction

Generalized linear models have been found very useful in statistical analysis (see [14]) for solving bounded regression problems. Classification problems as well are often solved by means of these models.

We use the Aggregating Algorithm in a way which is somewhat similar to the Aggregating Algorithm for Regression (AAR) introduced in [18]. Whereas the AAR only covers the class of linear experts, our new algorithm also covers other popular classes of experts, which are more efficient in that their predictions always belong to the interval  $[Y_1, Y_2]$  assumed to contain the label that is being predicted. When specialized to the case of linear experts, our general loss bound coincides with the known optimal bound for the AAR. A disadvantage of our algorithm is that we need to know  $[Y_1, Y_2]$  *a priori* to be able to apply it.

Another popular field related to competitive prediction is online convex optimization introduced in [20]. The two fields cover a common special case: a compact set of experts under loss functions of a specific form (the square loss for our application). The problems which cover in part generalized linear models are analyzed in [10]. In the general case, online convex optimization significantly relaxes the condition on loss functions, whereas competitive prediction removes the compactness requirement. Since many algorithms for competitive prediction



are designed for a narrower class of loss functions, they have a better coefficient in front of log in the regret term. Our algorithm can be applied to some non-convex functions, which is impossible for the methods of online convex optimization even for a compact set of experts.

Generalized linear models are popular in Bayesian statistics for solving classification problems (see Relevance Vector Machines in Section 7.2.3 of [4]). From the competitive prediction perspective, Bayesian mixtures are analogous to the Aggregating Algorithm competing under the logarithmic loss function. Upper bounds on the logarithmic loss are proved in [8], [12], [13], and [2] using different approaches. In this paper we prove upper bounds on the square loss, which is more often used in practice to compare the performances of different algorithms.

Our prediction problem does not appear to be analytically tractable. Therefore, we develop a prediction algorithm using the Markov chain Monte Carlo method, which is shown to be fast and reliable in many cases. Monte Carlo methods are well known in the Bayesian community [16]. They are also applied, for example, in [7] to explore exponential weighting schemes in problems with high dimension of the input vectors.

We give suggestions about choosing the parameters of our algorithm and perform experiments with it on a toy data set and two real world ozone level data sets.

Our paper is organized as follows. Section 2 describes the prediction protocol and the main theorem, and Section 3 covers a set of important examples. In Section 4 we derive the algorithm competitive with generalized linear models. Section 5 describes experiments with the new algorithm. In Section 6 we prove the main theorem.

## 2 Prediction Algorithm and Loss Bound

This is our prediction protocol:

---

### Protocol 1. Online bounded regression protocol

---

```

for  $t = 1, 2, \dots$  do
  Reality announces  $x_t \in \mathbb{R}^n$ 
  Learner predicts  $\gamma_t \in \mathbb{R}$ 
  Reality announces  $y_t \in [Y_1, Y_2]$ 
end for

```

---

Protocol 1 describes a perfect-information game between two players, Learner (male) and Reality (female). At each step  $t$  Reality announces a vector  $x_t \in \mathbb{R}^n$ , which is, intuitively, a hint that she gives to Learner to help him predict her next move  $y_t$ , called the *outcome*. The vector  $x_t$  will be called the *object*, or *input vector*, and its components  $x_{t,i}$ ,  $i = 1, \dots, n$ , will be called *attributes*. After that Learner announces his prediction  $\gamma_t$  for  $y_t$  and Reality announces the outcome  $y_t \in [Y_1, Y_2]$ . Sometimes we will say that  $y_t$  is the *label* of the object  $x_t$ . Learner's loss is measured by the square loss function. Learner wishes to compete with the

class of *generalized linear experts* indexed by  $\theta \in \Theta = \mathbb{R}^n$ . Expert  $\theta$ 's prediction at step  $t$  is denoted  $\xi_t^\theta$  and is equal to

$$\xi_t^\theta = Y_1 + (Y_2 - Y_1)\sigma(\theta' x_t). \tag{1}$$

Here  $\sigma : \mathbb{R} \rightarrow \mathbb{R}$  is a fixed *activation function*. We have  $\sigma : \mathbb{R} \rightarrow [0, 1]$  in all the cases except linear regression (see below). If the range of the function  $\sigma$  is  $[0, 1]$ , the experts necessarily give predictions from  $[Y_1, Y_2]$ .

Learner records the cumulative losses suffered by himself,  $L_T = \sum_{t=1}^T (\gamma_t - y_t)^2$ , and each expert  $\theta$ ,  $L_T^\theta = \sum_{t=1}^T (\xi_t^\theta - y_t)^2$ , over the first  $T$  steps. Intuitively, Learner's goal is to ensure that  $L_T \leq L_T^\theta$ , or at least  $L_T \approx L_T^\theta$ , for a vast majority of experts  $\theta \in \Theta$ .

Assume that the function

$$b(u, z) := \left( \frac{d\sigma(z)}{dz} \right)^2 + (\sigma(z) - u) \frac{d^2\sigma(z)}{dz^2} \tag{2}$$

is uniformly bounded:  $b := \sup_{u \in [0,1], z \in \mathbb{R}} |b(u, z)| < \infty$ . We will further see that this assumption holds for the most popular generalized linear regression models. We prove the following upper bound on Learner's loss.

**Theorem 1.** *Let  $a > 0$ . There exists a prediction strategy for Learner such that for every positive integer  $T$ , every sequence of outcomes of the length  $T$ , and every  $\theta \in \mathbb{R}^n$ , the cumulative loss  $L_T$  of Learner satisfies*

$$L_T \leq L_T^\theta + a\|\theta\|^2 + \frac{(Y_2 - Y_1)^2}{4} \ln \det \left( I + \frac{b(Y_2 - Y_1)^2}{a} \sum_{t=1}^T x_t x_t' \right), \tag{3}$$

where  $b := \sup_{u \in [0,1], z \in \mathbb{R}} |b(u, z)|$  and  $b(u, z)$  is defined by (2). If in addition  $\|x_t\|_\infty \leq X$  for all  $t$  then

$$L_T \leq L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln \left( 1 + \frac{b(Y_2 - Y_1)^2 X^2 T}{a} \right). \tag{4}$$

The order of the regret term in bound (4) is logarithmic in the number of steps. It matches the order of the best bounds for the linear regression problem under square loss proved in [18] and for the classification problem using generalized linear regression experts under logarithmic loss proved in [12].

The prediction strategy achieving (3) is formulated as Algorithm 1, calculated with the number of iterations  $M \rightarrow \infty$ ; we also call Algorithm 1 the Aggregating Algorithm for Generalized Linear Models (AAGLM). In Section 4 we derive it and describe its parameters. Even though Algorithm 1 is an online algorithm, it is easy to apply it in the batch setting: it suffices to consider each example in the test set as the next example after the training set.

### 3 Examples of the Models and Performance Guarantees

Now we give some examples of generalized linear models and bounds on the losses of the corresponding algorithms.

### 3.1 Linear Regression

We have for linear regression

$$\sigma(z) = \frac{z - Y_1}{Y_2 - Y_1}, \quad z \in \mathbb{R} \tag{5}$$

(so that the experts predict  $\xi_t^\theta = \theta'x_t$ ). The derivatives are equal to

$$\frac{d\sigma(z)}{dz} = \frac{1}{Y_2 - Y_1}$$

and

$$\frac{d^2\sigma(z)}{dz^2} = 0.$$

Using these expressions in the derivatives of  $\sigma$  in (2) we obtain

$$b(u, z) = \frac{1}{(Y_2 - Y_1)^2}.$$

Using  $b = \frac{1}{(Y_2 - Y_1)^2}$  in (4) we have the following corollary for the linear regression experts.

**Corollary 1.** *There exists a prediction strategy for Learner achieving*

$$L_T \leq L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln \left( 1 + \frac{X^2T}{a} \right)$$

for any expert  $\theta \in \mathbb{R}^n$  predicting according to (1) and (5) (i.e.,  $\xi_t^\theta = \theta'x_t$ ).

As we can see, the upper bound is the same as the upper bound for the AAR with  $Y_2 = Y$ ,  $Y_1 = -Y$  (see Theorem 1 in [18]). This bound is known to have the best possible order (see Theorem 2 in [18]) for the linear experts.

### 3.2 Logistic Regression

We have for logistic regression

$$\sigma(z) = \frac{1}{1 + e^{-z}}, \quad z \in \mathbb{R}. \tag{6}$$

The derivatives are equal to

$$\frac{d\sigma(z)}{dz} = \sigma(z)(1 - \sigma(z))$$

and

$$\frac{d^2\sigma(z)}{dz^2} = \sigma(z)(1 - \sigma(z))(1 - 2\sigma(z)).$$

Using these expressions in the derivatives of  $\sigma$  in (2) we obtain

$$b(u, z) = \sigma^2(z)(1 - \sigma(z))^2 + (\sigma(z) - u)\sigma(z)(1 - \sigma(z))(1 - 2\sigma(z)).$$

We have  $|b(u, z)| \leq b \leq \frac{5}{64}$ . Using this in (4) we have the following corollary for the logistic regression experts.

**Corollary 2.** *Let  $a > 0$ . There exists a prediction strategy for Learner achieving*

$$L_T \leq L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln \left( 1 + \frac{5(Y_2 - Y_1)^2 X^2 T}{64a} \right)$$

for any expert  $\theta \in \mathbb{R}^n$  predicting according to (1) and (6).

### 3.3 Probit Regression

We have for probit regression

$$\sigma(z) = \Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^z e^{-v^2/2} dv, \quad z \in \mathbb{R}, \tag{7}$$

where  $\Phi$  is the cumulative distribution function of the normal distribution with zero mean and unit variance. The derivatives are equal to

$$\frac{d\sigma(z)}{dz} = \frac{1}{\sqrt{2\pi}} e^{-z^2/2}$$

and

$$\frac{d^2\sigma(z)}{dz^2} = -\frac{z}{\sqrt{2\pi}} e^{-z^2/2}.$$

Using these expressions in the derivatives of  $\sigma$  in (2) we obtain

$$b(u, z) = \frac{1}{2\pi} e^{-z^2} - (\sigma(z) - u) \frac{z}{\sqrt{2\pi}} e^{-z^2/2}.$$

We have  $|b(u, z)| \leq b \leq \frac{25}{128}$ . Using this in (4) we have the following corollary for the probit regression experts.

**Corollary 3.** *Let  $a > 0$ . There exists a prediction strategy for Learner achieving*

$$L_T \leq L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln \left( 1 + \frac{25(Y_2 - Y_1)^2 X^2 T}{128a} \right)$$

for any expert  $\theta \in \mathbb{R}^n$  predicting according to (1) and (7).

### 3.4 Complementary Log-Log Regression

We have for the complementary log-log regression

$$\sigma(z) = 1 - \exp(-\exp(z)), \quad z \in \mathbb{R}. \tag{8}$$

When the argument  $z$  of the complementary log-log function  $1 - \exp(-\exp(z))$  approaches minus infinity, this function is similar to the logistic function  $\frac{1}{1+e^{-z}}$  and tends to zero. When the argument approaches infinity, the function tends to

one more quickly than the logistic function. This can be used in problems with asymmetry in outcomes. The derivatives of  $\sigma(z)$  are equal to

$$\frac{d\sigma(z)}{dz} = e^z(1 - \sigma(z))$$

and

$$\frac{d^2\sigma(z)}{dz^2} = e^z(1 - \sigma(z))(1 - e^z).$$

Using these expressions in the derivatives of  $\sigma$  in (2) we obtain

$$b(u, z) = e^{2z}(1 - \sigma(z))^2 + e^z(1 - \sigma(z))(1 - e^z).$$

We have  $|b(u, z)| \leq b \leq \frac{17}{64}$ . Using this in (4) we have the following corollary for the complementary log-log regression experts.

**Corollary 4.** *Let  $a > 0$ . There exists a prediction strategy for Learner achieving*

$$L_T \leq L_T^\theta + a\|\theta\|^2 + \frac{n(Y_2 - Y_1)^2}{4} \ln \left( 1 + \frac{17(Y_2 - Y_1)^2 X^2 T}{64a} \right)$$

for any expert  $\theta \in \mathbb{R}^n$  predicting according to (1) and (8).

## 4 Derivation of the Prediction Algorithm

Our prediction algorithm differs from many algorithms commonly used to fit a generalized linear model (see, for example, [14]). First, instead of fitting the data with the best parameter  $\theta$  (as in logistic regression) it uses the regularization parameter  $a > 0$  preventing  $\theta$  to be too large, and thus preventing overfitting to a certain extent. Second, it tries to minimize the square loss instead of the logarithmic loss. To predict at step  $T$  it works with the function

$$\sum_{t=1}^{T-1} (\xi_t^\theta - y_t)^2 + a\|\theta\|^2 \tag{9}$$

with  $\xi_t^\theta$  from (1). Third, it does not look for the best regularized expert  $\theta$ , but at each prediction step it mixes all the experts in a way which is similar to Bayesian scheme, thus preventing overfitting even further.

We use the Aggregating Algorithm (AA) to derive the prediction algorithm. The AA works as follows.

It is given a parameter  $\eta$  and an initial distribution on the experts. We choose the normal distribution

$$P_0(d\theta) = (a\eta/\pi)^{n/2} \exp(-a\eta\|\theta\|^2)d\theta \tag{10}$$

for some  $a > 0$ , the other parameter of the algorithm. After each step  $t$  the AA updates the weights of the experts according to their losses:

$$P_t(d\theta) = e^{-\eta(\xi_t^\theta - y_t)^2} P_{t-1}(d\theta)$$

and then normalizes the weights:  $P_t^*(d\theta) = \frac{P_t(d\theta)}{\int_{\Theta} P_t(d\theta)}$ . It is easy to see that at the step  $t$  the unnormalized weight of an expert  $\theta$  can be expressed as

$$P_t(d\theta) = (a\eta/\pi)^{n/2} e^{-\eta(L_t^\theta + a\|\theta\|^2)} d\theta. \tag{11}$$

To give a prediction the AA first calculates the *generalized prediction*

$$g_t(y) = -\frac{1}{\eta} \ln \int_{\Theta} e^{-\eta(\xi_t^\theta - y)^2} P_{t-1}^*(d\theta) \tag{12}$$

for any possible outcome  $y \in [Y_1, Y_2]$  using the normalized weights computed on the previous step. It then gives its prediction  $\gamma_t = \Sigma(g)$  using a *substitution function*  $\Sigma$  such that  $(\gamma_t - y)^2 \leq g_t(y)$  for any  $y \in [Y_1, Y_2]$ . It is known [18] that for the square loss game such a substitution function exists if  $\eta \in \left(0, \frac{2}{(Y_2 - Y_1)^2}\right]$ . The following substitution function outputs a permitted prediction even if the generalized predictions are calculated from the weights scaled by any constant (for example, unnormalized):

$$\gamma_t = \frac{1}{2} \left( Y_2 + Y_1 - \frac{G_t(Y_2) - G_t(Y_1)}{Y_2 - Y_1} \right) \tag{13}$$

for

$$G_t(y) = -\frac{1}{\eta} \ln \int_{\Theta} e^{-\eta((\xi_t^\theta - y)^2 + L_{t-1}^\theta + a\|\theta\|^2)} d\theta. \tag{14}$$

We obtain (14) by calculating the generalized prediction (12) with unnormalized weights (11) and omitting the factor  $(a\eta/\pi)^{n/2}$ . Normalization is avoided because calculating the normalizing constant is a computationally inefficient operation. We will use the maximum value for  $\eta$ ,  $\eta = \frac{2}{(Y_2 - Y_1)^2}$ . We denote

$$w_T(\theta) := \exp \left( -a\eta\|\theta\|^2 - \eta \sum_{t=1}^T (\xi_t^\theta - y_t)^2 \right). \tag{15}$$

Algorithm 1 is based on the MCMC technique of numerical integration in (14) at  $y = Y_1$  and  $y = Y_2$ ; a good MCMC survey is given in [1]. The function  $e^{-\eta(\xi_t^\theta - y)^2}$  needs to be integrated with respect to the unnormalized posterior distribution  $P_{t-1}(d\theta)$ . We choose to use the simple Metropolis sampling to sample  $\theta$  from the posterior.

Metropolis sampling from a distribution  $\mathcal{P}$  is an iterative process, with the initial value  $\theta^0$  and a simple proposal distribution. We choose the Gaussian proposal distribution  $N(0, \sigma^2)$  with the parameter  $\sigma^2$  chosen on the data. At each iteration  $i = 1, \dots, M$ , the update for  $\theta$  is sampled from the proposed distribution:

$$\theta^i = \theta^{i-1} + \zeta^i, \quad \zeta^i \sim N(0, \sigma^2).$$

The updated  $\theta^i$  is accepted with the probability  $\min \left( 1, \frac{f_{\mathcal{P}}(\theta^i)}{f_{\mathcal{P}}(\theta^{i-1})} \right)$ . Here  $f_{\mathcal{P}}(\theta)$  is the value of the density function for the distribution  $\mathcal{P}$  at the point  $\theta$ . By

accepting and rejecting the updates the values of the parameter  $\theta$  move closer to the value where the maximum of the density function  $f_{\mathcal{P}}$  is achieved. Thus sampling is performed from the area with high density of  $\mathcal{P}$  and covers the tails of it only occasionally. Therefore numerical integration using sampling with MCMC is often more efficient than the usual Monte Carlo sampling from the uniform distribution. Sometimes the updates are accepted even if they do not move the next  $\theta$  closer to the maximum (this happens when  $\frac{f_{\mathcal{P}}(\theta^i)}{f_{\mathcal{P}}(\theta^{i-1})} < 1$ ). This may allow the algorithm to move away from the local maxima of the density function.

---

**Algorithm 1.** AAGLM

---

**Require:** Bounds  $Y_1, Y_2$  for the outcomes,  
 maximum number  $M > 0$  of MCMC iterations,  
 standard deviation  $\sigma > 0$ ,  
 regularization coefficient  $a > 0$

calculate  $\eta := \frac{2}{(Y_2 - Y_1)^2}$

initialize  $\theta_0^M := 0 \in \Theta$

define  $w_0(\theta) := \exp(-a\eta \|\theta\|^2)$

**for**  $t = 1, 2, \dots$  **do**

$G_{t,1} := 0, G_{t,2} := 0$

read  $x_t \in \mathbb{R}^n$

initialize  $\theta_t^0 := \theta_{t-1}^M$

**for**  $m = 1, 2, \dots, M$  **do**

$\theta^* := \theta_t^{m-1} + N(0, \sigma^2 I)$

**if**  $w_{t-1}(\theta^*) \geq w_{t-1}(\theta_t^{m-1})$  **then**

$\theta_t^m := \theta^*$

**else**

flip a coin with success probability  $w_{t-1}(\theta^*)/w_{t-1}(\theta_t^{m-1})$

**if** success **then**

$\theta_t^m := \theta^*$

**else**

$\theta_t^m := \theta_t^{m-1}$

**end if**

**end if**

$G_{t,1} := G_{t,1} + e^{-\eta(\xi_t^{\theta_t^m} - Y_1)^2}, G_{t,2} := G_{t,2} + e^{-\eta(\xi_t^{\theta_t^m} - Y_2)^2}$

**end for**

output prediction  $\gamma_t := \frac{1}{2} \left( Y_2 + Y_1 + \frac{\ln G_{t,2} - \ln G_{t,1}}{\eta(Y_2 - Y_1)} \right)$

read  $y_t \in [Y_1, Y_2]$

define  $w_t(\theta) := w_{t-1}(\theta) \exp(-\eta(\xi_t^\theta - y_t)^2)$

**end for**

---

It is common when using the MCMC approach to have a “burn-in” stage, at which the integral is not calculated, but the algorithm is looking for the best “locality” for  $\theta$ . This stage is used to avoid the error accumulated while the algorithm is still looking for the correct location of the main mass of the distribution. Instead of that, at each prediction step  $t$  we take the new starting

point  $\theta^0$  for the Metropolis sampling to be the last point  $\theta^M$  achieved on the previous step  $t - 1$ .

In the case when the dimension  $n$  of input vectors is large and the sampling is not very efficient, one can use more advanced techniques, such as adaptive sampling or Slice sampling [16]. However, when we tried several versions of Slice sampling, the convergence speed on our data sets was slower than for Metropolis sampling.

The function (9) is not necessarily convex in  $\theta$ , so it may have several local minima. Thus we cannot use the Laplace approximation for the integral and obtain reliable Iteratively Reweighted Least Square estimation of  $\theta$ , the common approach to give predictions when working with generalized linear models. The MCMC approach to calculating similar integrals for Bayesian prediction models was analyzed in [15]. It is stated there that it takes  $O(n^3)$  operations to calculate a general integral.

## 5 Experiments

In this section we investigate empirical properties of our algorithm on toy and real data sets and suggest ways to choose the parameters for it.

### 5.1 Toy Data Set

In this experiment we aim to emphasize the main properties of competitive online algorithms: how they behave if the data follow the assumptions of one of the experts, and when the data fail to do so; how quickly online algorithms adjust to the substantial changes in the properties of the data.

Consider the following online classification problem. Let  $x_t \in \mathbb{R}$  be the input vectors  $x_1 = -50, x_2 = -49.9, \dots, x_T = 100$ , real numbers from  $-50$  to  $100$  with step  $0.1$ . Let the outcomes  $y_t$  be

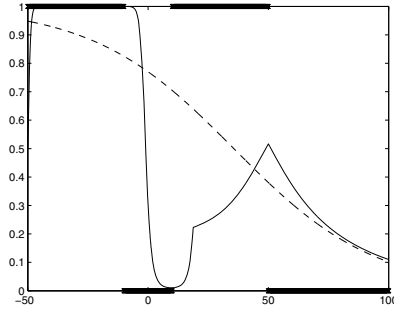
$$y_t = \begin{cases} 1 & \text{if } x_t < -10 \text{ or } 10 < x_t < 50, \\ 0 & \text{otherwise.} \end{cases}$$

We add the bias one as the second component of each input vector (input vector dimension becomes equal to 2). We try to predict this sequence online step by step by Algorithm 1 competing with logistic experts. The result is presented on Figure 1. We also show the best fitted logistic predictor achieving the minimum cumulative square loss. Notice three following interesting properties of the picture.

The predictions of the AAGLM tend to the predictions of the best logistic regression fitted to the whole data as  $T$  becomes large. This matches the fact that the mean loss of the AAGLM converges to the mean loss of the best logistic regression; see Corollary 2.

When  $x \in [-10, 50)$  both algorithms suffer large loss. Corollary 2 ensures that the AAGLM does not suffer much more than the best logistic regression.





**Fig. 1.** Sequential predictions of the AAGLM algorithm for the two-class classification problem. The dashed curve is the predictions of the best logistic regression (under square loss) on all the data. The horizontal axis contains the input vectors, the vertical axis contains the outcomes and the predictions of the outcomes by the two algorithms.

During each period the AAGLM tries to fit a logistic regression function in this and the previous periods. The dependence on the previous periods is due to the fact that the trained AAGLM is equivalent to the untrained AAGLM which starts predicting with initial weights distribution  $P_t^*(d\theta)$ , where  $t$  is the number of steps in the previous periods.

In order for expert-based algorithms to predict well on a certain type of data, the best expert should suffer small loss on these data. If the sequence of outcomes has several regimes which rarely switch from one to another, like in our figure, “tracking the best expert” [11, 17] may be a more suitable framework.

The parameters of the AAGLM used in these studies are  $Y_1 = 0$ ,  $Y_2 = 1$ ,  $M = 1000$  (we did not use “burn-in” stage),  $\sigma = 0.00001$ ,  $a = 10^{-100}$ . Increasing the regularization coefficient  $a$  leads to the regularization towards 0.5 (as expected). Increasing  $M$  accelerates somewhat the reaction in the very beginning of each turn, but the main trend does not change. Too low value for  $\sigma$  leads to slower convergence. Too high value of this parameter forces the algorithm to fluctuate between two classes, and never find a stable solution (this is expected as well since with large  $\sigma$  the numerical integration becomes less precise). It is common when applying the MCMC technique to use the following rule of thumb to determine the value of the parameter  $\sigma$ : the rejection rate should be 30–70%.

## 5.2 Ozone Data Set

We perform empirical studies on two real-world data sets described in [19].

**Data sets.** The data sets contain various meteorology and ozone data for the Houston, Galveston, and Brazoria (HGB) area in Texas, USA, day by day for 7 years, 1998–2004. We use both one-hour (ozone1) and eight-hour (ozone8) ozone data sets: they contain the the daily maximum of 1 hour (ozone1) ozone concentration and the daily maximum of the average over 8 consecutive hours (ozone8)

ozone concentration. Each observation is one day. Each observation has 72 features of various measures of air pollutant and meteorological information for the target area in the study. Each observation is assigned the label 1 (we say its outcome is equal to 1) if the ozone level exceeds the danger threshold, which is 120 parts per billion (ppb) for ozone1 and 80 ppb for ozone8; otherwise, the observation has the label 0 (outcome is equal to 0). The data are collected online, so online prediction algorithms are more appropriate for the study than batch algorithms. They are able to predict ozone levels day by day incorporating the information from all the previous days. Therefore we consider online two-class classification problems.

It is shown in [19] that all 72 features may be relevant to the prediction problem, and thus we decided to use all of them to train our algorithms. We replace the missing values of the features by the mean of the available values of them from the first year (we use the first year data as our training set).

The data sets are very skewed: the number of positive examples is very low (73 for ozone1 and 160 for ozone8 out of 2534 observations). It can be expected that for such data sets complementary log-log (comlog) regression performs better than logistic or probit regression. Indeed, the square loss suffered by logistic regression trained on the whole data set is 48.4178 for ozone1 and 96.2846 for ozone8. At the same time, the square loss suffered by comlog regression trained on the whole data set is 46.7815 and 94.8399 respectively. Thus we use the comlog activation function (8) in this experimental study.

**Algorithms and results.** We normalize all the features to have mean zero and maximum absolute value one over the first year. We also add an additional bias feature 1 to all the examples.

We compare different algorithms in two regimes: the online regime and the (incremental) batch regime. In the online regime the algorithms are retrained as soon as a new observation is obtained. In the batch regime the algorithms are only retrained yearly on all the past data. In [19] this regime is suggested as the most realistic for meteorologists (they did not consider the online regime though).

For the online regime the AAGLM parameters  $M$ ,  $\sigma$ ,  $a$ , and the length of the burn-in stage are chosen to suffer the least square loss over the first year. We choose  $\sigma$  from the range  $10^{-k}, 5 \cdot 10^{-k}$ ,  $k = 0, 1, \dots, 5$ . We choose  $a$  from the range  $10^{-k}, 5 \cdot 10^{-k}$ ,  $k = -1, 0, 1, \dots, 10$ . The best parameters are  $M = 2500$ ,  $\sigma = 0.01$ ,  $a = 0.1$ , and the length of the burn-in stage is 2000. It is interesting to note that for both data sets the best parameters are the same up to our precision. This may mean that the best parameters for the algorithm depend mostly on the input vectors, and not on the outcomes (because the input vectors are the same for our data sets).

The batch regime of the AAGLM can be understood as just one step of the online algorithm repeated for each new test example. During the training stage the batch algorithm does not calculate the integral but saves the values of  $\theta$  obtained at each iteration  $m = 1, 2, \dots, M$ . At the prediction stage the algorithm calculates the integral using the saved values of  $\theta$  computed on the iterations

between  $B < M$  and  $M$ , where  $B$  is the length of the burn-in stage. We choose the same parameters  $\sigma = 0.01$ ,  $a = 0.1$  as for the online version, and  $B = 5000$ ,  $M = 15000$  to ensure good convergence. There are no theoretical guarantees for the batch setting.

The first algorithm with which we compare the AAGLM is the online comlog regression minimizing logarithmic loss (standard generalized regression model): at each step  $t$  it uses all the previous steps to find the best parameter  $\hat{\theta}$  and then gives its prediction  $\sigma(\hat{\theta}'x_t)$  according to this parameter. In terms familiar from the online prediction literature, it corresponds to the Follow the Best Expert predictor under the logarithmic loss function, the natural competitor to our algorithm.

In the batch regime in the beginning of each year the best parameter  $\tilde{\theta}$  is found on all the previous years. All the predictions in the following year are made using this  $\tilde{\theta}$ .

We also calculate the performance of the linear Support Vector Machine (SVM) and the SVM with the RBF kernel [6]. The SVM with the RBF kernel showed the best performance on ozone8 in a different framework [19]. In the online regime the SVMs predict only one next outcome at each step and retrain after the actual outcome is announced. The parameter of the kernel and the parameter  $C$  are chosen to achieve the least square loss over the first year. Note that in the online regime one does not need the validation set: the training set at each step does not include the next test example at which prediction is made. Thus the risk of overfitting is less than if the testing was done on the training set.

In the batch regime at the beginning of each year the SVMs are retrained on all the previous years. All the predictions in the following year are made using these trained SVMs. The parameter of the kernel and the parameter  $C$  are chosen using 5-fold cross-validation: all the data from the previous years are randomly separated into 5 parts. Four parts are used to train the SVM, the fifth part is used for testing: the square loss is calculated over the fifth part. This procedure repeats 5 times with different combinations of the parts. The parameters of the SVMs are chosen to suffer the least average square loss.

Since the number of positive examples is small, it makes sense to compare the precision of the predicting algorithms with the precision of the zero predictor: the predictor which always predicts low ozone concentration.

The minimal square loss which is suffered by the comlog regression model trained on the whole data set is equal to 28.0001 for ozone1 and 75.0001 for ozone8. This loss is unrealistic since we do not know all the observations when start predicting, and included here to specify the limitations of the generalized regression model. The square loss of the online comlog regression over this period is equal to 105.1823 and 186.6147 respectively. The square loss of the AAGLM over this period is equal to 66.2727 and 120.8483 respectively. At the same time the upper bounds on its loss from Corollary 4 have the values 323.9632 for ozone1 and 371.3681 for ozone8. The zero predictor suffers the square loss 73 and 160 respectively.

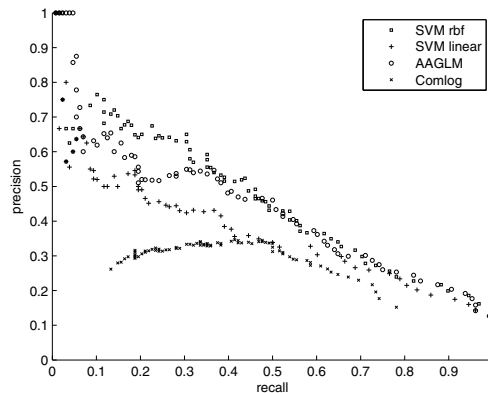
**Table 1.** Average MSEs of different algorithms over the last 6 years of ozone1/ozone8

Algorithm	MSE online	MSE batch
AAGLM	8.2159/15.3288	8.2163/15.7552
SVM rbf	9.7607/14.8806	9.4497/15.9679
SVM linear	8.8624/16.1532	8.8500/16.5264
Comlog	14.4578/24.2686	13.8096/27.9474
Zeros	9.6667/21.3333	9.6667/21.3333

Table 1 contains average mean square errors for different algorithms over the last 6 years (2–7) in the data sets.

Figure 2 presents *precision* (the number of correctly identified high ozone days divided by the total number of the predicted high ozone days) and *recall* (the number of correctly identified high ozone days divided by the total number of the actual high ozone days) for different threshold values calculated for the last 6 years of ozone8. It contains information for the four algorithms applied in the online regime. The area under the curve for the AAGLM is larger than the area under the curve for the online comlog regression and under the curve for the linear SVM. This shows the superiority of our algorithm over these competitors for the classification task. We can also see that there is a point on the curve for the online comlog regression where the reduction of the recall does not lead to the increase in the precision. This means that the threshold becomes larger than the predicted probability of many high-ozone days.

As we can see from the figures in Table 1, the algorithms in the online regime perform better than the same algorithms in the batch regime on ozone8 and usually worse on ozone1.



**Fig. 2.** Precision-recall curve for different threshold values for the algorithms applied in the online regime on ozone8

We can also see that the AAGLM significantly outperforms the simple zero predictor and the comlog predictor. Unlike the comlog predictor, the AAGLM is developed to work with the square loss and achieves better empirical performance in terms of the square loss. It performs a little worse on ozone8 than the SVM with the RBF kernel. It is not difficult to apply kernelization to the AAGLM as well (for the kernelization of standard generalized linear regression models see [5]) using an analogy with non-parametric Bayesian methods; the kernelized algorithm may achieve better performance. On ozone1 the AAGLM outperforms all the algorithms including SVMs.

The disadvantage of our algorithm against the competitors is in its training speed. Increasing the training speed of our algorithm is an interesting area of future research. It would be also interesting to apply our classifier to other data sets and find extreme data sets where the theoretical guarantee is tight.

### 6 Proof of Theorem 1

Let  $\eta := \frac{2}{(Y_2 - Y_1)^2}$  and set  $\beta = e^{-\eta}$ . The loss of the AA over the first  $T$  trials does not exceed the loss of the sum of the generalized predictions (12) over the first  $T$  trials, which in turn equals (see Lemma 1 in [18])

$$\log_{\beta} \int_{\Theta} \beta^{L_T(\theta)} P_0(d\theta) = -\frac{1}{\eta} \ln \left( (a\eta/\pi)^{n/2} \int_{\Theta} e^{-\eta Q(\theta)} d\theta \right), \tag{16}$$

where

$$Q(\theta) := \sum_{t=1}^T ((Y_2 - Y_1)\sigma(\theta'x_t) + Y_1 - y_t)^2 + a\|\theta\|^2.$$

We will lower bound the integral in (16) by upper bounding  $Q(\theta)$  using a quadratic form.

Because of the second addend in the definition of  $Q(\theta)$ ,  $Q(\theta) \rightarrow \infty$  as  $\|\theta\| \rightarrow \infty$ . Therefore  $\min_{\theta} Q(\theta)$  is attained at some point. Let  $\theta_0$  be the point where it is attained, and thus  $\nabla Q(\theta_0) = 0$ . We use Taylor expansion of  $Q(\theta)$  at the point  $\theta_0$ :

$$Q(\theta) = Q(\theta_0) + \frac{1}{2}(\theta - \theta_0)'H(\phi)(\theta - \theta_0),$$

where  $\phi$  is a convex combination of  $\theta_0$  and  $\theta$ . Here  $H$  is the Hessian matrix of  $Q(\theta)$ , the matrix of its second derivatives. By  $\delta_i^j$  we denote the Kronecker delta. The second partial derivative of  $Q$  is expressed as follows:

$$\begin{aligned} \frac{\partial^2 Q}{\partial \theta_i \partial \theta_j} &= 2a\delta_i^j + 2(Y_2 - Y_1)^2 \\ &\cdot \sum_{t=1}^T \left( \frac{\partial \sigma(\theta'x_t)}{\partial \theta_i} \frac{\partial \sigma(\theta'x_t)}{\partial \theta_j} - \left( \frac{y_t - Y_1}{Y_2 - Y_1} - \sigma(\theta'x_t) \right) \frac{\partial^2 \sigma(\theta'x_t)}{\partial \theta_i \partial \theta_j} \right) \\ &= 2a\delta_i^j + 2(Y_2 - Y_1)^2 \sum_{t=1}^T x_{t,i}x_{t,j}b \left( \frac{y_t - Y_1}{Y_2 - Y_1}, \theta'x_t \right). \end{aligned}$$

It is clear now that the matrix  $H(\phi)$  can be represented as follows:

$$H(\phi) = 2aI + 2(Y_2 - Y_1)^2 X' \Gamma(\phi) X,$$

where  $X$  is the design matrix  $T \times n$  consisting of the rows of the input vectors  $x'_1, \dots, x'_T$ . Here  $\Gamma(\phi)$  is the diagonal  $T \times T$  matrix which has the coefficients  $b(u_1, \phi' x_1), \dots, b(u_T, \phi' x_T)$  on the diagonal (with  $u_i = \frac{y_i - Y_1}{Y_2 - Y_1}$ ,  $i = 1, \dots, T$ ). Since  $\Gamma(\phi)$  is a symmetric matrix, we can see (Theorem 21.5.6 in [9]) that

$$\psi' \Gamma(\phi) \psi \leq \psi' \lambda_{\max} \psi \tag{17}$$

for any  $\psi \in \mathbb{R}^n$ , where  $\lambda_{\max}$  is the supremum over maximum eigenvalues of  $\Gamma(\phi)$ . Since  $b(u_t, \phi' x_t)$  is uniformly bounded, we have  $\lambda_{\max} \leq b$ .

We can take  $\psi = X(\theta - \theta_0)$  and obtain from (17) that

$$Q(\theta) \leq Q(\theta_0) + (\theta - \theta_0)'(aI + b(Y_2 - Y_1)^2 X' X)(\theta - \theta_0).$$

Thus the integral in (16) is lower bounded as follows:

$$\int_{\Theta} e^{-\eta Q(\theta)} d\theta \geq e^{-\eta Q(\theta_0)} \int_{\Theta} e^{-\eta(\theta - \theta_0)'(aI + b(Y_2 - Y_1)^2 X' X)(\theta - \theta_0)} d\theta.$$

The integral in the right-hand side can be analytically calculated (see Section 15.12 in [9]):

$$\int_{\Theta} e^{-\eta(\theta - \theta_0)'(aI + b(Y_2 - Y_1)^2 X' X)(\theta - \theta_0)} d\theta = \frac{(\pi/\eta)^{n/2}}{\sqrt{\det(aI + b(Y_2 - Y_1)^2 X' X)}}.$$

After taking the logarithm of  $e^{-\eta Q(\theta_0)}$ , we obtain  $L_T^{\theta_0} + a\|\theta_0\|^2$ . Substituting these expressions and  $\eta = \frac{2}{(Y_2 - Y_1)^2}$  to (16) we obtain the upper bound (3).

The determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Chapter 2, Theorem 7 in [3]):

$$\det \left( I + \frac{b(Y_2 - Y_1)^2}{a} \sum_{t=1}^T x_t x_t' \right) \leq \left( 1 + \frac{b(Y_2 - Y_1)^2 T X^2}{a} \right)^n.$$

This concludes the proof.

### Acknowledgments

We are grateful for useful comments to Alexey Chernov, Yuri Kalnishkan, and anonymous referees. This work was supported in part by EPSRC (grant EP/F002998/1).

## References

1. Andrieu, C., de Freitas, N., Doucet, A., Jordan, M.: An introduction to MCMC for machine learning. *Machine Learning* 50, 5–43 (2003)
2. Banerjee, A.: An analysis of logistic models: exponential family connections and online performance. In: *Proceedings of the 2007 SIAM International Conference on Data Mining*, pp. 204–215 (2007)
3. Beckenbach, E.F., Bellman, R.: *Inequalities*. Springer, Berlin (1961)
4. Bishop, C.M.: *Pattern Recognition and Machine Learning*. Springer, New York (2006)
5. Cawley, G.C., Janacek, G.J., Talbot, N.L.C.: Generalised kernel machines. In: *Proceedings of the International Joint Conference on Neural Networks*, pp. 1720–1725 (2007)
6. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001)
7. Dalalyan, A., Tsybakov, A.B.: Sparse regression learning by aggregation and Langevin Monte-Carlo. In: *Proceedings of the 22nd Annual Conference on Computational Learning Theory* (2009)
8. Forster, J.: On relative loss bounds in generalized linear regression. In: Ciobanu, G., Păun, G. (eds.) *FCT 1999. LNCS*, vol. 1684, pp. 269–280. Springer, Heidelberg (1999)
9. Harville, D.A.: *Matrix Algebra From a Statistician's Perspective*. Springer, New York (1997)
10. Hazan, E., Agarwal, A., Kale, S.: Logarithmic regret algorithms for online convex optimization. *Machine Learning* 69, 169–192 (2007)
11. Herbster, M., Warmuth, M.K.: Tracking the best expert. *Machine Learning* 32, 151–178 (1998)
12. Kakade, S.M., Ng, A.Y.: Online bounds for Bayesian algorithms. In: *Proceedings of the 18th Annual Conference on Neural Information Processing Systems* (2004)
13. Kivinen, J., Warmuth, M.K.: Relative loss bounds for multidimensional regression problems. *Machine Learning* 45, 301–329 (2001)
14. McCullagh, P., Nelder, J.: *Generalized Linear Models*, 2nd edn. Chapman and Hall, Boca Raton (1989)
15. Neal, R.M.: Regression and classification using gaussian process priors. In: Bernardo, J.M., Berger, J.O., Dawid, A.P., Smith, A.F.M. (eds.) *Bayesian Statistics*, vol. 6, pp. 475–501. Oxford University Press, Oxford (1999)
16. Neal, R.M.: Slice sampling. *The Annals of Statistics* 31, 705–741 (2003)
17. Vovk, V.: Derandomizing stochastic prediction strategies. *Machine Learning* 35, 247–282 (1999)
18. Vovk, V.: Competitive on-line statistics. *International Statistical Review* 69, 213–248 (2001)
19. Zhang, K., Fan, W.: Forecasting skewed biased stochastic ozone days: analyses, solutions and beyond. *Knowledge and Information Systems* 14, 299–326 (2008)
20. Zinkevich, M.: Online convex programming and generalized infinitesimal gradient ascent. In: *Proceedings of the 20th International Conference on Machine Learning*, pp. 928–936 (2003)

# Cross Validation Framework to Choose amongst Models and Datasets for Transfer Learning

Erheng Zhong<sup>1</sup>, Wei Fan<sup>2</sup>, Qiang Yang<sup>3</sup>,  
Olivier Verscheure<sup>2</sup>, and Jiangtao Ren<sup>1</sup>

<sup>1</sup> Sun Yat-Sen University, Guangzhou, China  
{sw04zheh@mail2, issrjt@mail}.sysu.edu.cn

<sup>2</sup> IBM T.J Watson Research, USA  
{weifan,ov1}@us.ibm.com

<sup>3</sup> Department of Computer Science, Hong Kong University of Science and Technology  
qyang@cse.ust.hk

**Abstract.** One solution to the lack of label problem is to exploit transfer learning, whereby one acquires knowledge from source-domains to improve the learning performance in the target-domain. The main challenge is that the source and target domains may have different distributions. An open problem is how to select the available models (including algorithms and parameters) and importantly, abundance of source-domain data, through statistically reliable methods, thus making transfer learning practical and easy-to-use for real-world applications. To address this challenge, one needs to take into account the difference in both marginal and conditional distributions in the same time, but not just one of them. In this paper, we formulate a new criterion to overcome “double” distribution shift and present a practical approach “Transfer Cross Validation” (TrCV) to select both models and data in a cross validation framework, optimized for transfer learning. The idea is to use density ratio weighting to overcome the difference in marginal distributions and propose a “reverse validation” procedure to quantify how well a model approximates the true conditional distribution of target-domain. The usefulness of TrCV is demonstrated on different cross-domain tasks, including wine quality evaluation, web-user ranking and text categorization. The experiment results show that the proposed method outperforms both traditional cross-validation and one state-of-the-art method which only considers marginal distribution shift. The software and datasets are available from the authors.

## 1 Introduction

Transfer learning works in the context that the number of labeled examples in target-domain is limited. It assumes that source-domain and target-domain are under different marginal and conditional distributions. Recently, a number of algorithms have been proposed to overcome the distribution shift, such as those reviewed in but not limited to [1]. Moreover, for a given target-domain in transfer learning, a likely large number of source-domains are available. For example, if we



**Table 1.** Definition of notations

Notation	Description	Notation	Description
$S$	Source-domain, $S = \{X_s, Y_s\} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$	$k$	Number of folds in cross validation
$S_i$	Data in $i$ -th fold	$r(\mathbf{x})$	Value of $\mathbf{x}$ got by reverse validation
$\bar{Y}_s^i$	Pseudo labels of $S_i$	$\ell^*(f)$	Expected loss of model $f$
$\bar{S}_i$	Remaining data in $i$ -th fold	$\ell(f)$	Empirical loss of model $f$ in $T$
$T$	Target-domain, $T = \{X_\ell, Y_\ell, X_u\}$	$\ell_w(f)$	Weighted empirical loss of model $f$ in $S$
$L$	Labeled data in $T$ , $L = \{X_\ell, Y_\ell\}$	$\varepsilon_u(f)$	Estimated accuracy of model $f$ by TrCV
$U$	Unlabeled data in $T$ , $U = X_u$	$\Theta_f$	Model complexity of $f$
$n$	Number of instances in $S$	$P(\mathbf{x})$	Marginal distribution of $\mathbf{x}$
$\ell$	Number of instances in $L$	$P(y \mathbf{x})$	Conditional distribution of $(\mathbf{x}, y)$
$u$	Number of instances in $U$	$\beta$	Density ratio vector of $X_s$

aim to classify the documents of 20-Newsgroup [2], RCV1 [3] and Reuters-21578 [2] or other text collections can be treated as the candidates of source-domain. Thus, for a transfer learning task, it is crucial to solve three problems effectively: (1) How to select the right transfer learning algorithms? (2) How to tune the optimal parameters? (3) How to choose the most helpful source-domain from a large pool of datasets? However, to the best of our knowledge, neither any analytical criterion nor efficient practical procedures have been proposed and reported.

Although some analytical techniques such as AIC (Akaike Information Criterion) [4], BIC (Bayesian Information Criterion) [5] and SRM (Structural Risk Minimization) principle [6] or sample re-use method (such as Cross Validation (CV)) to selecting the suitable model or training data (source-domain in transfer learning) have been studied, as reviewed later, they can not guarantee their performances in transfer learning for two reasons. First, due to the “double” distribution shift, including marginal and conditional distributions, the unbiasedness which guarantees the accuracy of these techniques does not hold anymore. Second, due to the very small number of labeled data in target-domain, it is unreliable to estimate the conditional distribution of target-domain directly.

To cope with these challenges, we first formulate a general criterion for model selection in transfer learning scenario, followed by a novel variant of CV method “Transfer Cross Validation” (TrCV) to solving the above three problems practically. Briefly, we introduce density ratio weighting to reduce the difference of marginal distributions between two domains. As proved in Section 4.1, it makes the estimation of TrCV unbiased. In addition, we exploit a method “Reverse Validation” (RV) to approximate the difference between the estimated and true conditional distribution of target-domain directly. As stated in Section 4.2, the value of RV is reliable to indicate the true difference. In summary, by eliminating the difference between two domains, the model selected by TrCV has a confidence bound on accuracy as shown in Section 4.3. In other words, the model or source-domain selected by TrCV is highly likely the best one among candidates as evaluated in Section 5.

## 2 Problem Statement

We review the limitation of traditional validation methods and then introduce a general criterion with transfer cross validation. The notations are summarized

in Table 1. Let  $S = \{X_s, Y_s\} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$  denote the source-domain and  $T = \{X_\ell, Y_\ell, X_u\} = \{(\mathbf{x}_i, y_i)\}_{i=1}^\ell \cup \{(\mathbf{x}_j)\}_{j=1}^u$  denote the target-domain, where  $n$  is the number of instances in source-domain,  $\ell$  and  $u$  are the number of labeled and unlabeled instances in target-domain respectively. Then, let  $P_s(\mathbf{x})$  and  $P_s(y|\mathbf{x})$  denote the marginal and conditional distribution of source-domain,  $P_t(\mathbf{x})$  and  $P_t(y|\mathbf{x})$  for target-domain. We use  $\hat{f}$  to represent the model expected to obtain.

### 2.1 Limitations of Existing Approaches

The model selected by analytical techniques is as follows:

$$\hat{f} = \arg \min_f \frac{1}{n} \sum_{\mathbf{x} \in X_s} \left| P_s(y|\mathbf{x}) - P(y|\mathbf{x}, f) \right| + \Theta_f \tag{1}$$

where the first term represents the empirical loss and  $\Theta_f$  is model complexity: the number of model parameters in AIC and BIC or the VC-Dimension in SRM. On the other hand,  $k$ -fold cross validation aims to select the model as:

$$\hat{f} = \arg \min_f \frac{1}{k} \sum_{j=1}^k \sum_{(\mathbf{x}, y) \in S_j} \left| P_s(y|\mathbf{x}) - P(y|\mathbf{x}, f_j) \right| \tag{2}$$

where  $k$  is the number of folds,  $S_j$  are the data in  $j$ -th fold and  $f_j$  is the model trained from the remaining data. However, these methods do not work as one would desire, for the following two reasons. First, because  $P_s(\mathbf{x}) \neq P_t(\mathbf{x})$ , Eq. (1) no longer provides consistent estimation [7]. In other words,  $\lim_{n \rightarrow \infty} (\hat{f}) \neq f^*$ , where  $f^*$  is the ideal hypothesis which achieves the minimal expected loss to approximate  $P_t(y|\mathbf{x})$ , regulated by model complexity:

$$f^* = \arg \min_f \mathbf{E}_{\mathbf{x} \sim P_t(\mathbf{x})} \left| P_t(y|\mathbf{x}) - P(y|\mathbf{x}, f) \right| + \Theta_f \tag{3}$$

To cope with similar problem in sample selection bias, previous work Weighted CV (WCV) [8] proposes to use density ratio to eliminate the difference in marginal distributions when performing cross-validation. It selects the model that minimizes the following objective.

$$\hat{f} = \arg \min_f \frac{1}{k} \sum_{j=1}^k \sum_{(\mathbf{x}, y) \in S_j} \frac{P_t(\mathbf{x})}{P_s(\mathbf{x})} \left| P_s(y|\mathbf{x}) - P(y|\mathbf{x}, f_j) \right| \tag{4}$$

However, neither of these explicitly considers the effect of conditional distribution shift between two domains, which is essential for most transfer learning problems. Because  $P_s(y|\mathbf{x}) \neq P_t(y|\mathbf{x})$  under transfer learning context, a model approximating  $P_s(y|\mathbf{x})$  is not necessarily close to  $P_t(y|\mathbf{x})$ . Thus, the model selected by Eq. (2) and Eq. (4) based on source-domain can not guarantee its performance in target-domain, as demonstrated experimentally in Section 5.

On the other hand, one may consider to perform CV on the labeled target-domain data  $L$  or to select the model trained using source-domain data  $S$  and has a high accuracy on  $L$ . But these methods fail to perform well on the whole target-domain, because the number of labeled data is so limited that they cannot reliably describe the true conditional distribution of target-domain,  $P_t(y|\mathbf{x})$ .

### 2.2 The Proposed Approach

As such, we have two observations. First, the estimation based on source-domain data need to be consistent with target-domain data. Second, the model should approximate the conditional distribution of target-domain, instead of the source-domain. Thus, we propose a new criterion by adding density ratio weighting and replacing the target conditional distribution as follows:

$$\hat{f} = \arg \min_f \frac{1}{n} \sum_{\mathbf{x} \in X_s} \frac{P_t(\mathbf{x})}{P_s(\mathbf{x})} \left| P_t(y|\mathbf{x}) - P(y|\mathbf{x}, f) \right| + \Theta_f \tag{5}$$

We notice that it is a general criterion extending Eq.(1). Under the traditional setting that marginal and conditional distributions do not shift, it is the same as Eq.(1). With the analysis in Section 4, we prove that Eq.(5) approximates an unbiased estimation to ideal hypothesis  $f^*$ . However, the model complexity term  $\Theta_f$  is usually hard to calculate in practice. Thus, following the same ideas, we propose a transfer cross validation (TrCV) method to solve the stated problems practically. It aims to select the model by minimizing the criterion:

$$\hat{f} = \arg \min_f \frac{1}{k} \sum_{j=1}^k \sum_{(\mathbf{x}, y) \in S_j} \frac{P_t(\mathbf{x})}{P_s(\mathbf{x})} \left| P_t(y|\mathbf{x}) - P(y|\mathbf{x}, f) \right| \tag{6}$$

Thus, algorithm selection, parameter tuning and source-domain selection in transfer learning can be solved using TrCV. For algorithm selection, it is intuitive. For other two problems, it is equivalent to pick a set of parameters or a source-domain which can build a model minimizing the value in Eq.(6).

### 3 Transfer Cross Validation (TrCV)

We discuss two main issues of TrCV in this section. The first one is that the density ratio of two domains  $\frac{P_t(\mathbf{x})}{P_s(\mathbf{x})}$  needs to be calculated based on the observed finite set. We let  $\beta = \{\beta(\mathbf{x}_1), \dots, \beta(\mathbf{x}_n)\}$  be the density ratio vector, where  $\beta(\mathbf{x}) = \frac{P_t(\mathbf{x})}{P_s(\mathbf{x})}$ . Some methods have been exploited for this problem [9, 10]. We adopt an existing one KMM from [10] which aims to find suitable values of  $\beta$  to minimize the discrepancy between means of two domains. Formally, it tries to minimize the following object by calculating the optimal  $\beta$ .

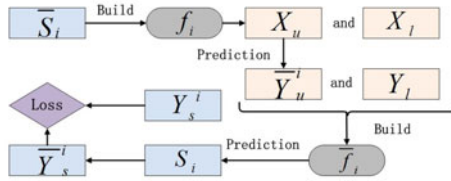
$$\begin{aligned} \min_{\beta} \quad & \frac{1}{2} \beta^T K \beta - \kappa^T \beta \\ \text{s.t.} \quad & \beta_i \in [0, B], \quad \left| \sum_{i=1}^n \beta_i - n \right| \leq n\epsilon \end{aligned}$$

where  $K_{ij} = \phi(\mathbf{x}_i, \mathbf{x}_j)$ ,  $\mathbf{x}_i, \mathbf{x}_j \in X_s$ ,  $\kappa_i = \frac{n}{\ell+u} \sum_{j=1}^{\ell+u} \phi(\mathbf{x}_i, \mathbf{x}_j)$ ,  $\mathbf{x}_i \in X_s, \mathbf{x}_j \in X_\ell \cup X_u$ ,  $\phi(*, *)$  is the kernel function,  $B$  is the upper bound for the ratio and  $\epsilon$  should be  $O(B/\sqrt{n})$ . In addition,  $\beta$  is restricted by two constraints: the first one limits the scope of discrepancy between  $p_t(\mathbf{x})$  and  $p_s(\mathbf{x})$  and the second one ensures that the measure  $\beta(\mathbf{x})p_s(\mathbf{x})$  is close to a probability distribution.

**Input:**  $S_i, \bar{S}_i, T$ , a learner  $\mathcal{F}$   
**Output:** The estimation of  $|P_t(y|\mathbf{x}) - P(y|\mathbf{x}, f_i)|$

- 1 Build a model  $f_i$  from  $\bar{S}_i$  using  $\mathcal{F}$ ;
- 2 Predict the labels of  $X_u, \bar{Y}_u^i$ ;
- 3 Build another model  $\bar{f}_i$  from  $\{X_u, \bar{Y}_u^i\} \cup \{X_\ell, Y_\ell\}$  using  $\mathcal{F}$ ;
- 4 Predict the labels of  $S_i, \bar{Y}_s^i$ ;
- 5 **for each instance**  $\mathbf{x}_{ij}$  **in**  $S_i$  **do**
- 6  $r(\mathbf{x}_{ij}) = |y_{ij} - \bar{y}_{ij}|$ , where  $\bar{y}_{ij} \in \bar{Y}_s^i$ ;
- 7 **end**
- 8 **return**  $r(\mathbf{x}_{ij}), \mathbf{x}_{ij} \in S_i$ ;

**Fig. 1.** Reverse Validation



**Fig. 2.** Flow chart of reverse validation

As follows, we focus on the second issue: how to calculate the difference between the conditional distribution estimated by model  $f$  and the true conditional distribution,  $|P_t(y|\mathbf{x}) - P(y|\mathbf{x}, f)|$ . Due to the limited number of labeled examples in target-domain, it is impossible to estimate the conditional distribution  $P_t(y|\mathbf{x})$  reliably. To overcome this challenge, we propose a novel method “Reverse Validation” which estimates the approximation difference directly and avoids computing the conditional distribution  $P_t(y|\mathbf{x})$ . To the best of our knowledge, this has not been well studied.

### 3.1 Reverse Validation (RV)

The main flow is presented in Figure 2 and the detail is stated in Figure 1. Let  $S_i$  be the source-domain data in  $i$ -th fold and  $\bar{S}_i$  be the remaining data. Firstly, for the given learner, we train a model  $f_i$  from  $\bar{S}_i$ , and then we use  $f_i$  to predict the labels of  $X_u$  and obtain  $\bar{Y}_u^i$ . Next, we combine  $\{X_u, \bar{Y}_u^i\}$  and  $\{X_\ell, Y_\ell\}$  to form a new set. Afterwards, a new model  $\bar{f}_i$  is built from the new set using the same algorithm and used to classify the instances in  $S_i$ . We denote the pseudo labels of  $S_i$  as  $\bar{Y}_s^i$ . Finally, for each instance  $\{\mathbf{x}_{ij}, y_{ij}\} \in S_i$ , we use the value of  $|y_{ij} - \bar{y}_{ij}|$  to estimate the difference, where  $\bar{y}_{ij}$  is the corresponding pseudo label of  $\mathbf{x}_{ij}$ . As analysed in Section 4.2, RV value  $r(\mathbf{x}_{ij}) = |y_{ij} - \bar{y}_{ij}|$  is related to  $|P_t(y_{ij}|\mathbf{x}_{ij}) - P(y_{ij}|\mathbf{x}_{ij}, f_i)|$  and can be used as an indicator.

TrCV can now be introduced using KMM and RV as stated in Figure 3. Briefly, we calculate the density ratio qualitatively and apply reverse validation to estimate the loss of conditional distribution approximation in each fold.

```

Input:  $S, T$ , a learner  $\mathcal{F}$ , number of fold  $k$ 
Output: The measure value of TrCV
1 Calculate  $\beta$  using KMM;
2 for  $i = 1$  to  $k$  do
3     Perform reverse validation,  $V_i = RV(S_i, \bar{S}_i, T, \mathcal{F})$ ;
4      $\ell = \ell + \sum_j v_{ij} \cdot \beta(\mathbf{x}_{ij})$ ,  $v_{ij} \in V_i$ ;
5 end
6 return  $\ell/n$ ;
    
```

**Fig. 3.** Transfer Cross Validation

## 4 Formal Analysis

We analyse three issues. First, does the general principle bound the risk in transfer learning? Second, is the loss calculated by reverse validation related to the true difference  $|P_t(y|\mathbf{x}) - P(y|\mathbf{x}, f)|$ ? Third, how is the confidence of the transfer cross validation?

### 4.1 Generalization Bound

We first demonstrate that the model selected by Eq. (5),  $\hat{f}$ , provides an unbiased estimator to  $f^*$  defined in Eq. (3). Let the expected loss of a model  $f$  be  $\ell^*(f)$ , the weighted empirical loss in source-domain be  $\ell_w(f)$  and  $n$  be the number of examples in  $S$ , then we get the lemma.

**Lemma 1.**  $\ell_w(\hat{f}) + \Theta_{\hat{f}} = \ell^*(f^*) + \Theta_{f^*}$ , when  $n \rightarrow \infty$  and  $f^*$  and  $\hat{f}$  belong to the same hypothesis class.

**Proof**

$$\begin{aligned}
 \ell_w(\hat{f}) &= \frac{1}{n} \sum_{\mathbf{x} \in X_s} \frac{P_t(\mathbf{x})}{P_s(\mathbf{x})} \left| P_t(y|\mathbf{x}) - P(y|\mathbf{x}, \hat{f}) \right| \\
 &= E_{\mathbf{x} \in X_s} \left[ \int_{\mathbf{x}} \frac{P_t(\mathbf{x})}{P_s(\mathbf{x})} \left| P_t(y|\mathbf{x}) - P(y|\mathbf{x}, \hat{f}) \right| P_s(\mathbf{x}) d\mathbf{x} \right] \\
 &= E_{\mathbf{x} \in X_s} \left[ \int_{\mathbf{x}} P_t(\mathbf{x}) \left| P_t(y|\mathbf{x}) - P(y|\mathbf{x}, \hat{f}) \right| d\mathbf{x} \right] \\
 &= \frac{1}{n} \sum_{\mathbf{x} \in X_s, X_s \sim P_t(\mathbf{x})} \left| P_t(y|\mathbf{x}) - P(y|\mathbf{x}, \hat{f}) \right| \\
 &= E_{\mathbf{x} \in X_s, X_s \sim P_t(\mathbf{x})} \left| P_t(y|\mathbf{x}) - P(y|\mathbf{x}, \hat{f}) \right|
 \end{aligned}$$

This means that, as  $n$  approaches infinity, the model minimizing the value of weighted empirical loss in source-domain also minimizes the expected loss in target-domain,  $\ell_w(\hat{f}) = \ell^*(f^*)$ . In addition, if  $f^*$  and  $\hat{f}$  belong to the same hypothesis class, it leads to  $\Theta_{f^*} = \Theta_{\hat{f}}$ . □

In addition, we conclude that the model minimizing the value of general principal in Eq. (5) is equal to the model minimizing the empirical error of target-domain data. In other words,

$$\begin{aligned} \ell(\hat{f}) &= \frac{1}{n} \sum_{\mathbf{x} \in X_s, X_s \sim P_t(\mathbf{x})} \left| P_t(y|\mathbf{x}) - P(y|\mathbf{x}, \hat{f}) \right| \\ &= \frac{1}{n} \sum_{\mathbf{x} \in X_s} \frac{P_t(\mathbf{x})}{P_s(\mathbf{x})} \left| P_t(y|\mathbf{x}) - P(y|\mathbf{x}, \hat{f}) \right| \end{aligned} \tag{7}$$

Next we demonstrate that if the estimator of  $\Theta_{\hat{f}}$  is related to VC-dimension,  $\hat{f}$  constructed from source-domain data has a generalization bound over target-domain data.

**Theorem 1.** *Let  $G(\hat{f})$  denote the generalization error of  $\hat{f}$  in the target-domain,  $n$  is the number of data in  $S$  and  $d_{vc}$  is the VC-dimension of the hypothesis class which  $\hat{f}$  belongs to, then with the probability at least  $1 - \delta$*

$$G(\hat{f}) \leq \ell_w(\hat{f}) + \sqrt{\left( \frac{d_{vc}(\log(2n/d_{vc}) + 1) - \log(\delta/4)}{n} \right)} \tag{8}$$

**Proof** As a conclusion from [6], for a given model  $f$ , it has a generalization bound:

$$G(f) \leq \ell(f) + \sqrt{\left( \frac{d_{vc}(\log(2n/d_{vc}) + 1) - \log(\delta/4)}{n} \right)} \tag{9}$$

In addition, let us recall Eq.(7), thus we obtain Eq.(8). □

### 4.2 Estimation by Reverse Validation

Due to the limited number of labeled examples in target-domain, we use reverse validation (RV) to estimate the difference between  $P_t(y|\mathbf{x})$  and  $P(y|\mathbf{x}, f)$  instead of estimating the conditional probability  $P_t(y|\mathbf{x})$  directly. As follows we provide some insights in RV. Let  $f_i$  be the model trained from  $\bar{S}_i$ ,  $\{X_u, \bar{Y}_u\}$  be the unlabeled data and corresponding pseudo labels predicted by  $f_i$  in the target-domain,  $\bar{f}_i$  be the model built from  $\{X_u, \bar{Y}_u^i\} \cup \{X_\ell, Y_\ell\}$  and  $\epsilon(f)$  be the approximation error of a model  $f$ . Thus, for a given instance  $\mathbf{x}$  from  $S_i$ , RV returns a value

$$r(\mathbf{x}) = \left| P_s(y|\mathbf{x}) - P(y|\mathbf{x}, \bar{f}_i) \right| \tag{10}$$

As an approximation to  $P_s(y|\mathbf{x})$ ,  $P(y|\mathbf{x}, f_i)$  can be rewritten as

$$P(y|\mathbf{x}, f_i) = P_s(y|\mathbf{x}) + \epsilon(f_i) \tag{11}$$

where  $\epsilon$  is the approximation error. In addition, because  $\bar{f}_i$  is trained from the label information  $\bar{Y}_u^i$  and  $Y_\ell$ ,  $P(y|\mathbf{x}, \bar{f}_i)$  can be treated as an approximation to the nuisance between  $P(y|\mathbf{x}, f_i)$  and  $P_t(y|\mathbf{x})$ .

$$P(y|\mathbf{x}, \bar{f}_i) = \alpha \cdot P(y|\mathbf{x}, f_i) + (1 - \alpha) \cdot P_t(y|\mathbf{x}) + \epsilon(\bar{f}_i) \tag{12}$$

where  $\alpha$  is the nuisance parameter related to the ratio between the size of  $X_u$  and  $X_l$ . Thus, by combining Eq.(10), (11) and (12),  $r(\mathbf{x})$  can be rewritten as

$$\begin{aligned}
 r(\mathbf{x}) &= \left| P_s(y|\mathbf{x}) - P(y|x, \bar{f}_i) \right| \\
 &= \left| P_s(y|\mathbf{x}) - \alpha P(y|\mathbf{x}, f_i) + (1 - \alpha)P_t(y|\mathbf{x}) - \epsilon(\bar{f}_i) \right| \\
 &= \left| (1 - \alpha) \left( P(y|\mathbf{x}, f_i) - P_t(y|\mathbf{x}) \right) - \epsilon(f_i) - \epsilon(\bar{f}_i) \right|
 \end{aligned} \tag{13}$$

This demonstrates that  $r(\mathbf{x})$  is related to  $|P(y|x, f_i) - P_t(y|\mathbf{x})|$  reliably. Thus, when the number of training data is large enough such that the model can approximate the true conditional probability reliably. In other words, when  $\epsilon(f_i)$  and  $\epsilon(\bar{f}_i)$  are small, RV can approach a confident estimation. In addition, when more labeled data obtained in target-domain,  $\alpha$  tends to be smaller. This implies that  $r(\mathbf{x})$  estimates  $|P(y|x, f_i) - P_t(y|\mathbf{x})|$  more precisely. On the other hand, if no labeled data in target-domain but  $P_t(y|\mathbf{x}) = P_s(y|\mathbf{x})$ ,  $r(\mathbf{x})$  becomes  $|\epsilon(f_i) + \epsilon(\bar{f}_i)|$  instead, which approximates as much as twice the error in traditional cross validation.

### 4.3 Confidence by TrCV

The discussion is based on the assumption that the classifiers are consistent: the classifiers built in each folds have the same predictability. Following Eq.(7), minimizing the weighted empirical loss of source-domain data in TrCV is equal to minimizing the empirical loss of target-domain data. In addition, combining Eq.(6) and Eq.(13), when model can approximate the true distribution well if obtaining enough labeled data, we rewrite the accuracy estimated by TrCV,  $\epsilon_u(f)$ , as

$$\begin{aligned}
 \epsilon_u(f) &= 1 - \frac{1}{k} \sum_{j=1}^k \sum_{\mathbf{x} \in S_j} \beta(\mathbf{x}) \left| r(\mathbf{x}) / (1 - \alpha) \right| \\
 &= 1 - \frac{1}{k} \sum_{j=1}^k \sum_{\mathbf{x} \in X_s, X_s \sim P_t(\mathbf{x})} \left| P_t(y|\mathbf{x}) - P(y|\mathbf{x}, f) \right|
 \end{aligned} \tag{14}$$

where  $r(\mathbf{x})$  is the value of reverse validation on data  $\mathbf{x}$  and  $\beta(\mathbf{x})$  is the density ratio of  $\mathbf{x}$ . Let  $\epsilon(f)$  be the true accuracy of  $f$ , based on the statement in [11], when the size of validation set is reasonably large, the distribution of  $\epsilon_u(f)$  is approximately normal with mean  $\epsilon(f)$  and a variance of  $\epsilon(f) \cdot (1 - \epsilon(f)) / n$ . By De Moivre-Laplace Limit theorem, we have

$$Pr \left\{ -z < \frac{\epsilon_u(f) - \epsilon(f)}{\sqrt{\epsilon(f) \cdot (1 - \epsilon(f)) / n}} < z \right\} \approx \lambda \tag{15}$$

where  $z$  is the  $(1 + \lambda) / 2$ -th quantile point of the standard normal distribution. The low and high confidence points of  $\epsilon(f)$  is calculated by inverting Eq.(15) as

$$\frac{2n \cdot \epsilon_u(f) + z^2 \pm z \cdot \sqrt{4n \cdot \epsilon_u(f) + z^2 - 4n \cdot \epsilon_u^2(f)}}{2(n + z^2)}$$

In addition, if the accuracy of  $f$  obtains the normal distribution in this interval with mean  $\mu = \frac{2n \cdot \varepsilon_u(f) + z^2}{2(n+z^2)}$  and variance  $\sigma = \frac{z \cdot \sqrt{4n \cdot \varepsilon_u(f) + z^2 - 4n \cdot \varepsilon_u^2(f)}}{2(n+z^2)}$ , the probability between two models,  $f_1$  and  $f_2$ ,  $P(\varepsilon(f_1) > \varepsilon(f_2))$  can be calculated.

$$\begin{aligned}
 &P(\varepsilon(f_1) > \varepsilon(f_2)) \\
 &= P(\varepsilon(f_1) - \varepsilon(f_2) > 0) \\
 &= P(x > 0), \quad x \sim N(\mu_1 - \mu_2, \sigma_1^2 + \sigma_2^2) \\
 &= 1 - \frac{1}{\sqrt{2\pi(\sigma_1^2 + \sigma_2^2)}} \int_{-\infty}^{\frac{\mu_2 - \mu_1}{\sqrt{\sigma_1^2 + \sigma_2^2}}} e^{-t^2/2} dt
 \end{aligned} \tag{16}$$

where  $\mu_1$  and  $\mu_2$  are the means of accuracy distributions obtained by  $f_1$  and  $f_2$  with TrCV and  $\sigma_1$  and  $\sigma_2$  are the corresponding variances. By calculating the means and variances based on the loss value of TrCV, the confidence of TrCV can be obtained by Eq. (16).

## 5 Experiment

TrCV criterion is evaluated to show if it can select the best algorithm for one task, can tune suitable parameters for one model and can choose the most useful source-domain over different candidates. For each task, several data collections have been utilized.

### 5.1 Experimental Setup

The proposal approach TrCV is compared against several other cross validation methods. The first two are the standard k-fold CV formulated by Eq. (2). One is on source-domain (SCV), another is on labeled data from target-domain (TCV). The third one is to build a model on the source-domain data and validate on labeled target-domain data (STV). Most importantly, we compare with Weighted CV (WCV) [8]. As discussed earlier, WCV is proposed for sample selection bias problems. It uses density ratio weighting to reduce the difference of marginal distribution between two domains, but ignores the difference in conditional probability, as shown in Eq. (6).

To test different criteria, we introduce five traditional classifiers, including Naive Bayes(NB), SVM, C4.5, K-NN and NNge(Ng), and three state-of-the-art transfer learning methods: TrAdaBoost(TA) [12], LatentMap(LM) [13] and LWE [14]. Among them, TrAdaBoost is based on instances weighting, LatentMap is through feature transform and LWE uses model weighting ensemble. As a comparison, the number of folds in SCV, TCV and TrCV is set to be the same: 10, and the number of labeled data in target-domain is fixed as the larger one between  $0.1 \times |T|$  and 20. As follows, we use ‘‘correlation’’ between the best classifiers and the selected classifiers by the criteria as the measure of evaluation.



**Table 2.** Dataset for Algorithm and Parameters Selection

Data Set	S	T	Description
Red-White(RW)	1599	4998	physicochemical variables
White-Red(WR)	4998	1599	
orgs vs. people(ope)	1016	1046	Documents from different subcategories
orgs vs. places(opl)	1079	1080	
people vs. places(pp)	1239	1210	
Sheep(Sp)	61	65	Web pages with different contents
Biomedical(BI)	61	131	
Goats(Gs)	61	70	

**Table 3.** Dataset for Source-domain Selection

Data Set	S	T	S	T
comp vs. rec	windows vs. motorcycles pc.hardware vs. baseball mac.hardware vs. hockey	graphics vs. autos	1596 1969 1954	1957
sci vs. talk	crypt vs. guns med vs. misc space vs. religion	electronics vs. mideast	1895 1761 1612	

Let  $f$  and  $g$  denote any two models, and  $\varepsilon(\cdot)$  and  $v(\cdot)$  are the accuracy and value of criteria (e.g. TrCV, standard CV, etc) on each model, respectively. Then the measure is

$$corr = C_{|\mathcal{H}|}^2 - \sum_{f,g \in \mathcal{H}} \left[ (\varepsilon(f) - \varepsilon(g)) \times (v(f) - v(g)) < 0 \right]$$

where  $[x]$  is 1 when  $x$  is true and 0 otherwise, and  $\mathcal{H}$  is the set of models. The first term  $C_{|\mathcal{H}|}^2$  is the number of comparisons where  $|\mathcal{H}|$  is the number of models and the second term indicates how many times the criterion selects the worse one among two models. This measure means that if one criterion can select the better model in the comparison, it gains a higher measure value. The main results can be found in Table 4 and 5.

Three data collections from three different domains are employed to evaluate the algorithm selection and parameter tuning by TrCV. Among them, Wine Quality dataset [2] contains two subsets related to red and white variants of the Portuguese “Vinho Verde” wine. The task is to classify wine’s quality according to their physicochemical variables. In the experiment, red-wine set and white-wine set are treated as source-domain and target-domain alternately. Reuters-21578 [2] is the primary benchmark of text categorization formed by different news with a hierarchical structure. It contains five top categories of news wire articles, and each main category contains several subcategories. Three top categories, “orgs”, “people” and “places” are selected in the study. All of the subcategories from each category are divided into two parts, one source-domain and one target-domain. They have different distributions and are approximately equal in size. The learning objective aims to classify articles into top categories. SyskillWebert [2] is the standard dataset used to test web page ratings, generated by the HTML source of web pages plus the user rating (“hot” or “not hot”) on those web pages. It contains four separate subjects belonging to

**Table 4.** Algorithm Selection and Parameters Tuning

Method	RW	WR	ope	opl	pp	Sp	Bl	Gs	RW	WR	ope	opl	pp	Sp	Bl	Gs	RW	WR	ope	opl	pp	Sp	Bl	Gs
	Algorithm Selection								Parameter Tuning (LatentMap)								Parameter Tuning (SVM)							
SCV	18	17	13	17	13	19	16	17	4	5	5	5	<b>8</b>	4	4	6	4	7	5	4	3	7	7	<b>8</b>
TCV	17	18	14	17	10	15	10	11	3	3	3	5	5	4	1	2	5	4	3	4	4	4	5	5
STV	16	15	13	15	14	18	<b>17</b>	<b>20</b>	4	5	4	4	7	<b>8</b>	1	6	4	7	4	7	3	<b>8</b>	7	5
WCV	20	19	17	19	18	18	15	15	4	5	5	<b>8</b>	<b>8</b>	4	3	<b>7</b>	<b>8</b>	7	6	6	5	<b>8</b>	6	7
TrCV	<b>22</b>	<b>23</b>	<b>22</b>	<b>20</b>	<b>22</b>	<b>20</b>	15	18	<b>5</b>	<b>7</b>	<b>8</b>	<b>8</b>	<b>8</b>	5	3	<b>7</b>	7	<b>8</b>	7	<b>8</b>	<b>6</b>	<b>8</b>	<b>8</b>	<b>8</b>

**Table 5.** Source-domain Selection

Method	NB	SVM	C45	KNN	Ng	TA	LM	LWE	Pr
SCV	5	<b>6</b>	<b>6</b>	5	4	4	1	<b>6</b>	436
STV	2	3	4	<b>6</b>	2	2	3	5	371
TCV	<b>6</b>	5	2	4	2	<b>5</b>	3	4	399
WCV	5	<b>6</b>	<b>6</b>	4	3	4	3	<b>6</b>	442
TrCV	<b>6</b>	<b>6</b>	<b>6</b>	<b>6</b>	<b>6</b>	<b>5</b>	<b>4</b>	<b>6</b>	<b>512</b>

different topics. The learning task is to predict the user’s preferences for the given web pages. In the experiment, we randomly reserve “Bands-recording artists” as source-domain and the three others as target-domain data. The details of datasets are summarized in Table 2. These datasets are chosen because they are highly representative of the real world data we typically encounter. For example, some of them have few instances but have high dimensions, while others have the opposite. In addition, to evaluate the performance of source-domain selection with TrCV, 20-Newsgroup [2] is chosen. It is another primary benchmark of text categorization similar to Reuters-21578. In our study, 16 subcategories from 4 top subjects, including “comp”, “rec”, “sci” and “talk”, are selected to form 8 different datasets of two tasks, “comp vs. rec” and “sci vs. talk”. Data of source-domain and target-domain come from the same top categories but different sub-topics. As shown in Table 3, for “comp vs. rec” task, “graphics vs. autos” is chose as the target-domain and three others are treated as source-domains. Similarly, “electronics vs. mideast” is target-domain in “sci vs. talk” task, while others are source-domains. Moreover, for SyskillWebert, Reuters-21578 and 20-Newsgroup, only 500 features with highest information gains are selected.

## 5.2 Experiment Procedure

*Selection among Different Algorithms.* As a comparison, the parameters of traditional classifiers are set as the default values in Weka<sup>1</sup> and those of transfer learning approaches are chosen as the values which are suggested in the corresponding papers. In addition, for TrAdaBoost, SVM with polynomial kernel is set as base model; for LWE, five traditional classifiers stated above with default parameters are the base models. There are 8 approaches, thus the number of comparison is  $C_8^2 = 28$ . Table 4 and Figure 4(a) present correlation measure values for each domain transfer datasets, given by five competitive approaches:

<sup>1</sup> [www.cs.waikato.ac.nz/ml/weka/](http://www.cs.waikato.ac.nz/ml/weka/)

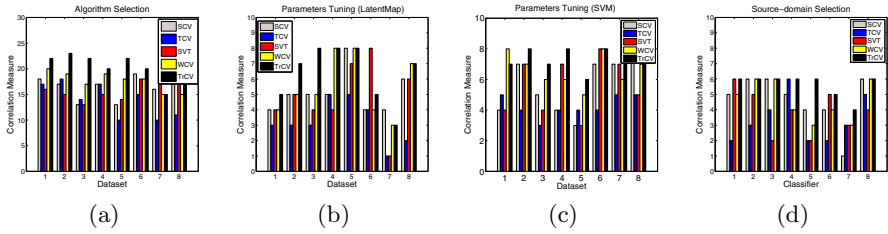
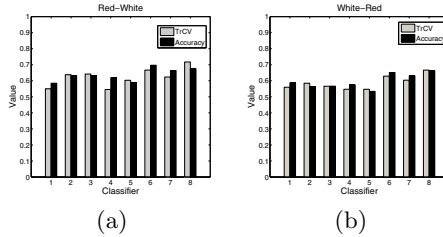


Fig. 4. The comparison of TrCV with other validation methods



Method	NB	SVM	C45	KNN	Ng	TA	LM	LWE
Red-White								
TrCV	0.550	0.638	0.642	0.546	0.603	0.667	0.624	0.717
Accuracy	0.585	0.633	0.632	0.618	0.590	0.697	0.664	0.675
White-Red								
TrCV	0.560	0.585	0.566	0.547	0.547	0.629	0.604	0.667
Accuracy	0.588	0.564	0.567	0.574	0.534	0.651	0.631	0.662

Fig. 5. The comparison between TrCV’s accuracy and the true accuracy

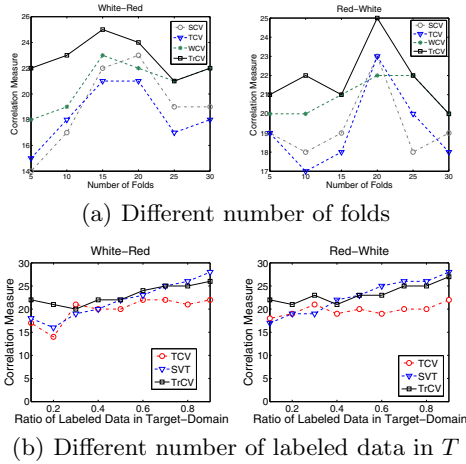
SCV, TCV, STV, WCV and TrCV. Dataset 1 ~ 8 correspond to those in Table 4. It is evident that TrCV achieves the best performance in 6 out of 8 runs. Due to “distribution gap” between source and target domains, SCV fails to select the better model among the comparisons most of the time. To be specific, the correlation value got by SCV is just 18 on the Red-White set and no more than 17 on the Reuters collection. In addition, TCV and STV also fail to select a better model. This can be ascribed to the limited number of labeled data in the target-domain. One classifier performing well in this small subset can not guarantees its generalizability over the whole target-domain collection. However, the proposed approach, TrCV, which considers the difference on marginal distribution and conditional possibility between source and target domains, has a much better performance. Specifically, we notice that TrCV performs better than SCV by at least 4 in correlation value on Wine Quality collection, and as high as 9 on the Reuters-21578 collection. Moreover, the performance of TrCV is better than WCV consistently. The main reason is that although WCV reduces the difference of marginal distribution, it still selects those models which approach conditional distribution of source-domain in stead of target-domain. Thus, as analysed in the section 2, these models can not guarantee their performances in

target-domain. This also affords an evidence that reverse validation is necessary under the context of transfer learning.

In addition, the advantage of TrCV over STV is that TrCV explores the power of unlabeled data and multiple validations, thus reducing the variance of the testing. As analyzed, these characteristics of TrCV reduce the distribution gap between two domains during algorithm selection. This, from the empirical perspective, provides justification to the analysis in Section 4. In addition, Figure 5 plots TrCV values and accuracies of each classifiers in Wine collection. It is intuitive that when one classifier achieves a higher TrCV value, it gets a higher accuracy with high confidence. In other words, accuracy obtained by TrCV is highly correlated to the true accuracy. Moreover, three transfer learning algorithms beat those traditional classifiers because they accommodate the distribution gap between two domains. Classifier 1 ~ 8 in the figure correspond to those list in the table.

*Parameter Tuning.* We select SVM and LatentMap as the learning models and generate two tasks. The first one is to select a suitable margin parameter  $C$  for SVM (from  $10^{-2}$  to  $10^2$ ) and the second one is to tune a good number of nearest neighbors for LatentMap (from 5 to 45). The size of these two parameter set is 5, so we get  $C_5^2 = 10$  comparisons. Table 4 and Figure 5(b) and (c) summarize the correlation values of baselines: SCV, TCV, STV and WCV and the proposed criteria TrCV on 8 datasets. Clearly, TrCV achieves higher correlation value (from 1 to 4 higher in 6 out of 8 datasets) than the corresponding baseline approaches on tuning the parameters of LatentMap and performs best in 7 out of 8 cases when we adjust the margin parameter in SVM. For example, on the Red-White dataset, the correlation value has been improved from 5 achieved by SCV and WCV to 7 by the proposed TrCV. More importantly, in total 16 comparisons, TrCV beats WCV consistently with only one exception in RW dataset when tuning margin parameter of SVM. On the other hand, two exceptions happened on the SyskillWebert collection. We observe that TrCV fails to tune the best parameters for LatentMap and does not have significant improvements to tune SVM. This can be ascribed to the limited number of data in both domains that makes the density ratio estimation imprecise and the reverse validation can not reflect the approximation error to the true conditional distribution significantly as shown in Eq.(13).

*Source-domains selection.* We aim to select a best source-domain among multiple candidates. Two comparisons are involved. One is to evaluate the ability of TrCV to select among source-domains when the model is fixed, another is to test whether TrCV can select the best pair of source-domain and classifier given a set of classifiers and a set of source-domains. The result is presented in Table 5 and Figure 5(d). For the first evaluation, both datasets have 3 candidate source-domains, thus the number of comparison is  $2 \times C_3^2 = 6$ . Among them, TrCV achieves the best performance over all 8 tasks in the correlation measure. In particular, TrCV beats SCV by as much as 5 times while it defeat WCV by 7 times. Table 5 also presents the second evaluation results over 2 data collections,



**Fig. 6.** Parameter Analysis

where  $2 \times C_{(8 \times 3)}^2 = 552$  comparisons are obtained. We denote the result of this comparison as “Pr”. Obviously, under this setting, TrCV still performs better than SCV, STV, WCV and TCV over two datasets, implying the TrCV can still select the best pairs of source-domains and algorithms. The performance improvement is due to density ratio weighting and reverse validation that effectively accommodate the difference between two domains. For WCV, although it boost the ability of SCV with density ratio weighting, it does not perform well due to the ignoring the conditional distribution shift.

*Parameter Analysis.* Two extended experiments were conducted on the Wine Quality collection to test the parameter sensitivity and the relationship between the number of labeled target-domain data and correlation value, *corr*. As shown in Section 3, the number of folds need be set before running TrCV. In addition, those labeled target-domain data affect the accuracy of TrCV to selecting a good model or a source-domain as shown in Eq. (13).

For sensitivity testing, we vary the value of folds from 5 to 30 with step size 5 to perform algorithm selection over 8 candidate approaches. As a comparison, we also attach the results obtained by SCV, TCV and WCV. The results are presented in Figure 5(a). Obviously, TrCV achieves the highest correlation value under all settings. This clearly demonstrates TrCV’s advantage over SCV, TCV and WCV. In addition, we test TrCV when the number of labeled data  $\ell$  increases from  $0.1 \times |T|$  to  $0.9 \times |T|$  by comparing with TCV, SVT.  $|T|$  is the number of data in target-domain. The results are presented in Figure 5(b). Overall, three criteria achieve a higher value with more labeled data and SVT performs better than TrCV when the number of labeled data is significantly large. With more labeled data in target-domain, SVT can obtain more precise estimate to the prediction accuracies of remaining target-domain data. However, when only a few labeled data ( $< 0.4 \times |T|$ ) can be obtained in the target-domain, the performance of TrCV is much better than both SVT and TCV.

## 6 Related Work

Many solutions for transfer learning have been proposed previously, such as but not limited to [12–14], while few approach has been studied to select models or source-domains. Though several existing standard techniques [4–6] can be applied for model selection, they fail to work in transfer learning due to the distribution shift between source and target domains. Two recent approaches [8, 15] have been proposed for model selection in covariant shift or sample selection bias. The method in [8] “WCV” adapts the density ratio into cross validation to handle unbiased estimation under covariant shift. The technique described in [15] performs “Reverse Testing” to select model under sample selection bias. We notice that “Reverse Testing” evaluates or rather “orders” the ability of one model based on another model and does not apply density ratio weighting that returns an estimated value, that is different from the method proposed in this paper. In addition, both of them do not consider the conditional distribution shift which may make them fail under transfer learning context as demonstrated in Section 2.1. Beside these, some techniques have been proposed to estimate the density ratio directly, including Kullback-Leibler importance estimation procedure [9] and nonparametric kernel mean matching (KMM) method [10]. The former one finds the density ratio to minimize the KL-divergence between two domains while the latter estimates by making the discrepancy between means of two domains small. On the other hand, works in [16] solved the similar problems under the context of meta-learning, including algorithm selection, parameter tuning and dataset selection.

## 7 Conclusion

Several challenges need to be resolved in order to make transfer learning methods practical: algorithm selection, parameter tuning and source-domain data selection. Traditional approach fails to solve these problems well due to the distribution gap between two domains. This paper firstly formulates a general criterion followed by proposing a transfer cross validation (TrCV) method. It works by applying density weighting to reduce the difference between marginal distributions of two domains, as well as utilizing reverse validation to measure how well a model approximates the true conditional distribution of target-domain. Formal analysis demonstrates that the newly proposed general criterion has a generalization bound on target-domain, and the confidence of transfer cross validation can also be bounded. Empirical studies under different tasks demonstrate that TrCV has higher chance to select the best models, parameters or source-domains than traditional approaches. In summary, it achieves the best in 28 out of 33 cases comparing with all baselines. Importantly, by considering both marginal and conditional distribution shift, the proposed TrCV approach outperforms in 23 out of 33 cases than WCV [8], a recently proposed method that only considers marginal distribution but ignores difference in conditional distribution.

## References

1. Pan, S.J., Yang, Q.: A survey on transfer learning. Technical Report HKUST-CS08-08, Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China (November 2008)
2. Asuncion, A., Newman, D.J.: uci machine learning repository (2007), <http://www.ics.uci.edu/mllearn/MLRepository.html>
3. Lewis, D.D., Yang, Y., Rose, T.G., Li, F.: Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research* 5, 361–397 (2004)
4. Akaike, H.: A new look at the statistical model identification. *IEEE Transactions on Automatic Control* 19(6), 716–723 (2003)
5. Schwarz, G.: Estimating the dimension of a model. *The Annals of Statistics* 6(2), 461–464 (1978)
6. Vapnik, V.: *Statistical Learning Theory*. Wiley, New York (1998)
7. Shimodaira, H.: Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference* 90(2), 227–244 (2000)
8. Sugiyama, M., Krauledat, M., Müller, K.R.: Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research* 8, 985–1005 (2007)
9. Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P.V., Kawanabe, M.: Direct importance estimation with model selection and its application to covariate shift adaptation. In: *NIPS '07: Proceedings of the 2007 Conference on Advances in Neural Information Processing Systems*, vol. 20, pp. 1433–1440. MIT Press, Cambridge (2008)
10. Huang, J., Smola, A.J., Gretton, A., Borgwardt, K.M., Schölkopf, B.: Correcting sample selection bias by unlabeled data. In: *NIPS '06: Proceedings of the 2006 Conference on Advances in Neural Information Processing Systems*, vol. 19, pp. 601–608. MIT Press, Cambridge (2007)
11. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: *IJCAI'95: Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 1137–1143. Morgan Kaufmann Publishers Inc, San Francisco (1995)
12. Dai, W., Yang, Q., Xue, G.R., Yu, Y.: Boosting for transfer learning. In: *ICML '07: Proceedings of the 24th International Conference on Machine Learning*, pp. 193–200. ACM, New York (2007)
13. Xie, S., Fan, W., Peng, J., Verscheure, O., Ren, J.: Latent space domain transfer between high dimensional overlapping distributions. In: *WWW '09: Proceedings of the 18th International Conference on World Wide Web*, pp. 91–100. ACM, New York (2009)
14. Gao, J., Fan, W., Jiang, J., Han, J.: Knowledge transfer via multiple model local structure mapping. In: *KDD '08: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 283–291. ACM, New York (2008)
15. Fan, W., Davidson, I.: Reverse testing: an efficient framework to select amongst classifiers under sample selection bias. In: *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 147–156. ACM, New York (2006)
16. Brazdil, P., Giraud-Carrier, C., Soares, C., Vilalta, R.: *Metalearning: Applications to Data Mining*. In: *Cognitive Technologies*. Springer, Heidelberg (2009)

# Fast, Effective Molecular Feature Mining by Local Optimization

Albrecht Zimmermann<sup>1</sup>, Björn Bringmann<sup>1</sup>, and Ulrich Rückert<sup>2</sup>

<sup>1</sup> Katholieke Universiteit Leuven, Celestijnenlaan 200A, B-3001 Leuven, Belgium  
{albrecht.zimmermann,bjoern.bringmann}@cs.kuleuven.be

<sup>2</sup> UC Berkeley, EECS Department, 750 Sutardja Dai Hall #1776, Berkeley, CA  
94720-1776, USA  
rueckert@eecs.berkeley.edu

**Abstract.** In structure-activity-relationships (SAR) one aims at finding classifiers that predict the biological or chemical activity of a compound from its molecular graph. Many approaches to SAR use sets of binary substructure features, which test for the occurrence of certain substructures in the molecular graph. As an alternative to enumerating very large sets of frequent patterns, numerous pattern set mining and pattern set selection techniques have been proposed. Existing approaches can be broadly classified into those that focus on minimizing correspondences, that is, the number of pairs of training instances from different classes with identical encodings and those that focus on maximizing the number of equivalence classes, that is, unique encodings in the training data. In this paper we evaluate a number of techniques to investigate which criterion is a better indicator of predictive accuracy. We find that minimizing correspondences is a necessary but not sufficient condition for good predictive accuracy, that equivalence classes are a better indicator of success and that it is important to have a good match between training set and pattern set size. Based on these results we propose a new, improved algorithm which performs local minimization of correspondences, yet evaluates the effect of patterns on equivalence classes globally. Empirical experiments demonstrate its efficacy and its superior run time behavior.

## 1 Introduction

The field of *structure-activity relationships* deals with the problem of predicting the biological or chemical activity of molecules. For example, a researcher might want to learn a model predicting whether or not a particular compound inhibits tumor growth. Such a model could then be used to reduce the amount of in-vitro experimentation. Since the molecular structure of a molecule can be easily encoded by a labeled graph, structure-activity learning problems have been a fertile field for structured data mining. Typically, data mining in this context starts from a data set of molecular compounds that fall into two activity classes, e.g. inhibiting cancer growth or not inhibiting it. Then, a first mining step generates a set of patterns. The patterns in this set are subsequences, subtrees, or



subgraphs of the molecular graphs in the database. This means that one can use the patterns as description features for classification. In this representation a molecular graph is re-encoded as a bit-vector where each bit indicates presence or absence of a specific pattern. Finally, this propositionalized data representation can then be handled by established machine learning techniques such as *Support Vector Machines* (SVM) to learn a classifier.<sup>1</sup>

The traditionally first approaches to the feature generation problem involved general substructure mining tools. For instance, some systems just generate all patterns whose occurrence frequency exceeds some predefined threshold. This was motivated by the use of *fingerprints* in bio-chemistry. In more recent developments, a variety of techniques [10,14,8,2,4,16,1,5] have been proposed to avoid the unnecessarily large feature sets and long runtimes of these early approaches. Newer approaches usually either post-process a set of patterns to select a relatively small subset or iteratively refine a candidate pattern set so that some quality measure is optimized.

Finding smaller pattern sets is motivated by the observation that popular similarity measures fail to work well, if the training instances are represented by many redundant features that check for highly similar substructures.

There is, however, less agreement on a second question connected to finding good pattern sets for the encoding of molecules: Is it more important that

1. instances belonging to different classes are encoded differently from each other *or that*
2. instances are generally encoded differently from each other, no matter the class label?

The approach whose selection strategy is driven strongest by the former option is the FCORK technique introduced in [16] while the post-processing techniques proposed in [10] and [2] mainly focus on the latter one. Other techniques consider the two extremes to differing degrees. More generally, it is not very well understood which quality measures a pattern set should optimize in order to lead to classifiers with high predictive accuracy.

Our contribution to deciding this question in this paper is two-fold:

- i We describe and compare several state-of-the-art approaches in Section 3 and evaluate the generated feature sets according to various quality measures in Section 4. In particular, we investigate whether the number of correspondences, a class-dependent measure, or the number of equivalence classes, a class-agnostic measure, are better indicators of good prediction accuracy.
- ii Based on the obtained insights (and the identification of the currently most successful technique), we propose a new iterative mining technique in Section 5 and show that it improves on existing techniques in a variety of ways in Section 6.

The quality measures used in the paper to rate pattern sets are introduced in Section 2 and we present our conclusions in Section 7.

<sup>1</sup> Some approaches to structural prediction [7,17,11] integrate the mining step and learning step.

## 2 Quality Measures for Pattern Sets

Let us start by introducing the notions and quality measures for pattern sets used in the rest of the paper. Given a data set  $\mathbb{D}$ , a pattern language  $\mathcal{L}$  and a pattern constraint  $c$ , the result of a constrained mining operation is a *theory*  $Th(\mathbb{D}, \mathcal{L}, c)$ , namely a set of patterns satisfying the constraint(s)  $c$  (cf. [11]). This is e.g. the usual approach in *frequent* pattern mining. If the pattern constraint employed is a measure  $\phi$  and the goal is to mine the top- $k$  patterns according to this measure, the resulting theory is denoted by  $Th_k(\mathbb{D}, \mathcal{L}, \phi)$ .

Let each pattern  $p$  be associated with a function  $p : \mathbb{D} \mapsto \{true, false\}$ . We define  $p(t) = true$  if  $p$  matches  $t$ , and  $p(t) = false$  otherwise. Associating a pattern with this boolean function does allow us to consider each pattern as a binary feature. In addition, we assume a binary class labeling function  $c : \mathbb{D} \mapsto \{+, -\}$ . Given a pattern set  $\mathbb{S} \subseteq \mathcal{L}$ , we define an equivalence relation  $\sim_{\mathbb{S}}$  on  $\mathbb{D}$  as:

$$\sim_{\mathbb{S}} \equiv \forall t_i, t_j \in \mathbb{D} : t_i \sim_{\mathbb{S}} t_j \Leftrightarrow \forall p \in \mathbb{S} : p(t_i) = p(t_j)$$

Thus, two data instances are considered to be equivalent under  $\mathbb{S}$  if they share exactly the same patterns. Using the equivalence relation  $\sim_{\mathbb{S}}$ , an *equivalence class* or *block* is defined in the following way:

$$[t] =: \{t' \in \mathbb{D} : t' \sim_{\mathbb{S}} t\}$$

The *partition* or *quotient set* of  $\mathbb{D}$  over  $\mathbb{S}$  finally, is defined as:

$$\mathbb{P} = \mathbb{D} / \sim_{\mathbb{S}} = \{[t] : t \in \mathbb{D}\}$$

The partition thus induced by a pattern (and therefore feature) set corresponds to the number of different bit-strings that can be presented to a machine learning technique attempting to build a classifier based on them. Instances assigned to the same block are effectively indistinguishable and will therefore be classified in the same way. Some systems therefore maximize the number of equivalence classes in the pattern sets:

$$eq_{\mathbb{D}}(\mathbb{S}) = |\{\mathbb{D} / \sim_{\mathbb{S}}\}|$$

While this might be slightly worrying in case of large blocks consisting of only one class (lump judgments can be problematic to generalize), it is clearly counter-productive if both classes are present in a block since this introduces unavoidable errors. The authors of [16] define the concept of *correspondence*: Two data instances  $t_1, t_2$  form a *correspondence* under the equivalence relation  $\sim_{\mathbb{S}}$  iff  $c(t_1) \neq c(t_2) \wedge t_1 \sim_{\mathbb{S}} t_2$ . Here,  $c(t)$  denotes the class label of a training instance. With this, the number of correspondences is:

$$corr_{\mathbb{D}}(\mathbb{S}) = \sum_{\mathbb{B}_i \in \mathbb{D} / \sim_{\mathbb{S}}} |\{t \in \mathbb{B}_i : c(t) = +\}| \cdot |\{t \in \mathbb{B}_i : c(t) = -\}|$$

A pattern set which minimizes the number of correspondences provides more information to the learning algorithm, because it encodes the specific properties

better that make each example differ from the other ones. On the other hand, this might lead to overfitting or large pattern sets. To overcome this, the authors of [10] use joint-entropy as a pattern set quality measure:

$$\text{je}_{\mathbb{D}}(\mathbb{S}) = - \sum_{\mathbb{B}_i \in \mathbb{D} / \sim_{\mathbb{S}}} \frac{|\mathbb{B}_i|}{|\mathbb{D}|} \log_2 \frac{|\mathbb{B}_i|}{|\mathbb{D}|}$$

Finally, the authors of [14] use a dispersion score to rate the quality of a pattern set:

$$\text{disp}_{\mathbb{D}}(\mathbb{S}) = \frac{1}{|\mathbb{D}|^2} \sum_{p_i, p_j \in \mathbb{S}} (|\{t \in \mathbb{D} : p_i(t) = p_j(t)\}| - |\{t \in \mathbb{D} : p_i(t) \neq p_j(t)\}|)^2$$

Since this score grows with the size of the pattern set, we use the following normalized version to compare pattern sets in Section 4:

$$\text{dispNorm}_{\mathbb{D}}(\mathbb{S}) = \text{disp}_{\mathbb{D}}(\mathbb{S}) / (|\mathbb{S}| \cdot |\mathbb{S}| - 1) / 2$$

To evaluate and compare pattern sets generated by different approaches we employ a SVM with the popular Tanimoto kernel. Given two molecules encoded as bit-vectors  $\mathbf{x}, \mathbf{y} \in \{0, 1\}^d$  using  $d$  mined patterns, it is defined as:

$$\mathcal{K}_T(\mathbf{x}, \mathbf{y}) = \frac{\sum_1^d \min\{x_i, y_i\}}{\sum_1^d \max\{x_i, y_i\} - \sum_1^d \min\{x_i, y_i\}}$$

Obviously, if both vectors' components are mostly 1 and only a few 0s distinguish them, even kernel values for different instance pairs will be close to each other. Such badly scaled kernels are known to be problematic for successful prediction.

### 3 Existing Approaches

The work concerned with finding good pattern sets for classification falls into two categories: 1) post-processing of a set of patterns, which is similar to feature selection techniques from machine learning, and 2) iterative pattern set mining techniques that extend and improve a candidate pattern set.

The first type of approaches has the advantage that only a single pattern mining run has to be performed. To ensure that the variety of patterns is high enough to enable the extraction of a suitable subset, it is usually necessary to mine a very large amount of patterns. This makes the post-processing step (and possibly the mining step) potentially expensive, especially if some features are not informative on their own, but lead to high predictive accuracy when considered together. In [10], this problem is addressed by assuming a size-constraint that is always present. The authors discuss several desirable measures and discuss their potential mutual exclusivity. In a related work, the authors gave an algorithm for finding pattern sets maximizing the *joint entropy* (*JE*) criterion. JE effectively measures the ability of a pattern set to induce a partition of equally sized equivalence classes.

Most existing systems differ in the search strategy and the criterion used to rate the quality of a pattern set. For instance, the system presented in [4] uses greedy search and a supervised quality measure, which is based on the correlation of a pattern with a class and the similarity to earlier patterns. The search uses a database coverage constraint to control pattern set size. In this manner, it minimizes the expected number of correspondences and maximizes the expected number of equivalence classes. The approaches discussed in [2] are also based on greedy search, but use an unsupervised quality measure, which quantifies how well sets of patterns partition the data, *without* any reference to a class label. The search strategy in [8], finally, is a local search technique, which selects a non-redundant set of patterns from randomly sampled maximal graph-structures.

The second style of structured pattern mining approaches uses an iterative approach, where a candidate pattern set is extended step by step. Iterative mining has the advantage that the mining runs in later iterations can be tailored towards the shortcomings of the previously generated pattern sets. This is in contrast to post-processing, where one assumes (or hopes) that the number of generated patterns is large enough to allow the extraction of an informative subset. On the other hand, there is often no clear way to identify how many iterations will be needed to mine a useful set, and the cumulative computational cost of several pattern mining runs can become rather high. Iterative mining can be performed in two settings: 1) as *sequential mining*: mining is performed strictly sequentially, only one mining process per iteration. This leads to a clear relationship among patterns: each pattern is influenced by those that were mined before it and influences those mined after it. For efficient handling, some algorithms modify the underlying graph database between iterations. The second setting is that of 2) *parallel mining*: in any iteration several mining processes can run in parallel. generally, the results of one mining process are used to split the database into two or more parts, so the mining steps in later iterations work on smaller subsets of data instances.

The two existing sequential approaches differ somewhat. The search strategy in [14] is stochastic local search to maximize the *class-correlated dispersion score* of patterns with regard to patterns mined in earlier iterations. This score, similar to the approach in [4], trades off class-correlation with the similarity of patterns with regard to coverage in the data, or in other words, the minimization of correspondences and the maximization of equivalence classes. In [16], mining for patterns themselves is performed exhaustively using an upper bound and minimum support criterion to heuristically optimize the submodular *correspondence-based quality criterion*. Maximization of equivalence classes, if it happens at all, is only a side effect of the process. The combination of exclusive focus on correspondence minimization and sequential mining leads to *very* compact sets of patterns.

The two approaches falling into the latter group [11,5], use the usual decision tree induction mechanism: a single pattern is mined using *information gain* and the data split according to matching of the pattern, before the algorithm recurses

on the derived subsets. Since information gain rewards class discrimination, repeated applications will lead to the minimization of correspondences while the fact that mining is performed on subsets of the data can lead to the split of instances from the same class in other parts of the data, thus increasing the number of equivalence classes.

## 4 Comparison of Systems and Quality Measures

In this section we describe experiments comparing the various approaches outlined earlier and the different quality measures that are used in literature to rate pattern sets for classification. In particular, we investigate *joint entropy*, the normalized and unnormalized *dispersion score*, the number of *correspondences* and the number of *equivalence classes*. The main goal here is to explore which approaches and quality measures lead to pattern sets with high predictive accuracy as measured by the AUC of the final classifier. The trends identified in these experiments can then be used to design fast algorithms generating small feature sets with good predictive accuracy. We investigate the following systems:

- Baseline. The 500 most general (shortest) free graph patterns mined under a minimum support threshold of 5%.
- PICKER\*. This is a technique introduced in [2], using the inference measure and no threshold. The underlying pattern set consists of all free graph patterns mined with minimum support 5%, i.e. a superset of the baseline.
- DISP. The stochastic local search technique from [14], optimizing class-correlated dispersion score.
- FCORK as introduced in [16], whose authors supplied us with an executable. Since unconstrained experiments using this technique did not terminate on the NCI data and the cancer data set, a 5% minimum support constraint was used.
- DTM. This is an implementation similar to MBT, introduced in [5]. We were unfortunately not able to obtain an executable of the algorithm from the authors of this paper. Since we had published a similar technique in 2005 under the name TREE<sup>2</sup> [1], however, we extended our implementation to work on graph-structured data and discarded the decision tree after mining. Based on past results [3], we chose to mine sequential patterns which perform as well as graph-structured patterns. We will refer to this technique as “decision tree-like miner” (DTM) in the following.

Generally, we chose the parameters so that only small frequency or weak selectivity constraints were imposed. This ensures that the systems have a large number of candidate patterns available for inclusion. Since different methods generate pattern sets of varying sizes, it is often difficult to compare them directly. To allow for a fair comparison, we thus sometimes cut back the number of features to match the number produced by other techniques. Cutting-back is done by keeping the  $k$  highest-ranked patterns/those mined in the  $k$  first iterations, with  $k$  derived from the size of competing techniques’ output.

For the experiments, we used the data sets shown in Table 1. This includes the NCI data sets first reported in [15] (specifically those on which FCORK with minimum support five percent terminated in reasonable time), as well as the *Blood Brain Barrier*, *NCTRER*, *Yoshida*, and *Cancer* data sets used in [14]. To evaluate the quality of the derived encodings, we performed a 10-fold cross-validation, using an SVM [9] with Tanimoto Kernel, with the C-parameter set to 1.0. This value gave good performance over the entire range of data sets.

**Table 1.** An overview of the used data sets

Data set	instances	majority class
NCI 786_0	3154	1648
NCI A549_ATCC	3359	1710
NCI CAKL1	3221	1678
NCI CCRF_CEM	3131	1995
NCI COLO_205	3279	1748
NCI SF_539	3045	1728
Blood Brain Barrier	373	248
NCTRER	208	125
Yoshida	238	143
Cancer	30796	15590

As a first experiment we investigated how the number of features affects classification accuracy. If prediction accuracy increases with the number of features, this would indicate that strict pattern set selection is misguided and permissive feature generation approaches should be preferred. In Figure 1 we plot the average number of features selected per fold against the AUC, labeled by data set. The plot shows that accuracy increases moderately or not at all with the number of features. It is remarkable that the Pearson correlation coefficient between number of features and AUC is actually positive for the larger datasets (0.4 for NCI and 0.57 for Cancer), but negative for the smaller data sets (-0.02 to -0.12 for Blood Brain Barrier, NCTRER, Yoshida). While this is not statistically significant, it seems to be consistent with the results in [13]: pattern set size should increase with the number of training instances, but overfitting is not as severe as in many other classification settings.

In the second experiments, we examine which pattern set criterion is a good indicator of prediction accuracy. Figures 2–5 give the scatter plots for the normalized dispersion score, average joint entropy, number of equivalence classes and number of correspondences. The results are mixed, but there are a few interesting insights. First of all, joint entropy and the number of equivalence classes appear to be fairly well correlated to prediction accuracy. Indeed, the following correlation coefficients are significant on the 99% significance level: The correlation between joint entropy and AUC is between 0.8 and 0.95, with the only outlier at 0.72 for the NCTRER data set. For the number of equivalence classes, the Pearson correlation coefficient is large for the bigger data sets (> 0.95 for Cancer and NCI), but still reasonable for the smaller ones (0.63–0.8 for NCTRER, Blood Brain Barrier, Yoshida). Looking at Figure 5, one can see that

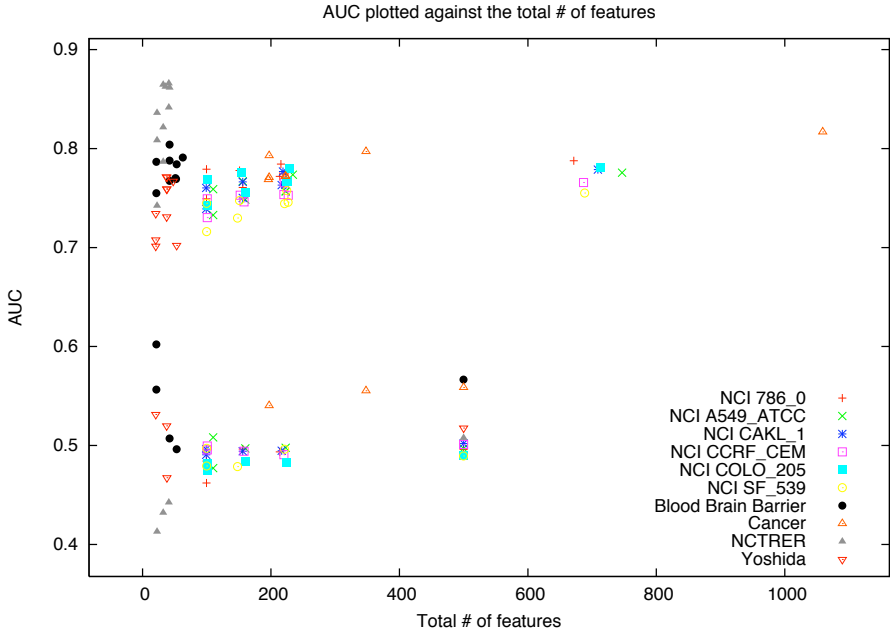
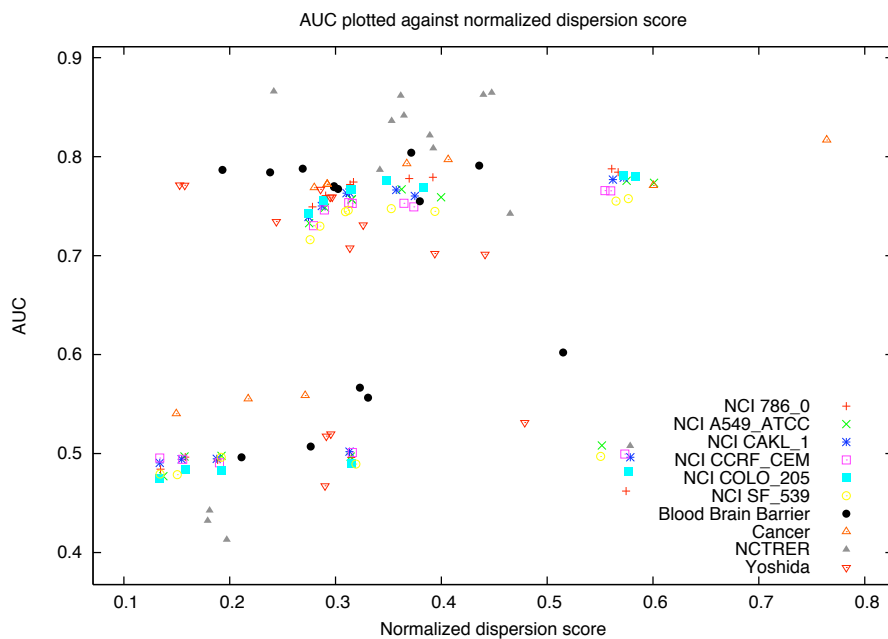


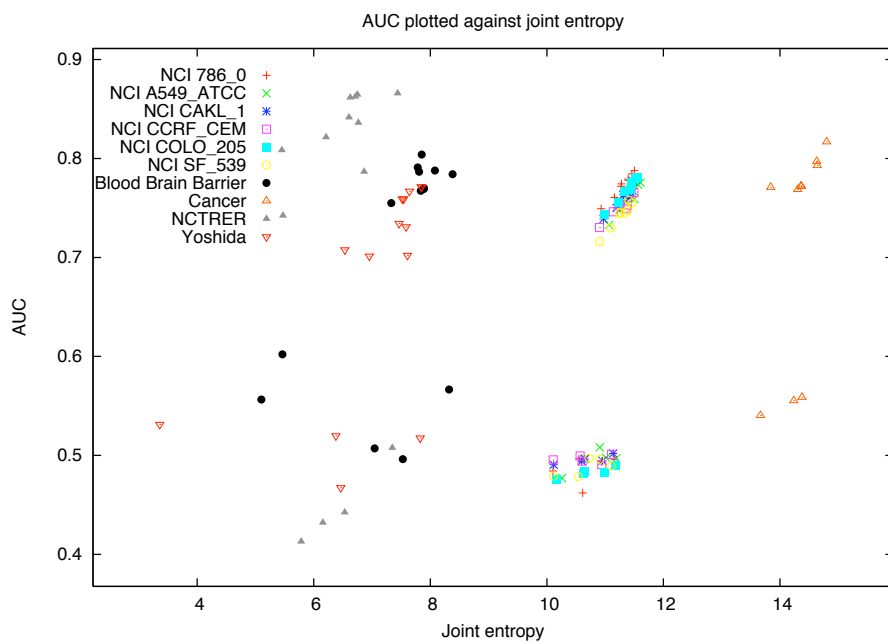
Fig. 1. AUC plotted against the average number of features selected

a large number of correspondences always indicates bad AUC performance. This means a pattern mining system needs a way to minimize the number of occurrences to be successful. Unfortunately, it is clearly not enough to only optimize for correspondences: there are a number of settings where pattern sets with low number of correspondences still lead to terrible predictive accuracy. It is clear that successful systems also need to optimize the variety within the instances of the same class. This is what joint entropy and the number of equivalence classes measure. Finally, the dispersion score is only slightly correlated with AUC. The dispersion score varies between around  $-0.10$  for the smaller data sets and around  $+0.25$  for the larger ones. It is noteworthy, though, that the dispersion score improves if class information is included.

Overall, one can conclude that supervised pattern mining scores tend to be better indicators of prediction accuracy and that the characteristics change between larger and smaller datasets. This means that successful techniques should include information about the class label during pattern generation and that the number of generated patterns should be in relation to the number of training instances. To see how the evaluated methods succeed in this regard, Table 2 shows the average rankings of different methods w.r.t. the four quantitative measures and AUC. Highest-ranked in terms of AUC is DTM, which also ranks very high in terms of equivalence classes *and* correspondences, coming second in terms of equivalence classes only to PICKER\* that directly optimizes those. Apparently, DTM's approach to class-sensitive pattern generation and its ability to relate

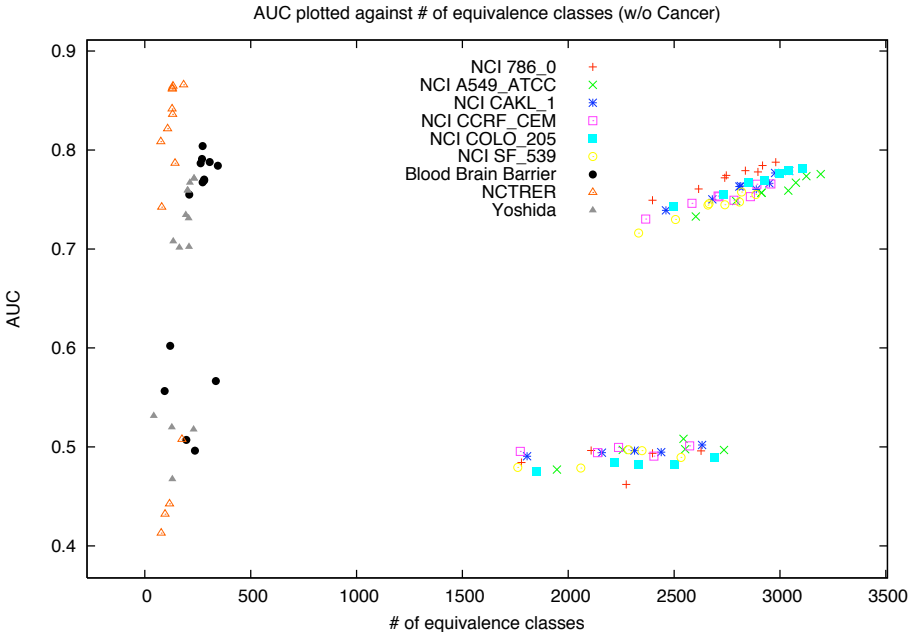


**Fig. 2.** AUC plotted against the average normalized dispersion score

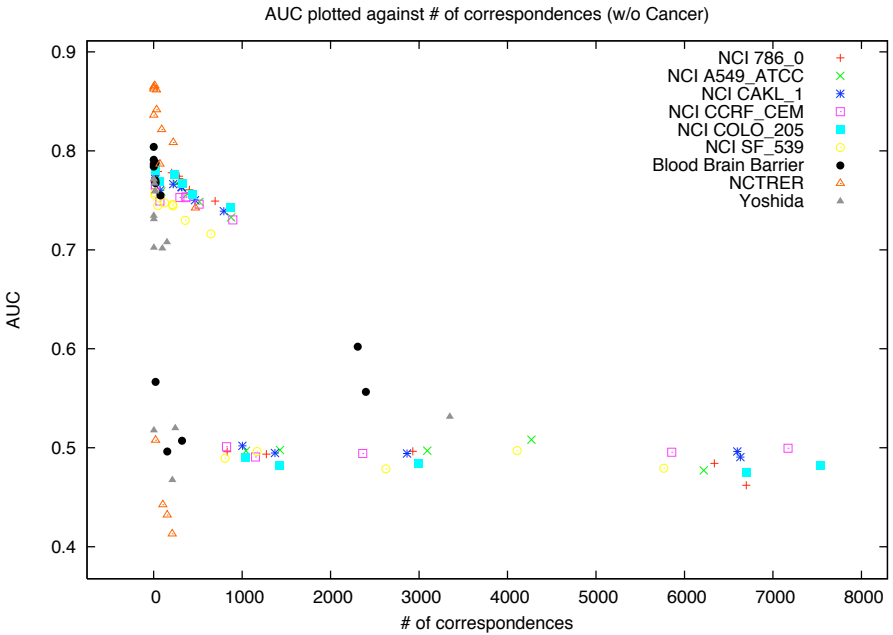


**Fig. 3.** AUC plotted against the average joint entropy





**Fig. 4.** AUC plotted against the average number of equivalence classes (w/o Cancer)



**Fig. 5.** AUC plotted against the average number of correspondences (w/o Cancer)

**Table 2.** The respective ranking of techniques for different criteria

AUC	# equivalence classes	# correspondences	dispersion score	joint entropy
DTM (1.44)	PICKER* (1.7)	DTM (1.6)	FCORK (1)	PICKER* (1.9)
PICKER* (2)	DTM (2)	FCORK (1.7)	PICKER* (2.1)	DTM (2)
FCORK (3.22)	FCORK (3.4)	PICKER* (2.8)	DISP (3)	FCORK (3.1)
DISP (3.33)	DISP (3.8)	DISP (4.2)	Baseline (4.3)	DISP (4)
Baseline (5)	Baseline (4.1)	Baseline (4.7)	DTM (4.6)	Baseline (4)

**Table 3.** Number of features divided by joint entropy of the feature set

Data set	Baseline	PICKER*	DISP	FCORK	DTM
NCI 786_0	44.89±0.0579	13.68±0.0048	19.71±0.1086	8.74±0.3560	58.39±0.7455
NCI A549_ATCC	44.77±0.1198	13.92±0.0071	19.66±0.1658	9.56±0.3645	64.33±1.1459
NCI CAKL1	44.98±0.1417	13.64±0.0071	19.79±0.0934	8.68±0.3641	61.49±1.1933
NCI CCRF_CEM	45.08±0.1046	13.85±0.0041	20.18±0.0840	8.87±0.4358	59.75±1.1532
NCI COLO_205	44.84±0.114	14.05±0.0078	19.89±0.0798	8.89±0.1390	61.66±0.7693
NCI SF_539	45.11±0.0663	13.02±0.0047	20.17±0.1445	8.79±0.2561	60.07±0.9540
Blood Brain Barrier	60.20±0.1036	6.32±0.0088	6.80±0.4355	2.74±0.1302	8.05±0.4570
NCTRER	68.17±0.3723	5.63±0.0403	6.52±0.1542	3.34±0.1906	5.33±0.3608
Yoshida	64.013±0.0963	4.84±0.0065	6.44±0.2702	2.74±0.1100	6.96±0.5243
Cancer	34.84±0.0059	23.78±0.0033	15.52±0.8953	13.46±0.2409	

feature set size to training set size lead to good overall performance. DTM’s bad dispersion score actually supports this point: the unnormalized dispersion score we report increases in the number of patterns involved. The fact that DTM finishes last in terms of it shows that it generates more features than most competing methods on the large datasets. This is also illustrated in Table 3. Joint entropy can be considered as giving the number of bits that are needed to encode the partition. Since each feature acts in fact as a bit in the encoding, the table essentially gives the average number of patterns per bit of information. One can see that DTM adapts better to the varying training set sizes than for instance FCORK or the Baseline. In fact, it adapts a bit too well and the drawback of its large pattern set sizes can be seen in Table 7, which gives the runtimes. Here, DTM performs worst by a large margin. For the cancer dataset, we had to cancel the run after 100 hours.

In order to find a highly predictive, yet fast technique, it is therefore desirable to keep the advantages of DTM – focus on supervised partition generation and sensitivity to the size and characteristics of the underlying dataset – while doing away with the drawbacks – the overly large number of features and long running times.

## 5 The REMINE Algorithm

DTM, shown in Algorithm 1 iteratively mines patterns in the same way as a binary decision tree is induced on class-labeled data: first, a single test that scores best according to *information gain* is extracted (line 2). In a standard

**Table 4.** Running times per fold for different techniques (the techniques that are not listed had negligible running times)

Data set	FCORK	DISP	DTM
NCI 786_0	4h40m±1h46m	1h2m±7m54s	4h46m±36m48s
NCI A549_ATCC	4h47m±1h53m	1h3m±6m39s	11h31m±4h0m
NCI CAKI_1	3h55m±1h7m	1h5m±6m28s	11h39m±3h17m
NCI CCRF_CEM	4h24m±1h7m	1h1m±9m55s	12h46m±4h55m
NCI COLO_205	3h22m±1h34m	1h5m±7m33s	10h43m±3h33m
NCI SF_539	5h54m±4h16m	1h0m±5m11s	11h37m±4h44m
Blood Brain Barrier	20m12s±11m26s	6.14s±0.98s	7m 2s±1m5.65s
NCTRER	1h10m±38m	1.52s±0.18s	4m31.6s±1m3.75s
Yoshida	1m42s±1m4s	3.16s±0.42s	7m7.5s±1m23.65s
Cancer	10h41m±1h	13h8m±20m	100h+

decision tree, such a test would be an attribute-value pair, while in DTM a graph-structured pattern is mined. The data is then split into two subsets, one on which the test matches, the second on which it does not (line 4), and the operation repeated on the derived subsets (line 5). A problem is that information gain, in contrast to e.g. minimum frequency, is not anti-monotone. However, it is possible to calculate an upper bound for information gain and use it for forward-pruning to make mining for the best pattern according to information gain ( $\mathcal{T}h_1(\mathbb{D}, \mathcal{L}, \phi)$ ) feasible [12].

---

**Algorithm 1.** The DTM algorithm

---

DTM( $\mathbb{D}$ )

- 1:  $\mathcal{F} = \emptyset$
  - 2:  $\mathcal{F}_{new} = \mathcal{T}h_1(\mathbb{D}, \mathcal{L}, \phi)$  – this yields zero or one feature
  - 3: **if**  $\mathcal{F}_{new} \neq \emptyset$  **then**
  - 4:    $\mathbb{P} = \{\mathbb{B}_1, \mathbb{B}_2\} = \mathbb{D} / \sim_{\mathcal{F}_{new}}$
  - 5:    $\mathcal{F} = \mathcal{F}_{new} \cup \text{DTM}(\mathbb{B}_1) \cup \text{DTM}(\mathbb{B}_2)$
  - 6: **end if**
  - 7: **return**  $\mathcal{F}$
- 

Information gain rewards patterns that separate instances with different class labels from each other, in this way reducing correspondences on the subset on which a pattern is mined. We aim at keeping this basic mining process of DTM, still mining a single best pattern from a subset. The difference lies in how we apply the patterns to derive *new* subsets for the following iteration: Instead of using each pattern to split only the subset on which it was mined, we use *all* patterns derived so far to induce a partition on the entire data (line 3 in Algorithm 2).

This can be illustrated by considering the first three iterations: in the first step, both DTM and REMINE mine the best pattern according to information gain and use it to split the whole data set into two subsets. In the second iteration, two more patterns are mined but when these are used to split the subsets, DTM

and REMINE behave differently. DTM splits each subset in two, according to the pattern mined from each, for a maximum of **four** subsets. REMINE, on the other hand, splits each subset according to *both* patterns. This can lead maximally to four *new* subsets from each current one, for a total of maximally **eight** subsets of the data. The maximum number of different blocks that can be encoded by three binary features is eight, which shows that REMINE can use patterns much more efficiently than DTM when it induces a partition. In the third iteration, DTM will therefore mine for four new patterns, while REMINE mines for eight.

Accordingly, the REMINE algorithm consists of a main loop (line 2-10) where in each iteration the whole data set  $\mathbb{D}$  is partitioned according to the current set of features  $\mathcal{F}$  (line 3). Then for each of the blocks  $\mathbb{B}_i$  of the partition  $\mathbb{P}$  the feature giving the highest information gain is extracted (line 6) which are then joined with the previous features (line 8). If the new partition  $\mathbb{P}'$  induced by the resulting enhanced feature set has not changed, the algorithm terminates and returns the found set of features  $\mathcal{F}$ .

---

**Algorithm 2.** The REMINE algorithm

---

```

1:  $\mathcal{F} = \emptyset$  – the initial set of features
2: repeat
3:    $\mathbb{P} = \mathbb{D} / \sim_{\mathcal{F}}$  – partition induced by the current feature set
4:    $\mathcal{F}_{new} = \emptyset$ 
5:   for all blocks  $\mathbb{B}_i \in \mathbb{P}$  do
6:      $\mathcal{F}_{new} = \mathcal{F}_{new} \cup Th_1(\mathbb{B}_i, \mathcal{L}, \phi)$ 
7:   end for
8:    $\mathcal{F} = \mathcal{F} \cup \mathcal{F}_{new}$ 
9:    $\mathbb{P}' = \mathbb{D} / \sim_{\mathcal{F}}$  – partition induced by the new feature set
10: until  $\mathbb{P}' = \mathbb{P}$ 
11: return  $\mathcal{F}$ 

```

---

## 6 Experimental Evaluation

Our goal in proposing the REMINE technique lies in keeping DTM’s good performance in terms of AUC while at the same time reducing the number of patterns mined which should help in achieving lower running times. We therefore repeat our earlier experiments and compare REMINE’s results against those of FCORK, PICKER\*, and DTM. As Table 5 shows, REMINE achieves on average second-best AUC (by a small margin), and performs at least as good as DTM in terms of the quantitative criteria, improving on it for the number of correspondences and the dispersion score. This means that we achieved our goal to keep the good performance and the ranking indicates improvements in the number of features as well. Table 6 confirms this indication, showing that while REMINE does not mine as few patterns as PICKER\* and FCORK, it strongly reduces the size of the feature set compared to DTM. The final aspect to be evaluated is that of running times and as Table 4 shows, along with the reduction in features mined comes a reduction in running times that is so pronounced that REMINE runs even faster than FCORK while mining more features (and leading to better AUC).

**Table 5.** Ranking of techniques for different criteria, including REMINE now

AUC	# equivalence classes	# correspondences	dispersion score	joint entropy
DTM (1.89)	DTM (2.2)	REMINe (1.7)	FCORK (1)	DTM (2.2)
REMINe (2)	PICKER* (2.4)	FCORK (2.4)	PICKER* (2.2)	PICKER* (2.6)
PICKER* (2.78)	REMINe (2.6)	DTM (2.5)	DISP (3.3)	REMINe (2.7)
FCORK (4.11)	FCORK (4.4)	PICKER* (3.6)	REMINe (3.7)	FCORK (4)
DISP (4.22)	DISP (4.6)	DISP (5.2)	Baseline (5.2)	DISP (4.7)
Baseline (6)	Baseline (4.8)	Baseline (5.6)	DTM (5.6)	Baseline (4.8)

**Table 6.** Number of features divided by joint entropy of the feature set, including REMINE now

Data set	PICKER*	FCORK	DTM	REMINe
NCI 786_0	13.68±0.0048	8.74±0.3560	58.385±0.745544	18.79±1.0739
NCI A549_ATCC	13.92±0.0071	9.56±0.3645	64.33±1.14595	20.23±0.9974
NCI CAKL1	13.64±0.0071	8.68±0.3641	61.49±1.19338	19.04±0.9252
NCI CCRF_CEM	13.85±0.0041	8.87±0.4358	59.75±1.15327	19.28±0.6363
NCI COLO_205	14.05±0.0078	8.89±0.1390	61.66±0.76938	19.85±0.7491
NCI SF_539	13.02±0.0047	8.79±0.2561	60.07±0.95406	19.68±1.059
Blood Brain Barrier	6.32±0.0088	2.74±0.1302	8.05±0.4570	5.36±0.4502
NCTRER	5.63±0.0403	3.34±0.1906	5.33±0.3608	4.77±0.2409
Yoshida	4.84±0.0065	2.74±0.1100	6.96±0.5243	4.95±0.2858
Cancer	23.78±0.0033	13.46±0.2409		71.55±2.5414

**Table 7.** Running times per fold for different techniques (the techniques that are not listed had negligible running times)

Data set	FCORK	DTM	REMINe
NCI 786_0	4h40m±1h46m	4h46m±36m48s	59m27s±13m56s
NCI A549_ATCC	4h47m±1h53m	11h31m±4h0m	59m27s±13m56s
NCI CAKL1	3h55m±1h7m	11h39m±3h17m	2h36m±1h57m
NCI CCRF_CEM	4h24m±1h7m	12h46m±4h55m	2h20m±1h51m
NCI COLO_205	3h22m±1h34m	10h43m±3h33m	2h47m±2h17m
NCI SF_539	5h54m±4h16m	11h37m±4h44m	2h59m±1h54m
Blood Brain Barrier	20m12s±11m26s	7m 2s±1m5.65s	2m16.4s±24.2s
NCTRER	1h10m±38m	4m31.6s±1m3.75s	1m48.5s±15.9s
Yoshida	1m42s±1m4s	7m7.5s±1m23.65s	1m53.9s±11.9s
Cancer	10h41m±1h	100h+	13h51m±2h14m

## 7 Conclusions and Future Work

In this paper we evaluated the importance (in terms of AUC) of several selection criteria for pattern set mining. We found that minimizing correspondences is necessary, but not sufficient for predictive classifiers, whereas general partitioning scores are good overall indicators of pattern set quality. We also found that successful methods need to adapt the pattern set size to the characteristics of the datasets. Unfortunately, large pattern sets require many expensive mining steps

and large runtimes. We thus designed a faster and more efficient adaptation of the DTM algorithm. REMINE performs the local pattern mining step in the same way as DTM, but uses *all* patterns instead of a single pattern per iteration on the *entire* data to induce the new partition. Our experimental evaluation showed that REMINE retains the good performance of DTM in terms of AUC, has similar characteristics in the number of equivalence classes and correspondences, reduces the number of patterns mined, and has lower runtimes.

The aforementioned partial redundancy among patterns mined by REMINE could actually go as far as producing completely redundant, e.g. complementary, patterns. Whether removing such redundant patterns or other redundancy control methods would improve or impair the usefulness of the derived encoding is an open question. There is also the issue that decision-tree like iterative miners like REMINE (or MBT and TREE<sup>2</sup>) can be parallelized, giving them a further speed advantage over sequential miners like FCORK. Finally, we were only concerned with graph-structured data in this work, other representations, including unstructured data such as itemsets, remain a topic for future work.

## Acknowledgements

We thank Marisa Thoma for making an executable of the FCORK algorithm available to us. We thank Luc De Raedt for helpful comments and discussion, as well as the anonymous reviewers for their valuable input. The work presented here was partially supported by the European Commission under the 7th Framework Programme FP7-ICT-2007-C FET-Open, contract no. BISON-211898 and by Deutsche Forschungsgemeinschaft, contract no. RU 1589/1-1.

## References

1. Bringmann, B., Zimmermann, A.: Tree<sup>2</sup> - Decision trees for tree structured data. In: Jorge, A., Torgo, L., Brazdil, P., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 46–58. Springer, Heidelberg (2005)
2. Bringmann, B., Zimmermann, A.: One in a million: picking the right patterns. Knowledge and Information Systems 18(1), 61–81 (2009)
3. Bringmann, B., Zimmermann, A., De Raedt, L., Nijssen, S.: Don't be afraid of simpler patterns. In: Fürnkranz, et al. (eds.) [6], pp. 55–66 (2006)
4. Cheng, H., Yan, X., Han, J., Hsu, C.W.: Discriminative frequent pattern analysis for effective classification. In: Proceedings of the 23rd International Conference on Data Engineering, pp. 716–725. IEEE, Los Alamitos (2007)
5. Fan, W., Zhang, K., Cheng, H., Gao, J., Yan, X., Han, J., Yu, P.S., Verscheure, O.: Direct mining of discriminative and essential frequent patterns via model-based search tree. In: Li, Y., Liu, B., Sarawagi, S. (eds.) Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 230–238. ACM, New York (2008)
6. Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.): PKDD 2006. LNCS (LNAI), vol. 4213. Springer, Heidelberg (2006)

7. Geamsakul, W., Matsuda, T., Yoshida, T., Motoda, H., Washio, T.: Performance evaluation of decision tree graph-based induction. In: Grieser, G., Tanaka, Y., Yamamoto, A. (eds.) DS 2003. LNCS (LNAI), vol. 2843, pp. 128–140. Springer, Heidelberg (2003)
8. Hasan, M.A., Chaoji, V., Salem, S., Besson, J., Zaki, M.J.: Origami: Mining representative orthogonal graph patterns. In: Ramakrishnan, N., Zaiane, O. (eds.) ICDM, pp. 153–162. IEEE Computer Society, Los Alamitos (2007)
9. Joachims, T.: Making large-scale support vector machine learning practical. In: Advances in Kernel Methods: Support Vector Learning, pp. 169–184. MIT Press, Cambridge (1999)
10. Knobbe, A.J., Ho, E.K.Y.: Pattern teams. In: Fürnkranz, et al. (eds.) [6], pp. 577–584 (2006)
11. Mannila, H., Toivonen, H.: Levelwise search and borders of theories in knowledge discovery. *Data Mining and Knowledge Discovery* 1(3), 241–258 (1997)
12. Morishita, S., Sese, J.: Traversing itemset lattice with statistical metric pruning. In: Proceedings of the 19th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (2000)
13. Rückert, U.: Capacity control for partially ordered feature sets. In: ECML PKDD '09: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 318–333. Springer, Heidelberg (2009)
14. Rückert, U., Kramer, S.: Optimizing feature sets for structured data. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 716–723. Springer, Heidelberg (2007)
15. Swamidass, S.J., Chen, J.H., Bruand, J., Phung, P., Ralaivola, L., Baldi, P.: Kernels for small molecules and the prediction of mutagenicity, toxicity and anti-cancer activity. In: ISMB (Supplement of Bioinformatics), pp. 359–368 (2005)
16. Thoma, M., Cheng, H., Gretton, A., Han, J., Kriegel, H.P., Smola, A.J., Song, L., Yu, P.S., Yan, X., Borgwardt, K.M.: Near-optimal supervised feature selection among frequent subgraphs. In: Proceedings of the SIAM International Conference on Data Mining, SDM 2009, pp. 1–12. SIAM, Philadelphia (2009)
17. Zaki, M.J., Aggarwal, C.C.: XRules: an effective structural classifier for XML data. In: Getoor, L., Senator, T.E., Domingos, P., Faloutsos, C. (eds.) Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 2003, pp. 316–325. ACM, Washington (2003)

# AnswerArt - Contextualized Question Answering

Lorand Dali, Delia Rusu, Blaž Fortuna, Dunja Mladenić, and Marko Grobelnik

Jožef Stefan Institute, Department of Knowledge Technologies, Jamova 39,  
1000 Ljubljana, Slovenia  
{Lorand.Dali, Delia.Rusu, Blaz.Fortuna,  
Dunja.Mladenic, Marko.Grobelnik}@ijs.si

**Abstract.** The focus of this paper is a question answering system, where the answers are retrieved from a collection of textual documents. The system also includes automatic document summarization and document visualization by means of a semantic graph. The information extracted from the documents is stored as subject-predicate-object triplets, and the indexed terms are expanded using Cyc, a large common sense ontology.

**Keywords:** question answering, summarization, ontology.

## 1 Introduction

We describe an enhanced question answering system that integrates two important functionalities: providing answers to questions and browsing through the document that supports the answer. The documents have to undergo several natural language processing steps before they are indexed. Moreover, the indexed terms are semantically enhanced by inference using WordNet<sup>1</sup> and the Cyc [1] ontology. Section 3 will explain the preprocessing steps in more detail. Another feature of the system is that the user can choose at query time in which document collection the answer should be searched. Also, the user can upload his own document collection. To our knowledge such flexibility is not provided by other similar systems.

Previous work has typically focused on a single topic (question answering, summarization, semantic representation and visualization of documents) and we see the advantage of the proposed system in combining these topics together. Many of the previous approaches, like Aqualog [2] and QuestIO [3], query structured data stored in ontologies. Aqualog has a restricted grammar and restricted vocabulary to which the query has to be compatible. QuestIO does not require a fixed grammatical structure of the question, but the words which it can handle are limited because of the dependency on an underlying ontology. Our system derives the answers only from unstructured text, which means that the range of questions is not limited or domain specific. However the questions must be in fixed grammatical forms for our system to “understand” them. TextRunner [4] is similar to our system in the way that it also consists of structured queries on unstructured text but the difference is that we also

---

<sup>1</sup> <http://wordnet.princeton.edu/>



provide a natural language interface to the search. Powerset<sup>2</sup> enables search and discovery in Wikipedia and Freebase, by entering keywords, phrases or simple questions. What distinguishes our system from Powerset is the way we describe the answer: by a visual representation of the document in the form of a semantic graph and by the document summary, which is automatically extracted based on the semantic graph of the document.

## 2 System Overview

AnswerArt [5] combines question answering, summarization and document visualization. Firstly, in a step performed offline, facts (consisting of subject - predicate - object triplets) are extracted from text and then stored in a triplet store. The user queries these triplets by asking a natural language question which is transformed into a structured query for the triplet store. The result consists of a list of matching triplets and the list of documents in which they occurred. In a detailed overview of the document the user can also see the list of all triplets from that document, the semantic graph (made out of triplets) and an automatically generated summary. Fig 1 shows a possible use case, while Fig 2 is a screenshot showing as an example the results we get for the question *What could pollution have affected?*

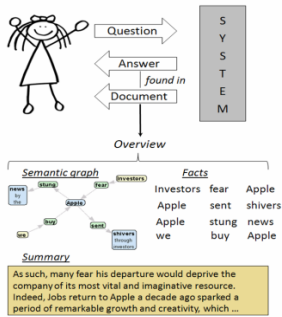


Fig. 1. System Overview

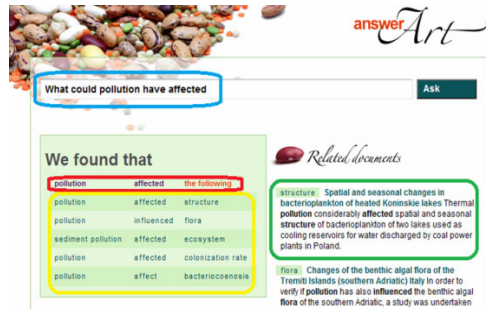


Fig. 2. Example Screenshot

## 3 Document Preprocessing

We shall now describe the preprocessing steps which are necessary to make the functionality described in Section 2 possible. The starting point is a document collection, containing unstructured text which needs to be preprocessed. The central part of preprocessing is triplet extraction [6]. Triplets, which are made of *subject*, *predicate* and *object*, are extracted from each sentence. It is in this form that the knowledge contained in the text is stored and made available for searching. Before triplets can be extracted, certain other preprocessing steps have to be done: Part of speech tagging, noun phrase chunking, parsing and named entity extraction. Thus

<sup>2</sup> <http://www.powerset.com/>

having obtained the triplets we construct a semantic graph by merging equivalent terms in the separate triplets. The semantic graph is used in document visualization [7] and summarization [8]. To make the semantic graph more connected, co-reference resolution, anaphora resolution and normalization has to be done on the triplet terms.

To make the search efficient, the terms contained in the triplets are indexed. Before indexing, the terms are expanded with related terms which are found from WordNet and the Cyc ontology. The related terms are found from the synonyms and related concepts in the ontology. For example the term *water* would be expanded with: *lake, sea, ocean, stream, freshwater* etc. Indexing related terms results in a semantic enhancement of the stored knowledge and has the goal of increasing the recall of the system.

## 4 A Machine Learning Approach to Document Summarization

In order to automatically generate document summaries, we consider a machine learning approach, where we aim at learning which sentences belong to the summary. More exactly, we describe a set of features for each triplet extracted from the document sentences, and train an SVM model for binary classification of triplets as belonging or not to the summary. Further we identify the corresponding sentences which yielded the triplets, and, relying on them, construct the document summary. The features used are of three kinds: *document features* (for e.g. position of the sentence in the document, position of the triplet in the sentence, words in the triplet elements), *linguistic features* (for e.g. part of speech tags, location of the triplet in the parse tree) and *graph features* (for e.g. hub and authority weights, page rank, node degrees, connected components).

For training the linear SVM model and for evaluating the triplet ranking, we use the *DUC (Document Understanding Conferences)*<sup>3</sup> datasets from 2002 and 2007, respectively. The 2002 dataset comprises 300 newspaper articles on 30 different topics and for each article we have a 100 word human written abstract. The DUC 2007 dataset comprises 250 articles for the update task and 1125 articles for the main task.

We evaluated our summarization system, by comparing our results to the ones obtained by other systems participating in the DUC 2007 update task; we refer to [8] for more details regarding the evaluation outcome.

## 5 Evaluation

To evaluate the contribution of the triplet enhancement with ontologies to the performance of the question answering, we have conducted the following experiment. We have asked 27 questions to which the system responded both with and without using inference from ontologies. To each question the system gave a number of answers. The correctness or relevance of the given answers was determined according to the judgement of the authors. On average a question was answered with 8 answers out of which on average 3 were due to using ontologies. Hence the usage of ontologies

---

<sup>3</sup> <http://duc.nist.gov/>

increases the number of answers retrieved by about 60%. However the number of answers that are actually correct increases by only 40% when using ontologies. This shows that the precision of answers obtained using ontologies is lower and that trying to obtain more answers by inference has a negative effect on the precision. Indeed, the precision of the system drops from 84.17% to 76.61% when adding answers obtained from ontologies, because the answers using ontologies have a precision of only 63.29%. Although the size of the experiment is too small to base any solid conclusions on it, we can argue that the AnswerArt system cannot find an important number of correct answers unless it uses ontologies. On the negative side however, ontologies introduce more mistakes and decrease the precision of the system.

## 6 Conclusions

We have presented a question answering system enhanced with summarization and document visualization functionalities. The information from which the answers are retrieved is stored as subject-predicate-object triplets. The indexed terms are expanded using inference from the Cyc ontology and WordNet. Evaluation shows that the use of ontologies increases recall but decreases precision.

## References

1. Cyc, L.D.B.: A Large-Scale Investment in Knowledge Infrastructure. *Comm. of the ACM* 38(11) (November 1995)
2. Lopez, V., Uren, V., Motta, E., Pasin, M.: AquaLog: An ontology-driven question answering system for organizational semantic intranets. *Journal of Web Semantics*, 72–105 (2007)
3. Damljanovic, D., Tablan, V., Bontcheva, K.: A text-based query interface to owl ontologies. In: *The 6th Language Resources and Evaluation Conference (LREC)*, Marrakech, Morocco, ELRA (May 2008)
4. Banko, M., Etzioni, O.: The Tradeoffs Between Open and Traditional Relation Extraction. In: *Proceedings of the Association for Computational Linguistics*, Columbus, Ohio (2008)
5. Dali, L., Rusu, D., Fortuna, B., Mladenic, D., Grobelnik, M.: Question Answering Based on Semantic Graphs. In: *LTC 2009*, Poznan, Poland (November 6, 2009)
6. Rusu, D., Dali, L., Fortuna, B., Grobelnik, M., Mladenic, D.: Triplet extraction from sentences. In: *SiKDD 2007*, Ljubljana, Slovenia (October 2009)
7. Rusu, D., Fortuna, B., Mladenic, D., Grobelnik, M., Sipos, R.: Document Visualization Based on Semantic Graphs. In: *Proceedings of the 13th International Conference Information Visualisation (IV'09)*, Barcelona, Spain, pp. 292–297 (2009)
8. Rusu, D., Fortuna, B., Grobelnik, M., Mladenic, D.: Semantic Graphs Derived From Triplets with Application in Document Summarization. In: *Proceedings of the 11th International Multiconference Information Society - IS 2008*, Ljubljana, Slovenia, pp. 198–201 (2008)

# Real-Time News Recommender System

Blaž Fortuna, Carolina Fortuna, and Dunja Mladenić

Jožef Stefan Institute, Jamova 39, 1000, Ljubljana, Slovenia  
{blaz.fortuna, carolina.fortuna, dunja.mladenic}@ijs.si

**Abstract.** In this demo we present a robust system for delivering real-time news recommendation to the user based on the user's history of the past visits to the site, current user's context and popularity of stories. Our system is running live providing real-time recommendations of news articles. The system handles overspecializing as we recommend categories as opposed to items, it implicitly uses collaboration by taking into account user context and popular items and, it can handle new users by using context information. A unique characteristic of our system is that it prefers freshness over relevance, which is important for recommending news articles in real-world setting as addressed here. We experimentally compare the proposed approach as implemented in our system against several state-of-the-art alternatives and show that it significantly outperforms them.

**Keywords:** Recommender system, SVM, collaborative filter, real-time, news.

## 1 Introduction

Recommender systems [1] and, more specifically, news recommendation [2, 3] is a popular topic in machine learning and data mining. This is also reflected in the well known competition for Netflix prize<sup>1</sup> with hundreds of competing teams. News recommendation is a younger and less researched branch of recommender systems posing unique challenges such as real-time requirements and the lack of explicit user ratings. Therefore, a news recommender system should recommend relevant new stories as soon as they are published and handle the fact that ratings are most of the times binary – read/unread – and do not necessarily express the preference of a user – reading an article doesn't mean one likes it and, not reading it might mean that one did not notice it. Also, in general the time spent on a story cannot be taken as indicative of the preference for that story.

News recommender systems have to balance between long term user preferences – driven by professional activity, education, etc – and short term trends – driven by some discontinuity in the public or personal context. Long term preferences are best captured by content based recommendation systems where content can be defined by a mix of features such as history of read topics and registration data. Short term user interests are best captured by collaborative systems using features such as the context of the user (i.e. referring page) and popular stories (i.e. stories that are of general interest and outside the user's long term preferences).

---

<sup>1</sup> <http://www.netflixprize.com/>

In this demo, we present a hybrid system which combines long term user profile, the current context describing the last visit and the most popular story. The long term user profile is built based on the user’s long term browsing habits. The context is described by the user’s local time and location, last read story and referring link to the last read story. Based on these features, the system predicts the most likely news categories of interest to the user. From the top three predicted categories it recommends the most popular news stories appearing in the last six hours. We show that the system can outperform several base-line systems: most popular, item-to-item collaborative filter, and contextual. Furthermore the system is running live providing real-time recommendations of news articles to a large user base.

## 2 Methodology

The proposed methodology consists of several steps. (1) First, we analyzed the user base data and split it into two groups: “old” users which have a history of more than 50 visits and, “new” users which have a history of less than 50 visits (the threshold 50 was selected based on preliminary experiments). History giving a long term user interest is included in feature representation for “old” users only. (2) Then, a separate model is trained for each of the two user groups. To avoid overspecializing, we define a machine learning problem as predicting the most interesting news category rather than specific news articles. In our specific setting the news stories were manually classified by domain experts into a taxonomy of 40 categories. Alternatively, one can use machine learning to automatically classify the articles into taxonomy.

Experimental evaluation was conducted using different combinations of feature sets (as input for the SVM) to predict the top three categories of news articles the user would be interested in. Table 1 shows that, for “old” users, the precision<sup>2</sup> using context information, such as referring page and time is relatively low (36%). Including requested page is more predictive (41%), while using history information gives the highest precision (48%). However, if we include all the features (i.e. history, Geo, requested and referring page and time), we obtain the highest precision (52%).

**Table 1.** Precision for sets of features for “old” users (more than 50 visits) on top 3 categories

	All	History	Geo	Requested	Referring	Time
Top precision [%]	<b>52</b>	48	43	41	36	36

Table 2 shows that our system can do good predictions even for “new” users, and that the quality of the predictions using all the features again gives the highest precision.

(3) After predicting the top 3 categories, the system selects the top new article in each category and suggests this set of news to the user. This approach provides robustness and diversity since we are only predicting interesting categories for each user and are not linking him/her directly to stories.

<sup>2</sup> From the categories the SVM model predicts for user A, which were actually interesting enough so user A read an article from that category in the next step.

**Table 2.** Precision for sets of features for “new” users (less than 50 visits) on top 3 categories

	All	Geo	Requested	Referring	Time
Top precision [%]	45	36	35	37	37

We deployed our system (see Fig. 1) in real-world setup with millions of requests and hundreds of new articles per day. In our server logs each user visit and updates the index in real-time (batches of 1000 visits). The logs are also archived and, along with the rest of the data, used daily for training new SVM models. The co-visitation matrix of the collaborative filtering module is updated on the fly. Newly published stories are crawled as soon as they get the first visit. The average time to compute and serve the recommendation is 20 ms<sup>3</sup>.

**Fig. 1.** Real-time recommender system architecture (only SVM and CF modules represented)

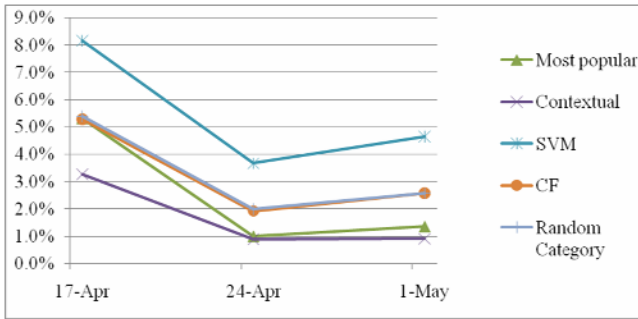
We compared the performance of our system against 4 other approaches (1) item-to-item collaborative filtering based co-visitation matrix (users who read this, also read that), (2) random category which selects three random categories and recommends the most popular stories in these categories (3) most popular which recommends the top most popular stories at the moment of the visit, and (4) contextual which recommends based on the last read story. Fig. 2 presents the results over 2 weeks, it shows that our system outperformed all the others in terms of transition probabilities. This means that, provided with the set of recommended news articles, the user is most likely to click on the articles recommended by the SVM module. Random category and collaborative filter modules performed on par over the two weeks of the trials. The third ranked was most popular and the fourth was contextual.

### 3 Overview of the Demonstration

The demonstration will consist of a system running on a large live feed of news articles and page-views with user profiles and visit information being updated on the

<sup>3</sup> This may vary according to location and connection speed. The reported value is an average of loading times reported by Chrome.

fly. We will show the three best performing modules SVM recommender, the CF recommender, the Random category recommender and the logging server running live. They will be exposed to millions of requests and hundreds of new stories. We will also provide a web client allowing interested participants to delete cookies, construct their own history and assess the quality of the recommendations.



**Fig. 2.** Transition probabilities for the five modules (i.e number of users who read a recommended story divided by the total number of users who saw the recommendations)

## 4 Conclusions

In this work, we proposed an SVM based news article recommendation system and we compared its performance against several state-of-the-art alternatives. We proved via large scale real-world experimental tests that it outperforms the alternatives. The system is robust against overspecializing as we recommend categories as opposed to individual stories. It implicitly uses collaboration by taking into account context and popular items (i.e. users who are interested in the same categories visited these stories the most) and it can handle new users by using context information. A unique characteristic of our system is that it prefers freshness (6 hours time window) over relevance.

## References

1. Adomavicius, G., Tuzhilin, A.: Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering* 17(6) (June 2005)
2. Das, A.S., Datar, M., Garg, A., Rajaram, S.: Google news personalization: scalable online collaborative filtering. In: *Proceedings of the 16th International Conference on World Wide Web*, Banff, Alberta, Canada, May 08-12 (2007)
3. Liu, J., Dolan, P., Pedersen, E.R.: Personalized news recommendation based on click behavior. In: *Proceeding of the 14th International Conference on Intelligent User Interfaces*, Hong Kong, China, February 07-10 (2010)

# CET: A Tool for Creative Exploration of Graphs

Stefan Haun<sup>1</sup>, Andreas Nürnberger<sup>1</sup>, Tobias Kötter<sup>2</sup>,  
Kilian Thiel<sup>2</sup>, and Michael R. Berthold<sup>2</sup>

<sup>1</sup> Data and Knowledge Engineering Group,  
Faculty of Computer Science, Otto-von-Guericke-University, Germany

<http://www.findke.ovgu.de>

<sup>2</sup> Nycomed-Chair for Bioinformatics and Information Mining  
University of Konstanz, Germany

<http://www.inf.uni-konstanz.de/bioml>

**Abstract.** We present a tool for interactive exploration of graphs that integrates advanced graph mining methods in an interactive visualization framework. The tool enables efficient exploration and analysis of complex graph structures. For flexible integration of state-of-the-art graph mining methods, the viewer makes use of the open source data mining platform KNIME. In contrast to existing graph visualization interfaces, all parts of the interface can be dynamically changed to specific visualization requirements, including the use of node type dependent icons, methods for a marking if nodes or edges and highlighting and a fluent graph that allows for iterative growing, shrinking and abstraction of (sub)graphs.

## 1 Introduction

Today's search is still concerned mostly with keyword-based searches and the closed discovery of facts. Many tasks, however, can be solved by mapping the underlying data to a graph structure and searching for structural features in a network, e.g. the connection between certain pages in Wikipedia<sup>1</sup> or the environment of a specific document. Exploring a hyperlink structure in a graph representation enables these tasks to be fulfilled much more efficiently. On the other hand, graph visualization can handle quite large graphs, but is rather static, i.e. the layout and presentation methods calculate the graph visualization once and are well suited for interactions, such as adding or removing nodes. One of the famous graph layout methods, the *Spring Force Layout*, can yield very chaotic results when it comes to small changes in the graph, leading to a completely different layout if just one node is removed. Since a user relies on the node positions during interaction with the graph, such behavior is not desirable.

With the *Creative Exploration Toolkit (CET)*, we present a user interface with several distinct features:

- support of interactive graph visualization and exploration,
- integration of a modular open source data analytics system,

---

<sup>1</sup> <http://www.wikipedia.org>



- easy configuration to serve specific user requirements.

In the following sections, we will describe these features in more detail.

## 2 The Creative Exploration Toolkit

The Creative Exploration Toolkit (CET) is the user interface that visualizes the graph and allows interaction. The global design, shown in Figure 1, consists of

- a *dashboard* at the top, where the controls are located,
- a *logging area*, below, to show information on running processes and the tool status,
- a *sidebar* on the right which displays detailed information about a node,
- and the *workspace* in the center, which is used for visualization.

We currently use the *Stress Minimization Layout* [3] to determine the initial graph layout, which enables the user to interact with the graph: Nodes can be moved to create certain arrangements, nodes can be selected, and nodes can be expanded by double-clicking them. Additionally, the user may issue keyword-based queries. The corresponding results consists of graphs and can be visualized as well. Subsequent query results are added to the graph, enabling the user to explore the graph itself and the structures between the query results.

While CET takes care of graph visualization and presentation, special semantics are not supported. For example, the shortest path between two nodes is displayed by highlighting all nodes on the path. However, the user interface is not aware of the path-property, but only displays the highlight attribute of the

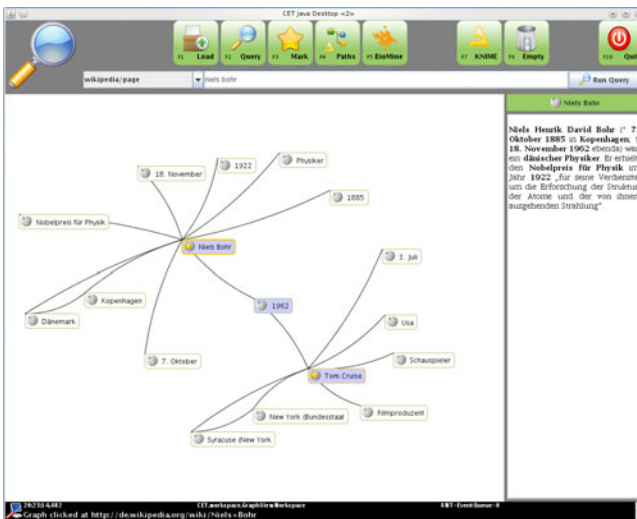


Fig. 1. Screenshot of the Creative Exploration Toolkit (CET)

nodes, while the actual calculation takes place in the underlying data analysis platform described in the next section. The user interface is therefore very flexible when it comes to tasks from different domains.

As described in the next section, any KNIME workflow may be called from inside the user interface. However, this is also meant for design and development purposes. Finalized workflows can be integrated more directly into the UI to provide a more natural and convenient user experience. In our current setup, we integrate calls to a shortest path calculation and queries to the BioMine<sup>2</sup> database. More workflow and interaction schemes will follow in the course of future work.

### 3 The KNIME Information Mining Platform

KNIME<sup>3</sup>, the Konstanz Information Miner, was initially developed by the Chair for Bioinformatics and Information Mining at the University of Konstanz, Germany. KNIME is released under an open source license (GPL v3) and can be downloaded free of charge<sup>4</sup>. KNIME is a modular data exploration platform that enables the user to visually create data flows (often referred to as pipelines), selectively execute some or all analysis steps, and later investigate the results through interactive views on data and models. The KNIME base version already incorporates hundreds of processing nodes for data I/O, preprocessing and cleansing, modeling, analysis and data mining as well as various interactive views, such as scatter plots, parallel coordinates and others. It integrates all analysis modules of the well known Weka data mining environment and additional plugins allow, among others, R-scripts<sup>4</sup> to be run, offering access to a vast library of statistical routines. Within the frame of the EU FP7 project “BISON”, KNIME was extended to also allow the flexible processing of large graphs. Combined with the already existing nodes, KNIME can therefore be used to model complex network processing and analysis tasks.

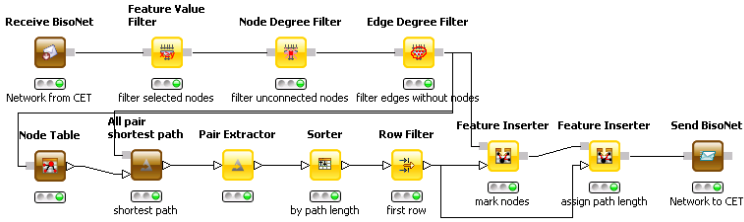
CET offers a very generic access to KNIME, enabling the user to make arbitrary calls without adapting the user interface. CET can be configured to directly call a KNIME workflow via a pre-configured button. CET also provides a list of all available workflows plus a list of parameters for a selected work, which can be edited by the user. Essentially, all information that would be sent by the user interface can be provided to start a KNIME workflow. The result is then visualized in the graph. New analysis methods can therefore be integrated easily into CET by simply adding a new workflow providing the corresponding functionality.

Figure 2 shows an example of a workflow computing the network diameter. In this workflow, firstly all nodes with a certain feature value are filtered, i.e. to take only those into account that have been selected and marked by the user. Secondly degree filters are applied on nodes and edges to filter unconnected

<sup>2</sup> <http://www.cs.helsinki.fi/group/biomine/>

<sup>3</sup> <http://www.knime.org>

<sup>4</sup> <http://www.r-project.org>



**Fig. 2.** An example KNIME workflow for calculating the network diameter which is called from CET

nodes. The shortest paths of all node pairs are subsequently computed and a feature is assigned consisting of the path length to those nodes of the longest of shortest paths. Finally the graph is sent back to the CET.

## 4 Conclusion and Future Work

We demonstrate a novel user interface for generic graph visualization with special emphasis on extensibility by integration with data and graph analysis. The presented interface allows for easy interaction with the visualized graphs. This setup is particularly interesting for researchers in the area of Data Mining and Network Analysis, as it is very simple to plug in new approaches and visualize the results, even if there is interaction involved.

Extensions of the CET aim towards the integration of more workflows, thus adding to the available interaction and analysis features. We will also further improve graph visualization by incorporating constraint-based graph layout (for a first discussion see [2]).

*Acknowledgement.* The work presented here was supported by the European Commission under the 7th Framework Programme FP7-ICT-2007-C FET-Open, contract no. BISON-211898.

## References

- Berthold, M.R., Cebron, N., Dill, F., Gabriel, T.R., Kötter, T., Meinl, T., Ohl, P., Sieb, C., Thiel, K., Wiswedel, B.: KNIME: The Konstanz Information Miner. In: Data Analysis, Machine Learning and Applications - Proceedings of the 31st Annual Conference of the Gesellschaft für Klassifikation e.V., Studies in Classification, Data Analysis, and Knowledge Organization, pp. 319–326. Springer, Heidelberg (2007)
- Haun, S., Nitsche, M., Nürnberger, A.: Interactive Visualization of Continuous Node Features in Graphs. In: Proc. of Workshop on Explorative Analytics of Information Networks (EIN), part of ECML/PKDD 2009, pp. 98–106 (2009)
- Koren, Y., Çivril, A.: The Binary Stress Model for Graph Drawing. In: Tollis, I.G., Patrignani, M. (eds.) GD 2008. LNCS, vol. 5417, pp. 193–205. Springer, Heidelberg (2009)

# NewsGist: A Multilingual Statistical News Summarizer

Mijail Kabadjov, Martin Atkinson, Josef Steinberger,  
Ralf Steinberger, and Erik van der Goot

Joint Research Centre, European Commission,  
Via E. Fermi 2749, Ispra (VA), Italy  
firstname.lastname@jrc.ec.europa.eu

**Abstract.** In this paper we present NewsGist, a multilingual, multi-document news summarization system underpinned by the Singular Value Decomposition (SVD) paradigm for document summarization and purpose-built for the Europe Media Monitor (EMM). The summarization method employed yielded state-of-the-art performance for English at the Update Summarization task of the last Text Analysis Conference (TAC) 2009 and integrated with EMM represents the first online summarization system able to produce summaries for so many languages. We discuss the context and motivation for developing the system and provide an overview of its architecture. The paper is intended to serve as accompaniment of a live demo of the system, which can be of interest to researchers and engineers working on multilingual open-source news analysis and mining.

## 1 Introduction

On a daily basis, the Europe Media Monitor (EMM) gathers over 100k news articles in several dozens of languages from thousands of on-line news sources worldwide [1, 8]. It clusters all these articles into major news stories and plots in real time news clusters' sizes along a time line to provide, as opposed to standard search engines, a visual overview of the current state of affairs. It also automatically identifies spikes on the graph and sends out relevant breaking-news alerts, where 'relevance' is user-defined by using a set of intuitive semantic categories (called EMM categories), to the thousands of subscribed users.

Additionally, EMM recognizes references to entities (locations, persons and organizations) in the news [4], detects sentiment, monitors the development of news stories over time and links news clusters across languages [7].

Currently, however, EMM does not provide succinct summaries for the, potentially large, news clusters. This is clearly a desirable feature since these clusters may contain hundreds of news articles which would be impossible to read in full within a short time frame. Yet, this is often the need of EU decision makers who make use of the EMM system on a daily- or even hourly-basis and based on the information they receive they must produce timely responses to complex issues. Therefore, providing high quality

---

<sup>1</sup> EMM's news analysis applications are NewsBrief, NewsExplorer, MedISys and EMM labs accessible from: <http://emm.newsbrief.eu>

summaries would substantially improve the usability of EMM as a news aggregation and trend visualization system.

In this paper we describe the summarization system currently under development for EMM, which we have named NewsGist. In the search for a suitable summarization method we adopted a general processing model for summarization foreseeing three phases [5]: interpretation, transformation and generation, and we chose the Singular Value Decomposition (SVD) paradigm to underpin it [2, 6]. The SVD approach has the advantage of being language-independent and has proven to be an effective summarization method yielding state-of-the-art performance in international evaluation efforts such as those of the Text Analysis Conference<sup>2</sup> (TAC) [6]. Furthermore, high-performance implementations of SVD taking advantage of heavily parallel architectures such as GPUs are already available.

The rest of the paper is organized as follows. In the next section we provide a brief overview of the Europe Media Monitor. Then, we describe the SVD model to summarization, section 3. After that, in section 4, we present NewsGist. Finally, we conclude the paper with pointers to further work.

## 2 Europe Media Monitor

The Europe Media Monitor is a web-based multilingual news aggregation system that collects over 100k news articles per day in about 50 languages from more than 2500 web news sources. The system employs text mining techniques to provide a picture of the present situation in the World (as conveyed in the media). Every ten minutes it automatically clusters all the collected news articles and displays the ten largest clusters per language by plotting them on a time-by-size graph. It also provides all the necessary hyperlinks to navigate through the clusters and to go to the source for a detailed exploration. In addition, it applies some deeper information analysis techniques, as for example, to automatically detect violent events, derive reported social networks and analyze media impact.

The public website provides a user interface to all this information. This public website is visited on a regular basis by some 30000 human users, and gets some 1.2M hits per day<sup>3</sup>.

## 3 Multi-document Summarization Based on SVD

As mentioned above, we chose the SVD paradigm to build our summarizer on. Next, we describe how each one of the three processing phases of interpretation, transformation and generation are realized.

In SVD-based summarization the interpretation phase takes the form of building a term-by-sentence matrix  $A = [A_1, A_2, \dots, A_n]$ , where  $A_j = [a_{1j}, a_{2j}, \dots, a_{nj}]^T$  represents the weighted term-frequency vector of sentence  $j$  in a given set of documents.

<sup>2</sup> <http://www.nist.gov/tac/>

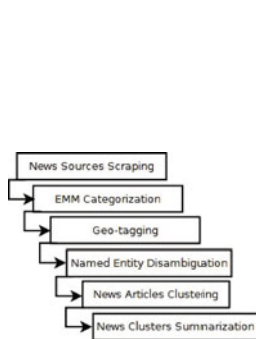
<sup>3</sup> For more details on EMM see [1, 8].

The transformation phase is done by applying singular value decomposition (SVD) to the initial term-by-sentence matrix and is defined as  $A = U\Sigma V^T$ .

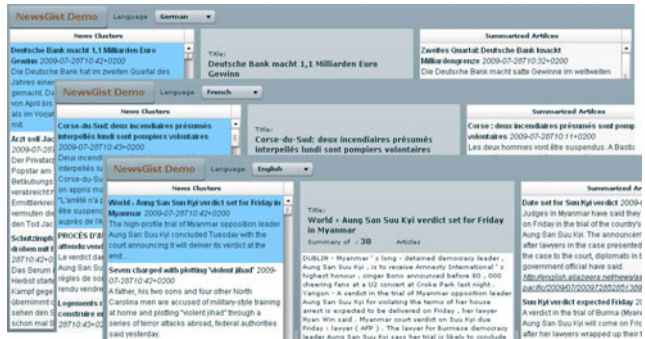
Finally, the generation phase takes place in the form of prominent sentence selection<sup>4</sup>.

## 4 NewsGist

In this section we present NewsGist<sup>5</sup>. We provide a brief overview of its architecture and show screenshots for several languages.



(a) EMM's main processing phases.



(b) NewsGist's overlaid screenshots for English, French and German.

Fig. 1. EMM and NewsGist

As mentioned earlier, NewsGist was developed as part of EMM which is built on a pipeline architecture, where an input text document undergoes several processing phases during which the source is augmented with several layers of metadata such as named entities recognized in the text and semantic categories triggered by the text. The data interchange format between processing phases is RSS, a light-weight type of XML typically used by on-line news providers.

Thus, the input to NewsGist is an RSS file enriched with information acquired by previous processing phases. Most importantly, by the time the RSS file reaches NewsGist, it already contains the outcome of the clustering of news articles and as output NewsGist produces a summary for each distinct news cluster (see fig. 1(a)).

The core system is pretty compact and is implemented as a Java servlet, running on top of Apache's Tomcat web server<sup>6</sup>.

Language-specific tokenization and sentence splitting is provided by CORLEONE [3]. Reading and writing of RSS files is provided by EMM utility libraries. Matrix

<sup>4</sup> See [6] for full details of the method.

<sup>5</sup> Online demo of the system is available at

<http://emm-labs.jrc.it/EMMLabs/NewsGist.html>

<sup>6</sup> <http://tomcat.apache.org/>

operations, and in particular singular value decomposition, is provided by the matrix-toolkits-java libraries<sup>7</sup> and also, alternatively, by the Java Matrix Package (JAMA)<sup>8</sup>.

In figure 1(b) we show overlaid screenshots of NewsGist's online demo (<http://emm-labs.jrc.it/EMMLabs/NewsGist.html>) for three languages of the European Union: English, French and German.

## 5 Conclusion

In this paper we presented NewsGist, a multilingual multi-document summarization system purpose-built for the Europe Media Monitor (EMM). We provided an overview of EMM, briefly discussed the underlying summarization method based on the SVD paradigm and described the architecture of the system.

In future work we intend to carry out a comprehensive evaluation of the quality of the summaries for languages other than English.

## Acknowledgments

We are grateful to the EMM development team for their support.

## References

- [1] Atkinson, M., Van der Goot, E.: Near real-time information mining in multilingual news. In: Proceedings of the 18th International World Wide Web Conference (WWW 2009), Madrid, Spain, pp. 1153–1154 (April 2009)
- [2] Kabadjov, M., Steinberger, J., Pouliquen, B., Steinberger, R., Poesio, M.: Multilingual statistical news summarisation: Preliminary experiments with English. In: Proceedings of IAP-WNC at the IEEE/WIC/ACM WI-IAT (2009)
- [3] Piskorski, J.: CORLEONE - core linguistic entity online extraction. Tech. Rep. EN 23393, Joint Research Centre of the European Commission (2008)
- [4] Pouliquen, B., Steinberger, R.: Automatic construction of multilingual name dictionaries. In: Goutte, C., Cancedda, N., Dymetman, M., Foster, G. (eds.) Learning Machine Translation. NIPS series, MIT Press, Cambridge (2009)
- [5] Spärck-Jones, K.: Automatic summarising: Factors and directions. In: Mani, I., Maybury, M. (eds.) Advances in Automatic Text Summarization. MIT Press, Cambridge (1999)
- [6] Steinberger, J., Kabadjov, M., Pouliquen, B., Steinberger, R., Poesio, M.: WB-JRC-UT's participation in TAC 2009: Update Summarization and AESOP tasks. In: National Institute of Standards and Technology (eds.) Proceedings of TAC, Gaithersburg, MD (November 2009)
- [7] Steinberger, R., Pouliquen, B., Ignat, C.: Using language-independent rules to achieve high multilinguality in text mining. In: Fogelman-Soulié, F., Perrotta, D., Piskorski, J., Steinberger, R. (eds.) Mining Massive Data Sets for Security. IOS-Press, Amsterdam (2009)
- [8] Steinberger, R., Pouliquen, B., van der Goot, E.: An introduction to the europe media monitor family of applications. In: Gey, F., Kando, N., Karlgren, J. (eds.) Proceeding of the SIGIR Workshop on Information Access in a Multilingual World (SIGIR-CLIR 2009), Boston, USA (July 2009)

<sup>7</sup> <http://code.google.com/p/matrix-toolkits-java/>

<sup>8</sup> <http://math.nist.gov/javanumerics/jama/>

# QUEST: Query Expansion Using Synonyms over Time\*

Nattiya Kanhabua and Kjetil Nørvåg

Dept. of Computer Science  
Norwegian University of Science and Technology  
Trondheim, Norway

**Abstract.** A particular problem of searching news archives with named entities is that they are very dynamic in appearance compared to other vocabulary terms, and synonym relationships between terms change with time. In previous work, we proposed an approach to extracting time-based synonyms of named entities from the whole history of Wikipedia. In this paper, we present QUEST (Query Expansion using Synonyms over Time), a system that exploits time-based synonyms in searching news archives. The system takes as input a named entity query, and automatically determines time-based synonyms for a given query wrt. time criteria. Query expansion using the determined synonyms can be employed in order to improve the retrieval effectiveness.

## 1 Introduction

News archives are publicly available nowadays, e.g., Google News Archive and The Times Online. Nevertheless, searching for information in such resources is not straightforward because their contents are strongly time-dependent. To increase precision, a user can narrow down search results by extending query keywords with the creation or update date of documents (called temporal criteria). Two ways of obtaining temporal criteria relevant to a query are 1) having them provided by the user [16], or 2) determined by the system [5]. One way of increasing recall is to perform query expansion using synonyms. However, when queries are named entities (people, organizations, locations, etc.), a problem of expanding the queries is the effect of rapidly changing synonyms<sup>1</sup> over time, e.g., changes of roles or alterations of names. For example, “Cardinal Joseph Ratzinger” is a synonym of “Pope Benedict XVI” before 2005, and “United States Senator from New York” is a synonym of “Hillary R. Clinton” between 2001 and 2008. Instead of referring to a synonym alone, we have to always refer to an entity-synonym relationship because a term can be a synonym of one or more entities. In this paper, we present QUEST (Query Expansion using Synonyms over Time), a system that exploits changing synonyms over time in searching news archives. To the best of our knowledge, this has never been done before in the existing news archive search systems. Our system consists of two parts: 1) the offline module for extracting time-based synonyms as depicted in Fig. 1 and 2) the online module for searching news archive as

\* This work has been supported by the LongRec project, partially funded by the Norwegian Research Council.

<sup>1</sup> In general, synonyms are different words with very similar meanings, but in our context synonyms are name variants (other names, titles, or roles) of a named entity.



illustrated in Fig. 2. With a web-based interface, the system can take as input a named entity query. It automatically determines time-based synonyms for a given named entity, and ranks the synonyms by their time-based scores. Then, a user can expand the named entity with the synonyms in order to improve the retrieval effectiveness.

Our news archive search system is mainly driven by entity-synonym relationships, which can be automatically created based on the whole history of Wikipedia. Evolving relationships are detected using the most current version of Wikipedia, while relationships for particular time in the past are discovered through the use of snapshots of previous Wikipedia versions. Using our approach, future relationships with new named entities can be also discovered simply by processing Wikipedia as new contents are added. Further, we employ the New York Times Annotated Corpus<sup>2</sup> in order to extend the covered time range as well as improve the accuracy of time of synonyms. The rest of the paper is organized as follows. In Sect. 2 we describe an approach to extracting synonyms from Wikipedia, and ranking synonyms based on their temporal characteristic. In Sect. 3 we outline the online search system and our proposed demo.

## 2 Extracting Time-Based Synonyms from Wikipedia

We extract entity-synonym relationships in an offline manner as depicted in Fig. 1. We downloaded the complete dump of English Wikipedia from the Internet Archive<sup>3</sup>, which is composed of all pages and all revisions. Each revision of a page has the time period that it was in use before being replaced by the succeeding version. In other words, the associated time of a revision is the period when it was a current version.

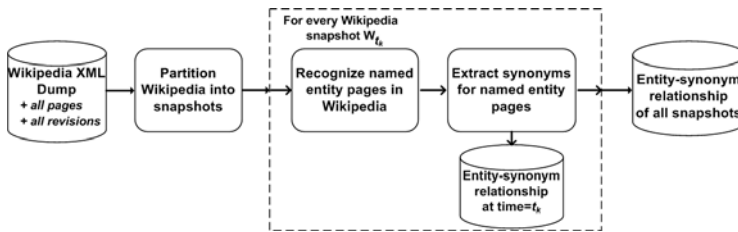


Fig. 1. Extracting time-based synonyms from the history of Wikipedia

First, we partition Wikipedia into snapshots  $\{W_{t_1}, \dots, W_{t_z}\}$  with *1-month* granularity. For each Wikipedia snapshot  $W_{t_k}$ , we identify all named entities in the snapshot  $W_{t_k}$  using the approach described by Bunesco and Paşca in [3]. After identifying an entity page  $p_e$  from a snapshot  $W_{t_k}$ , we will have a set of entity pages  $\mathcal{P}_{e,t_k} = \{p_e | p_e \in W_{t_k}\}$ . From this set, we will create a set of named entities  $E_{t_k}$  at time  $t_k$  by simply extracting a title from each named entity page  $p_e \in \mathcal{P}_{e,t_k}$ . For each named entity in  $E_{t_k}$ , we will find synonyms by extracting anchor texts from article links, as described by Bøhn and Nørvåg [2]. For a page  $p_i \in W_{t_k}$ , we extract all internal links in

<sup>2</sup> [http://www ldc.upenn.edu/Catalog/docs/LDC2008T19/new\\_york\\_times\\_annotated\\_corpus.pdf](http://www ldc.upenn.edu/Catalog/docs/LDC2008T19/new_york_times_annotated_corpus.pdf)

<sup>3</sup> <http://www.archive.org/details/enwiki-20080103>

$p_i$ , but only those links that point to an entity page  $p_e \in \mathcal{P}_{e,t_k}$  are interesting. In other words, the system extracts as synonyms all anchor texts for the associated entity, and these synonyms are weighted by their frequencies of occurrence. We then obtain a set of entity-synonym relationships. By accumulating the set of entity-synonym relationships from every page  $p_i \in W_{t_k}$ , we will have a set of entity-synonym relationships at time  $t_k$ , i.e., a synonym snapshot  $S_{t_k} = \{\xi_{1,1}, \dots, \xi_{n,m}\}$ . Named entity recognition and synonym extraction steps are processed for every snapshot  $W_{t_k}$ . Finally, we will have obtained the set of entity-synonym relationships from all snapshots  $\mathbb{S} = \{S_{t_1}, \dots, S_{t_z}\}$ , and the set of synonyms for all entities  $\mathcal{S} = \{s_1, \dots, s_y\}$ . Note that, the time periods of synonyms are timestamps of Wikipedia articles in which they appear, not the time extracted from the contents. To discover the more accurate time, we need to analyze a document corpus with the longer time period, i.e., the New York Time Annotated Corpus. Due to the size limitation of the paper, the reader can refer to [4] for more detail about improving time of entity-synonym relationships. Given a named entity  $e_i$  and temporal criteria  $[t_a, t_b]$ , we can retrieve a set of synonyms of  $e_i$  wrt.  $[t_a, t_b]$  from  $\mathbb{S}$ . The synonyms can be ranked by time-based scores defined as a mixture model of a temporal feature and a frequency feature as follows.

$$TB(s_j, [t_a, t_b]) = \mu \cdot pf(s_j, [t_a, t_b]) + (1 - \mu) \cdot \overline{tf}(s_j, [t_a, t_b]) \tag{1}$$

where  $pf(s_j, [t_a, t_b])$  is a time partition frequency or the number of time partitions (or time snapshots) in which a synonym  $s_j$  occurs within  $[t_a, t_b]$ .  $\overline{tf}(s_j, [t_a, t_b])$  is an averaged term frequency of  $s_j$  in all time partitions within  $[t_a, t_b]$ ,  $\overline{tf}(s_j, [t_a, t_b]) = \frac{\sum_{t_i \in [t_a, t_b]} tf(s_j, p_{t_i})}{pf(s_j, [t_a, t_b])}$ .  $\mu$  underlines the importance of a temporal feature and a frequency feature, and  $\mu=0.5$  gave the best performance in our experiments.

### 3 Online Demo

The time-based synonyms extracted using our approach can be applied to any news archive collection. In this demo, we use the New York Times Annotated Corpus as an illustrative example of such a news archive. This collection contains over 1.8 million articles from January 1987 to June 2007. We use the enterprise search platform Solr from Apache Lucene. The system screenshots are shown in Fig. 2, and the online demo is publicly available at <http://research.idi.ntnu.no/wislab/quest/>. In this demo, we find over 2.5 million named entities and 3 million entity-synonym relationships. Given a query  $q$  and temporal criteria  $[t_a, t_b]$ , the system has to verify whether  $q$  is a named entity. We do this by searching Wikipedia with  $q$ , and the first page in the result list will be used as the associated named entity for  $q$ . Subsequently, the system determines synonyms for the associated named entity, and the user can select synonyms to expand the original  $q$  in order to improve the retrieval effectiveness. In addition, the user can choose whether to show time periods and scores associated to synonyms. In the following, we will give two search scenario as examples.

*First scenario:* A student studying the history of the Roman Catholic Church wants to know about the Pope Benedict XVI during the years before he became the Pope (i.e. before 2005). The student searches using the query “Pope Benedict XVI” and the publication dates “01/1987” and “04/2005”. The system retrieves documents for the

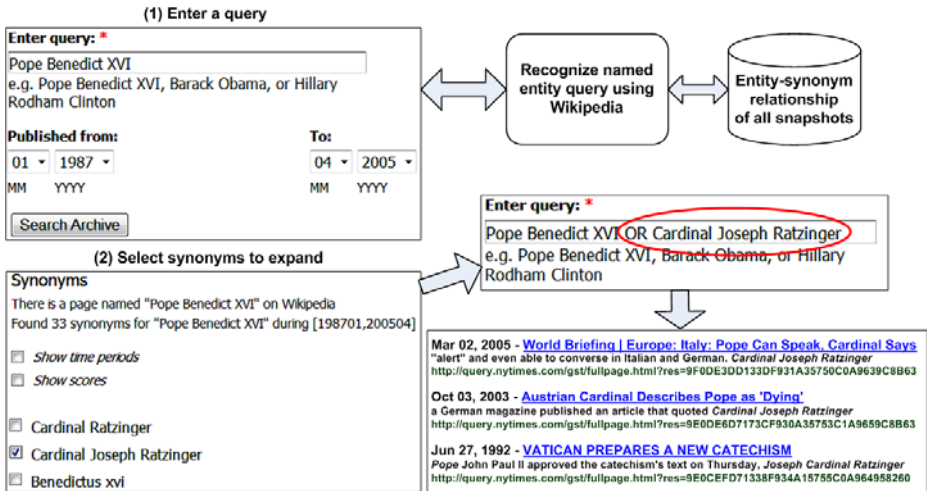


Fig. 2. QUEST online demo at <http://research.idi.ntnu.no/wislab/quest/>

query “Pope Benedict XVI”, and also determines synonyms for the query wrt. time criteria. The student then selects the synonyms “Cardinal Joseph Ratzinger” to expand the query. The new query becomes “Pope Benedict XVI OR Cardinal Joseph Ratzinger”. He performs search again, and the system retrieves documents which are relevant to both “Pope Benedict XVI” and “Cardinal Joseph Ratzinger”.

*Second scenario:* A marketing journalist wants to search for past information about Kmart, or a chain of discount department stores in the United States. She enters the query “Kmart” and the publication dates “01/1987” and “01/2000”. The system retrieves documents for the query “Kmart”, and also determines synonyms for the query wrt. time criteria. She selects the synonyms “Kresge” to expand the query (Kmart was founded as the S. S. Kresge Company in 1899, and it was named to Kmart in 1962.). The new query becomes “Kmart OR Kresge”. She performs search again, and the system retrieves documents which are relevant to both “Kmart OR Kresge”.

## References

- Berberich, K., Bedathur, S.J., Neumann, T., Weikum, G.: A time machine for text search. In: Proceedings of SIGIR 2007 (2007)
- Bøhn, C., Nørvåg, K.: Extracting named entities and synonyms from wikipedia. In: Proceedings of AINA 2010 (2010)
- Bunescu, R.C., Paşca, M.: Using encyclopedic knowledge for named entity disambiguation. In: Proceedings of EACL 2006 (2006)
- Kanhabua, N., Nørvåg, K.: Exploiting time-based synonyms in searching document archives. In: Proceedings of JCDL 2010 (2010)
- Kanhabua, N., Nørvåg, K.: Determining Time of Queries for Re-ranking Search Results. In: Proceedings of ECDL 2010 (2010)
- Nørvåg, K.: Supporting temporal text-containment queries in temporal document databases. *Journal of Data & Knowledge Engineering* 49(1), 105–125 (2004)

# Flu Detector - Tracking Epidemics on Twitter

Vasileios Lampos, Tijn De Bie, and Nello Cristianini

Intelligent Systems Laboratory  
University of Bristol, UK

**Abstract.** We present an automated tool with a web interface for tracking the prevalence of Influenza-like Illness (ILI) in several regions of the United Kingdom using the contents of Twitter’s microblogging service. Our data is comprised by a daily average of approximately 200,000 geolocated tweets collected by targeting 49 urban centres in the UK for a time period of 40 weeks. Official ILI rates from the Health Protection Agency (HPA) form our ground truth. Bolasso, the bootstrapped version of LASSO, is applied in order to extract a consistent set of features, which are then used for learning a regression model.

## 1 Introduction

Monitoring the diffusion of an epidemic disease such as seasonal influenza is a very important task. Various methods are deployed by the health sector in order to detect and constrain epidemics, such as counting the consultation rates of general practitioners (GPs) [1], school or workforce absenteeism figures [2], etc. The need of a proper infrastructure and the time delays due to the necessary data processing are the main drawbacks of those methodologies.

We argue that information available on the web can provide an additional means for tackling this problem. In [3,4] it has been demonstrated that user queries on web search engines can be used to provide an early warning for an epidemic. Furthermore, recent work [5,8] has shown that the social web media have a predictive power on different domains. In particular, article [5] presents a method for inferring ILI rates for several regions in the UK by using data from Twitter [4]. The core of the method performed feature selection and regression with L1 regularisation by applying the LASSO [7]. This paper extends our previous methodology and presents a complete pipelined application of it: the “Flu detector” (<http://geopatterns.enm.bris.ac.uk/epidemics/>).

Short and geolocated messages from users on Twitter, commonly known as ‘tweets’ on the one side, and weekly ILI reports from the HPA on the other for 3 UK regions, namely Central England & Wales ( $r_1$ ), South England ( $r_2$ ) and North England ( $r_3$ ), are the sources of our information. For the experimental part of this work, we use 40 weeks of the Twitter corpus and the respective ground truth, from 22/06/2009 to 28/03/2010. We apply Bolasso, the bootstrapped version of LASSO [6], for performing feature selection on the vector

---

<sup>1</sup> Twitter micro-blogging social network, <http://twitter.com/>

space representation of the Twitter corpus, *i.e.* for extracting a consistent set of keywords that can be used for inferring HPA’s ILI rates optimally. The selected features are then used in a regression model; evaluating the performance of our model on unseen data yields inferences which are very well correlated with the ground truth.

## 2 Methodology

Our aim is to compute a flu-score from Twitter corpus on a daily basis. For this purpose, we learn a set of weighted keywords (we refer to them as markers or features) via an automated process. Given a set of markers  $\mathcal{M} = \{m_i\}$ ,  $i \in [1, n]$ , their respective weights  $\mathcal{W} = \{w_i\}$ ,  $i \in [1, n]$ , and a set of tweets  $\mathcal{T} = \{t_j\}$ ,  $j \in [1, k]$ , Twitter’s flu-score  $f_S$  is defined as  $f_S(\mathcal{T}, \mathcal{W}, \mathcal{M}) = \sum_j \sum_i w_i \times g(t_j, m_i) / k$ , where  $g(t_j, m_i) = 1$ , if a tweet  $t_j$  contains a marker  $m_i$ , otherwise  $g(t_j, m_i) = 0$ .

We start by creating a pool of candidate features by using encyclopedic and informal references related to influenza as well as some flu-related word clusters created by Google Sets. After the necessary preprocessing (tokenisation, stemming, stop-word and name removal), we end up with a set of  $\theta = 2675$  candidate features, denoted by  $\mathcal{C} = \{c_u\}$ ,  $u \in [1, \theta]$ <sup>2</sup>. Given a set  $\mathcal{T}$  of  $k$  tweets, each candidate feature  $c_u \in \mathcal{C}$  has its own normalised and unweighted flu score  $f_{\mathcal{C}}(\mathcal{T}, c_u) = \sum_j g(t_j, c_u) / k$  (denoted also as  $f_{c_u}$  for keeping the notation short). For a time period of  $h$  days, the flu-score time series of each candidate feature (denoted by  $f_{c_u}^{(h)}$ ) forms an  $h \times \theta$  array,  $X^{(h)} = [f_{c_1}^{(h)} \dots f_{c_\theta}^{(h)}]$ . HPA ILI rates for the same time period are denoted by  $y^{(h)}$ . We use Bolasso method for extracting a consistent set of markers with respect to the ground truth. Internally, Bolasso uses the LASSO method for performing regression with L1-regularisation which provides a sparse solution [7]. In our case, LASSO is formulated as the following optimisation problem:

$$\min_w \|X^{(h)}w - y^{(h)}\|_2^2 \quad \text{s.t. } \|w\|_1 \leq t,$$

where vector  $w$  is guaranteed to be a sparse solution and  $t$  is the regularisation parameter. Bolasso decides automatically the optimal value for  $t$ . A soft version of Bolasso is used, *i.e.* we select the markers that have non zero weights in  $s = 65\%$  to  $75\%$  of the bootstraps. Then, we perform linear least squares regression for learning the weights of the selected markers. During testing, each inferred daily flu score is smoothed (averaged) with the flu scores of the past 6 days to infer a weekly trend.

The performance of the described method is evaluated by computing the mean absolute error (MAE) between the inferred and the target values. When the ground truth signal is clearly present, we additionally compute its linear correlation with the inferences. The predictability of the selected markers is assessed by testing on the last 3 weeks (training is performed on the preceding weeks); this gives out MAEs equal to 5.27, 4.26 and 2.18 for regions  $r_1$ ,  $r_2$

<sup>2</sup> More information on the candidate features is available at <http://goo.gl/7TZA>

and  $r_3$  respectively. Based on the fact that the actual flu rates are high in the beginning of the investigated time period (when the epidemic was emerging), we also test our method's inferences for the first 3 weeks (training is performed on the succeeding weeks); for regions  $r_1$ ,  $r_2$  and  $r_3$ , the MAEs are equal to 18.34, 9.38 and 27.29, and the corresponding linear correlations are equal to 0.94, 0.84 and 0.87 (with p-values  $< 10^{-5}$ ). As an overall performance quantification, 10-fold cross validation is performed, where each fold is formed by 4 contiguous weeks; the MAE is on average equal to 11.1 with a standard deviation of 10.04<sup>3</sup>

### 3 Data Collection and Back-End Operations

We focus our data collection on tweets geolocated within a 10 Km radius from the 49 most populated urban centres in regions  $r_1$ ,  $r_2$  and  $r_3$ . Twitter's Search API is used for querying the social network periodically in order to get the most recent tweets per urban centre. The posts are retrieved in Really Simple Syndication (RSS) format and are being parsed using the ROME Java API. Data collection is a non-stop process which is performed automatically by our crawling software; the retrieved data are stored in a MySQL database.

For the experimental part of this work we are using approximately 50 million tweets, *i.e.* 200K tweets per day. Vector space representations from the corpus are produced on demand using our Java libraries; tokenisation, stop word removal, and stemming by applying Porter's algorithm [9] are embedded in the whole process. Our ground truth is formed by weekly epidemiological reports from the HPA for several regions in the UK<sup>4</sup>. The reports are based on data gathered by the Royal College of General Practitioners (RCGP) and express the number of GP consultations per  $10^5$  citizens, where the the diagnosis result was ILI. In order to retrieve an equal representation between the weekly HPA rates and the daily Twitter flu-scores, firstly, we expand each value of the former over a 7-day period, and then we smooth the expanded ground truth time series with a 7-point moving average. Feature selection and learning of the weights are performed offline; we use an implementation of Bolasso made available in the Probabilistic Modeling Toolkit (PMTK)<sup>5</sup>.

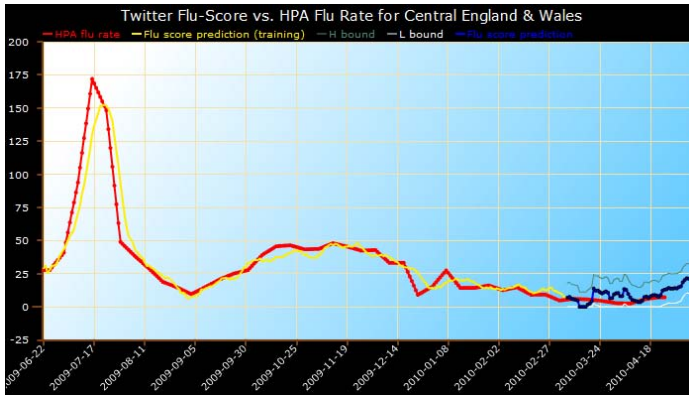
### 4 Web Interface

The Flu detector website presents the outcomes of our method (see Figure 11). An automated procedure updates the flu-score inferences on a daily basis. Visitors are able to view the inferred regional flu scores (with corresponding error bounds) in comparison with the actual HPA's ILI rates (which become available with a 7 to 10 day time lag). Apart from the aforementioned regions, we display a flu-score inference for the merged region of England & Wales as well as for the entire

<sup>3</sup> A detailed reference on evaluation procedures is available at <http://goo.gl/yZkG>

<sup>4</sup> HPA epidemiological reports, <http://goo.gl/wJex>

<sup>5</sup> Probabilistic Modeling Toolkit v.3, <http://code.google.com/p/pmtk3/>



**Fig. 1.** Flu detector's regional predictions as they appear on the website

UK - for each regional plot, the exact locations which contributed to the score are listed. The periods of training are denoted with distinct colours in every time line and the MAE between the inferred and the actual flu scores is computed and displayed.

**Acknowledgements.** The authors would like to thank Twitter Inc. for making its data publicly available. This work is partially supported by EC through the PASCAL2 NoE (FP7-216866) and EPSRC grant EP/G056447/1. V. Lampos is supported by EPSRC (DTA/ SB1826) and Nokia Research and N. Cristianini is supported by a Royal Society Wolfson Merit Award.

## References

1. Fleming, D., Elliot, A.: Lessons from 40 years surveillance of influenza in England and Wales. *Epidemiology and Infection* 136(7), 866–875 (2007)
2. Neuzil, K.M., Hohlbein, C., Zhu, Y.: Illness among schoolchildren during influenza season: effect on school absenteeism, parental absenteeism from work, and secondary illness in families. *Arch. Pediatr. Adolesc. Med.* 156(10), 986–991 (2002)
3. Ginsberg, J., Mohebbi, M.H., et al.: Detecting influenza epidemics using search engine query data. *Nature* 457(7232), 1012–1014 (2008)
4. Polgreen, P.M., Chen, Y., et al.: Using internet searches for influenza surveillance. *Clinical Infectious Diseases* 47, 1443–1448 (2008)
5. Lampos, V., Cristianini, N.: Tracking the flu pandemic by monitoring the Social Web. In: 2nd IAPR Workshop on Cognitive Information Processing, pp. 411–416 (2010)
6. Bach, F.R.: Bolasso: model consistent Lasso estimation through the bootstrap. *ICML* 25, 33–40 (2008)
7. Tibshirani, R.: Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society* 58B, 267–288 (1996)
8. Asur, S., Huberman, B.A.: Predicting the Future with Social Media. Arxiv preprint arXiv:1003.5699 (2010)
9. Porter, M.: An algorithm for suffix stripping. *Program* 14(3), 130–137 (1980)

# X-SDR: An Extensible Experimentation Suite for Dimensionality Reduction

Panagis Magdalinos, Anastasios Kapernekas,  
Alexandros Mpiratsis, and Michalis Vazirgiannis

Athens University of Economics and Business Athens, Greece  
{[pmagdal](mailto:pmagdal@aub.gr),[kapernekas](mailto:kapernekas@aub.gr),[mvazirg](mailto:m vazirg@aub.gr)}@aub.gr, [abiratsis@gmail.com](mailto:abiratsis@gmail.com)

**Abstract.** Due to the vast amount and pace of high-dimensional data production, dimensionality reduction emerges as an important requirement in many application areas. In this paper, we introduce X-SDR, a prototype designed specifically for the deployment and assessment of dimensionality reduction techniques. X-SDR is an integrated environment for dimensionality reduction and knowledge discovery that can be effectively used in the data mining process. In the current version, it supports communication with different database management systems and integrates a wealth of dimensionality reduction algorithms both distributed and centralized. Additionally, it interacts with Weka thus enabling the exploitation of the data mining algorithms therein. Finally, X-SDR provides an API that enables the integration and evaluation of any dimensionality reduction algorithm.

**Keywords:** dimensionality reduction, data mining, knowledge discovery.

## 1 Introduction

Data pre-processing is a crucial step of data mining that enables the abduction of irrelevant values from a dataset. An important aspect of data pre-processing is dimensionality reduction (DR). DR methods address challenges that rise from the large number of variables describing each observation. In high dimensional spaces typical knowledge discovery tasks, such as clustering or classification become ineffective [1]. DR algorithms apply transformations on the original dataset and embed it from the original space  $R^n$  to a new, low dimensional space  $R^k$  ( $k \ll n$ ). The objective of the methodology is to retain the distances between points or other statistical properties (i.e. variance) in the new space. Recently, due to the advent of large distributed applications, distributed dimensionality reduction (DDR) has also emerged as a decentralized pre-processing step.

Despite the large number of centralized [4] and distributed [8] and references therein) approaches we lack a software tool that integrates them towards experimenting with them in a user friendly manner. The latter, has inspired the design and implementation of X-SDR [2], a prototype that enables the integration and evaluation of any DR algorithm. X-SDR is open source and supports the evaluation of

---

<sup>1</sup> X-SDR is an acronym for eXtensible Suite for Dimensionality Reduction.



DR methods through experimentation on artificial and real world datasets. Its key features are its extensibility and user friendliness. From a user's perspective it is easy to use while from a developer's point of view, it is straightforward to extend. X-SDR supports numerous experimentation scenarios, all aligned with the evaluation methodology of applying a linear or non-linear DR method in a centralized or network environment and then assessing its quality through visualization or further experimentation with well known data mining techniques ([6]).

## 2 Related Work

The aim of this section is to present a brief outline of the various DR software packages. Due to space limitations we focus only on prototypes that primarily target on the evaluation and assessment of DR algorithms.

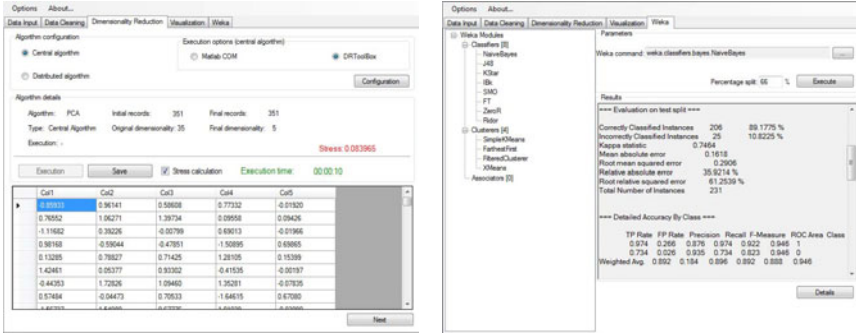
XGvis ([2], [3]) was the first attempt to offer a toolkit for experimenting with DR methods as well as visualizing their results. Unfortunately, XGvis has been confined to a number of variations of MDS ([4]). The application utilized XGobi ([9]) as a visualization frontend. XGvis evolved to GGobi ([5]), a powerful, open source suite that provides a large number of data visualization and knowledge extraction techniques specifically designed for high dimensional data. The most prominent software package in the area is the Matlab Toolbox for Dimensionality Reduction (MTDR [7]). The latter contains a vast amount of DR techniques implemented in MATLAB. Additionally, the toolbox provides implementations of methods for intrinsic dimensionality estimation, as well as functions for out-of-sample extension, prewhitening of data, and the generation of toy datasets.

Unfortunately, none of the afore described efforts assesses DR algorithms in the context of knowledge discovery. Although GGobi incorporates a large number of data mining techniques, it utilizes only a small number of DR methods. On the other hand, MTDR focuses on the algorithms rather than the evaluation of their results. Consequently, the ideal software package should combine the salient features of these two efforts and provide a user friendly frontend enabling the deployment and experimental driven assessment of any DR technique.

## 3 The X-SDR Suite

In this section we present the extensible experimentation Suite for Dimensionality Reduction (X-SDR). We commence by outlining its software architecture followed by various implementation details. Furthermore we analyze its key aspects and highlight its novel features. Finally we provide a small experimentation scenario that highlights the merits of X-SDR.

X-SDR is structured in three layers. The first comprises the data input layer that enables interaction with various data sources ranging from simple text files to database management systems (i.e. MySQL, MS SQL Server). Its main purpose is to transform the underlying data into  $n$ -dimensional vectors. These vectors are provided as input to the DR layer which incorporates a large number of DR techniques as well as the whole MTDR suite. Additionally, it supports experimentation with distributed DR algorithms, a feature that to the best of our knowledge



(a) Dimensionality reduction form (b) Assessment through classification

**Fig. 1.** Dimensionality Reduction and Data Mining forms

uniquely characterizes our framework. In order to simulate the decentralization procedure, a network profile, in the form of an adjacency matrix, should also be provided. The latter is used for the definition of a star overlay network where the central node undertakes all tasks that need to be computed centrally.

The DR layer provides two outputs. The first is a set of assessment metrics directly related to the effectiveness and efficiency of the evaluated algorithm while the second is the low dimensional embedding of the initial dataset in vector format. The assessment metrics are related to the time requirements and stress value [4] that the algorithm exhibits on a particular dataset. Moreover, for distributed algorithms, an estimation of the communication cost is also provided. The third layer aims at the assessment of the algorithms, which is accomplished either through visualization of the low dimensional dataset or further experimentation. Towards the latter we have integrated X-SDR with numerous algorithms provided by Weka [2] through proper exploitation of the OS service calls. Consequently the results of each DR algorithm are evaluated with respect to the performance of prominent clustering and classification approaches. Moreover the combination of a large number of DR methods with data mining tools promotes it as an ideal candidate for teaching as well as research activities.

The key feature of X-SDR is its extensibility which is achieved by a simple programming interface that enables any researcher to design his own algorithm and experiment through X-SDR. All required parameters are defined in XML format and incorporated as comments in the header of each MATLAB source file. Each input parameter is identified by the triplet {name, type, value}; afterwards the header of the file is parsed and the input form is dynamically created. During execution, the overarching application formulates the required calls to the MATLAB server which in turn executes the identified algorithm.

In order to provide a short demonstration scenario we use the 35-dimensional ionosphere dataset from the UCI repository [3]. The dataset is stored in a comma

<sup>2</sup> Weka <http://www.cs.waikato.ac.nz/ml/weka/>  
<sup>3</sup> <http://archive.ics.uci.edu/ml/>

separated file and is loaded by the corresponding X-SDR module. We use the PCA [4] implementation available from MTDR and embed the dataset in a 5-dimensional space (Fig. 1(a)). The produced dataset can be visualized and even compared against the initial one. We finally evaluate the embedded dataset with the use of Naive Bayes (Fig. 1(b)) and derive a classification accuracy of 89%. The obtained value is marginally equal to the one exhibited by the same algorithm in the original space, thus concluding that PCA has successfully retained data properties while projection.

X-SDR (available through <http://www.db-net.aueb.gr/panagis/X-SDR>) is implemented in C# while all algorithms are implemented in MATLAB. The interfacing of these technologies is accomplished via the COM Automation Server. X-SDR's deployment requires MATLAB and .NET framework (version 3.5 SP1) together with Microsoft Chart Controls library.

## 4 Conclusion

We have presented X-SDR, an extensible experimentation suite for dimensionality reduction algorithms. X-SDR is an open source tool, integrating a large number of DR algorithms and well known knowledge discovery tool. Moreover, X-SDR enables the simulated execution of distributed DR approaches. These features promote it as an ideal candidate platform for research and teaching in academia. Future enhancements will primarily focus on incorporating distributed data mining techniques in the package thus providing an open source application for distributed knowledge discovery experimentation.

## References

1. Beyer, K.S., Goldstein, J., Ramakrishnan, R., Shaft, U.: When is nearest neighbor meaningful? In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 217–235. Springer, Heidelberg (1998)
2. Buja, A., Swayne, D.: Visualization methodology for multidimensional scaling. *Journal of Classification* 19 (2001)
3. Buja, A., Swayne, D., Littman, M., Dean, N., Hofmann, H.: Xgvis: Interactive data visualization with multidimensional scaling. Technical Report (2001)
4. Carreira-Perpinan, M.A.: A review of dimension reduction techniques. Technical Report, CS-96-09, University of Sheffield (1997)
5. Dianne, D.C., Swayne, D., Deborah, F.: Interactive and dynamic graphics for data analysis with r and ggobi. *Journal of Computational and Graphical Statistics* (2007)
6. Gabriela, H., Martin, F.: Cluster-preserving embedding of proteins. Tech. rep., Center for Discrete Mathematics and Theoretical Computer Science (1999)
7. van der Maaten, L.: An introduction to dimensionality reduction using matlab. Technical Report MICC 07-07, Maastricht University (2007)
8. Magdalinos, P., Doulkeridis, C., Vazirgiannis, M.: K-landmarks: Distributed dimensionality reduction for clustering quality maintenance. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 322–334. Springer, Heidelberg (2006)
9. Swayne, D., Cook, D., Buja, A.: Vxgobi: Interactive dynamic data visualization in the x window system. *Journal of Computational and Graphical Statistics* 7 (1998)

# *SOREX*: Subspace Outlier Ranking Exploration Toolkit

Emmanuel Müller, Matthias Schiffer, Patrick Gerwert, Matthias Hannen,  
Timm Jansen, and Thomas Seidl

Data management and data exploration group  
RWTH Aachen University, Germany  
{mueллер,mschiffer,gerwert,hannen,jansen,seidl}@cs.rwth-aachen.de  
<http://dme.rwth-aachen.de>

**Abstract.** Outlier mining is an important data analysis task to distinguish exceptional outliers from regular objects. In recent research novel outlier ranking methods propose to focus on outliers hidden in subspace projections of the data. However, focusing only on the detection of outliers these approaches miss to provide reasons why an object should be considered as an outlier.

In this work, we propose a novel toolkit for exploration of subspace outlier rankings. To enable exploration of subspace outliers and to complete knowledge extraction we provide further descriptive information in addition to the pure detection of outliers. As witnesses for the outlier-ness of an object, we provide information about the relevant projections describing the reasons for outlier properties. We provided *SOREX* as open source framework on our website<sup>1</sup> it is easily extensible and suitable for research and educational purposes in this emerging research area.

## 1 Challenges in Outlier Exploration

In general, the task of knowledge discovery in databases is twofold. On the one side data mining methods try to *detect* meaningful patterns, while on the other side knowledge is extracted out of the data by *providing descriptions* of these patterns. Especially, for the unsupervised outlier mining task, knowledge discovery does not end with the detection of the highly deviating objects. In applications like fraud detection, health surveillance, customer segmentation or sensor monitoring, one is interested in additional descriptions about the reasons why an object seems outlying. For example in health surveillance, a young patient might be considered as outlier due to high risk of dehydration (cf.  $o_1$  in Figure 1). While dehydration is quite normal for elderly people, it is quite rare for young persons. By looking at a subset of measured attributes (*subspace projection*), one might detect this outlying patient showing high deviation from the residual patients in the attributes “age” and “skin humidity”. While traditional outlier methods measure deviation using all attributes (*full data space*), subspace outlier ranking focuses on object deviation in subspaces.

---

<sup>1</sup> <http://dme.rwth-aachen.de/OpenSubspace/SOREX>

However, not only the detection of such a high risk patient but also the underlying outlier properties are important. Providing information about the high deviation in age and skin humidity while showing normal measurements in all other attributes assists health professionals in verifying this automatically detected outlier. Thus, an obvious aim for outlier detection methods is to provide additional information about outlier properties such as the relevant attributes and to which extend a deviation from the regular objects can be observed. In Figure 1, we depict two outliers in three possible subspace projections for our toy example. As illustrated the hidden outliers show up only in specific projections while are hidden in other projections. For each outlier these specific projections can be seen as witnesses for its outlier properties.

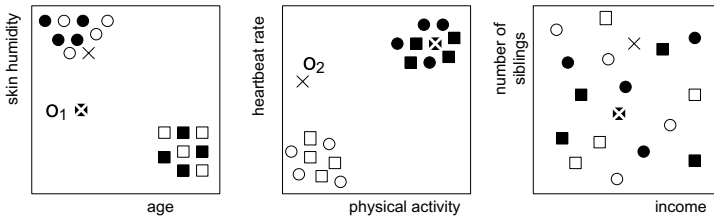


Fig. 1. Toy example (health surveillance): outliers hidden in subspaces

## 2 SOREX

Our novel SOREX toolkit provides such additional descriptions for each object. With SOREX we provide a repository of subspace outlier ranking algorithms [1–4], extending the popular WEKA framework. In addition, descriptive components provide the reasons why an object seems to be outlying. Overall, the main contributions of *SOREX* are:

- Enhancing subspace outlier ranking by descriptive components.
- Open source toolkit for outlier exploration based on the WEKA framework

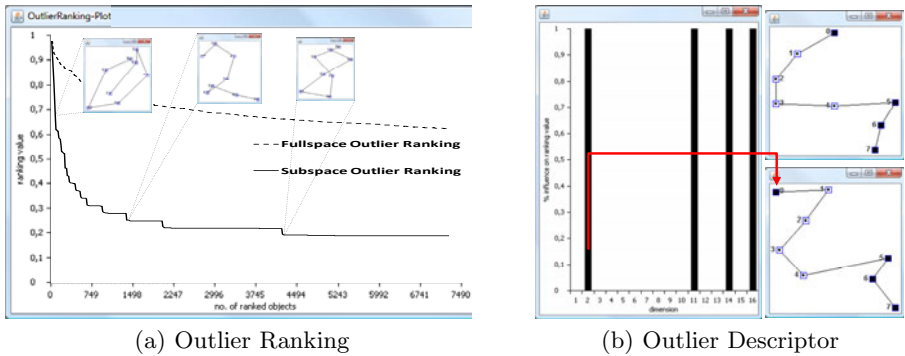
For our algorithm repository we include also some traditional outlier mining methods [5, 6]. In contrast to these full space methods, outlier detection in subspaces was first specified by [1], but without considering any additional outlier descriptions. Recent approaches have focused on outlier detection by considering projections of the data [2–4]. Their key idea is that outliers show high deviation from clustered objects in some subspaces. All of the proposed outlier ranking approaches have their focus on outlier *detection*, they provide only the ranking values as descriptive components. Thus, they are limited to providing a sorted list of most probable outliers, without giving explanations why an object seems to be an outlier. SOREX solves this drawback by additional descriptions.

Considering both mentioned contributions, *SOREX* is the first data mining framework for outlier mining that provides witnesses for the object’s outlier properties. It enables the exploration of outliers and assists in reasoning of their

outlier properties. In contrast to existing data mining toolkits such as WEKA, RapidMiner, KNIME and RATTLE our toolkit focuses on the emerging research area of subspace outlier ranking not included in these frameworks. As part of our open source initiative “*OpenSubspace*” for subspace mining covering cluster detection, evaluation and visualization in our previous work [7-9], SOREX is a key component for exploration of outliers hidden in subspace projections.

*Exploration based on descriptive components*

In addition to the ranking values provided by the implemented outlier ranking algorithms SOREX provides descriptive components about relevant subspaces and the deviation in local neighborhoods. As post-processing to any subspace outlier ranking method SOREX can be used also for future algorithm enhancements in this emerging research area. In the following we describe the descriptive information in SOREX, illustrated also for an example in Figure 2.



**Fig. 2.** Descriptive outlier ranking on Pendigits data set from UCI ML repository

In general, the ranking value has to provide a clear distinction between outliers and regular objects. As depicted in Figure 2(a), subspace outlier rankings achieve high ranking values for the few objects supposed to be outliers. Thus, they can be clearly distinguished by the rapid decrease to the regular objects showing low ranking values. Using the full space or all possible subspaces for ranking results in the depicted bad ranking [2, 5, 6], where no clear distinction is possible. Their uniform decrease in this ranking lacks a clear support for the outlier detection. State-of-the-art ranking plots with additional visualization of ranked objects form the first descriptive components in SOREX. However, they do not yet provide reasons about ranking values, especially not about the relevant subspaces.

The relevant subspaces provide knowledge about the reasons why an object should be considered outlying. As depicted in Figure 2(b) for an outlying “four” in the pendigits database we derive a histogram describing the relative contribution of each attribute to the overall ranking value. One can observe which are the most deviating attributes for the considered object. This additional knowledge can be used to verify each outlier or even to provide its outlying properties.

In addition, one is not only interested in knowledge about the responsible attributes but also about the local neighborhoods of each outlier. Objects in these local neighborhoods appear as witnesses for the outlier properties as the outlier is highly deviating from this specific set of objects. By comparing the detected outlier with these objects one can extract the differences between a set of regular objects and one rare outlier as depicted for two version of the digit “four”.

### *Demonstration of SOREX*

The demo will illustrate the exploration of outlier ranking results for several data sets. It will allow conference attendees to explore the diverse subspace outlier ranking approaches implemented in the *SOREX* system, thus raising research interest in the area. Furthermore, the interfaces of our open source toolkit will facilitate the extension with further outlier mining algorithms and visual exploration paradigms by other researchers.

Our demonstration will raise interest for future work where the extracted knowledge about reasons for outlier properties could be used for interactive subspace outlier mining. Users may interact with the resulting descriptive outlier ranking by providing specific attributes which are supposed to be the reasons for outliers or focusing on specific sets of suspicious objects. This leads to in-depth analysis of the detected outliers.

## Acknowledgments

This research was funded by the cluster of excellence on Ultra-high speed Mobile Information and Communication (UMIC) of the DFG (German Research Foundation grant EXC 89).

## References

1. Aggarwal, C.C., Yu, P.S.: Outlier detection for high dimensional data. In: SIGMOD, pp. 37–46 (2001)
2. Lazarevic, A., Kumar, V.: Feature bagging for outlier detection. In: KDD, pp. 157–166 (2005)
3. Müller, E., Assent, I., Steinhausen, U., Seidl, T.: Outrank: Ranking outliers in high dimensional data. In: DBRank Workshop at ICDE, pp. 600–603 (2008)
4. Kriegel, H.P., Schubert, E., Zimek, A., Kröger, P.: Outlier detection in axis-parallel subspaces of high dimensional data. In: PAKDD, pp. 831–838 (2009)
5. Breunig, M., Kriegel, H.P., Ng, R., Sander, J.: LOF: identifying density-based local outliers. In: SIGMOD, pp. 93–104 (2000)
6. Kriegel, H.P., Schubert, M., Zimek, A.: Angle-based outlier detection in high-dimensional data. In: KDD, pp. 444–452 (2008)
7. Müller, E., Assent, I., Krieger, R., Jansen, T., Seidl, T.: Morpheus: Interactive exploration of subspace clustering. In: KDD, pp. 1089–1092 (2008)
8. Müller, E., Günemann, S., Assent, I., Seidl, T.: Evaluating clustering in subspace projections of high dimensional data. In: VLDB, pp. 1270–1281 (2009)
9. Assent, I., Krieger, R., Müller, E., Seidl, T.: VISA: Visual subspace clustering analysis. ACM SIGKDD Explorations 9(2), 5–12 (2007)


# KDTA: Automated Knowledge-Driven Text Annotation

Katerina Papantoniou<sup>1</sup>, George Tsatsaronis<sup>2,\*</sup>, and Georgios Paliouras<sup>1</sup>


<sup>1</sup> Institute of Informatics Telecommunications, NCSR “Demokritos”, Greece




<sup>2</sup> Dept. of Computer and Information Science,  
Norwegian University of Science and Technology

kpapantoniou@iit.demokritos.gr, gbt@idi.ntnu.no,  
paliourg@iit.demokritos.gr

**Abstract.** In this paper we demonstrate a system that automatically annotates text documents with a given domain ontology’s concepts. The annotation process utilizes lexical and Web resources to analyze the semantic similarity of text components with any of the ontology concepts, and outputs a list with the proposed annotations, accompanied with appropriate confidence values. The demonstrated system is available online and free to use, and it constitutes one of the main components of the *KDTA (Knowledge-Driven Text Analysis)* module of the *CASAM* European research project .

## 1 Introduction

Reasoning about the content of text documents constitutes a key challenge to every semantics-aware document management system. One step towards this direction is the design and development of new methods that enable the automated annotation of plain text with ontology concepts. Such techniques enable the transfer of useful information from text documents to ontology structures, and vice versa. Motivated by this effort, the *CASAM* research project introduces the concept of computer-aided semantic annotation to accelerate the adoption of semi-automated multimedia annotation by the industry. In previous work, we presented part of the *KDTA (Knowledge-driven Text Analysis)* module of the overall project architecture , that is responsible for the automated annotation of text documents. In this work we demonstrate the component that automatically annotates text documents with ontology concepts.

The contributions of this work lie in the following: (a) a prototype implementation of a system that automatically annotates plain text with domain ontology concepts, using thesauri (e.g., *WordNet*) and Web resources (e.g., *Wikipedia*), and (b) a Web demo that is, publicly accessible. The rest of the paper is organized as follows: Section  summarizes the methodology of the text annotation with ontology concepts, while Section  presents the on-line demo that is publicly available. Finally, Section  concludes and provides pointers to future work.

---

\* The author conducted part of this work while in the Bioinformatics Group, Biotechnological Center, Technische Universität Dresden.

<sup>1</sup> *Computer-Aided Semantic Annotation of Multimedia* -  
<http://www.casam-project.eu>



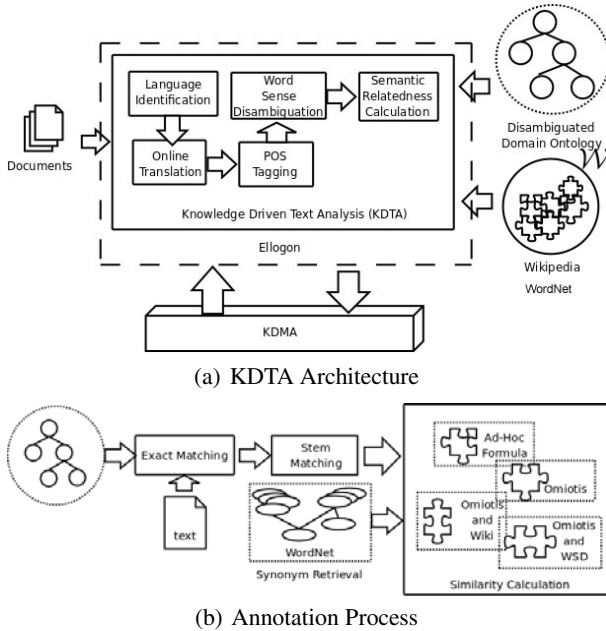


Fig. 1. The KDTA overall architecture (a), and the details of the annotation process (b)

## 2 Automated Annotation of Text with Domain Ontology Concepts

Text annotation with ontology concepts constitutes a fundamental technology for intelligent Web applications, e.g., Semantic Web, and for this reason, there has been a large focus in this research direction over the past few years, e.g., [11, 12]. However, most of the previous approaches required a lot of human intervention, or were able to annotate only specific parts of text, like named entities. In our recent work [5], we presented a new method for the automated annotation of plain text with concepts residing in a given domain ontology. The method combines a pre-processing and a semantic annotation phases (Fig. 1). In the pre-processing phase the text is processed syntactically and semantically by generic tools. The semantic annotation phase, which is the core of the method, utilizes the WordNet thesaurus<sup>2</sup> and the Wikipedia electronic encyclopedia<sup>3</sup>. The proposed method combines measures of semantic relatedness and word sense disambiguation (WSD) algorithms to annotate text words with ontology concepts.

In Fig. 1 the architecture of the *KDTA* component of *CASAM*, as well as the details of the semantic annotation process are shown (Fig. 1(a) and 1(b) respectively). The latter is the core of the proposed method and utilizes *WordNet*, and *Wikipedia* as knowledge bases, as well as two respective measures of semantic relatedness: a dictionary-based measure, namely *Omiotis* [4], which is based on *WordNet*, and a *Wikipedia*-based

<sup>2</sup> <http://wordnet.princeton.edu/>

<sup>3</sup> <http://www.wikipedia.org/>

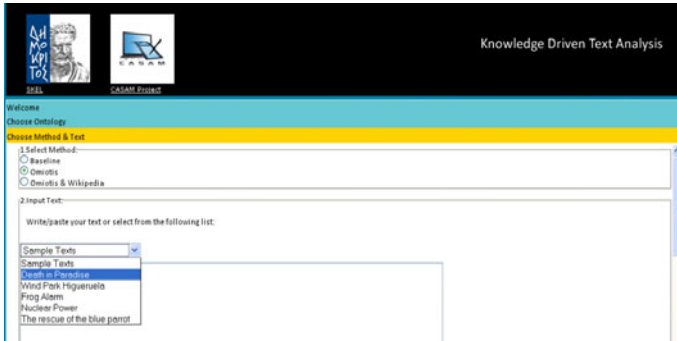


Fig. 2. Loading text in the KDTA annotation demo

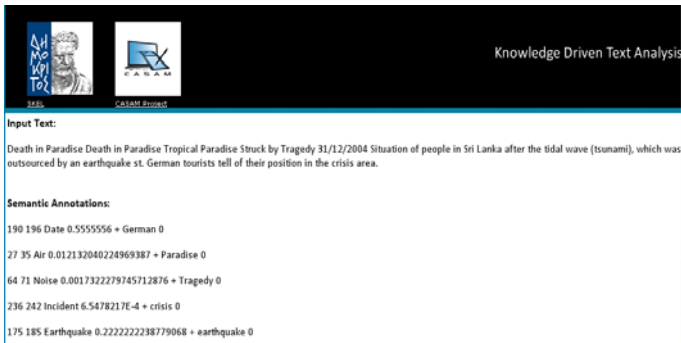


Fig. 3. Ontology-based annotation suggestions in the provided text document

measure [3]. Both measures have been shown to achieve state-of-the-art performance in measuring word-to-word semantic relatedness [4]. Both methods are also very efficient.

### 3 System Demonstration

A Web application that implements KDTA is publicly available online<sup>4</sup>. Firstly, the user may upload an ontology in *OWL* format or may select an already existing ontology by leaving the corresponding browsing path empty. This will load the default ontology of the environmental domain. Next, the user may select the text to be annotated. As Fig. 2 shows, there is already a list of text documents in the server that can be used, pertaining to environmental news. Alternatively, the user may write the text to be annotated in the provided text area. Finally, in Fig. 3 the results of the suggested annotations are shown. Column 1 is the id of the annotation, columns 2 and 3 are the start and end offsets of the annotation measured in characters, column 4 is the ontology concept that matches the respective part of the text, column 5 is a confidence value from 0 to 1, and column

<sup>4</sup> <http://phoebe.iit.demokritos.gr:8480/KDTA/>

6 is the source term from the given text, that led to the choice of the annotation. The method was experimentally evaluated in two domains, environmental and biomedical. For the environmental domain two datasets were examined provided by LUSA<sup>5</sup> and Deutsche Welle<sup>6</sup>, while for the biomedical domain MEDLINE abstracts were used. The performance was measured in terms of Macro Average Precision, Recall and F-measure with the aid of a manually created gold standards for each dataset. The results show that the method can achieve high F-measures, up to 73% in some cases. Analytical results and a detailed description of the evaluation process have been presented in [5].

## 4 Conclusions and Future Work

In this paper we have presented briefly an on-line web application of a system that annotates automatically text documents with concepts from a given domain ontology. The annotation system has already been embedded in the *CASAM* prototype successfully, and evaluation results have shown that provides accurate text annotation with ontology concepts. In the future, we plan to investigate the usage of more measures of semantic relatedness, and attempt to ensemble their confidence.

## Acknowledgements

This work has been supported by the EU, in the context of the *CASAM* project (Contract number FP7-217061, Web site: [www.casam-project.eu](http://www.casam-project.eu)). We would like to thank our partners in the project for providing us with the environmental ontology and the corresponding data.

## References

1. Cimiano, P., Ladwig, G., Staab, S.: Gimme' the context: context-driven automatic semantic annotation with c-pankow. In: WWW, pp. 332–341 (2005)
2. El-Beltagy, S., Hazman, M., Rafea, A.: Ontology based annotation of text segments. In: SAC (2007)
3. Milne, D., Witten, I.: An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In: AAAI Workshop on Wikipedia and Artificial Intelligence (2008)
4. Tsatsaronis, G., Varlamis, I., Vazirgiannis, M.: Text relatedness based on a word thesaurus. *Journal of Artificial Intelligence Research* 37, 1–39 (2010)
5. Zavitsanos, E., Tsatsaronis, G., Varlamis, E., Paliouras, G.: Scalable semantic annotation of text using lexical and web resources. In: Konstantopoulos, S., Perantonis, S., Karkaletsis, V., Spyropoulos, C.D., Vouros, G. (eds.) SETN 2010. LNCS, vol. 6040, pp. 287–296. Springer, Heidelberg (2010)

<sup>5</sup> <http://www.lusa.pt/lusaweb/>

<sup>6</sup> <http://www.dw-world.de/>

# Detecting Events in a Million New York Times Articles

Tristan Snowsill<sup>1</sup>, Ilias Flaounas<sup>2</sup>, Tijl De Bie<sup>1</sup>, and Nello Cristianini<sup>1,2</sup>

<sup>1</sup> Department of Engineering Mathematics, University of Bristol

<sup>2</sup> Department of Computer Science, University of Bristol

tristan.snowsill@bristol.ac.uk

**Abstract.** We present a demonstration of a newly developed text stream event detection method on over a million articles from the New York Times corpus. The event detection is designed to operate in a predominantly on-line fashion, reporting new events within a specified timeframe. The event detection is achieved by detecting significant changes in the statistical properties of the text where those properties are efficiently stored and updated in a suffix tree.

This particular demonstration shows how our method is effective at discovering both short- and long-term events (which are often denoted topics), and how it automatically copes with topic drift on a corpus of 1 035 263 articles.

**Keywords:** event detection, suffix tree, New York Times.

## 1 Introduction

In recent years we have witnessed an explosion of information through the ever-growing presence of the internet, the World Wide Web and more recently blogs and microblogs (*e.g.*, Twitter, Facebook statuses). There is a clear and present need for computational methods for processing the overwhelming amount of data which may be of interest to any of us.

Event detection and document summarisation methods can prove invaluable to people from all walks of life: journalists need to report the latest developments in current events, many of which are first reported online; investors need to very quickly assess the potential effect of breaking news on the values of their investments; companies need to monitor complaints and abuses of their products and services and many more uses.

One of the potential difficulties in identifying effective computational methods is in finding a dataset which is realistic in both content and scale. To the authors' knowledge the largest public test dataset for event detection is the TDT2 dataset with under 50 000 documents. To put this in perspective we have a system which indexes over 10 000 documents from the WWW every day.

Recently the New York Times released an annotated corpus of their articles from 1987 to 2007 with over 1.8 million documents [1]. This corpus is potentially an excellent testbed for computational methods as it is many orders of magnitude larger than existing corpuses and will truly test the ability of methods to scale.

## 2 Our Method

Our method is described in [2], so we summarise our method here. As articles from a text stream are observed they are added to a *generalised suffix tree*, a data structure which efficiently indexes the substrings of a set of texts. We annotate the suffix tree at its nodes with sufficient statistical information to perform an hypothesis test for every node (and due to the suffix tree structure for every  $n$ -gram in the text). Each hypothesis test gives the extent to which the observed frequency of an  $n$ -gram has increased from its expected frequency (calculated as a weighted average of its observed frequency in the past) in the form of a  $p$ -value.  $n$ -grams with a lower  $p$ -value have undergone a more significant increase in frequency. By ranking  $n$ -grams in order of ascending  $p$ -value we see the most significant  $n$ -grams which are indicators of events.

The suffix tree of a text of length  $n$  (or a set of texts with a combined length of  $n$ ) can be stored in  $\mathcal{O}(n)$  space. The suffix tree can be built in  $\mathcal{O}(n \log |\Sigma|)$  time, where  $\Sigma$  is the *alphabet* (the set of symbols from which documents are drawn). In many applications of suffix trees  $|\Sigma|$  is very small and  $\mathcal{O}(n)$  construction time is often given, *e.g.*, for DNA,  $\Sigma_{\text{DNA}} = \{a, c, g, t\}$ . In our method  $|\Sigma|$  is very large as it is the set of all words and punctuation symbols observed in the text as the text is tokenized at word boundaries and punctuation symbols. Our construction is a modification of Ukkonen's method [3] designed to be more efficient when the suffix tree is stored in external memory (*i.e.*, on a hard disk).

## 3 The Dataset

We considered articles from the New York Times annotated corpus from 1<sup>st</sup> January 1996 to 18<sup>th</sup> June 2007. We used only the *Body* attribute of each article (*i.e.*, we did not include the headlines or any tags). There are 1 035 263 articles in this time period with a total size of 3 401 MB and a length of 793 500 772 symbols after tokenization with an alphabet containing 1 422 974 symbols (stored in 12 MB).

## 4 Results

The suffix tree constructed from the corpus was 36 216 MB in size, a 10.6-fold increase in size over the original corpus text. Table 1 gives further details of the suffix tree.

We computed the  $p$ -value for each  $n$ -gram by comparing different weighted averages of the frequency of the  $n$ -gram. The weighted averages used were exponentially weighted averages,

$$\hat{x}_i = \alpha \hat{x}_{i-1} + (1 - \alpha) x_i,$$

where  $\alpha$  is a weighting parameter. We used six different exponential weighting parameters:  $\alpha = \{0, 0.215, 0.518, 0.720, 0.858, 0.950\}$ ; these correspond to 99%

**Table 1.** The suffix tree created from the New York Times corpus

Sequences	1 035 263
Suffixes	793 500 772
Alphabet size	1 422 974 (12.1 MB)
Nodes	
Internal	206 163 152
Leaf	777 468 706
Size on hard disk	36 216 MB

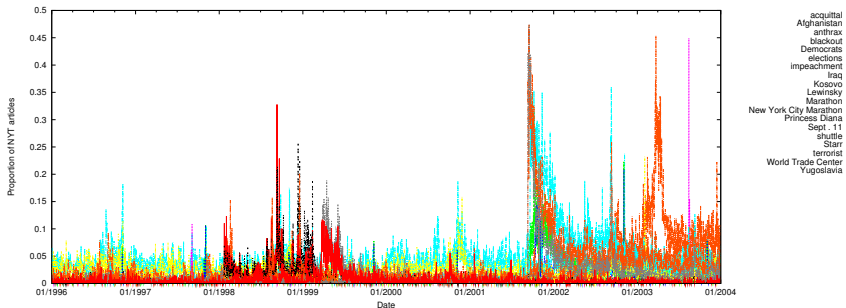
decay after 0 (*i.e.*, memoryless), 3, 7, 14, 30 and 90 timeframes respectively. For each value of  $\alpha$  we stored a weighted average of the number of articles containing each  $n$ -gram, and the total number of articles in the timeframe. In addition we stored a timestamp for each  $n$ -gram recording the last timeframe it was observed in, allowing for just-in-time updates of the weighted averages. These weighted averages allowed for efficient computation of a  $p$ -value for each  $n$ -gram for each pairing of  $\alpha$  values.

The statistics were stored in C++ floats for real values and in C++ ints for integer values. The total space requirement for each node in the suffix tree for the statistics was therefore

$$[4n_\alpha + 8] \text{ bytes} = 32 \text{ bytes.}$$

We chose not to compute  $p$ -values for leaf nodes as their frequencies are almost always 0 or 1. The total space requirement for the statistics is then 6.14 GB and the *total* space requirement of our method was 41.5 GB.

We considered three time periods, 1996–2000, 2000–2004 and 1996–2004. For each of these time periods and for each pairing of  $\alpha$  values we calculated the forty most significant  $n$ -grams. We show a selected subset of these in Table 2<sup>1</sup>. We also show the timelines of these  $n$ -grams for the entire duration in Figure 1.



**Fig. 1.** Timelines of the most significant  $n$ -grams

<sup>1</sup> We believe it will be possible to automate this selection based on linguistic properties.

**Table 2.** The top five events detected in the three time periods using different comparison methods

Short term comparison method ( $\alpha_1 = 0, \alpha_2 = 0.215$ )		
1996–2000	2000–2004	1996–2004
New York City Marathon	Afghanistan	Afghanistan
Starr	Marathon	New York City Marathon
elections	terrorist	terrorist
Democrats	World Trade [Center]	World Trade [Center]
impeachment	Iraq	Iraq
Medium term comparison method ( $\alpha_1 = 0.518, \alpha_2 = 0.720$ )		
1996–2000	2000–2004	1996–2004
elections	World Trade [Center]	World Trade [Center]
Lewinsky	blackout	blackout
Iraq	elections	elections
acquittal	terrorist	terrorist
New York City Marathon	shuttle	Lewinsky
Long term comparison method ( $\alpha_1 = 0.858, \alpha_2 = 0.950$ )		
1996–2000	2000–2004	1996–2004
elections	World Trade Center	World Trade Center
impeachment	Sept. 11	Sept. 11
Kosovo	elections	elections
Yugoslavia	blackout	blackout
Princess [Diana]	anthrax	anthrax

## 5 Discussion

We have demonstrated our method for event detection on a large publicly available corpus of text. The method draws from areas of hypothesis testing and combinatorial pattern matching to solve a text stream mining problem.

Further results will be shown on the website for the project.<sup>2</sup>

## References

1. Sandhaus, E.: The New York Times Annotated Corpus. In: Linguistic Data Consortium, Philadelphia (2008)
2. Snowsill, T., Nicart, F., Stefani, M., De Bie, T., Cristianini, N.: Finding surprising patterns in textual data streams. In: 2010 IAPR Workshop on Cognitive Information Processing (2010)
3. Ukkonen, E.: On-line construction of suffix trees. *Algorithmica* 14(3), 249–260 (1995)

<sup>2</sup> [http://memewatch.enm.bris.ac.uk/new\\_york\\_times/](http://memewatch.enm.bris.ac.uk/new_york_times/)

# Experience STORIES: A Visual News Search and Summarization System

Ilija Subašić and Bettina Berendt

Department of Computer Science, K.U. Leuven, Belgium

**Abstract.** Using data collections available on the Internet has for many people become the main medium for staying informed about the world. Many of these collections are dynamic by nature, evolving as the subjects they describe change. We present the STORIES system for (a) learning an abstracted story representation from a collection of time-indexed documents; (b) visualizing it in a way that encourages users to interact and explore in order to discover temporal “story stages” depending on their interests; and (c) supporting the search for documents and facts that pertain to the user-constructed story stages.

## 1 Introduction

Search engines, RSS feeds, micro-blogging tools, and many other services support Internet users in *story tracking*: following the developments of topics over time. This is usually done by manually or automatically issuing a series of same queries about an event (“Haiti Earthquake”), a person (“Britney Spears”), or a scientific area (“text mining”). This creates a challenge for information seekers because large numbers of new documents from different sources keep arriving at a fast rate. There is clearly a need for different search interfaces and search experience, which provide a concise abstracted representation of information from all the pertinent parts of a source or source set. Users will be able to profit most from summarising services that provide convenient interfaces to both the abstracted summary and the underlying documents, and that allow for and encourage a flexible, (inter)active exploration of the space of the abstracted “stories” and at the same time searches of the space of documents.

To enhance the story tracking search experience, we pursue two goals: (a) give users a more active role in the search process, and (b) break away from traditional “top-10” ranked document search interfaces. As a part of this we present the STORIES system for news-stories tracking, which instantiates these ideas. It consists of story learning (done by the system) and graphical support for story understanding and story search (provided to the user). The paper builds on the detailed explanation in [1]; it describes a re-implementation with new interface features and examples from a new corpus.

## 2 Related Work

Apart from “classical” news search engines like Google News or Yahoo! News, recently many alternative ways of tracking and browsing news collections have been



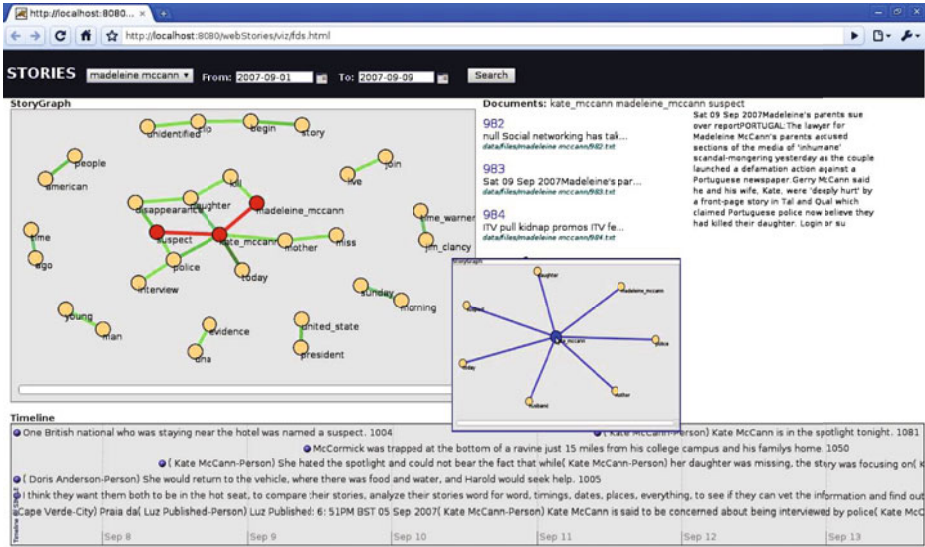
developed. *Google News Timeline* (<http://newstimeline.googlelabs.com/>) provides a pre-set time period (day, week, month, year) overview of news using a timeline interface. It allows for the tracking of news sources, arbitrary queries or entities such as movies, books, music... Another Google system, named *Fast Flip* (<http://fastflip.googlelabs.com/>), provides an interface for browsing news articles resembling hard-copy newspaper reading. The *Yahoo! Correlator* (<http://correlator.sandbox.yahoo.net/>) associates a search term with all its related “events”. *EMM NewsExplorer* (<http://emm.newsexplorer.eu>) and *EMM NewsBrief* (<http://emm.newsbrief.eu>) are news search and summarization services tracking news from a number of multi-lingual sources. The *SearchPoint* (<http://searchpoint.ijs.si/>) system allows users to focus on a specific sub-topic of search results. *MemeTracker* (<http://www.memetracker.org/>) tracks quotes from news and visualises their “burstiness” using interactive charts. These deployed systems rest on a growing body of work on topic detection and tracking, temporal text mining, and visual web search surveyed in [1].

In contrast to these other tools, our system combines visual search, summarization, and burst-pattern detection into a single interface which provides an interactive inspection of temporal changes in a corpus.

### 3 Method

First, a corpus of text-only documents is transformed into a sequence-of-terms representation. Subsequently, basic term statistics are calculated to identify candidates for story basics. We applied different measures to obtain the story basics including the top ranked words based on term frequency, TF.IDF weight, combining “regular” terms with terms referencing some named entities, and all terms form a corpus.

For *story understanding*, we analyse a text corpus and its (user-definable) time-indexed subsets. For each time-indexed subset of the whole corpus  $c_i$ , the *frequency* of the co-occurrence of all pairs of content-bearing terms  $b_j$  in documents is calculated as the number of occurrences of both terms in a window of  $w$  terms, divided by the number of all documents in  $c_i$ . This measure of frequency and therefore relevance is normalised by its counterpart in the whole corpus  $C$  to yield *time relevance* as the measure of burstiness:  $TR_i(b_1, b_2) = (freq_i(b_1, b_2))/(freq_C(b_1, b_2))$ . Thresholds are applied to avoid singular associations in small sub-corpora and to concentrate on those associations that are most characteristic of the period and most distinctive relative to others: This gives rise to the *story graphs*  $G_i = \langle V_i, E_i \rangle$  for time periods  $i$ . The edges  $E_i$  of  $G_i$  are the *story elements*: all pairs  $(b_1, b_2)$  with absolute frequencies and  $TR$  above the respective thresholds. The nodes  $V_i$  of  $G_i$  are the *story basics*, the terms involved in at least one association in this symmetric graph:  $\{b_j | \exists b_k : (b_j, b_k) \in E_i\}$ . From each document, we extract sentences containing “facts”, short statements with semantic role labelling, as returned by Open Calais (<http://www.opencalais.com/>). The full set of these sentences for each



**Fig. 1.** Web interface: A *story graph* (left) is built based on articles about the disappearance of a person. By marking the edges connecting the person’s name (top node of the subgraph marked in red) with another name (middle node) as a “suspect” (left node), the user obtains a list of pertinent documents (centre), whose text can be inspected (right). The *timeline* (bottom) shows the important “facts” from a selected time period. The overlaid *tracking story graph* shows how the search can be focused on the chosen node over different time periods.

time period is indexed using Lucene (<http://lucene.apache.org>). We then use story graphs to filter the most important facts: for each of the graph’s edges, we query the index, using node names of the edge as query terms, and select the top sentences as defined by Lucene. We treat the resulting set of short textual statements as a summary of the story.

*Story search* can be constrained by the nodes of a subgraph of the story graph. Retrieval is then restricted to documents relevant to these subgraphs. The selection of documents of the starting corpus  $C$  corresponds to a top-level query; this query is expanded by the information from the subgraph and the time restriction. STORIES then uses all the nodes  $n$  as a query (restriction) for the documents inside  $c_i$  to obtain the pertinent document subset, as identified by a search over a Lucene index.

## 4 Tool

Figure 1 shows a screenshot of our STORIES front-end. We apply the method to news articles obtained from different sources on the Web. Corpora can be compiled either on a continuing basis (e.g., subscribed-to feeds) or in response to a top-level query to a search engine. Our indexing service crawls a number

of news aggregators and retrieves documents for a given top-level query. The top-level query describes the whole story (e.g., “Haiti Earthquake” or “*person name*” for crime cases or celebrity reporting). Data cleaning and other data preparation steps are then applied, in particular HTML wrapper induction and removal, tokenisation, cross-document named-entity recognition, lemmatisation, and stopword removal. Finally, document and term measures as described in the previous section are computed. The system backend is developed in Java and the front-end using GUESS (<http://graphexploration.cond.org/>) library for the stand-alone version, and a JavaScript implementation of the spring-layout algorithm for the web version of the tool.

The primary representations are visualisations of story graphs. They provide functionalities to: (1) scan over time to track the global story evolution, (2) zoom in and out by time, by adapting the period-window size, (3) zoom in and out by detail, uncovering more details about the story by setting the *TR* values (a configurable colour coding schema of edges points to the different values of time relevance), (3) tracking certain terms or entities in time, by selecting the corresponding node. This outputs a graph of bursty co-occurrences including this “tracking node” as its central node.

By clicking on a single edge, the user can select documents associated with the term pair. For easier and more flexible search, users may also select an edge and then highlight a subgraph which contains the selection’s adjoining edges and neighbouring nodes. Each selected edge expands the query and restricts the document set based on it. In this way, the user incrementally builds the query and at the same time can discover and learn about the story. Search provides a list of documents, and a set of sentences visualized along the time-line using one day as an atomic period. This allows users to interact with the story using different views: patterns, sentences (“facts”), and documents.

## 5 Evaluation

Evaluations of *search quality* demonstrated that STORIES finds coherent subsets of documents, that its quality is comparable to or better than state-of-the-art clustering, and that the tool enables people to answer questions on ground-truth events accurately and quickly [1]. We also *compared our method with other “bursty-pattern discovery” methods*, with a framework that leverages sentence retrieval and internal pattern structure and evaluates the sentences against a ground truth. Our experiments showed that different methods perform at similar levels overall, but provide distinctive advantages in some settings [2]. In a third, ongoing round of evaluations we perform a *user study* evaluating how users can discover information using our and other search interfaces.

## 6 Outlook

In future work, we will investigate more advanced language processing (linguistic parsing, semantic role labelling for story graphs, etc.), the use of lexical

resources and other background knowledge, as well as different sources of media bias/viewpoints. We also plan to explore aggregation and analysis dimensions other than time, such as multilinguality. Further quantitative and qualitative evaluations will be carried out.

## References

1. Subašić, I., Berendt, B.: Discovery of interactive graphs for understanding and searching time-indexed corpora. *Knowledge and Information Systems* 23(3), 293–319 (2010)
2. Subašić, I., Berendt, B.: From bursty patterns to bursty facts: The effectiveness of temporal text mining for news. To appear in *Proc. ECAI'2010* (2010)

# Exploring Real Mobility Data with M-Atlas

R. Trasarti, S. Rinzivillo, F. Pinelli, M. Nanni, A. Monreale,  
C. Renso, D. Pedreschi, and F. Giannotti

Pisa KDD Laboratory, ISTI - CNR, Italy

## 1 Introduction

Research on moving-object data analysis has been recently fostered by the widespread diffusion of new techniques and systems for monitoring, collecting and storing location aware data, generated by a wealth of technological infrastructures, such as GPS positioning and wireless networks. These have made available massive repositories of spatio-temporal data recording human mobile activities, that call for suitable analytical methods, capable of enabling the development of innovative, location-aware applications [3]. The M-Atlas is the evolution of the system presented in [5] allows to handle the whole knowledge discovery process from mobility data. The analysis capabilities of M-Atlas system have been applied onto a massive real life GPS dataset, obtained from 17,000 vehicles with on-board GPS receivers under a specific car insurance contract, tracked during one week of ordinary mobile activity in the urban area of the city of Milan; the dataset contains more than 2 million observations leading to a set of more than 200,000 trajectories (see Fig. 1).

## 2 The M-Atlas System

A system able to master the complexity of the knowledge discovery process over mobility data needs to support at least four functionalities: (i) trajectory data need to be created, stored and queried through spatio temporal primitives; (ii) trajectory models and patterns representing collective behavior have to be extracted using trajectory mining algorithms; (iii) such patterns and models have to be represented and stored in order to be re-used or combined; (iv) new mining algorithms may be added. The M-Atlas system allows the user to combine all these aspects through an innovative Data Mining Query Language (DMQL). This language can be used to express the whole knowledge discovery process as a sequence of queries to be submitted to the system. The GUI interface gives the user the possibility to use pre-defined analysis (i.e. O/D Matrix) or to use the console to write down his/her own DMQL queries.

In the next sections we will give a short example of the capabilities of the system on the real dataset of Milan described above.

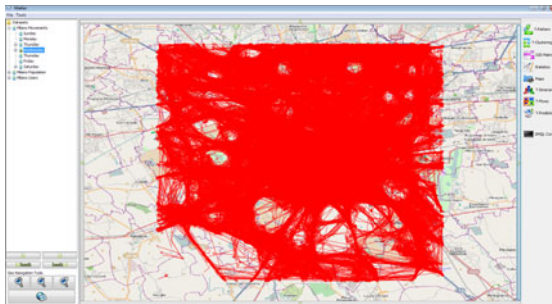
## 3 Understanding Mobility

To grasp a general vision on the dataset we performed a series of statistical analysis on the dataset (the charts representing the results are shown in Figure 2):

**The Movements Distribution analysis** estimates the active movements in each hour of the week. From the figure it is evident the drop in the movements during the night hours, clearly separating the different days, hence the seven groups represent the seven days from Sunday to Saturday. The shapes of the days are very similar especially during the week-days: each day contains two peaks, one in the morning and one in the late afternoon. This analysis validates the dataset comparing it with a survey obtained from the Mobility agency of Milan in 2005.

**The Cumulative Lengths Distribution** represents the cumulative number of trajectories having the same length. From this analysis it is clear that in the city there are many short movements and few long movements obtaining a distribution which follows a power law.

**The Density of Length over Speed** represents in cold colors a low density and the warm colors an high density. In this way we can see the variance of lengths for each speed value, looking at the *existence* of points and the common value looking for the warmer ones. For Example the trajectories with speed around 20 km/h have length from 1 to 190 km with a very high density zone between 1 and 30 km.



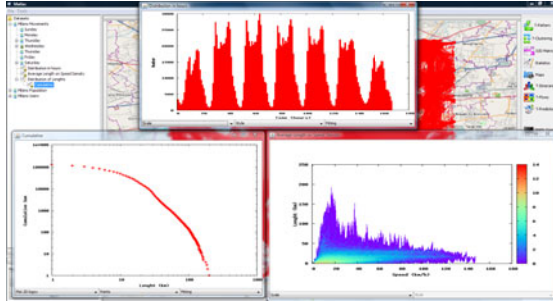
**Fig. 1.** The Trajectory Dataset of Milan

In the next section we show some analytical tools used in combination with data mining algorithms in order to answer to a specific request of the mobility manager.

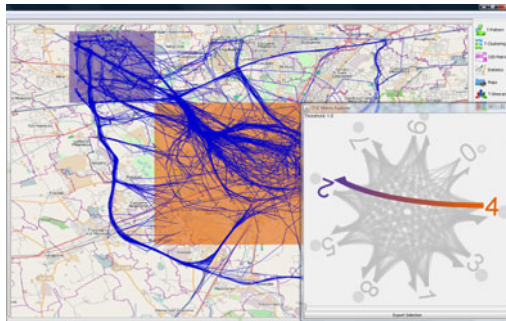
## 4 Discovering Mobility Behaviors

The M-Atlas system integrates a set of analytical and data mining tools such as the construction of Origins-Destinations Matrix, the construction of georeferenced density maps according to different measures, extracting of T-Patterns [2], T-Clustering [6], T-Itineraries [1] and T-Prediction [4]. In this section we present an example of these tools as basic bricks to build a discovery process. The presented analysis answers to a specific requirements of the mobility manager: *How the people leave the city center during the day?*.

**Computation of O/D Matrix:** The common exits of the city has been discovered building an O/D Matrix between the zones of the city represented in a big scale by a  $3 \times 3$



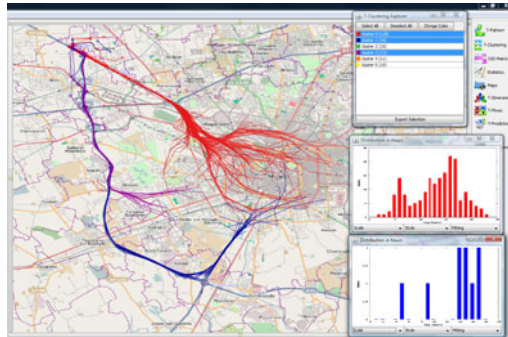
**Fig. 2.** Statistical Analysis of the dataset: Movements Distribution (top), Cumulative Lengths Distribution (left) and Density of Length over Speed (right)



**Fig. 3.** The O/D Matrix between zones of the city

matrix. Each zone in the border corresponds to an exit and the zone in the middle corresponds to the city center. In Fig 3 in the bottom right corner we can see a conceptual representation of the flows between the zones (in gray) and a particular flow from the zone n.4 (the city center, in orange) to the zone n.2 (the north-west exit, in violet) highlighted as the *most dense flow starting from the city center* (according to the size of the arrow). Hence the set of trajectories which is part of the selected flow can be extracted (see Fig 3) which can be used for further analysis.

**T-Clustering:** Given the trajectories leaving the city center toward the north-west exit, we want to group together similar trajectories in order to discover common behaviors. Therefore a data mining tool called *T-Clustering* is used. M-Atlas provides different methods to describe the similarity between two trajectories, one of them is *route similarity* which performs a temporal alignment and then compares spatially the two trajectories. The result of this computation is shown in Fig 4 where each cluster is represented by a different color. There are three major clusters: (i) the *red* one is the group of people which leaves the very center of the city via the north-west city gate and then goes straight to the north-west exit, (ii) the *purple* one represents the people leaving the center from a peripheral part of the city center and (iii) the *blue* one is the group of people which prefers to avoid the crowd and opts for a



**Fig. 4.** The result of T-Clustering tools and the distribution of movements during the day for the red and blue clusters

longer way but probably faster. This last hypothesis is supported also by the analysis of the temporal distribution of the two cluster (red and blue) which highlight the simultaneity of the peaks of the movements distribution in the afternoon as shown in Fig. 4 (right).

The same analysis can be applied also for the other exit directions, presenting to the mobility manager a complete view of behaviors.

## 5 Conclusion

The presented process of analysis is just a sketch of the power of the M-Atlas system which allows the analyst to combine tools in order to build his/her own discovery knowledge process in an iterative and interactive way.

*Acknowledgments.* The authors wish to acknowledge the Mobility Agency of Milano and Octotelematics S.p.A. for providing the anonymized dataset of private cars with on-board GPS receivers.

## References

1. Benkert, M., Gudmundsson, J., Hbner, F., Wolle, T.: Reporting flock patterns. *Computational Geometry* 41, 111–125 (2008)
2. Giannotti, F., Nanni, M., Pinelli, F., Pedreschi, D.: Trajectory pattern mining. In: *KDD*, pp. 330–339 (2007)
3. Giannotti, F., Pedreschi, D. (eds.): *Mobility, Data Mining and Privacy - Geographic Knowledge Discovery*. Springer, Heidelberg (2008)
4. Monreale, A., Pinelli, F., Trasarti, R., Giannotti, F.: Wherenext: a location predictor on trajectory pattern mining. In: *Proc. ACM KDD*, pp. 637–646 (2009)
5. Trasarti, R., Giannotti, F., Nanni, M., Pedreschi, D., Renso, C.: A Query Language for Mobility Data Mining. *International Journal of Data Warehousing and Mining* (2010) (print on)
6. Andrienko, G., Andrienko, N., Rinzivillo, S., Nanni, M., Pedreschi, D.: A Visual Analytics Toolkit for Cluster-Based Classification of Mobility Data. In: *SSTD*, pp. 432–435 (2009)



# Author Index

- Abdelzaher, Tarek II-35  
Airola, Antti II-499  
Akoglu, Leman III-354  
Aldinucci, Marco I-7  
Almeida, Jussara M. II-402  
Ang, Hock Hee I-24  
Anil Kumar, V.S. II-111  
Arias, Marta III-338  
Atkinson, Martin III-591  
Attenberg, Josh I-40  
Auer, Peter I-554  
Aussem, Alex III-164
- Bach, Francis III-515  
Baeza-Yates, Ricardo I-168  
Bai, Bing II-128  
Baldassarre, Luca I-56  
Barat, Cécile I-72  
Barla, Annalisa I-56  
Bartlett, Peter L. II-66  
Batal, Iyad I-87  
Bavaud, François I-103  
Belém, Fabiano II-402  
Bennett, Kristin P. II-145  
Berendt, Bettina III-619  
Berthold, Michael R. III-587  
Bertini, Matteo II-259  
Besada-Portas, Eva I-119  
Bhattacharya, Indrajit I-409  
Bifet, Albert I-135  
Blockeel, Hendrik II-369  
Böhm, Christian I-151, III-245  
Böhm, Klemens I-425  
Borboudakis, Giorgos III-322  
Bordino, Ilaria I-168  
Bringmann, Björn III-563  
Buchwald, Fabian III-213  
Buhmann, Joachim M. III-83
- Carmona, J. I-184  
Castro, Pablo Samuel I-200  
Chen, Qing II-337  
Chen, Ye II-483  
Cheng, Weiwei I-215, I-280  
Chou, Bin-Hui III-306
- Cléménçon, Stéphan I-248  
Collobert, Ronan II-128  
Cortadella, J. I-184  
Cramer, Tobias II-353  
Cristianini, Nello III-599, III-615
- Dai, Guang I-361  
Dali, Lorand III-579  
d'Amato, Claudia I-442  
Danafar, Somayeh I-264  
Danilevsky, Marina I-570  
De Baets, Bernard I-215, II-499  
De Bie, Tijn III-599, III-615  
DeJong, Gerald II-243  
de la Cruz, Jesus M. I-119  
del Coz, Juan José III-115  
Dembczyński, Krzysztof I-280  
de Morais, Sérgio Rodrigues III-164  
de Moura, Edleno Silva II-402  
de Vries, Gerben I-296  
Di Castro, Dotan I-312  
Diethe, Tom I-328  
Dijkstra, Tjeerd M.H. I-506  
Ding, Chris II-451, III-451  
Domeniconi, Carlotta III-435  
Donato, Debora I-168  
Dong, Bing I-587  
Doppa, Janardhan Rao I-344  
Dou, Wenjun I-361  
Du, Jun I-377  
Du, Nan I-393  
Dubey, Avinava I-409  
Ducottet, Christophe I-72  
Dupont, Pierre I-522
- Eichinger, Frank I-425  
Esposito, Floriana I-442
- Faloutsos, Christos I-1, I-393,  
III-99, III-354  
Faloutsos, Michalis III-99  
Fan, Wei III-483, III-547  
Fanizzi, Nicola I-442  
Fern, Alan III-467  
Fiedler, Frank I-151  
Flaounas, Ilias III-615

- Fortuna, Blaž III-579, III-583  
 Fortuna, Carolina III-583  
 Frasconi, Paolo II-259  
 Fromont, Elisa I-72
- Gao, Jing I-570, II-337  
 Garcke, Jochen I-458  
 Gerwert, Patrick III-607  
 Getoor, Lise I-344  
 Ghodsi, Ali II-19  
 Giannotti, F. III-624  
 Girschick, Tobias III-213  
 Godbole, Shantanu I-409  
 Gonçalves, Marcos André II-402  
 Gopalkrishnan, Vivekanand I-24  
 Graepel, Thore II-1  
 Gretton, Arthur I-264  
 Grobelnik, Marko III-579  
 Gunopulos, Dimitrios II-195  
 Guns, Tias II-467
- Hachiya, Hirotaka I-474  
 Han, Jiawei I-2, I-570, II-35, II-337  
 Hanczar, Blaise I-490  
 Hannen, Matthias III-607  
 Hardoon, David Roi I-328, I-554  
 Haun, Stefan III-587  
 Hauskrecht, Milos I-87  
 Helleputte, Thibault I-522  
 Helma, Christoph II-353  
 Herbrich, Ralf II-1  
 Hernández-Lobato, Daniel I-522  
 Hernández-Lobato, José Miguel I-506, I-522  
 Hoi, Steven I-24  
 Holmes, Geoff I-135  
 Hottinen, Ari III-1  
 Huang, Heng II-451, III-451  
 Hüllermeier, Eyke I-215, I-280  
 Huopaniemi, Ilkka I-538  
 Hussain, Zakria I-554
- Jakubowicz, Jérémie I-248  
 Jansen, Timm III-607  
 Jebara, Tony III-261  
 Ji, Ming I-570  
 Jiang, Xiaoqian I-587  
 Joachims, Thorsten III-499  
 Jouve, Pierre-Emmanuel III-67  
 Jung, Tobias I-601
- Kabadjov, Mijail III-591  
 Kaelbling, Leslie Pack I-3  
 Kanhabua, Nattiya III-595  
 Kapernekas, Anastasios III-603  
 Kashima, Hisashi III-131  
 Kaski, Samuel I-538, III-370  
 Kasneci, Gjergji II-1  
 Kersting, Kristian II-178, II-434, III-402, III-499  
 Khan, Latifur II-337  
 Khardon, Roni III-418  
 Khot, Tushar II-434  
 Kim, Hyungsul II-35  
 Kim, Minyoung II-51  
 Kim, Sangkyum II-35  
 Kimura, Masahiro III-180  
 Klami, Arto III-370  
 Kloft, Marius II-66  
 Klos, Tomas II-82  
 Klug, Roland I-425  
 Kopanakis, Ioannis III-17  
 Kötter, Tobias III-587  
 Kramer, Stefan II-353, III-213  
 Krogmann, Klaus I-425  
 Kubera, Elżbieta II-97  
 Kuhlman, Chris J. II-111  
 Kuksa, Pavel II-128  
 Kunapuli, Gautam II-145
- Lacasse, Alexandre II-162  
 Lacerda, Anísio II-402  
 Lallich, Stéphane II-227  
 Lampos, Vasileios III-599  
 Lane, Terran I-119  
 Lang, Tobias II-178  
 Lappas, Theodoros II-195  
 Laskey, Kathryn Blackmond III-435  
 Laviolette, François II-162  
 Law, Edith II-211  
 Le Bras, Yannick II-227  
 Lee, Kee-Khoon I-231  
 Legrand, Anne-Claire I-72  
 Lenca, Philippe II-227  
 Leung, Alex P. I-554  
 Levine, Geoffrey II-243  
 Lillo-Le Louët, Agnès III-386  
 Ling, Charles X. I-377  
 Lippi, Marco II-259  
 Lipson, Hod I-4  
 Liu, Fei Tony II-274

- Loog, Marco II-291  
Lopes, Manuel II-385  
Loureiro, Antonio A.F. III-354  
Lowd, Daniel II-434  
Luaces, Oscar III-115  
Luo, Dijun II-451
- Maclin, Richard II-145  
Magdalinos, Panagis III-603  
Maillard, Odalric-Ambrym II-305  
Mampaey, Michael II-321  
Mannor, Shie I-312  
Marathe, Madhav V. II-111  
Marchand, Mario II-162  
Masud, Mohammad M. II-337  
Maunz, Andreas II-353  
McCallum, Andrew III-148  
Meert, Wannes II-369  
Melo, Francisco S. II-385  
Melville, Prem I-40  
Menezes, Guilherme Vale II-402  
Meyer, Patrick II-227  
Mitchell, Tom II-211  
Mladenčić, Dunja III-579, III-583  
Monreale, A. III-624  
Montañés, Elena III-115  
Moschitti, Alessandro III-229  
Mosci, Sofia II-418  
Motoda, Hiroshi III-180  
Mpiratsis, Alexandros III-603  
Müller, Emmanuel III-607  
Munos, Rémi II-305
- Nadif, Mohamed I-490  
Nalbantov, Georgi III-277  
Nanni, M. III-624  
Natarajan, Sriraam II-434  
Ng, Wee Keong I-24  
Nie, Feiping II-451  
Nijssen, Siegfried II-467  
Nikolaev, Nikolay III-277  
Ning, Xia II-128  
Nørvåg, Kjetil III-595  
Nürnbergger, Andreas III-587
- Ohara, Kouzou III-180  
Ong, Cheng Soon III-83  
Ong, Yew-Soon I-231  
Orešič, Matej I-538  
Oswald, Annahita I-151  
Ovsjanikov, Maks II-483
- Pahikkala, Tapio II-499  
Pajarinen, Joni III-1  
Paliouras, Georgios III-611  
Panagiotakis, Costas III-17  
Papantoniou, Katerina III-611  
Pappa, Gisele L. II-402  
Pasupa, Kitsuchart I-554  
Pavlovic, Vladimir II-51, II-128  
Pedreschi, D. III-624  
Pelekis, Nikos III-17  
Peltonen, Jaakko III-1  
Pennerath, Frédéric III-34  
Pernkopf, Franz III-50  
Pfahring, Bernhard I-135  
Pinelli, F. III-624  
Pisetta, Vincent III-67  
Plant, Claudia I-151, III-245  
Pletscher, Patrick III-83  
Plis, Sergey M. I-119  
Poggio, Tomaso I-5  
Prakash, B. Aditya III-99  
Precup, Doina I-200  
Protopapas, Pavlos III-418  
Provost, Foster I-40
- Qi, Yanjun II-128  
Quevedo, José Ramón III-115
- Rademaker, Michaël I-215  
Raś, Zbigniew II-97  
Ravi, S.S. II-111  
Raymond, Rudy III-131  
Ren, Jiangtao III-483, III-547  
Renso, C. III-624  
Riedel, Sebastian III-148  
Rinzivillo, S. III-624  
Rish, Irina III-196  
Rolet, Philippe III-293  
Rosasco, Lorenzo I-56, II-418  
Rosenkrantz, Daniel J. II-111  
Roth, Dan II-243  
Rückert, Ulrich II-66, III-563  
Ruggieri, Salvatore I-7  
Rusu, Delia III-579
- Saito, Kazumi III-180  
Salakoski, Tapio II-499  
Samdani, Rajhans II-243  
Santoro, Matteo II-418  
Scheinberg, Katya III-196

- Schiffer, Matthias III-607  
 Schmidhuber, Jürgen I-6, I-264  
 Seah, Chun-Wei I-231  
 Sebban, Marc I-72  
 Seeland, Madeleine III-213  
 Seidl, Thomas III-607  
 Settles, Burr II-211  
 Sevryn, Aliaksei III-229  
 Shabbeer, Amina II-145  
 Shao, Hao III-306  
 Shao, Junming III-245  
 Shavlik, Jude II-145, II-434  
 Shawe-Taylor, John I-328, I-554  
 Shivaswamy, Pannagadatta K. III-261  
 Skrzypiec, Magdalena II-97  
 Smirnov, Evgueni III-277  
 Snowsill, Tristan III-615  
 Steinberger, Josef III-591  
 Steinberger, Ralf III-591  
 Stone, Peter I-601  
 Subašić, Ilija III-619  
 Sugiyama, Masashi I-474  
 Sun, Yizhou I-570  
 Suviataival, Tommi I-538  
 Suzuki, Einoshin III-306  
 Sweeney, Latanya I-587
- Tadavani, Pooyan Khajehpour II-19  
 Tadepalli, Prasad I-344, II-434, III-467  
 Taghipour, Nima II-369  
 Teytaud, Olivier III-293  
 Theodoridis, Yannis III-17  
 Thiel, Kilian III-587  
 Thuraisingham, Bhavani II-337  
 Ting, Kai Ming II-274  
 Tong, Bin III-306  
 Tong, Hanghang III-99  
 Torquati, Massimo I-7  
 Toussaint, Marc II-178  
 Toussaint, Yannick III-386  
 Trasarti, R. III-624  
 Tsamardinos, Ioannis III-322  
 Tsang, Ivor W. I-231  
 Tsatsaronis, George III-611  
 Turgeon-Boutin, Francis II-162  
 Tuyls, Karl II-82
- Ukkonen, Antti III-338  
 Uusitalo, Mikko A. III-1
- Valler, Nicholas III-99  
 van Ahee, Gerrit Jan II-82  
 van der Goot, Erik III-591  
 Van Gael, Jürgen II-1  
 van Someren, Maarten I-296  
 Vaz de Melo, Pedro O.S. III-354  
 Vazirgiannis, Michalis III-603  
 Veloso, Adriano II-402  
 Vembu, Shankar II-243  
 Verri, Alessandro I-56, II-418  
 Verscheure, Olivier III-483, III-547  
 Vert, Jean-Philippe III-515  
 Viinikanoja, Jaakko III-370  
 Villa, Silvia II-418  
 Villerd, Jean III-386  
 Vovk, Vladimir III-531  
 Vreeken, Jilles II-321
- Wackersreuther, Bianca I-151  
 Wackersreuther, Peter I-151  
 Waegeman, Willem I-280, II-499  
 Wahabzada, Mirwaes III-402  
 Wang, Hao I-393  
 Wang, Hua III-451  
 Wang, Li-Lun II-243  
 Wang, Pu III-435  
 Wang, Yuyang III-418  
 Weninger, Tim II-35  
 Weston, Jason II-128  
 Wiczorkowska, Alicja II-97  
 Wilson, Aaron III-467  
 Wohlmayr, Michael III-50
- Xie, Sihong III-483  
 Xu, Congfu I-361  
 Xu, Zhao III-402, III-499
- Yang, Qiang III-547  
 Yang, Qinli III-245  
 Yao, Limin III-148  
 Yu, Jun I-344
- Zaslavskiy, Mikhail III-515  
 Zhang, Zhihua I-361  
 Zhdanov, Fedor III-531  
 Zhong, Erheng III-547  
 Zhou, Zhi-Hua II-274  
 Zighed, Djamel A. III-67  
 Zimmermann, Albrecht III-563  
 Ziviani, Nivio II-402